
# Intelligent Decision Support System for Selection of Learning Apps to Promote Critical Thinking in First Year Programming Students

by

Kesarie Singh

20928764

A dissertation submitted in fulfilment of the requirement for the Master of Information and Communications Technology degree

Faculty of Accounting and Informatics, Department of Information Technology, Postgraduate Studies, Durban University of Technology

2021

Supervisor: Dr N. Naicker (PhD)

Co-Supervisor: Dr M. Rajkoomar (PhD)

# DECLARATION

I, *Kesarie Singh*, declare that:

(i) The research reported in this dissertation, except where otherwise indicated, is my original research.

(ii) This dissertation has not been submitted for any degree or examination at any other university.

(iii) This dissertation does not contain other persons' data, pictures, graphs or other information, unless specifically acknowledged as being sourced from other persons.

(iv) This dissertation does not contain other persons' writing, unless specifically acknowledged as being sourced from other researchers. Where other written sources have been quoted, then:

- Their words have been re-written but the general information attributed to them has been referenced.
- Where their exact words have been used, their writing has been placed inside quotation marks, and referenced.

(v) This dissertation does not contain text, graphics or tables copied and pasted from the Internet, unless specifically acknowledged, and the source being detailed in the dissertation and in the Reference Section of this dissertation.

**Signature:** *Kesarie Singh*
Date: 30th August 2021

Approved Supervisor
Date: 9 December 2021

# ACKNOWLEDGEMENTS

This incredible journey of research has been fraught with an insatiable desire for self-discovery, combined with a multitude of mixed emotions. The route to reach the destination of this difficult road was made easier through the relentless encouragement and support of my supervisory team.

I wish to commend and express my genuine appreciation to Dr Nalen Naicker, my main supervisor, who not only motivated me through his positive energy and purposeful direction but his work ethic and disciplined approach, which set an amazing example for me to follow. Thank you to my co-supervisor Dr Mogie Rajkoomar, who was nurturing and understanding, as well as deeply caring and compassionate. Together they provided me with a complementary team of mentorship that exuded confidence and a search for knowledge through constructive feedback, while systematically enabling me to achieve more than my expectations within a relatively short period of time.

I would also like to express my sincere gratitude to the first year programming students, whose passion and excitement to be a part of this project made this study all the more worthwhile. Their curiosity, motivation and eagerness to participate, earned me the drive to fulfill my obligation as the researcher on this study.

A special thank you to my head of department, Dr Jeanette Wing, whose vision to support the circumstances I found myself in, provided me with the strength and opportunity to see this project to its completion.

Finally, thank you from the bottom of my heart to my partner Pradesh and my boys Jushen and Kimeshen, whose personal sacrifices afforded me the time and space to give this research the commitment it deserved.

# ABSTRACT

The disruption on higher education across the globe through adverse events such as student strikes, natural disasters and pandemics like Coronavirus Disease 2019 (Covid-19), can have catastrophic long-term effects on its sustainability unless there are significant and innovative research endeavours to mitigate this impact. Never before has the desire to keep learners motivated, engaged and successful in advancing their knowledge and perfecting their 21st century skills through student-centred, technology-rich teaching and learning practices, become so imperative across disciplines and job profiles. In particular, the problem associated with teaching programming to novice learners is further exacerbated by the complex and abstract nature of the field and the heavy reliance on 21st century skills such as critical and computational thinking. As a result, a kaleidoscope of research into programming self-efficacy, the complexity of the field, teaching methods and a variety of teaching tools, have emerged over the recent past. In response, the aim of this research was to use decision support systems to obtain student-centred preferences for learning applications to promote critical thinking in first year programming students. This study focuses on the visual programming environment and critical thinking as the gateway skill for student success in understanding programming. The extensive literature review has revealed an array of learning Apps and a multiplicity of critical thinking criteria that serve a diverse set of needs and expectations. Therefore, research to develop a multiple attribute decision-making model is needed to assist academics make quick, scientifically-proven, accurate and collective decisions about which learning App to choose from the range of available alternatives. The study used decision theory and Diane Halpern's 4-part model for critical thinking as the theoretical frameworks for evaluating and selecting learning Apps on the basis of its capacity to promote critical thinking. As a quantitative study, it randomly selected 217 students from a population of 500 programming students to rate four learning Apps, namely, Scratch, Alice, Blockly and MIT App Inventor, against critical thinking criteria and established Scratch as the App that best promotes critical thinking among first year programming students. Consequently, its distinctiveness lies in its use of the Fuzzy TOPSIS (Technique for Order Preference by Similarity to Ideal Situation) multi-criteria decision-making algorithm to rank criteria for critical thinking, calculate their weights on the basis of informed opinion and hence scientifically deduce the best rated App among the available alternatives that promote critical thinking among first year programming students. Furthermore, the study offers useful, insightful and ranked critical thinking criteria to formulate a user-friendly, transparent and evidence-based framework for App selection among academics teaching programming in higher education institutions.

# Table of Contents

# LIST OF ANNEXURES

# LIST OF FIGURES

# LIST OF TABLES

# LIST OF ABBREVIATIONS

| | | |
|---|---|---|
| 4IR | : | 4th Industrial Revolution |
| AHP | : | Analytic Hierarchy Process |
| Apps | : | Applications |
| CCTDI | : | California Critical Thinking Dispositions Inventory |
| CT | : | Computational Thinking |
| DTM | : | Decision Thinking Model |
| DUT | : | Durban University of Technology |
| ERD | : | Entity Relationship Diagram |
| FL | : | Fuzzy Logic |
| FPIS | : | Fuzzy Positive Ideal Solution |
| FNIS | : | Fuzzy Negative Ideal Solution |
| ICT | : | Information Communications Technology |
| IDE | : | Integrated Development Environment |
| IOS | : | iPhone Operating System |
| ITS | : | Integrated Tertiary Software |
| LMS | : | Learning Management System |
| MatLab | : | Matrix Laboratory |
| MCA | : | Multi-criteria Analysis |
| MCDA | : | Multi-criteria Decision Analysis |
| MCDM | : | Multi-criteria Decision-Making |
| MIT | : | Massachusetts Institute of Technology |
| P-VALUE | : | Probability Value |
| SPSS | : | Statistical Package for the Social Sciences |
| TOPSIS | : | Technique for Order Preference by Similarity to an Ideal Solution |
| UOT | : | University of Technology |
| VPE | : | Visual Programming Environment |
| WGCTA | : | Watson-Glaser Critical Thinking Appraisal |

# TERMINOLOGY

**Abstraction**: mental process of identifying the relevant parts and filtering out the inapt parts in the problem-solving process

**Algorithmic thinking**: to solve computer problems through a logical and systematic sequence of steps

**Block-based programming**: a type of visual programming that enables a person to use puzzle-pieces when designing a problem solution aimed to promote programming learning among novice students of various ages

**Computational thinking**: mental process used in problem-solving that involves abstraction, algorithmic thinking and pattern recognition

**Constructivist learning**: when learners actively contribute to the construction of their own knowledge using their experiences

**Critical thinking**: a higher-order mental process of questioning, analysing, synthesising, evaluating and applying logic, reasoning and diverse perspectives when finding creative and collaborative problem solutions

**Criterion**: a requirement or standard upon which a decision is made

**Decision theory**: knowledge domain that focusses on an approach to taking optimal decisions in an uncertain environment

**Fuzzy set theory**: mathematical framework devised to address ambiguity, vagueness and inaccuracy that emerges in decision-making judgements made in an uncertain environment

**Fuzzy TOPSIS**: a multi-criteria decision-making technique used to rank alternatives in an uncertain or fuzzy problem environment

**Learning App**: software tool designed to make learning easy

**Metacognition**: higher-order cognitive process of intentionally thinking about one's own thinking and learning

**Multi-criteria decision-making**: evaluating multiple conflicting specified criteria against several choices to determine the optimal alternative in a decision-making problem

**Text-based programming**: a type of programming that forces a person to learn the syntax of the language when designing the problem solution which is very close to how computers think

**Twenty-first century skills**: modern-day abilities such as critical thinking, collaboration and creativity that are required for success in the global economy

**Visual programming environment**: programming software that integrates graphical tools such as drop-down menus, icons and other screen components when designing the problem solution which is very close to how human beings think

# CHAPTER ONE

# INTRODUCTION TO THE STUDY

## 1.1 Context of the study

Learning to program can be perceived as both tedious and difficult to the novice programmer for reasons such as the abstract nature of computational thinking and the burdensome syntax and semantics associated with many programming languages (Xinogalos, Satratzemi and Malliarakis 2015; Buitrago Flórez *et al.* 2017; Medeiros, Ramalho and Falcão 2018). This has resulted in an upsurge of educational programming environments and tools aimed at stimulating the learners' interest, making programming accessible to people of all interests and ages and minimizing the challenges of understanding programming (Ivanović *et al.* 2017). A case study conducted by Pinto-Llorente *et al.* (2018) presented the impact of the visual programming environment (VPE) in promoting critical thinking and problem-solving skills, albeit among primary school learners. Other studies have revealed that visual programming environments support the constructivist approach to learning and contribute to enhancing one's skills of independent learning, creative thinking, problem-solving, reflection and collaboration (Dekhan, Xu and Tsoi 2013; Grover and Pea 2013; Kurilovas 2014; Papadakis and Orfanakis 2016; Harrison and Lee 2018; Ropii, Hardyanto and Ellianawati 2019; Jahnke and Liebscher 2020).

The twenty-first century witnessed the transition from the information age to the digital age to the knowledge economy, an era characterised by the rapid adoption of new technologies, requiring proficiencies in problem-solving, critical thinking, communication and collaboration (Dewi *et al.* 2019). According to Al-Mubaid, Abukmail and Bettayeb (2016), critical thinking and communication are two of the most crucial competences of the 21$^{st}$ century, while logistical inferencing, collaborative learning and creative and innovative thinking are considered its basic skills (Wang 2017; Zubaidah, Corebima and Mahanal 2018). Almerich *et al*. (2019) include adaptability, digital literacy, and computational thinking for an information- and knowledge-rich society. The 4th industrial revolution (4IR), while building on the digital revolution, are also demanding new skills from graduates. Industry requires students with creativity, originality, critical reasoning, problem- solving and analytical skills, all of which represent the trending graduate skills for the 2022 workforce (Van den Berg 2020).

According to Ahmad *et al*. (2017), the 3 highest levels of the widely-accepted Bloom's Taxonomy supports the notion that critical thinking skills are shaped by cognitive skills. The taxonomy contains six levels of learning arranged in order of difficulty as follows: "knowledge, comprehension, application, analysis, synthesis and evaluation" (Su, Ricci and Mnatsakanian 2016: 192). The highest level, evaluation, is a measure of the ability to make reasonable, reflective judgments (Smetanová, Drbalová and Vitáková 2015). Furthermore, there exists some consensus amongst notable authors, that critical thinking skills encompasses *many* cognitive skills. For example, Davies (2015) postulated that it also involves assessing statements, making logical arguments, constructing and interpreting inferences, as well as identifying flaws in reasoning. This is supported by Al-Mubaid, Abukmail and Bettayed (2016), who concluded that deep thinking can yield a better solution or an improved decision. From a practical point of view, deep thinking can be contextualised as an exercise in which a learner robustly solves a problem in different ways. This idea is also supported by the formally accepted definition of critical thinking as being "the mental process of actively and skilfully conceptualizing, applying, analysing, synthesizing, and evaluating information to reach an answer or conclusion" (Al-Mubaid, Abukmail and Bettayeb 2016: 69).

Critical thinking is also viewed as an active cognitive process that involves metacognition "thinking about one's thinking" (Davies 2015: 53). Su, Ricci and Mnatsakanian (2016) defined metacognition in the learning context as higher order thinking that leads to understanding, analysis and control of one's cognitive process. Consequently, metacognition asserts that a student will become aware of her own style of learning and will be able to recognize and implement problem-solving strategies. This approach is highlighted by Smetanová, Drbalová and Vitáková (2015: 725), who expanded the meaning of critical thinking as "thinking about our own thinking in a way that allows us to find out the strengths as well as weaknesses of our own thinking and then improve it".

Adding a further dimension to the afore-mentioned definition of critical thinking is one that includes one's disposition (Davies 2015), which can be defined as "a probing inquisitiveness, a zealous dedication to reason, and a hunger or eagerness for reliable information" (Ahmad *et al*. 2017: 232). This suggests that a critical thinker is identified by his enquiring mind, desire to solve problems and a passion to seek the necessary information using any available resources. Interestingly, other authors have noted the two interrelated components of critical

thinking as cognitive skills and dispositional attitudes (Halpern 1998; Abrami *et al.* 2015; Davies 2015; Quinn *et al.* 2020).

The Knowledge Society, combined with the unabated technological evolution, requires university students becoming proficient in these 21st century competences to become effective citizens of society (Almerich *et al.* 2019). Critical thinking is one of these. As it has been traditionally defined in the critical thinking movement, cognitive ability is key to critical thinking (Kules 2016; Ahmad *et al.* 2017; Almerich *et al.* 2019); however, this definition will not be complete without purposeful action. Therefore, performing reflective analyses, making logical arguments and constructing and evaluating judgements, are *only* the precursors to being future active agents of change.

In educating for criticality within the higher education landscape, much effort is needed to develop and sustain a higher-order sense of critical thinking that extends beyond educating into a process of facilitating radical growth and enabling transformative critique, to become a role model for what it means to be a critical being (Davies 2015).

## 1.2 Statement of the Problem

The demands associated with the teaching and learning of computer programming have attracted widespread research spanning several decades. The complex and difficult nature of the field (Elshiekh and Butgerit 2017; Medeiros, Ramalho and Falcão 2018; João *et al.* 2019; Tuloli, Latief and Rohandi 2019) that relies predominantly on the capacity for 21st century skills such as creative thinking and computational thinking, encompassing data abstraction, algorithmic, logic and reasoning skills, has therefore contributed to the high failure and dropout rates among novice learners (Medeiros, Ramalho and Falcão 2018; Figueiredo and García-Peñalvo 2019; Mathew, Malik and Tawafak 2019; Tuloli, Latief and Rohandi 2019). Furthermore, there is a lack of motivation among learners who find programming boring and tedious (Figueiredo and García-Peñalvo 2019; Mathew, Malik and Tawafak 2019). An additional challenge in the South African context is that learners whose mother tongue is not English experienced difficulty engaging in programming content (Anyango and Suleman 2018). A multitude of technological and pedagogical interventions have consequently emerged, such as constructionist techniques (Arawjo *et al.* 2017), problem-based collaborative learning (Chis *et al.* 2018), hands-on practice, visual programming environments (Arawjo *et al.* 2017;

Elshiekh and Butgerit 2017), digital gamification (Mathew, Malik and Tawafak 2019; Tuloli, Latief and Rohandi 2019) and mobile technologies, in response to the challenges experienced by novice programmers.

The literature has repeatedly revealed that visual block-based programming environments have a positively powerful impact on the performances of novices learning to program (João *et al.* 2019). An online learning application (App) becomes especially appropriate post COVID, where higher education institutions are strategically searching for alternate teaching methodologies to the traditional teaching approach. However, a number of learning Apps originated over the years with a variety of different characteristics, each serving a slightly different need (João *et al.* 2019). For example, visual environments designed specifically to nurture the development of pre-school children such as Bee-bot Emulator, Scratch Jr., Bricks and Lego Bits, and some activities of Tynker Hour of Code, include an interface design using blocks and symbolic language so that young learners who are yet to learn how to read and write, can be taught computational thinking skills (João *et al*. 2019). Notwithstanding, none of these Apps are deemed suitable for the study as they target the younger age group of learners and do not facilitate the teaching of content associated with the first year programming module. The focus of RiTa is to apply computational concepts and tools to create, analyse, edit, and generate text (Howe 2009). Berland *et al.* (2013) describes Scratch as a user-friendly application software utilizing design blocks to access the programming interface in order for "Scratchers" to create games, animations, stories, music videos, and many other applications that they can then share. Accordingly, the first problem to be addressed in the current study was to identify those Apps which are suitable for a first year programming course.

Choosing an optimal learning App is therefore a complex problem requiring an intelligent decision support system that will use multiple decision-makers to simultaneously consider multiple criteria such as the prior skills of the learner, the technical infrastructure of the learning environment and the 21$^{st}$ century skills required for success in the higher order thinking of computer programming (Kules 2016). There are no studies in the extant literature that use a decision support system that is able to find the optimal choice of learning App to suit the needs of the instructors and learners in acquiring improved student success not only to encourage their interest and excitement for programming, but also to acquire a proficiency to create their own applications. Therefore, sufficient scope exists for this research study.

## 1.3 Aim and Objectives

The **Aim** of the study is to propose a decision support system (DSS) that uses Fuzzy TOPSIS for multi-criteria decision-making to assist higher education institutions explicate preferences for learning applications to promote critical thinking in first year programming students.

This broadly stated aim was further divided into the following practical and achievable **Research Objectives** [RO]:

**[RO1]**: Synthesize the criteria that are used to promote critical thinking skills in students.

**[RO2]**: Assess the impact of visual programming on critical thinking

**[RO3]**: Select the most efficient Apps using multi-criteria decision-making for novice programming students using live data

**[RO4]**: Design an automated intelligent decision support prototype using Fuzzy TOPSIS.

## 1.4 Significance of the Study

This study was significant in the realm of multi-criteria decision analysis when applied to choosing learning Apps for the promotion of critical thinking skills for programming, and its contribution exists in its ability to empower programming lecturers across higher education institutions to use scientific methods to make decisions around key learning resources. The definition and promotion of critical thinking for learning has become increasingly important in the 21$^{st}$ century information and digital age, where organisations are competing for applicants who have the ability to investigate, apply, and transform data, to collaborate and innovate, to reflect, analyse and think critically, and to arrive at optimal decisions (Duran and Sendag 2012). The literature defines the higher order thinking processes associated with programming as closely aligned with the concept of computational thinking (CT), where problem-solving is implemented through the mental process of abstraction, logic, analysis, synthesis and constructive thinking (Romero, Lepage and Lille 2017; Pinto-Llorente *et al.* 2018; Turker and Pala 2019). It has been justified through various studies that CT skills should not just be accessible to everyone at a much younger age, but also have the potential to be applied to various disciplines and hence to be incorporated into the basic curriculum (Buitrago Flórez *et al.* 2017; Pinto-Llorente *et al.* 2018).

There are a variety of learning resources and learning Apps to develop CT (Weintrop and Wilensky 2017; Šiaulys 2020) and each of these include aspects of robotics (Buitrago Flórez *et al.* 2017; Atmatzidou *et al.* 2018; Figueiredo and García-Peñalvo 2019; João *et al.* 2019), gaming (Papadakis and Kalogiannakis 2017; Tuloli, Latief and Rohandi 2019), augmented reality simulation (Martín-Gutiérrez *et al.* 2015) or multimedia such as graphics, sound and animation, with the latter referred to as visual programing environments (Papadakis *et al.* 2016; João *et al.* 2019; Turker and Pala 2019). This study is significant as it aims to extend the application of multi-criteria decision-making using Fuzzy TOPSIS, with multiple decision-makers choosing the most appropriate learning App based on well-researched critical thinking criteria. This study took a holistic view by combining Diane Halpern's 4-part model for critical thinking and the multi-criteria decision-making (MCDM) framework for Fuzzy TOPSIS to resolve the problem of choosing the optimal learning App that promotes critical thinking for programming. The results of this study provide a foundation for programming lecturers to improve the overall performances of novice programming students with technological resources. The analysis and projected findings of this research study are anticipated to provide valuable insight, knowledge and guiding techniques for decision-makers at higher education institutions to enhance the students' enabling environments for novice programmers. In addition, the criteria to measure critical thinking for programming could provide ideas for future studies on improving the prospects and performances of higher education IT graduates.

## 1.5 Scope and Delimitations of Study

The study is restricted to first year students at a university of technology with limited or no prior exposure to programming.

## 1.6 Architecture of the Research Study

The theoretical framework underpinning the study, the processes undertaken by the researcher and the flow of the entire research study, are depicted in Figure 1.1 and expanded upon throughout this research report.

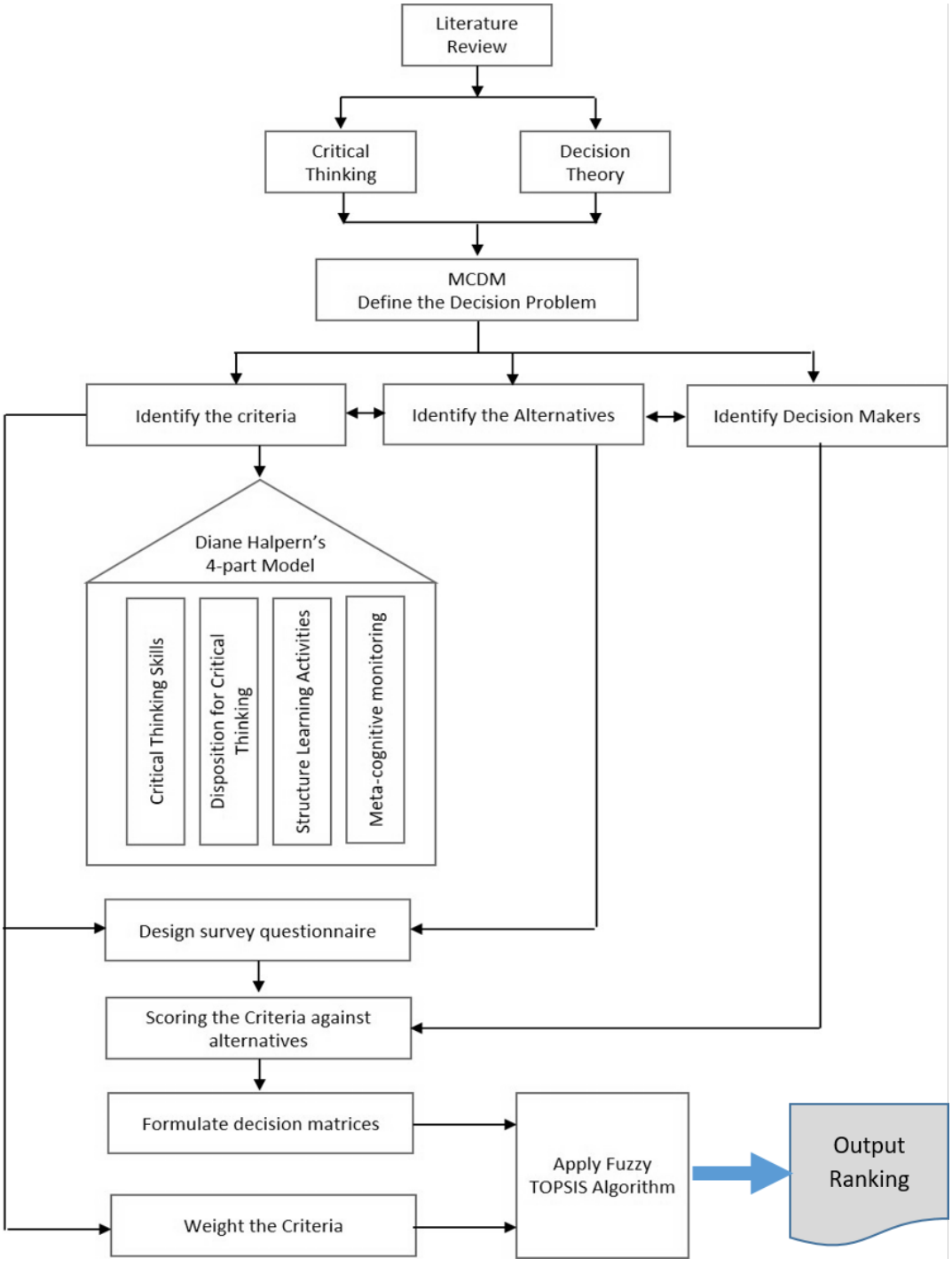


**Figure 1.1 The Architecture of the Research Study**

Source: Researcher's Own Construction (2021)

## 1.7 Structure of the Dissertation

This study is presented in six chapters, which are arranged in the following manner:

**Chapter One: Introduction to the Study**

A context of the research, rationale for the study, background and a general overview of the key elements underpinning the study are introduced. This chapter also highlights the significance of the research, its contributions and structure of the study.

**Chapter Two: Literature Review**

This chapter provides a comprehensive review of the literature in the area of critical thinking, its relevance and importance for programming, the measures for critical thinking and the selection of learning applications to promote critical thinking for programming.

**Chapter Three: Theoretical Frameworks and Research Methodology**

Section A of this chapter describes the theoretical frameworks underpinning the study in an effort to understand decision support systems and critical thinking within the realm of the aim and objectives of the research. Section B describes the research methodology that guided the entire research process in this study, including various aspects pertaining to sampling, data collection, ethics, validity and reliability. This became necessary due to the high impact of the theoretical frameworks on the entire research design developed by the researcher for the study.

**Chapter Four: Fuzzy TOPSIS**

The multi-criteria decision-making technique called Fuzzy TOPSIS methodology is used in the data analysis and is discussed in this chapter. An evaluation of selection techniques with a subsequent motivation for Fuzzy TOPSIS is provided. An analysis of Fuzzy TOPSIS compared to conventional TOPSIS is also made when applied to the selection of learning Apps.

**Chapter Five: Presentation of Results and Discussion**

This chapter discusses the statistical analysis of the data collected using the Fuzzy TOPSIS technique. The various steps in the technique will be applied in detail and analysed in MatLab R2020a.

**Chapter Six: Summary, Conclusions and Recommendations**

This chapter provides a summary of the whole study, including the developed framework and the results from the data analysis together with the contribution of the study to knowledge in research and practice. Furthermore, the chapter highlights some limitations of the study and suggests possible recommendations for future research.

## 1.8 Research Output

This study has contributed towards a research publication in the Volume 12 Issue 10 October 2021 issue of the International Journal of Advanced Computer Science and Applications (IJACSA). The article is published at the following link https://thesai.org/Publications/ViewPaper?Volume=12&Issue=10&Code=IJACSA&SerialNo=42.

## 1.9 Chapter Summary

This chapter highlights the research problem being investigated and addressed through a literature review on critical thinking and an appropriate theoretical framework for multi-criteria decision-making. The aim and objectives provide an important focus and direction for the study, while the architecture of the study presents a step-by-step implementation of the research design. The research methodology used to accomplish these objectives was presented with a brief explanation of the overall research process. This chapter also highlights the importance and impact of this research. In the next chapter, a more in-depth review of critical thinking when applied to the field of Information Technology and Programming will be undertaken to identify the measures for critical thinking and to rationalise the choice of learning Apps that will be used in solving the research problem.

# CHAPTER TWO
# LITERATURE REVIEW

## 2.1 Introduction

This chapter presents an overview of the key theoretical concepts in the literature related to critical thinking and the selection of learning Apps to promote critical thinking for programming. It starts by introducing the importance of critical thinking in the field of Computer Science and programming, and its relationship to computational thinking. In addition, the chapter discusses several criteria or measures for critical thinking when applied to the domain of programming and provides a brief description of the rationale for each of the criteria. The last section of the chapter offers a more detailed review of the literature on visual programming environments and provides a comprehensive rationalization for the four chosen learning Apps, based on their potential to promote critical thinking skills.

## 2.2 Importance of Critical Thinking

Critical thinking research dating back to the early 1900s have continued into the current century with various studies recognizing critical thinking as one of the important 21st century learning skill relevant across various disciplines (Al-Mubaid, Abukmail and Bettayeb (2016), Almerich *et al*. (2019), Duran and Sendag 2012). Critical thinking in the domain of computer programming supplements computational thinking. This is supported by Kules (2016), who defined computational thinking as the rational process of problem-solving, the mental process of making decisions and interacting with our world. It embodies the notions of "decomposition, algorithmic design, abstraction, generalization, evaluation and iteration from computer and information science" (Kules 2016: 1). Moreover, computational thinking was popularised by several authors in the literature (Buitrago Flórez *et al*. 2017; Rose, Habgood and Jay 2018; Agbo *et al.* 2019; Figueiredo and García-Peñalvo 2019; João *et al.* 2019), and described as the thinking that symbolizes the framing of computational problems and the formation of solutions.

Historically, research on computational skills dates back to studies on programming with children by Papert (1980). The author highlighted the essence of constructionism where learners actively construct knowledge through creative and innovative acts of accomplishment. For example, Wong and Cheung (2020) described the practical and concrete activity of building computational artefacts through active teamwork within the framework of problem-

solving as a creative process. Therefore, creative thinking is also somewhat interrelated with critical thinking.

Binkley *et al.* (2012) views critical thinking and problem thinking as synonymous with decision-making. Critical thinking skills are also useful in the computational problem-solving process, which requires decisions around the logical sequencing of instructions, extracting the key components of a complex problem scenario and developing and maintaining program structure, while considering the impact of society. Furthermore, Tee, Leong and Abdul Rahim (2018) noted that critical thinking skills also embody the intellectual strategies used by learners when thinking of alternate solutions, seeking innovative ideas and justifying these ideas through critical questions. Therefore, critical thinking (logical reasoning, problem-solving, making judgements and decisions) and programming (algorithm design, testing and debugging) are closely associated.

Rational and logical decisions throughout the various stages of programming requires sound critical thinking skills. This is supported by Irwanto, Rohaeti and Prodjosantoso (2018), whose article refers to studies that confirm a positive correlation between learners critical thinking skills and their problem-solving abilities. In a similar way, Kong and Wang (2020) use the A-E programming conceptual framework as a basis to demonstrate that the five stages of programming can be linked to their associated critical questions. Moreover, according to Wong and Cheung (2020), a learner's competence in critical thinking is directly proportional to the quality and robustness of the computational problem solutions they develop.

It has also been established by Ennis (1985), and reported in Chapter 1 that there exists two interconnected components required for the process of critical thinking, namely cognitive skills and dispositional attitudes which has been widely supported by various authors in the literature (Halpern 1998; Abrami et al. 2015; Davies 2015; Quinn et al. 2020; Facione 2010). The researcher has consequently captured the essence of this relationship within the domain of programming as illustrated in Figure 2.1. Interestingly, the last 3 cognitive skills featured in the diagram directly map to the 3 highest order levels of Blooms Taxonomy.

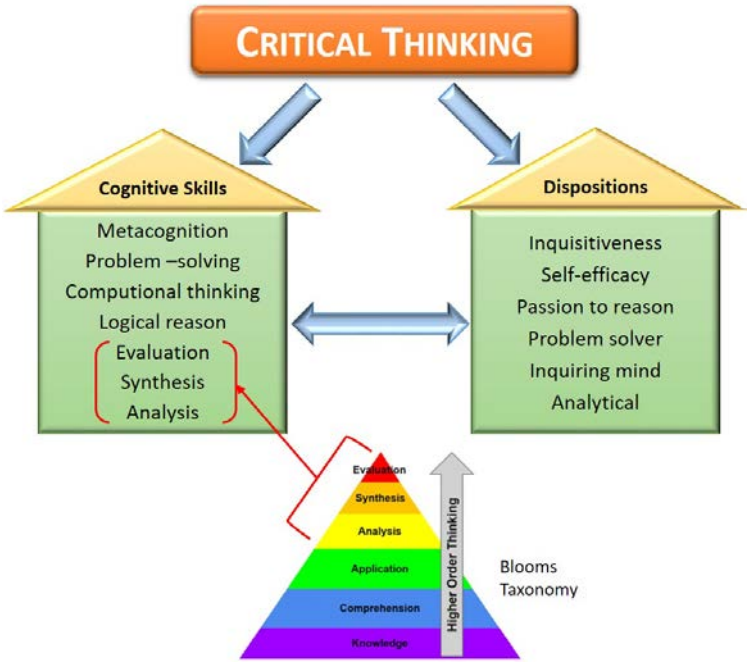


**Figure 2.1 Interconnected Components of Critical Thinking**
Source: Researcher's own construction (2021)

## 2.3 Measures for Critical Thinking

The extensive literature on critical thinking reveals the varying nature and context of its meaning. There exist philosophical, educational and psychological viewpoints in the way critical thinking is applied in the real world (Abrami et al. 2015; Hyytinen and Toom 2019). This study focusses on critical thinking as a deep higher order mental process with particular emphasis on the cognitive and metacognitive skills that complement analysis, synthesis, evaluation, logical reasoning and computational thinking, which are profound competences required in the discipline of computer science, more specifically computer programming, in an online higher education setting. Some of the more concrete learning outcomes derived from critical thinking include data abstraction, acquiring a comprehensive understanding of the problem, being able to develop multiple approaches to solving the problem, constructive engagement towards collaborative learning invoking review and reflection, and creatively producing a useful artefact through iterative evaluations (Romero, Lepage and Lille 2017; Agbo *et al.* 2019; Turker and Pala 2019).

Despite a significant investment on critical thinking research spanning a few decades, measuring the multi-dimensional nature of critical thinking remains a challenging task. A number of interventions emerged as a consequence of the various flaws that emerged from critical thinking assessments and the test results, such as unreliable test scores, ambiguous

definitions and non-comparable test forms (Liu *et al.* 2016). As one intervention, Shively, Stith and Rubenstein (2018) explored the assessment rubric as a means to assess critical thinking skills, but warned of the emphasis on the end product and the tendency to *only* superficially interpret process skills. The authors provide sample rubrics for independently measuring creative and critical thinking skills, and additionally illustrated the Design Thinking Model (DTM) used for complex project assessment, in which creative and critical thinking skills are embedded into the design process. As another intervention, Liu *et al.* (2016) introduced the HEIghten critical thinking test aimed at determining the cognitive and non-cognitive skills of college students, focussing more specifically on the two crucial aspects of critical thinking, analysis and synthesis.

Critical thinking is fundamentally important for increasing problem-solving skills and their association to computational thinking and creative thinking. Therefore, the following criteria will represent the most appropriate measures for critical thinking within the context of computer programming.

### 2.3.1 Feedback

An interactive enabling learning environment that cultivates two-way constructive dialogue of critique and sound purposeful logic and reasoning in which learners give and receive input on their problem solutions, will institute a measure of critical thinking. Peer-review, flipped classroom, inquiry-method through higher-order questioning and group work are some of the teaching strategies that demonstrate such an environment (McMahon 2009; Abrami *et al.* 2015; Davenport 2018; Turker and Pala 2019). Learners can use probing questions as a means to provide an assessment or review of a peer's attempt at a solution (Swart 2017).

### 2.3.2 Nature and Complexity of the Problem-solving

Evidence of critical thinking is embodied in multiple levels of abstraction, algorithmic thinking and analysis of the problem specification, which comprise the metacognitive and computational subsets of critical thinking (Buitrago Flórez *et al.* 2017). Therefore, assessing critical thinking may entail large authentic complex problem scenarios, where extracting the detail (abstraction) towards solution finding becomes the key focus. Analysis and synthesis, two of Bloom's higher order thinking skills, are required to logically interpret, reason and obtain a series of steps that will need to be accurately communicated to the computer. A study

by Abrami *et al.* (2015) revealed that interaction among peers, and between instructors and learners, together with authentic problem examples, enhanced learners' critical thinking.

### 2.3.3 Alternate Solutions

Critical thinking also entails thinking about one's thinking (metacognition), hence an opportunity for learners to provide alternate solutions to case studies will invoke deep thinking. An attempt to justify the choice of solution also entails a deeper mental activity. It can therefore be concluded that exploring the consequences of a solution or evaluating a solution from various points of view such as from that of the user or from the technical efficiency of the program, like speed and storage capacity, will also initiate higher-order thinking. Opportunities for debugging or correcting bad programming practices will require a deeper understanding and therefore constitute elements of critical thinking (Rose, Habgood and Jay 2018). Analytical thinking also involves comparing two or more solutions and providing a rationale for the best solution.

### 2.3.4 Visual Environment

In a study with secondary education students, McMahon (2009) explored the relationship between critical thinking and ICT competency and noted a positive correlation. In particular, González-González and Jiménez-Zarco (2015) reported that technology-enriched educational experiences enhance university students' critical thinking skills. A visual multimedia online learning environment that simulated the active constructivist learning paradigm developed independent thinking, critical thinking, problem-solving, collaboration and creativity among learners. For example, in a study by Knoesel (2017) on nursing graduates to investigate the impact of simulation on critical thinking skills, positive critical thinking scores were found to emerge after using simulation as an active teaching-learning strategy.

### 2.3.5 Creativity

Creativity refers to the propensity to use novel ideas to construct something useful and invaluable. Creativity in the context of computer science and programming refers to a process that generates an inventive and valuable artefact produced from the software development process. Accordingly, collaboration and creativity generally complement one another. This view is supported by Romero, Lepage and Lille (2017) who also link problem-solving and technology with the creative process. They argue that the active collaborative, design and creative processes form part of the cognitive and metacognitive strategies, inherent in

computational thinking. Shively, Stith and Rubenstein (2018) used a rubric to assess creativity and included flexibility (different types of ideas), originality (uniqueness of ideas), fluency (quantity of ideas) and elaboration (building upon ideas) as assessment criteria. These were extended to include usefulness as a component of creativity, indicating that creativity must serve a purpose within a social context (Shively *et al.* 2018). Consequently, the artefact in programming will represent the creative design and deliverable of the critical thinking process.

### 2.3.6 Logic and Reasoning

In the context of programming, clearly and precisely interpreting a problem statement, selectively using what is relevant in the problem solution and applying the appropriate knowledge concepts, represent logic and reasoning. The process flow and logical sequencing of steps is a culmination of connecting the problem statement with the relevant content knowledge through analysis and synthesis (Bernstein and Isaac 2018).

### 2.3.7 Interpretation of Data

One of the skills tested by the Watson-Glaser Critical Thinking Appraisal (WGCTA) is the interpretation of data. In the context of programming, a critical thinker would use pattern recognition in the data to synthesise a solution.

### 2.3.8 Deduction and Inference

According to Spicer and Hanks (1995), deduction and inference are two types of skills that measure critical thinking. Complex problem statements would require a clear identification of the problem that refutes any biases (Bernstein and Isaac 2018), a deeper understanding of its requirements and the ability to make inferences that would successfully lead to a working solution.

### 2.3.9 Dialogue

Effective critical thinking instruction would engage the learners in dialogue (Chen *et al.* 2019). This implies that learners will need to provide justifications for their rationale, disprove the arguments of their peers where necessary, and remember the knowledge content that will substantiate their logical reasoning.

### 2.3.10 Context

The learning environment must provide a context for critical thinking and opportunities for practice (Chen *et al.* 2019). Learners must be allowed multiple opportunities to self-correct

their solutions through the process of debugging (Al-Mubaid, Abukmail and Bettayeb 2016). A setting that includes peer review will enable learners to see a problem from different perspectives, reflect on their own solutions based on the comments from peers, and elaborate on their thinking, therefore providing a platform to strengthen the opportunity for critical thinking (Wang 2017). A public space for learners to share their work would further encourage reflection and deep learning (Pierce 2011).

### 2.3.11 Problem-solving

A recent study by Atmatzidou *et al.* (2018) concluded that when learners are provided with strong guidance in solving problems, it can have a positive impact on their problem-solving and metacognitive skills. Additionally, the problem scenario specifications that are embedded in programming and problem-solving represent a catalyst for critical thinking development (Pierce 2011).

### 2.3.12 Multimedia

According to Turker and Pala (2019), a visual block-based programming environment that includes rich media such as animation, sound, painting and music, develops a learners problem-solving, algorithmic thinking, creativity and computational skills.

### 2.3.13 Collaboration

Peer instruction and project-based learning using authentic real-life case studies offer an active learning technique that enables learners to collaborate with their peers in an organised, cooperative and supportive learning environment (Buitrago Flórez *et al.* 2017; Davenport 2018).

### 2.3.14 Disposition

The California Critical Thinking Dispositions Inventory (CCTDI) is a survey tool designed to assess the extent to which a person demonstrates the mind set for critical thinking, and includes analyticity, inquisitiveness, maturity, open-mindedness, self-confidence, systematicity and truth-seeking (Spicer and Hanks 1995).

## 2.4 Measures of Critical Thinking for Programming

Logical reasoning, problem-solving, application, reflective analysis and synthesis are some of the necessary critical thinking skills required in the programming process (algorithm design, testing and debugging). In an article by Irwanto, Rohaeti and Prodjosantoso (2018), the authors

demonstrate a positive relationship between the learners critical thinking skills and their problem-solving abilities. According to Kong and Wang (2020) the five stages of programming generally yield robust solutions to programming problems when the correct critical questions are being asked. Therefore, this section aims to explore the measures of critical thinking required for first year programming.

The testing and debugging stage of programming requires the learner to reflect on the current problem solution to determine and correct its errors. Possible measures for critical thinking will therefore include basic logic and reasoning skills as well as deduction and inferencing (Bernstein and Isaac 2018). Interpretation skills, collaboration and innovation are key critical thinking skills and predispositions required in the requirements and specifications stage of programming. According to Davenport (2018) encouraging active engagement and collaboration increases learners chances of success in the problem solving aspect of programming. The design stage of programming will require the learners creativity, cognitive ability and synthesis skills while the implementation stage of programming relies strongly on the learners ability to apply ones knowledge, metacognition, use of multimedia and evaluation. In an article by Shively, Stith and Rubenstein (2018), the authors identified innovation and flexibility as assessment criteria to measure creativity.

### 2.5 Selection of Learning Applications to Promote Critical Thinking for Programming

The growth of learning Apps to supplement higher education teaching and learning practices is expected to escalate exponentially during these unprecedented times of the global pandemic. The shift from face-to-face teacher dependence to maximising the potential of the learning App, especially on mobile devices, serves more than just introducing first-time programmers to computer programming (Xu *et al.* 2019). It is also purported to be an engaging tool to enhance student learning experiences (Nguyen, Barton and Nguyen 2015). A study by Xinogalos, Satratzemi and Malliarakis (2015) further confirms that an educational programming environment has the added benefit of assisting students who previously struggled to cope with an introduction to programming module in the traditional text-based programming environment. Therefore, the choice of learning App and the type of learning App is of paramount importance.

Kelleher and Pausch (2005) discuss a myriad of 86 programming environments intended to make the understanding of programming to both children and novice adults more manageable.

Some of these date back to the 1960s, with examples illustrating the strategy that environments use for visualisation. More recently in a similar analysis, João *et al.* (2019) explored the pedagogical usefulness of 26 most popular VPEs across 27 categories. In parallel, a study undertaken by Šiaulys (2020) offers a categorisation of visual programming environments arranged according to the age of prospective users and types of concepts covered. However, the current study aims to review those learning Apps that aim to promote critical thinking among programming students. Therefore, the study focusses on visual programming environments that remove unnecessary complex syntax so that the learner can focus on logic and problem-solving, uses graphics and visual contexts to promote creativity, enables immediate results of their creations through feedback and supports an open sharing platform for collaboration and reflective thinking. Consequently, this study draws heavily on both Joao and Šiaulys in evaluating which Apps are best suited for the study.

One of the earlier responses to making programming available and of interest to people of varying ages, was visual programming using flowcharts (Noone and Mooney 2018). A study by Greyling, Cilliers and Calitz (2006) assessed the use of B# which enables learners to build a flowchart, while code is generated concurrently in a text-based programming language like Pascal or Java. Although it worked well to introduce basic concepts to novice programmers, higher-order concepts became progressively difficult and complex to implement. Furthermore, initial assessments revealed that few learners were successful in developing adequate coding skills using B# (Noone and Mooney 2018). Another example shown in the literature is RAPTOR, which allows learners to represent their problem solutions visually without the burden of needing to learn the syntax of a text-based language. However, in both instances, questions remain on whether these learning environments make it less difficult for students to migrate to the authentic text-based programming languages.

A review of the literature precipitated the researcher to capture the relationship between the characteristics of visual programming environments and critical thinking skills as illustrated in Figure 2.2. The characteristic of the VPE on the left, provides the learner with an opportunity to use the critical thinking skill on the right, indicated by the arrow. This conceptual understanding of the VPE as needing to promote critical thinking skills, assisted the researcher to appropriately focus the decision around the choice of learning Apps to be used in this study.

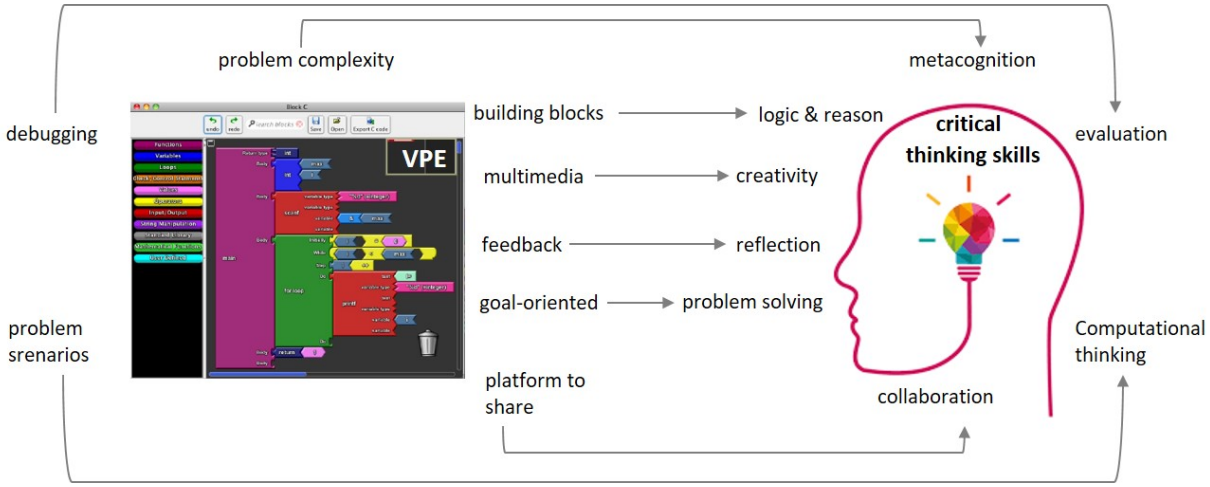


**Figure 2.2 Relationship between the VPE and Critical Thinking Skills**
Source: Researcher's own construction (2021)

Scratch (Bala and Alacapinar 2021, Snap! (Ball *et al*. 2019), and Blockly (Bau *et al.* 2017; Pasternak, Fenichel and Marshall 2017) represent the first in its generation of tools to provide an enabling environment to allow learners to use logic and reasoning to drag-and-drop blocks together to form scripts. Many of the VPE activities use gamification such as games and puzzles in the problem-solving process. Blockly Games is a visual block-based environment developed in 2012 that teaches children from ages 8 and above the concepts of conditions, loops, variables and functions through game puzzles using moving sprites, animation, drawings and music (João *et al.* 2019). Code Monkey developed in 2014 allows children to develop their own games using moving sprites (Šiaulys 2020). Kodu, on the other hand enable children to create games on the personal computer (PC) using picture blocks; however, it also allows a community of novices to share their efforts (João *et al.* 2019). The former two environments require the continuous use of the internet and can be run on computers and smartphones, while Kodu runs on Windows on a computer. All these visual Apps are freely available.

Scratch, BlueJ, Alice, objectKarel and MIT App inventor were investigated in a detailed study undertaken by Xinogalos, Satratzemi and Malliarakis (2015). BlueJ is a visual development environment supporting object-oriented programming in Java, enabling classes to be tested individually (McIver 2002). Some of the advantages of the BlueJ environment include the visualization of object-oriented programming (Fincher *et al.*) concepts and the interactive creation of objects and method calling, while the drawbacks include the pre-requisite of prior knowledge, the poor graphics and its failure to support program development (Xinogalos, Satratzemi and Malliarakis 2015).

The same study also revealed that the object Karel used one robotic environment to teach various concepts, was not user-friendly, difficult to follow at the beginning, and did not enable many programming capabilities. Scratch was perceived as simple to use, with nice graphics, interactive and entertaining, enabling many programming capabilities, promoting critical thinking, an easy creation of games which facilitated a community of sharing. The learners' feedback pertaining to Alice in the same study revealed its ease of use, interactivity, interesting and stimulating environment, 3-dimensional (3D) functionality, promoting creative thinking, graphical visualisation of program development, while simultaneously highlighting its relatively complex environment. Various advantages have been identified with MIT App Inventor, such as its simplicity and exciting functionality in designing mobile Apps, enabling good quality App design and implementation, easy testing, good tutorials and promoting creative thinking, while the disadvantages have been listed as too few graphics and being inappropriate for novice programmers.

A number of visual programming environments (VPE) can be considered for programming with robots. According to João *et al.* (2019), the two most appropriate VPEs are Micro-bit and mBlock. Tinker IDE allows programmers to program drones based on robot typology and architecture. Additionally, MIT App Inventor, Thunkable and Kodular focus on the development of mobile applications. Code Combat also poses an alternative which focusses on advanced programming concepts in a game environment. However, for the purposes of this study, the chosen Apps must reflect a best fit for the curriculum of the first year programming module, which is offered to novice programmers and does not include robotics programming.

The popularity of the visual block-based programming environment has seen growing development in various other domains. Various visual programming development environments are mentioned in the literature, such as the mobile application development environment using MIT App Inventor and Pocket Code (Weintrop and Wilensky 2015); the modelling and simulation tools using CodeBug, NetLogo and Open Roberta Lab (Šiaulys 2020); game-based learning environments like Code Combat, Cargo-Bot and Kodu (João *et al.* 2019; Kaya and Yildiz 2019), and creative and artistic drawing tools like Pencil Code (Bau *et al.* 2017).

Other learning Apps also considered include the graphic IOS environment of GameSalad and Hour of Code (Roy, Rousse and DeMeritt 2012; Dekhan, Xu and Tsoi 2013; Hutchison, Nadolny and Estapa 2016). In 2012, Code.org was created as a non-profit organization operating predominantly in the virtual space. Their popular initiative termed Hour of Code is intended to encourage learners to complete short programming tutorials. This initiative was predicated by Code.org's philosophy of making opportunities of learning to code available to every student in every school. A study conducted by Liu, Wimmer and Rada (2016) showed that the Hour of Code initiative improved students' attitudes toward programming, but warned that interaction with the environment *alone* does not necessarily enhance their skills for coding.

According to Howe (2009) Scratch is a graphical-programming environment aimed at the younger age group, and provides a platform for users to create animated games and interactive stories which can be shared among other users on the internet for their creative and collaborative endeavours. It also engages learners to reason systematically in a meaningful and social context. Alice offers an interactive IDE in which interactive narratives are created by learners by dragging and dropping graphic tiles on a canvas. These correspond to standard statements in object-oriented programming languages like Java, C++, or C#. It engages students' critical and creative thinking through a 3D programming environment that facilitates the creation of animations for storytelling, interactive games, and web videos. The study by Howe (2009) also showed that the features offered by Scratch make it an excellent VPE for both learning and enhancing the development of computational thinking skills.

Alice, Blockly and Scratch are similar in that all offer an interactive, graphic, simulated and visual environment for the purpose of introducing abstract programming concepts to novices. However, Alice is a 3D drag-and-drop development environment enabling animations, Blockly supports games and simulations, and Scratch is a block-based multi-media rich networked environment (Fincher *et al.* 2010; Xinogalos, Satratzemi and Malliarakis 2015; Buitrago Flórez *et al.* 2017; Noone and Mooney 2018; Xu *et al.* 2019). MIT App Inventor and Scratch support graphics and sound use building blocks for problem-solving, thus eliminating the possible syntax errors that may emerge from novices needing to remember the rules of a language (Papadakis *et al.* 2016). Scratch 3.0 enables project development on the browser and supports while MIT App Inventor allows for the development of mobile Apps on the smartphone in real-time. These VPEs provide a platform that facilitates immediate feedback, collaborative

learning and creative thinking through animations, text, stories and music which many students perceive as pertinent and exciting (Papadakis and Orfanakis 2016).

In the context of this study, various visual programming environments were considered and four were chosen relative to their potential to promote critical thinking skills, to support learning by novice programmers, their alignment to the introductory programming module content and their accessibility. In particular, an introductory programming environment should motivate students through active engagement connected to their interests, offer an exciting user-friendly visual interface for real-life relevant problem-solving, support mobility, and promote and develop computational skills rather than focus on complex language syntax (Xinogalos, Satratzemi and Malliarakis 2015). Therefore, the four open source learning Apps chosen for this study are the Scratch (for interactive stories and games), Alice (for 3D animation and story-telling), MIT App Inventor (for mobile applications development in real-time), and Blockly (the visually interactive environment). The suitability of these Apps is discussed throughout this chapter supported by a number of authors, including Fincher *et al.* (2010), Grover and Pea (2013), Bau *et al.* (2017), Papadakis and Orfanakis (2018), and Ropii, Hardyanto and Ellianawati (2019).

## 2.6 Chapter Summary

This chapter explored the concept of critical thinking when applied to computer programming, amidst the diverse and multifaceted definitions that exist within the literature in the psychological, educational and philosophical contexts. An extensive synthesis of the literature was required to formulate and rationalize the various criteria that could be used to measure critical thinking for programming. The chapter includes an overview of visual programming environments, their purpose and benefits, with the aid of examples and analogies. Four learning Apps that purport to support major aspects of critical thinking criteria, were selected by the researcher. Analysing these learning Apps, namely, Blockly, Alice, Scratch and MIT App Inventor, revealed their adequacy and relevance when introducing programming to novice learners. The next chapter will introduce the two key theoretical frameworks underpinning the study as well as discuss in detail the methodology charted to realize the objectives of this research study.

# CHAPTER THREE

# THEORETICAL FRAMEWORK AND RESEARCH METHODOLOGY

## 3.1 Introduction

This chapter comprises two sections: section A, describes the two theoretical frameworks applicable to the study and section B discusses the research methodology and design for the study. Decision theory and Diane Halpern's 4-part model are highlighted as the two main theoretical frameworks in the literature that underpins this study, providing insights into the research domain. Section A starts by introducing decision theory as a discipline of knowledge and provides a strong rationale for choosing the multi-criteria decision analysis model as the decision-making technique to be applied in solving the decision problem. In addition, the section discusses Diane Halpern's 4-part model for critical thinking instruction as a guide to determining the various measures for critical thinking for the purpose of being evaluated against the four learning Apps through a quantitative research design, is elaborated in section B.

Section B presents the research design applicable to the study and systematically applies the design in methodological steps for the purpose of meeting the objectives of the study. A survey questionnaire is used as the research instrument to collect data about the four learning Apps when evaluated against various critical thinking criteria. Therefore, details pertaining to sampling, data collection, ethics, validity and reliability are subsequently addressed in this section of the chapter.

## SECTION A: THEORETICAL FRAMEWORK

## 3.2 Decision Theory and its Application to the Study

Universities worldwide are faced with strategic decision-making when introducing educational technology post Covid-19 and beyond. Decisions around the choice of educational technologies become significant and may have far-reaching consequences unless critical factors such as availability of resources, future needs, challenges and risk are considered against a climate of uncertainty (Bradley 2014). The rapidly increasing need for technology to support learning has precipitated the need for important and convincing decisions when applied

to the use of educational learning tools, and these decisions will often have to be made without all knowledge of the outcomes.

Decision theory as a discipline of knowledge, established and advanced through the years arising from significant contributions within various fields of disciplines, will be used as the theoretical framework to guide the decision-making process. Decision theorists focus on how choices are made in a non-random way and therefore embody purposeful action where options are available (Hansson 2011). Earlier decision-making models was started by Dewey (1910) and modified by Simon (1960) to reflect the decision-making process as sequential stages. This model was applied in a healthcare setting in an article by Liu (2014).

Notwithstanding the above, the model was criticized by several authors who disagreed with the view that the decision-making stages are executed in sequence and shared the perspective that the stages are in parallel rather than in sequence. This consequently gave rise to a more realistic non-sequential model proposed by Mintzberg, Raisinghani, and Theoret (1976). These authors proposed a three-phase seven-routine plan of action with identification, development and selection representing the major phases of decision-making. The last phase of this model encompasses evaluation and selection concluding with making a distinctive choice among the available alternatives. According to Bradley (2014), this process, defined in the literature as normative decision theory, addresses the questions of what and how decisions should be made: how each alternative should be evaluated, what criteria should be selected, and what procedures should be followed. These are illustrated in Figure 1 as the information around the decision-making process (Aristodemou and Tietze 2019).

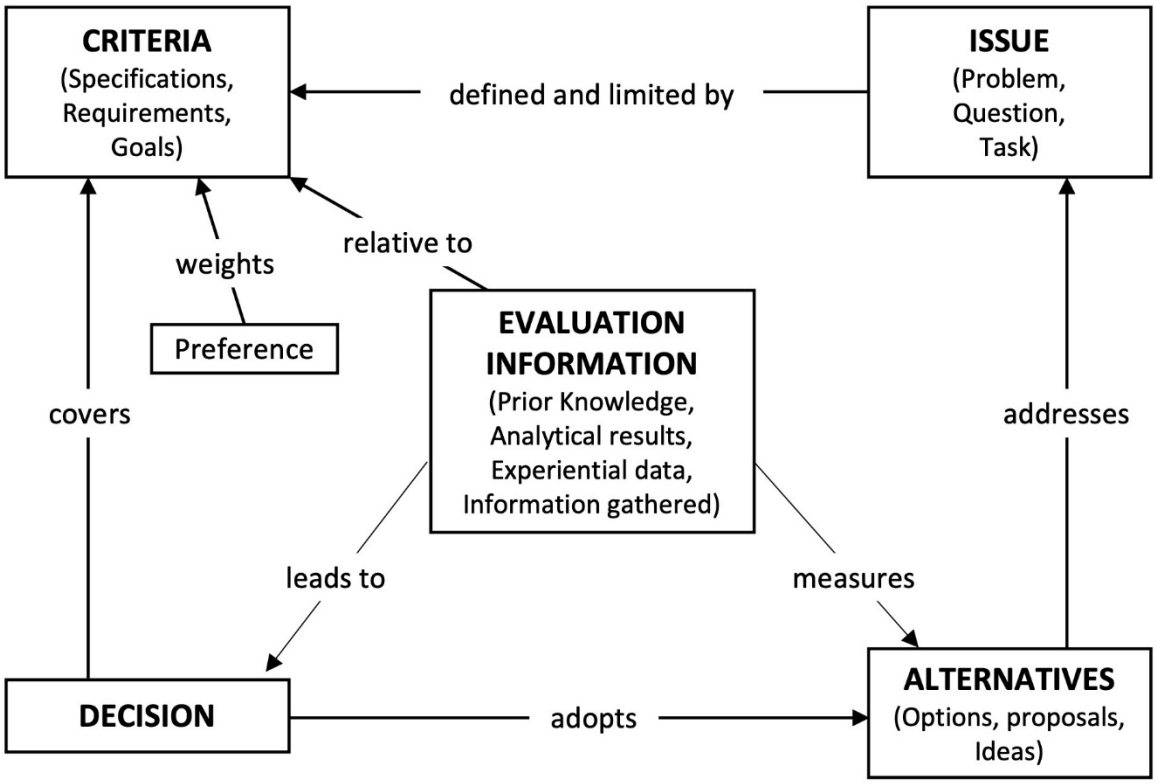


**Figure 3.1 Decision-making Process**
Source: Extracted from Aristodemou and Tietze (2019)

Normative decision theory in particular refers to norms or decision rules intended to provide guidance for decision-making, and includes aspects around uncertainty and lack of information around the decision problem. This is especially pertinent to the selection of learning Apps where risk and uncertainty is prevalent in the early stages of new technology adoption. Normative decision theory also enables the framing of decision problems in a way that identifies different alternatives, with each alternative evaluated relative to a pre-determined set of unique criteria, in the form of an outcome or consequence. The standard representation of the decision problem requires that the possible outcomes of each alternative against each criterion be specified as a numerical utility value within a utility matrix (Hansson 2011). The rows of the matrix represent the alternatives and the columns indicate the criteria. Therefore, in any decision problem, the alternatives are measured against information that represents the criteria used to stipulate the problem. Consequently, the decision will emerge through an evaluation process that assesses each alternative against all the criteria.

In general, if $A_1$ through $A_m$ represent the $m$ choices available to the decision-maker, $s_1$ through $s_n$ are the $n$ selection criteria, and $C_{11}$ through $C_{mn}$ are the $m$ x $n$ consequences that might follow from the choice, then a decision problem can be represented by a state-consequence matrix, as illustrated in Table 3.1.

| | States of the world | | | |
|---|---|---|---|---|
| Options | $s_1$ | $s_2$ | ... | $s_n$ |
| $A_1$ | $C_{11}$ | $C_{12}$ | ... | $C_{1n}$ |
| ... | ... | ... | ... | ... |
| $A_m$ | $C_{m1}$ | $C_{21}$ | ... | $C_{mn}$ |

**Table 3.1 State-Consequence Matrix**
Source: Extracted from Bradley (2014)

In classical decision theory, people are considered to be rational agents seeking to maximize their subjective utility with each decision. This implies that the decision-maker must seek the result of a combination of two assessment processes, namely, discerning the value of choosing an alternative and assessing the probability of an outcome after pursuing each option. The option chosen is the one that offers the optimal combination of probability and utility (Bradley 2014). However, in decision theory, when the decision-maker is unsure of the outcome when an alternative is chosen, then the decision is being made under non-certainty, which is usually divided into further categories, such as risk, uncertainty, and ignorance (Hansson 2011).

The theory contends that subjectivity associated with the decision-maker assessing the value of an alternative and estimating the probability of the outcome for each alternative, can result in risk. Notwithstanding, the theory makes the assumption that human rationality always prevails. The strength of the decision-making will rely on the likelihood of the outcomes associated with an option being predicted (risk) or the probability of the decision-maker's judgements relative to the information they are made against (uncertainty), and the possibility of there being no known information surrounding the consequences (ignorance). Intermediate cases are also crucial, especially in the event of the decision-maker being partly ignorant of the germane probabilities - a position generally regarded as ambiguity (Bradley 2014).

The decision problem for this study uses a population of 500 first year programming students to evaluate four learning Apps against several criteria derived from the literature as contributing to the promotion of critical thinking. Decision theory has been identified as the theoretical framework underpinning the study. This is aimed at following a defined list of procedures to analyse large quantities of data within an environment climate of uncertainty, risk and ignorance in analysing the complex decision to identify the most preferred option. The multi-criteria analysis (MCA) approach supports the processing of a large amount of information (quantitative) in a consistent way, promotes decision-making in a transparent way and permits multiple stakeholders in the decision-making process which can be fully verified (Ellis, Deutsch and Legret 2006). The purpose of the full MCA approach is to assist academic decision-makers to identify the most optimal alternative through ranking of carefully chosen criteria that will promote critical thinking in first year programming students. Figure 3.2 illustrates the multi-criteria decision analysis model that has been selected as the decision-making technique to determine the optimal choice.

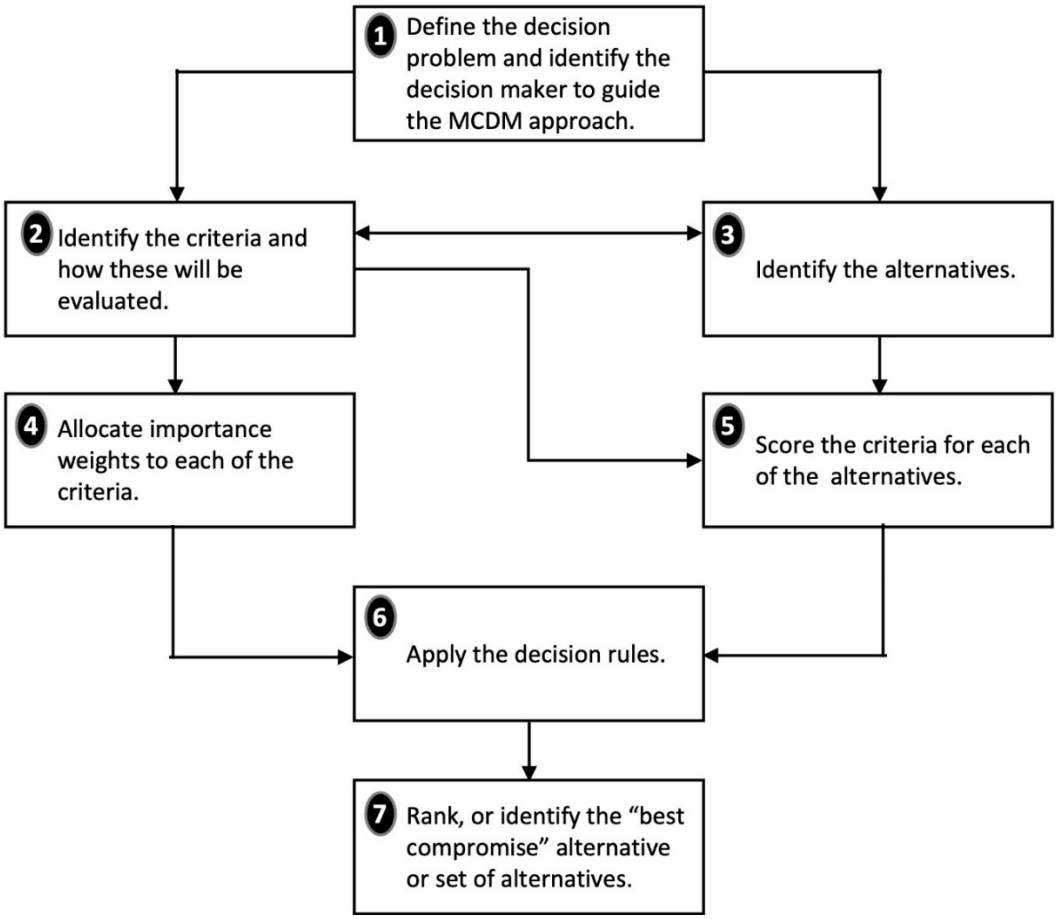


**Figure 3.2 Multi-criteria decision analysis model**
Source: Extracted from Aristodemou and Tietze (2019)

## 3.3 Diane Halpern's 4-Part Model on Critical Thinking and its Application to the Study

In present society, technology plays a fundamental role in higher education and has become even more imperative during unprecedented times such as the Covid-19 pandemic. Equally so, the expansive literature on critical thinking research spanning several decades has highlighted its significance in various forms across disciplines, with many higher education institutions across the globe initiating proposals to explicitly include critical thinking as an instructional module or to implicitly include critical thinking as a teaching outcome. According to Halpern (1998), the ability and desire to think critically will hold students in good stead decades from now, when the current job and technologies they now know, may not exist, but the capacity to intellectually engage in rational thought and the disposition to participate in the vigorous process of thinking, will always remain one of the fundamental and most anticipated outcomes in education.

Halpern (1999: 70) presented a definition of critical thinking as "the use of those cognitive skills or strategies that increase the probability of a desirable outcome. It is used to describe thinking that is purposeful, reasoned, and goal directed—the kind of thinking involved in solving problems, formulating inferences, calculating likelihoods, and making decisions". This definition may be applied to the context of programming where to think critically would mean using the correct rationale to make decisions about the choice of instructions and its sequence designed to deliver a particular output. There is also widespread evidence in the research that critical thinking skills can be identified, taught, learned and measured (Halpern 1998; Ng'ambi and Johnston 2006; McMahon 2009; Butler 2012; Liu *et al.* 2016; Knoesel 2017; Swart 2017; Irwanto, Rohaeti and Prodjosantoso 2018; Shively, Stith and Rubenstein 2018), hence this study also aims to use the proposed four-part theoretical model for critical thinking instruction proposed by Halpern (2013) as a paradigm the research. Figure 3.3 illustrates five categories of skills required in the critical thinking process.

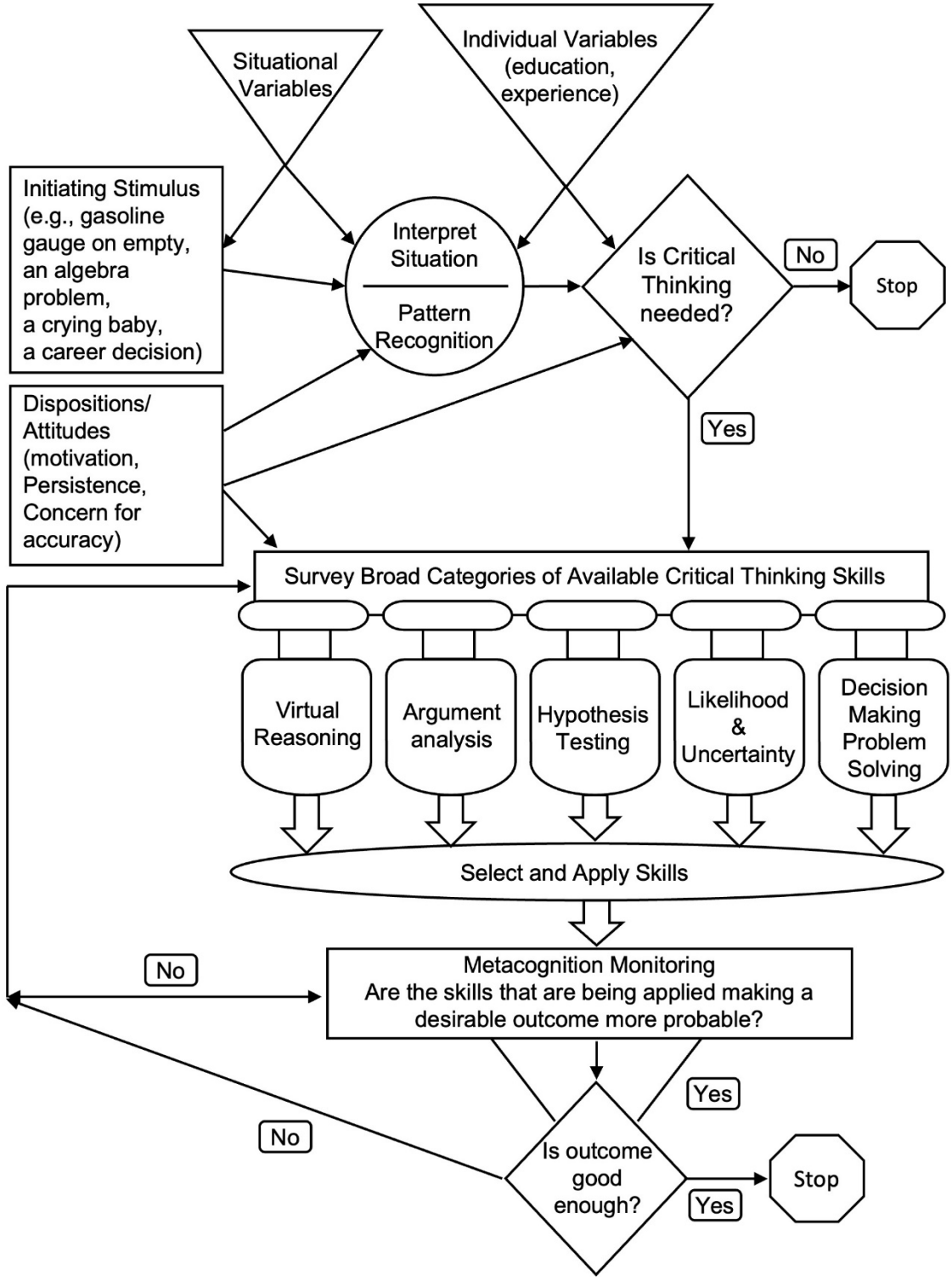


**Figure 3.3 Critical Thinking Skills Categories**
Source: Extracted from Halpern (2013)

Analysing and rationalising situations in a decision-making scenario is affected by the dispositions, individual background variables and situational variables. These elements also indicate if the deliberate application of critical thinking is warranted in the decision-making process. A critical thinker is one who will identify and use the critical thinking skills that are

most apt for a specific context. According to Davies (2015), critical thinking skills and choosing which skill to apply in particular contexts, are learned. Davies (2015) further explains that the process of metacognition entails an iterative cycle of evaluation as individuals determine whether the selected skills being used delivers a desirable outcome or whether selecting additional skills will further enhance the outcome. The iteration is terminated when a "good enough" outcome is achieved.

The purpose of the questionnaire in this study is to evaluate the four learning Apps against the defensible criteria that measures critical thinking. Therefore, the design of the questionnaire will be informed by Diane Halpern's model for critical thinking instruction, which consists of four parts: explicitly learn the skills of critical thinking, develop the disposition for effortful thinking and learning, structure learning activities designed to facilitate transfer across contexts (structure training) and monitor metacognitive processes openly and explicitly (Halpern 2013).

The four components of the model take into account how people recollect, absorb and reflect, and in particular, how learners obtain, consolidate, apply, and retrieve information. Key factors that instigate higher order thinking can be identified in the realm of problem-solving, making inferences, understanding probabilities, making decisions and thinking creatively. However, critical thinking is more than designing and directing activities that will enforce the demonstration of these higher order thinking skills. It also relies on the willingness and flexibility of an incumbent to make a deliberate and concerted mental effort to consider and reflect on multiple ways to apply it. The third component of the model examines the incumbent's ability to recognise what thinking skill is required, when the skill is needed and within which context. Metacognitive Monitoring refers to monitoring ones thinking process towards an improved decision or solution and iteratively validating ones thought process towards accuracy in achieving one's goal. A study by Swart (2017) reveals that there exists potential for the instructional strategies and purposeful integration of technology-enhanced learning to develop and stimulate critical thinking skills in learners, which further reinforces the appropriateness of the four-part model proposed by Halpern (2013).

The components of the model will guide the design of the questionnaire required to measure the critical thinking drivers of the learning Apps. All of the salient aspects of critical thinking will be included in the line of questions but informed by the 4 components of the model. The literature has also identified an association between critical thinking and everyday decision-

making (Butler 2012). This model therefore has the added advantage of informing decision-making which is crucial for the study, as the definition of critical thinking by Halpern (2013) embodies purposeful thinking with logic and reason similar to the aims of decision theory. Therefore, questions that measure the potential of the App to demonstrate good versus bad decision-making will also be included.

## SECTION B: RESEARCH METHODOLOGY

### 3.4 Research Paradigm

The research paradigm appropriate for this study is post-positivism which allows for the study of human behaviour and actions. According to Creswell and Creswell (2017), the knowledge that is developed through this lens requires careful investigation and evaluation of the real world. The post-positivism paradigm with quantitative analysis takes into account that human beings cannot be perceived as perfect respondents but it does view them as diverse and complex beings (Ryan 2006), therefore allowing the scientific method to be probabilistic and provisional (Panhwar, Ansari and Shah 2017). Notwithstanding the study being depicted as numerically significant with the aim of extracting readily available and unambiguous information from respondents (Khaldi 2017), post-positivism provides a vantage point that extends the scientific approach to include ongoing critical reflection that not only examines patterns across data but also interprets meaning through experience and knowledge of participants (Ryan 2006). Therefore, the post-positivist paradigm to analyse the responses of the student's assessment of the critical thinking criteria against the various learning Apps is well suited to this study.

### 3.5 Quantitative Research Method

The design for this research study employs the quantitative approach which uses the survey questionnaire designed to enable questions to be answered authentically, objectively and accurately (Kumar 2018). According to Ajoodha, Jadhav and Dukhan (2020), quantitative research involves the statistical analysis of phenomena using mathematically based methods. It is also possible to collect linguistic phenomena that do not obviously appear in quantitative form in a quantitative way. However, a fundamental aim of quantitative research is to build precise and reliable metrics for further statistical analysis. Therefore, this research study utilizes a survey questionnaire designed with specific, well-structured, direct and quantifiable questions for a 5-point Likert scale rating, so that the data collected from respondents can be scientifically tested for its validity and reliability.

One of the advantages of quantitative research is that the study can be designed for replication and retesting to support validation and trust (Creswell and Creswell 2017). Quantitative methods also allow the research findings to be generalised to a larger specific population from a smaller sample thereby preventing the prohibitive costs and time associated with processing the entire population. However, the extent to which each characteristic is present in the sample must closely resemble that of the population, and developing a reliable measurement instrument that meets the objectives of the research study allows the researcher to objectively make sense of the physical world. Therefore, probability sampling in particular simple random sampling, sample size and the administration of a pilot study have all been carefully considered for this study in an attempt to address the key concepts of validity, reliability and generalisability which are the mainstay of a successful research outcome in a quantitative research study.

## 3.6 Research Setting and Target Population

This study is conducted at a South African University of Technology. A population refers to the entire set of cases which is of interest to the study or a relatively large group of subjects from which gathering information is both impractical and inconvenient to manage the research (Creswell 2012; Taherdoost 2016a). The target population best suited to the study represents the 500 first year programming students aged between 17 and 19 years old and registered for an introductory programming module in the Information and Communications Technology Business Analysis programme of the department of Information Systems at the Durban University of Technology. The first year programming students registered for the Foundation programme and Applications Development programme have been excluded from the study as these students may have been accepted with different selection criteria which may impact the validity and reliability of the results.

## 3.7 Sampling Strategy

The inconvenience associated with processing and analysing the entire population of students that would further require impractical time constraints and inadequate financial and human resources requires researchers to apply sampling techniques to the population of the study. However, sampling is used to generalise from the population and therefore should be designed to genuinely and statistically represent the study population, strategized to be unbiased yet cost effective and should represent, with a satisfactorily high degree of probability and maximum

accuracy, real associations with the population chosen for the study (Kumar 2018). Therefore, determining an applicable sample size is a vital part of quantitative studies.

The study executes a sampling strategy aimed to invoke minimum bias and therefore uses probabilistic sampling, where the likelihood of selecting a participant in the sampling frame is the same for all participants (Creswell 2012). The simple random sampling technique of probability sampling was administered, as all respondents had an equal opportunity to participate and were chosen based on their convenience and availability (Creswell and Creswell 2017). Randomisation offers a good rationale for optimising reliability of the instrument, making subsequent inferences or generalisations to the population group as well as enabling adequate data collection to fulfil the objectives of this study.

## 3.8 Sample Size

The research design also includes making decisions about the sample size from a population. This decision must be considered carefully and thoroughly to avoid any possible financial or scientific ramifications. Although it is advisable to ensure that the sample is large enough to warrant making appropriate estimates of the population, too large a sample may result in wasteful resources. Therefore, this study uses a simple table to allow calculation of a sample size that optimizes decision-making. According to Sekaran's (2000) sample size table below, the suggested sample size for a given population of 500 is approximately 217 and therefore a total of 217 students will be used as the sample size for the study (Sekaran and Bougie 2016).

| *N* | *S* | *N* | *S* | *N* | *S* | *N* | *S* | *N* | *S* |
|---|---|---|---|---|---|---|---|---|---|
| 10 | 10 | 100 | 80 | 280 | 162 | 800 | 260 | 2800 | 338 |
| 15 | 14 | 110 | 86 | 290 | 165 | 850 | 265 | 3000 | 341 |
| 20 | 19 | 120 | 92 | 300 | 169 | 900 | 269 | 3500 | 346 |
| 25 | 24 | 130 | 97 | 320 | 175 | 950 | 274 | 4000 | 351 |
| 30 | 28 | 140 | 103 | 340 | 181 | 1000 | 278 | 4500 | 354 |
| 35 | 32 | 150 | 108 | 360 | 186 | 1100 | 285 | 5000 | 357 |
| 40 | 36 | 160 | 113 | 380 | 191 | 1200 | 291 | 6000 | 361 |
| 45 | 40 | 170 | 118 | 400 | 196 | 1300 | 297 | 7000 | 364 |
| 50 | 44 | 180 | 123 | 420 | 201 | 1400 | 302 | 8000 | 367 |
| 55 | 48 | 190 | 127 | 440 | 205 | 1500 | 306 | 9000 | 368 |
| 60 | 52 | 200 | 132 | 460 | 210 | 1600 | 310 | 10000 | 370 |
| 65 | 56 | 210 | 136 | 480 | 214 | 1700 | 313 | 15000 | 375 |
| 70 | 59 | 220 | 140 | 500 | 217 | 1800 | 317 | 20000 | 377 |
| 75 | 63 | 230 | 144 | 550 | 226 | 1900 | 320 | 30000 | 379 |
| 80 | 66 | 240 | 148 | 600 | 234 | 2000 | 322 | 40000 | 380 |
| 85 | 70 | 250 | 152 | 650 | 242 | 2200 | 327 | 50000 | 381 |
| 90 | 73 | 260 | 155 | 700 | 248 | 2400 | 331 | 75000 | 382 |
| 95 | 76 | 270 | 159 | 750 | 254 | 2600 | 335 | 1000000 | 384 |

***N*** is population size. ***S*** is sample size.

**Table 3.2 Table for Determining Sample Size from a Given Population**
Source: Extracted from Sekaran (2000)

## 3.9 Recruitment of Participants

As part of the participant recruitment drive, a letter of information was sent electronically to students using university email addresses. Furthermore, students were introduced to the background of the study, its importance, relevance and benefits as well as subsequent expectations of the participants through an online meeting forum aligned with the University's protocols that emerged since Covid-19. All communique was instituted through participants' dut4life email addresses and the university's official learning management system (LMS) platform.

## 3.10 Research Instrument

The purpose of a survey questionnaire for a quantitative study is to acquire feedback from a set of questions offering possible answers with the intention of performing statistical analysis (Creswell 2012). An online survey questionnaire which allows a respondent to provide answers quickly and easily through a computer or smart device is inexpensive, time-consuming, relatively convenient and requires minimal financial and human resources to administer (Kumar 2018). It also preserves anonymity as there is no face-to-face interaction, thereby offering greater data security and accuracy.

The quantitative approach using the survey questionnaire as the data collection instrument is the optimal choice for the research design, as the research problem requires that the data be concisely represented in numerical form for the subsequent implementation of the MCDA model (Kurilovas 2014). A survey questionnaire was used with specific, close-ended questions and the 5-point Likert-type rating scale, to gather discernible and quantifiable data on the critical thinking criteria when evaluated against the four learning Apps (Chen *et al.* 2019). The options available on the questionnaire were Very Poor, Poor, Fair, Good and Very Good. The students dut4life email addresses extracted from the University's ITS system were used as a means to communicate with potential participants on the study. Student email addresses and the university's official LMS were used to circulate the online questionnaire designed using google docs. It was taken into account that all first year students are also computer literate and would therefore be able to easily and accurately engage with an online questionnaire. The survey questionnaire comprised two sections, Section A, that requested demographic information and Section B that assessed the four learning Apps using attributes for critical

## 3.11 Pilot Study

In quantitative research, pre-testing or a pilot study is a practice whereby the researcher administers the questionnaire to a small group of individuals before its actual use (Kumar 2018) to critically examine possible problems with the instrument with the intention of validating its content (Creswell 2012). This would subsequently enable the researcher to make the necessary adjustments towards improving questions, format and scales based on the feedback received (Creswell and Creswell 2017). The sample for a pilot can also serve to establish trust and respect with the participants. The sample size for the pilot study must be feasibly large while being representative of the population to warrant a test run aimed to assess viability of the main study. Although the subjects for the pilot must be excluded from the main study, they must still fulfil the same inclusion/exclusion criteria as the main study (Thabane *et al.* 2010).

It is necessary to establish the sample size of the pilot study prior to data collection to establish adequate precision or statistical power. Johanson and Brooks (2010) suggest that sample representation for a pilot study has a greater impact than sample size when assessing the accuracy and viability of the research instrument for the main study. For a preliminary survey, the authors recommend a minimum of 30 representative participants from the population of interest. The recommendation from the authors is aligned to prevailing analysis in the literature with regard to confidence intervals.

Accordingly, the pilot phase of this study was undertaken among 30 participants from a population of 500 first year programming students. These first year programming students were chosen from a different programme and not included in the sample but carefully selected to include diversity in terms of English as a 1st or 2$^{nd}$ language, gender, race, educational background and computer literacy exposure. This was conducted via an online session, where the researcher used the opportunity to explain the objectives and relevance of the pilot and encourage participants to list ambiguities and other queries on the questionnaire. During this live session, the participants were requested to complete the questionnaire and to raise any questions of clarity or ambiguity pertaining to the questionnaire via the Teams chat. Arising from this engagement, the researcher made minor grammatical changes to 6 questions in the pilot questionnaire to the satisfaction of all participants. Feedback during this engagement also assisted the researcher to improve the information disseminated to students prior to completing the actual survey.

## 3.12 Data Collection

The purpose of the survey questionnaire (Annexure F) was to use a sample of 217 first year students to determine the performance of each of four learning Apps when assessed against various critical thinking criteria. It was therefore imperative that the students had sufficient exposure to each of the learning Apps that would prepare them to adequately answer the various questions appropriately. The researcher spent at least two weeks tutoring students on each learning App while providing them with an opportunity to reinforce their understanding and potential of the App through various problem-solving practice exercises. These supplementary tutoring sessions were administered via MS Teams and through Moodle, the university's official learning management system.

The researcher underwent an online certification training course on introduction to ethics (Annexure C) in preparation to conduct research. In keeping with sound ethical principles and practice, all students were introduced to the research study and briefly exposed to its background, purpose, benefits, their rights, confidentiality and anonymity features of the survey administration, technical requirements for the online distribution and the intent of publishing all research findings, prior to acquiring permission from the DUT (Annexure E) and formal approval from the faculty research ethics committee (Annexure D). This was implemented through the first year orientation programme and through the letter of consent (Annexure B) and information letter which was also translated into isiZulu for the 2$^{nd}$ language learners. Communication with respondents transpired via the students official dut4life email accounts.

## 3.13 Data Analysis

The decision-making dataset was imported and analysed in Matlab R2020a. The code corresponding to the Fuzzy TOPSIS method was implemented in MatLab. The code was executed on a computer running Windows 10, 64 bit operating system with the configuration of Intel(R) Core(TM) i7-8650U CPU @ 1.90GHz 2.11 GHz, 16 GB RAM memory and 500 Gigabytes Hard Disk drive.

### 3.13.1 Fuzzy TOPSIS

The ratings of alternatives with respect to multiple selection criteria was assessed by 217 students, and the different weights of all criteria determined by academic experts in the field

through a feature questionnaire, using more logical interval judgements based on linguistic values which can be parameterized by triangular Fuzzy numbers (Han and Trimi 2018). The purpose of the triangular Fuzzy set is an attempt to effectively address vagueness, uncertainty and imprecision in the linguistic human decisions (Rouyendegh, Yildizbasi and Üstünyer 2020).

The Fuzzy TOPSIS technique encompasses 6 steps: "constructing the normalized decision matrix after ascertaining the decision criteria, the decision-makers and the alternatives; deciding on the linguistic variables for the weighting of the criteria and for the evaluation of the alternatives according to the criteria and subsequently mapping these to their corresponding triangular fuzzy numbers; calculating the weighted normalized fuzzy decision matrix; calculating the fuzzy positive ideal solution (FPIS) and fuzzy negative ideal solution (FNIS); calculating the relative distances of each alternative from the FPIS and FNIS and calculating the closeness coefficient for each of the alternatives" (Sodhi and T V 2012: 3; Ece and Uludag 2017: 110-111).

### 3.13.2 Biographical and Descriptive Analysis

The first section of the survey questionnaire sought information on the demographic profile of respondents, including their age, gender, digital access medium, internet connectivity and experience with computers for the purpose of assisting in the analysis of the survey response. Descriptive analysis is presented by means of tables and graphs. This entails a comprehensive analysis of the sections in the main study questionnaire.

### 3.13.3 Inferential Analysis

In general, quantitative research is designed to test hypotheses, examine cause and effect or make findings by reviewing specific variables objectively and identifying and analysing statistical relationships between variables, culminating in a final report with possible correlations and results from means and significance tests in support of findings (Apuke 2017). In univariate analysis, frequency distribution in the form of tables and graphs of individual variables such as age or gender can be depicted. The literature identifies levels of measurement comprising three types of variables, namely, nominal, ordinal and continuous, which have significant implications for the types of statistical analyses that can be performed and the way these are interpreted (Muijs 2010). For example, the most appropriate measure for nominal variables such as gender, is the mode which represents the most common value in the dataset;

for ordinal variables such as the App has various features that enable me to use my more creative side is the median which represents the middle value in a dataset ordered from lowest to highest; and for continuous variables such as age is the mean which is the result of the quotient of sum of all values and number of values. According to Muijs (2010), other measures of spread around the centre such as range and interquartile range are good measures for ordinal variables, while standard deviation is best for continuous variables.

In bivariate analysis, a statistical method called cross tabulation compares two nominal variables, a nominal and an ordinal variable or two ordinal variables (Muijs 2010). Combinations of two or more categories of variables are cross-tabulated and reflected in tables, for example the number of female respondents, with home Wi-Fi, who had prior exposure to programming. In this way, the researcher is able to make a comparison between actual data collected through the sample and what would have been anticipated barring any relationship between variables.

When a sample is drawn from a population in quantitative analysis, it is crucial to support the findings of the sample data through sufficient evidence that merits making an extrapolation of the findings to the population. Therefore, calculating the significance level or probability value (*p-value*) using the chi-square test will inform the researcher the degree to which the difference or relationship being studied is statistically significant. The smaller the *p-value*, the more confident the researcher is of the findings being representative of the population. The default value of less than 0.05 (corresponding to a confidence level of 95 per cent) is usually used to declare a relationship statistically significant. However, the significance level is also determined by sample size and not just by weight of the relationship. Therefore, another measure is required to assess the weight of the relationship, or the effect size of the chi square test, called *phi*. This is calculated by taking the square root of the chi square value and divide it by the sample size. This result can fall within a range of 0 (no relationship) and 1 (perfect positive relationship). Therefore, a *phi* value close to 1 indicates a stronger relationship to one that is close to 0.

### 3.13.4 Reliability of the Research Instrument

According to Delice (2010), the research instrument is deemed reliable when the same results are generated from repeated trials conducted by different researchers under constant conditions. Therefore, reliability tests are conducted to establish the internal consistency across the survey

instrument revealing whether the scores are consistent and stable. This study used the Cronbach Alpha metric to measure the reliability and consistency of the data collected through the measuring instrument. Johanson and Brooks (2010) refers to Cronbach's coefficient alpha in survey research as one of the most popularly used measure for internal consistency in survey research when using Likert scales. The literature suggests a score of 0.60 or above to indicate reliability of an instrument for pilot studies (Straub, Boudreau and Gefen 2004). According to Larkin (2015), Cronbach alpha scores greater than 0.7 indicate a high degree of internal consistency, while Hinton, McMurray and Brownlow (2014) recommend four ranges of scores to depict reliability: excellent reliability (0.90 and above), high reliability (0.70-0.90), moderate reliability (0.50-0.70) and low reliability (0.50 and below).

## 3.14 Inclusion and Exclusion Criteria for the Learning Apps

The learning Apps chosen for the study needed to comply with various dimensions pertaining to programming content, computer requirements, user interaction, tangible objects and pedagogical adequacy (João *et al.* 2019). The features of the learning Apps must support the curriculum guidelines for the first year programming module for which the study is being conducted. The computer requirements that need to be considered are internet connectivity, operating system, type of equipment, language and cost. This study supported learning Apps that run in a browser, windows, android, IOS, stand-alone desktop machines, smartphones, are freely accessible and available in the English language. Apps that run on the IOS platform only, incur hardware, software or connectivity costs and do not align with the introductory programming module content, will be eliminated from the study. The quality of interaction with the user was also considered in the choice of learning Apps. Those Apps that support rich multimedia, invoke users' interest and enable various types of scenario applications that are both complex and stimulating, were included in the study. Visual block-based environments that focused on programming robots and drones were eliminated from the study because the focus of the programming module is on non-technical programming. Apps that were more suited to children or school pupils were also eliminated from the study. The emphasis was on choosing visual programming environments that were more suited to novice adults.

## 3.15 Ethical Considerations

In the interest of sound research ethical policy and practice, university protocols were complied with throughout the research process. A formal application of ethical clearance from the faculty

research ethics committee is mandatory prior to conducting the survey. Although the respondents email accounts were be used to circulate the questionnaire, the identities and responses of participants were kept confidential during the various phases of the study. A letter of information and consent form aligned to sound ethical practices were distributed to potential participants and included an assurance that the study would pose no potential harm or disrespect to any participant. As part of the University's postgraduate guidelines in maintaining and practicing ethical standards, an official letter of information and consent form aimed at guaranteeing and protecting the rights of participants was outlined. This included details confirming that their participation was voluntary, that they do own the right to withdraw from the study at any time without recourse, and that they also deserve to know the background, purpose, benefits and findings from the study prior to their agreement to participate. All these details were communicated to the participants via their official email accounts. The signed written consent form was a requirement for participation in the study.

## 3.16 Validity

Creswell (2012) defines validity as the extent to which the evidence aligns itself to the anticipated interpretation of test scores against its proposed purpose. The measurement instrument must measure what it claims to measure in order for the results to be accurately applied and interpreted. The literature indicates the five key sources of evidence to support validity as process, content, structure, relations to other variables and consequences of testing. Kumar (2018) identifies three types of validity in quantitative research namely "face and content validity, concurrent and predictive validity and construct validity".

Each question on the questionnaire must have a logical rationale (face validity) and all the criteria being measured (content validity) must be covered in the questions. These types of validity can be logically justified by the researcher yet can be subjectively challenged by experts and respondents; hence the study will use independent experts to validate the questions against the research objectives prior to being administered. Predictive validity is judged by whether scores predict a criterion measure and concurrent validity is determined by whether results correlate with results from other studies (Creswell 2012). The study can explore the use of a validity coefficient to express predictive validity. Kumar (2018) defines construct validity as important in determining the value of the scores when applying the theory in practice which can possibly be substantiated through factor analysis (Creswell and Creswell 2017).

In order to test for validity, the survey questionnaire was reviewed by two ICT professors prior to the pilot study. The professors were also provided with a synopsis of the study including the aim and objectives together with a description of each criterion (Section 2.2) on which the questionnaire was based. Feedback received from this process did not require any amendments to the questionnaire. Furthermore, the results in Chapter 4 indicate that Scratch is the App that promotes critical thinking the best among the first year novice programming learners. This result aligns to other research studies in the literature, in which Scratch generally performed well when viewed through the lens of various characteristics and features of learning Apps (Kalelioglu and Gülbahar 2014; Erol and Kurt 2017; João et al. 2019; Kaya and Yildiz 2019; Durak 2020). Therefore, concurrent validity was also achieved in this study.

## 3.17 Chapter Summary

This chapter presented the two theoretical frameworks underpinning the study followed by the research methodology designed to attain the objectives of the study. Each framework systematically explained the focus firmly embedded in the rationale for and description of the framework when applied to the selection of learning Apps to promote critical thinking among first year programming students. Next, the chapter highlighted the details surrounding the research methodology, with particular emphasis on the research paradigm, research method, research instrument, participants in the study (population and sample), identification of variables, measures of the variables, overall design, ethical considerations and selection of statistical analysis tools. The next chapter presents a comprehensive analysis of Fuzzy TOPSIS.

# CHAPTER FOUR
# FUZZY TOPSIS

## 4.1 Introduction

The significant growth in diverse learning Apps in the last decade and the varying commercial aspirations these Apps may be designed to serve, have impacted on the need for university academics to make rigorous and well-considered decisions around their choice of learning App. Multi-criteria group decision-making (MCDM) is a procedure that derives the optimal alternative when multiple decision-makers asses the performance of each decision alternative against multiple, conflicting, quantitative decision criteria (Yavuz 2016; Nursal, Omar and Nawi 2018; Salih *et al.* 2019). The various criteria will be identified from a thorough review of the literature in the relevant field. According to Başaran and Haruna (2017), manually selecting these Apps can be biased, tedious, inconsistent and time-consuming, ultimately resulting in a premature selection. Therefore, to address ambiguity and uncertainty associated with human judgements, this quantitative study will use the Technique for Order Preference by Similarity to an Ideal Solution (TOPSIS), a scientific and mathematical technique in multi-criteria decision-making (MCDM) that preserves integrity and objectivity of process (Yavuz 2016; Rajak and Shaw 2019). The TOPSIS method, developed in 1981 by Yoon and Hwang, was based on the premise that the optimal alternative is the one with the shortest geometric distance from the positive ideal solution and farthest from the negative ideal solution (Balioti, Tzimopoulos and Evangelides 2018).

Fuzzy set theory, which was introduced by Zadeh in 1965, can be used to manage the uncertainties and subjectivity of the evaluation process (Salih *et al.* 2019). More specifically in this study, an extension of the TOPSIS method to a fuzzy environment will be adopted. Fuzzy TOPSIS is widely and extensively applied among researchers in diverse fields of economics, business management, engineering, ICT and education, to solve various MCDM problems associated with site selection in mining and engineering, equipment selection, risk analysis for complex infrastructure projects, portfolio selection, selection of e-Learning approaches, supplier selection and software selection (Yavuz 2016; Başaran and Haruna 2017; Ece and Uludag 2017; Balioti, Tzimopoulos and Evangelides 2018; Mohammed, Kasim and Shaharanee 2018; Memari *et al.* 2019; Rajak and Shaw 2019; Ranganath *et al.* 2020). The advantages of Fuzzy TOPSIS include its ease of implementation, capacity to compare the best and worst alternatives quantitatively, that it enables linguistic expressions to be represented as

fuzzy numbers, and its practicality and ability to handle incomplete or partial quantitative data (Han and Trimi 2018). The ratings of alternatives with respect to multiple selection criteria are disseminated through a questionnaire to 217 students, and the different weights of all criteria are assessed by academic experts in the field, using more logical interval judgements based on linguistic values which can be parameterized by triangular fuzzy numbers (Han and Trimi 2018).

## 4.2 Evaluation of Selection Techniques and Motivation for Fuzzy TOPSIS

Multi-criteria decision analysis (MCDA) has successfully been used to solve a wide range of complex real-life decision problems in a variety of fields (Palczewski and Sałabun 2019), despite researchers encountering many challenges associated with partial ignorance and unquantifiable, incomplete, unobtainable, uncertain, ambiguous and vague information (Salih *et al.* 2019). The choice of MCDA technique rests predominantly on the nature of the decision problem and its ability to address many of the afore-mentioned challenges. The fuzzy approach can be used to effectively represent and process any ambiguities, subjectivity and indistinctness associated with the decision-maker's evaluation (Safari, Faghih and Fathi 2012; Rajak and Shaw 2019). Furthermore, the technique for order of preference by similarity to ideal solution (TOPSIS) suggested by Hwang and Yoon in 1981, is a popular method that can rank the best alternatives quickly, address conflicting situations, is easy to use and can be integrated with other decision-making methods (Rajak and Shaw 2019). The fuzzy set theory combined with TOPSIS has the added advantage of addressing any uncertainty in the assessment process (Salih *et al.* 2019).

A variety of refined MCDA methods have been developed over the years, such as the analytic hierarchy process (AHP) and the best-worst method that depends on human preferences; the technique for order of preference by similarity to ideal solution (TOPSIS) and simple additive weighting (SAW), which depend on mathematical operations, and the outranking models like ELECTRE and PROMETHEE, which are premised on the assumption that the decision is a process whereby decision-makers can alter their preferences after thorough reflection. The hierarchical model of the AHP method requires the decision-maker to pairwise-compare multiple criteria and uses ratio and semantic scales to arrive at the decision-maker’s preference. Memari *et al.* (2019) compared the TOPSIS method with two outranking methods namely, PROMETHEE and ELECTRE. According to the authors, TOPSIS is easy to learn and apply

compared to outranking methods which proved to be more complex and less transparent to decision-makers. Zanakis *et al.* (1998) included TOPSIS in their study due to its unique approach to the problem and its intuitively appealing and easy to understand qualities. Palczewski and Sałabun (2019) further highlighted simplicity, computational efficiency and comprehensive mathematical concept as well as its support for group decision-making as key contributing factors to the popularity of the Fuzzy TOPSIS technique.

In a study conducted by Junior, Osiro and Carpinetti (2014), a comparison of the Fuzzy AHP and Fuzzy TOPSIS methods was undertaken for a supplier selection problem. The criteria central to the comparison were: adequacy to changes of alternatives or criteria; agility in the decision process; computational complexity; adequacy to supporting group decision-making; the number of alternatives and criteria, and modelling of uncertainty. The study revealed that Fuzzy TOPSIS provided greater agility in the decision process, lower time complexity when there was a large number of decision-makers, offered no restrictions on the number of criteria and alternatives and became the preferred choice when there were changes to alternatives or criteria as rank reversal posed a potential problem with Fuzzy AHP.

### 4.3 Fuzzy TOPSIS versus Conventional TOPSIS

In the learning App selection problem, the number of decision-makers, the nature of the various criteria to promote critical thinking, the weightings of the various criteria and the degree of uncertainty and vagueness have to be carefully considered in the decision on the multi-criteria decision-making approach to be applied. This study involves a sample of 217 students and questions on the performance of each alternative against multiple critical thinking criteria, some more easily measurable than others. According to Nursal, Omar and Nawi (2018), the performance ratings and the weights of the criteria in the process of TOPSIS are given as crisp or exact numerical values which are also inadequate to simulate human judgements in real-life scenarios. Therefore, to accommodate the imprecise or vague nature of the group decision-making assessment pertaining to critical thinking criteria, this study will use a more realistic approach and use linguistic variables to capture the decision-maker's rating more accurately and consistently (Sevkli *et al.* 2010). Hence the Fuzzy TOPSIS approach is being more suitably applied to model the linguistic ratings as a fuzzy triplet (a, b, c), where the parameters of a, b, and c indicate the smallest possible value, the most promising value, and the largest possible

value that describe a fuzzy event, respectively, considering the fuzziness in the decision data and group decision-making process (Chen 2000; Sevkli *et al.* 2010).

Fuzzy logic has been successfully applied to a wide variety of MCDM problems ranging from facility location selection, robot selection, machine tool selection problem and plant layout design problem (Kahraman *et al.* 2007), to ERP software selection (Sevkli *et al.* 2010), equipment selection (Yavuz 2016) and supplier selection (Memari *et al.* 2019; Palczewski and Sałabun 2019). Although this study uses the quantitative approach, it also identifies some subjective criteria that are difficult to measure, hence the need for fuzzy sets which are capable of representing vague data (Chen 2000). According to (Kahraman *et al.* 2007), the strength of fuzzy logic lies in its ability to enable approximate human reasoning capabilities associated with human cognitive processes, such as thinking and reasoning, to be captured mathematically and more precisely, resulting in a better expected performance in this case scenario.

### 4.4 Fuzzy Set Theory

In real-life situations which are frequently not deterministic, it is often inadequate to describe phenomena in crisp and precise terms. Therefore, fuzziness occurs in many areas of human judgement, reasoning, evaluations and decision-making. In this study, for example, the evaluation of learning Apps against various subjective critical thinking criteria are usually expressed in linguistic terms, such as very good, poor or satisfactory, which may affect vagueness and ambiguity. However, fuzzy set theory, introduced by Zadeh in 1965, provides a strict mathematical framework to efficiently resolve the indistinctness associated with the subjectivity of human judgements, and in which vagueness and uncertainty can be precisely and rigorously studied (Zimmermann 2010).

Sevkli *et al.* (2010) highlighted fuzzy sets and fuzzy logic (FL) as powerful mathematical tools for modelling uncertain systems in industry, while Zadeh (1975b) and Hellmann (2001) define fuzzy logic as a multivalued logic, that uses intermediary values between conventional evaluations such as true/false, yes/no or high/low. According to Zadeh (1975a), the language of linguistics is helpful in describing uncertain and ill-defined qualitative data, which otherwise cannot be subjected to quantitative analysis. These linguistic variables can then be represented as computational-efficient fuzzy-triples for appropriately quantifying vague information (Safari, Faghih and Fathi 2012).

A fuzzy set can be defined mathematically by a membership function, which assigns each element x in the universe of discourse X a real number in the interval [0,1] (Safari, Faghih and Fathi 2012). In fuzzy set theory, an uncertain situation can be modelled using different types of membership functions, for example, triangular or trapezoidal. According to Rajak and Shaw (2019), modelling the decision problem using triangular fuzzy numbers yields a better result. The membership function F(x) of a triangular fuzzy number Ã, illustrated in Figure 4.1 can be defined by a triplet (a, b, c), where the parameters of a, b, and c indicate the smallest possible value, the most promising value, and the largest possible value that describe a fuzzy event, respectively (Sevkli *et al.* 2010).

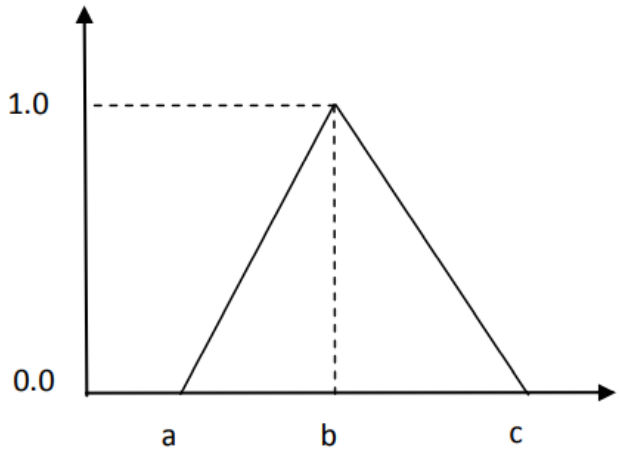


**Figure 4.1 A membership function F(x) of a triangular fuzzy number Ã**
Source: Extracted from Han and Trimi (2018)

A triangular fuzzy number is defined in Figure 4.1 and expressed as a membership function in Equation 4.1, extracted from Safari, Faghih and Fathi (2012).

$$F(x) = \begin{cases} \dfrac{x-a}{b-a}, & a \le x \le b \\ \dfrac{c-x}{c-b}, & b \le x \le c \\ 0, & otherwise \end{cases} \tag{4.1}$$

In fuzzy set theory, conversion scales are applied to transform the linguistic terms into fuzzy numbers. Usually, a scale of 1 to 9 is applied for weighting the criteria and ranking the alternatives, as illustrated in Table 4.1. The intervals are chosen so as to have a uniform

representation from 1 to 9 for the fuzzy triangular numbers used for the five linguistic ratings (Ranganath *et al.* 2020).

Linguistic terms for criteria and alternatives ratings.

| Criteria | Alternatives | |
|---|---|---|
| | Linguistic term | Triangular fuzzy numbers |
| Very low (VL) | Very poor (VP) | (1, 1, 3) |
| Low (L) | Poor (P) | (1, 3, 5) |
| Medium (M) | Fair (F) | (3, 5, 7) |
| High (H) | Good (G) | (5, 7, 9) |
| Very high (VH) | Very good (VG) | (7, 9, 9) |

**Table 4.1 Linguistic terms for criteria and alternatives ratings**
Source: Extracted from Han and Trimi (2018)

## 4.5 Fuzzy TOPSIS Method

In this research study, fuzzy set theory is introduced to model ambiguity and uncertainty in a MCDM problem and integrated with TOPSIS to appreciate the benefits of its practicality and ease of use, enabling evaluations to be expressed in a linguistic language and then converted to triangular fuzzy numbers and implementing the algorithm using a software tool.

Assume the decision problem has $k$ decision-makers ($D_1$, $D_2$ … $D_k$), with $m$ possible alternatives ($A_1$, $A_2$ … $A_m$), which is evaluated against $n$ criteria ($C_1$, $C_2$ … $C_n$). The rating of criteria and weight with respect to each criterion can be accurately represented in the form of matrices for each decision-maker.

$$\tilde{D} = \begin{array}{c} \\ A_1 \\ A_2 \\ \\ A_m \end{array} \begin{array}{c} \begin{array}{cccc} C_1 & C_2 & & C_n \end{array} \\ \begin{pmatrix} \tilde{x}_{11} & \tilde{x}_{12} & \dots & \tilde{x}_{1n} \\ \tilde{x}_{21} & \tilde{x}_{22} & \dots & \tilde{x}_{2n} \\ \dots & \dots & \tilde{x}_{ij} & \dots \\ \tilde{x}_{m1} & \tilde{x}_{m2} & \dots & \tilde{x}_{mn} \end{pmatrix} \end{array} \tag{4.2}$$

$$\tilde{W} = (\tilde{w}_1, \tilde{w}_2, \dots, \tilde{w}_n)$$

Where for all $x_{ij}$ and $w_j$, i = 1, 2 … m and j=1, 2 … n.

Hence $x_{ij} = (a_{ij},\ b_{ij},\ c_{ij})$ *and* $w_j = (a_j, b_j, c_j)$ are triangular fuzzy numbers representing linguistic variables.

The Fuzzy TOPSIS procedure includes the following steps (Han, 2018: 137):

"Step 1: Assign ratings to the criteria and alternatives. The criteria weights are denoted by $W_{NK}$ (N = 1,2,…, n; K = 1,2,…, k) and the performance ratings of alternatives with respect to criteria by experts are denoted as $X_{NKM}$ (N = 1,2,…, n; K = 1,2,…, k; M = 1,2,…, m).

Step 2: Aggregate the evaluation of the criteria and alternatives. fuzzy ratings $W_{NK}$ and $X_{NKM}$ is described as triangular fuzzy numbers ($a_K$, $b_K$, $c_K$) where K = 1,2,…, k, then the aggregated importance can be evaluated as:

$$a = \min\{a_K\} \quad b = \frac{1}{k}\sum_{K=1}^{k} b_K \quad c = \max\{c_K\} \tag{4.3}$$

Step 3: Normalize triangular fuzzy numbers. The raw data is normalized using linear scale transformation to bring the various criteria scales into a comparable scale.

If $W_{NK}$ represents benefit criteria, then : $\left(\frac{a_K}{C}, \frac{b_K}{C}, \frac{c_K}{C}\right)$ (4.4)

If $W_{NK}$ represents cost criteria, then : $\left(\frac{a}{c_K}, \frac{a}{b_K}, \frac{a}{a_K}\right)$ (4.5)

$(a = min\{a_K\}, c = max\{c_K\})$

Step 4: Compute weighted normalized fuzzy values. $W_{NK}$* becomes $W_{NK}$ after normalization, $X_{NKM}$* is new $X_{NKM}$ after aggregation. Let the weighted normalized value be $V_{NKM}$.

$V_{NKM} = W_{NK}$* x $X_{NKM}$* (4.6)

where N = 1,2 … n; K = 1,2 … k; M = 1,2 … m;

The corresponding triangular fuzzy number of $V_{NKM}$ is

$$\left(a_{V_{NKM,}}\ b_{V_{NKM}},\ c_{V_{NKM}}\right) \tag{4.7}$$

Step 5: Calculate fuzzy positive ideal solutions (FPIS) and fuzzy negative ideal solutions (FNIS);

$$FPIS = \left(C_{V,}\ C_{V,}\ C_V\right) where\ C_V = \max\left\{C_{V_{NKM}}\right\} \tag{4.8}$$

$$FNIS = \left(a_{V,}\ a_{V,}\ a_V\right) where\ a_V = \min\left\{a_{V_{NKM}}\right\} \tag{4.9}$$

Step 6: Calculate the distance of each alternative from FPIS ($d^+$) and FNIS ($d^-$) which is calculated, respectively, as follows:

$$d^+ = \sqrt{\frac{1}{3}\left[\left(a_{V_{NKM}} - c_V\right)^2 + \left(b_{V_{NKM}} - c_V\right)^2 + \left(c_{V_{NKM}} - c_V\right)^2\right]} \tag{4.10}$$

$$d^- = \sqrt{\frac{1}{3}\left[\left(a_{V_{NKM}} - a_V\right)^2 + \left(b_{V_{NKM}} - a_V\right)^2 + \left(c_{V_{NKM}} - a_V\right)^2\right]} \tag{4.11}$$

Step 7: Calculate the closeness coefficient ($CC_M$)

$$CC_M = \frac{\sum_1^n d_{NM}^-}{\sum_1^n d_{NM}^+ + \sum_1^n d_{NM}^-}, N = 1,2 \dots n; M = 1,2 \dots m \quad (4.12)$$

The $CC_M$ value is then used to determine the ranking order of all alternatives for the purpose of selecting the best one from among a set of feasible alternatives."

## 4.6 Application of Fuzzy TOPSIS in a small-scale Example

This section aims to use a subset of the data gathered through the survey questionnaire and feature questionnaire to illustrate the step-by-step process of Fuzzy TOPSIS as outlined in Section 4.5. This small-scale example is explained and implemented using the equations defined in the Fuzzy TOPSIS method.

In this scenario example, four student decision-makers must select from four alternatives, namely, Alice, Scratch, Blockly and MIT App Inventor, using four criteria, namely, feedback, problem-solving, collaboration and logic to rate the alternatives. Nine academic experts rated the criteria in linguistic terms according to the levels of importance. A step-by-step analysis using the Fuzzy TOPSIS method is explicated below.

**Step 1: Define Linguistic Ratings for Criteria and Alternatives**

The first step in the Fuzzy TOPSIS technique is to derive the weights of the various critical thinking criteria and the performance ratings of the learning Apps against these criteria. The researcher chose the 5-point Likert scale because of its practicality and accessibility to collect and operationalise complex phenomena easily, enabling numerical values in order to facilitate statistical testing and analyses (Li 2013).

In this study, the importance weights of the critical thinking criteria and the performance ratings of the learning Apps are considered as linguistic variables chosen appropriately and defined as fuzzy triples ranging from 1 to 9, as illustrated in Tables 4.2 and 4.3 respectively.

**Criteria:**

**C1** Feedback

**C2** Problem-Solving

**C3** Collaboration

**C4** Logic

| Linguistic term | Membership function |
|---|---|
| Not Important (NI) | (1,1,3) |
| Slightly Important (SI) | (1,3,5) |
| Moderately Important (MI) | (3,5,7) |
| Important (I) | (5,7,9) |
| Very Important (VI) | (7,9,9) |

**Table 4.2 Linguistic Language for Criteria Weightings**

**Learning Apps (Alternatives)**

**A1** Alice

**A2** Scratch

**A3** Blockly

**A4** MIT App Inventor

| Linguistic term | Membership function |
|---|---|
| Very Poor (VP) | (1,1,3) |
| Poor (P) | (1,3,5) |
| Fair (F) | (3,5,7) |
| Good (G) | (5,7,9) |
| Very Good (VG) | (7,9,9) |

**Table 4.3 Linguistic Language for Learning App Ratings**

The four decision-makers rated the four learning Apps against the four critical thinking criteria using linguistic terms as represented in Table 4.4.

| Criteria | Alternatives | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | A1 (Alice) | | | | A2 (Scratch) | | | | A3 (Blockly) | | | | A4 (MIT App Inventor) | | | |
| | D1 | D2 | D3 | D4 | D1 | D2 | D3 | D4 | D1 | D2 | D3 | D4 | D1 | D2 | D3 | D4 |
| C1 | P | G | VG | VG | G | P | G | VG | G | G | F | G | VG | G | VG | VG |
| C2 | F | G | G | F | G | G | VG | G | G | F | F | G | VG | G | VG | VG |
| C3 | P | F | G | VG | VG | G | G | VG | VG | F | F | G | VG | G | G | VG |
| C4 | VG | VG | G | G | VG | VG | VG | VG | VG | VG | VG | G | VG | VG | G | G |

**Table 4.4 Linguistic Language Rankings for Learning Apps**

The nine academic experts rated in linguistic terms the four critical thinking criteria as represented in Table 4.5.

| Criteria | Expert Weightings on Criteria | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | D1 | D2 | D3 | D4 | D5 | D6 | D7 | D8 | D9 |
| C1 | I | I | VI | I | VI | VI | SI | I | VI |
| C2 | VI | MI | VI | I | MI | MI | I | VI | I |
| C3 | MI | I | I | I | VI | MI | MI | MI | SI |
| C4 | VI | VI | I | VI | VI | VI | I | VI | VI |

**Table 4.5 Linguistic Language Expert Weightings of Criteria**

The corresponding fuzzy triple representations for the data collected and represented in Table 4.4 and as defined in Table 4.3, are used to formulate the App rankings matrix shown in Table 4.6.

| Criteria | Alternatives | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | A1 (Alice) | | | | A2 (Scratch) | | | | A3 (Blockly) | | | | A4 (MIT App Inventor) | | | |
| | D1 | D2 | D3 | D4 | D1 | D2 | D3 | D4 | D1 | D2 | D3 | D4 | D1 | D2 | D3 | D4 |
| C1 | 1,3,5 | 5,7,9 | 7,9,9 | 7,9,9 | 5,7,9 | 1,3,5 | 5,7,9 | 7,9,9 | 5,7,9 | 5,7,9 | 3,5,7 | 5,7,9 | 7,9,9 | 5,7,9 | 7,9,9 | 7,9,9 |
| C2 | 3,5,7 | 5,7,9 | 5,7,9 | 3,5,7 | 5,7,9 | 5,7,9 | 7,9,9 | 5,7,9 | 5,7,9 | 3,5,7 | 3,5,7 | 5,7,9 | 7,9,9 | 5,7,9 | 7,9,9 | 7,9,9 |
| C3 | 1,3,5 | 3,5,7 | 5,7,9 | 7,9,9 | 7,9,9 | 5,7,9 | 5,7,9 | 7,9,9 | 7,9,9 | 3,5,7 | 3,5,7 | 5,7,9 | 7,9,9 | 5,7,9 | 5,7,9 | 7,9,9 |
| C4 | 7,9,9 | 7,9,9 | 5,7,9 | 5,7,9 | 7,9,9 | 7,9,9 | 7,9,9 | 7,9,9 | 7,9,9 | 7,9,9 | 7,9,9 | 5,7,9 | 7,9,9 | 7,9,9 | 5,7,9 | 5,7,9 |

**Table 4.6 Fuzzy Triple Representations for App Rankings Matrix**

The corresponding fuzzy triple representations for the data collected and represented in Table 4.5 and as defined in Table 4.2 are used to formulate the expert weightings matrix shown in Table 4.7.

| Criteria | Expert Weightings on Criteria | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | D1 | D2 | D3 | D4 | D5 | D6 | D7 | D8 | D9 |
| C1 | 5,7,9 | 5,7,9 | 7,9,9 | 5,7,9 | 7,9,9 | 7,9,9 | 1,3,5 | 5,7,9 | 7,9,9 |
| C2 | 7,9,9 | 3,5,7 | 7,9,9 | 5,7,9 | 3,5,7 | 3,5,7 | 5,7,9 | 7,9,9 | 5,7,9 |
| C3 | 3,5,7 | 5,7,9 | 5,7,9 | 5,7,9 | 7,9,9 | 3,5,7 | 3,5,7 | 3,5,7 | 1,3,5 |
| C4 | 7,9,9 | 7,9,9 | 5,7,9 | 7,9,9 | 7,9,9 | 7,9,9 | 5,7,9 | 7,9,9 | 7,9,9 |

**Table 4.7 Fuzzy Triple Representations for Expert Weightings Matrix**

**Step 2: Aggregate the evaluation of Alternatives and Criteria**

The aggregated importance of the data for the App rankings matrix is calculated using Equation 4.3 in Section 4.5 and reflected in the aggregated App rankings matrix shown in Table 4.8.

| | A1 | A2 | A3 | A4 |
|---|---|---|---|---|
| C1 | 1,7,9 | 1,6.5,9 | 5,6.5,9 | 5,8.5,9 |
| C2 | 3,6,9 | 5,7.5,9 | 3,6,9 | 5,8.5,9 |
| C3 | 1,6,9 | 5,8,9 | 3,6.5,9 | 5,8,9 |
| C4 | 5,8,9 | 7,9,9 | 5,8.5,9 | 5,8,9 |

**Table 4.8 Aggregated App Rankings Matrix**

The aggregated importance of the data for the criteria weightings matrix is calculated using Equation 4.3 in Section 4.5, and reflected in the aggregated criteria weightings matrix shown in Table 4.9.

| Criteria | Weight |
|---|---|
| C1 | 1,7.4,9 |
| C2 | 3,7,9 |
| C3 | 1,4.5,9 |
| C4 | 5,8.5,9 |

**Table 4.9 Aggregated Criteria Weightings Matrix**

**Step 3: Normalise the Fuzzy Matrices**

The aggregated triangular fuzzy values illustrated in Table 4.8 must be normalised using linear scale transformation to bring the various criteria scales into a comparable scale. This is executed on the triangular fuzzy numbers in Table 4.8 using Equation 4.4 of Section 4.5, resulting in the normalized App rankings fuzzy matrix illustrated in Table 4.10.

| Criteria | Alternatives | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | A1 (Alice) | | | A2 (Scratch) | | | A3 (Blockly) | | | A4 (MIT App Inventor) | | |
| C1 | 0,11111 | 0,77778 | 1,00000 | 0,11111 | 0,72222 | 1,00000 | 0,55556 | 0,72222 | 1,00000 | 0,55556 | 0,94444 | 1,00000 |
| C2 | 0,33333 | 0,66667 | 1,00000 | 0,55556 | 0,83333 | 1,00000 | 0,33333 | 0,66667 | 1,00000 | 0,55556 | 0,94444 | 1,00000 |
| C3 | 0,11111 | 0,66667 | 1,00000 | 0,55556 | 0,88889 | 1,00000 | 0,33333 | 0,72222 | 1,00000 | 0,55556 | 0,88889 | 1,00000 |
| C4 | 0,55556 | 0,88889 | 1,00000 | 0,77778 | 1,00000 | 1,00000 | 0,55556 | 0,94444 | 1,00000 | 0,55556 | 0,88889 | 1,00000 |

**Table 4.10 Normalised App Rankings Fuzzy Matrix**

**Step 4: Compute weighted normalized fuzzy values**

In this step of the Fuzzy TOPSIS method, the importance of each criterion must be factored into the triangular fuzzy ratings of the Apps to determine the weighted fuzzy values. Therefore, Equation 4.6 in Section 4.5 has been computed using the aggregated criteria weightings matrix shown in Table 4.9 and the normalised App rankings fuzzy matrix shown in Table 4.10 to determine the weighted normalised App rankings fuzzy matrix shown in Table 4.11.

| Criteria | Alternatives | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | A1 (Alice) | | | A2 (Scratch) | | | A3 (Blockly) | | | A4 (MIT App Inventor) | | |
| C1 | 0,11111 | 5,75556 | 9,00000 | 0,11111 | 5,34444 | 9,00000 | 0,55556 | 5,34444 | 9,00000 | 0,55556 | 6,98889 | 9,00000 |
| C2 | 1,00000 | 4,66667 | 9,00000 | 1,66667 | 5,83333 | 9,00000 | 1,00000 | 4,66667 | 9,00000 | 1,66667 | 6,61111 | 9,00000 |
| C3 | 0,11111 | 3,00000 | 9,00000 | 0,55556 | 4,00000 | 9,00000 | 0,33333 | 3,25000 | 9,00000 | 0,55556 | 4,00000 | 9,00000 |
| C4 | 2,77778 | 7,55556 | 9,00000 | 3,88889 | 8,50000 | 9,00000 | 2,77778 | 8,02778 | 9,00000 | 2,77778 | 7,55556 | 9,00000 |

**Table 4.11 Weighted Normalised App Rankings Fuzzy Matrix**

**Step 5: Calculate fuzzy positive ideal solutions (FPIS) and fuzzy negative ideal solutions (FNIS)**

In the TOPSIS approach, an alternative that is nearest to the Fuzzy Positive Ideal Solution (FPIS) is composed as the best performance values for each alternative, and an alternative that is farthest from the Fuzzy Negative Ideal Solution (FNIS) consists of the worst performance values. These alternatives represent optimal choices and are calculated using Equations 4.8 and 4.9 in Section 4.5. The resulting values are depicted in Table 4.12.

| FNIS and FPIS | | | | | |
|---|---|---|---|---|---|
| FNIS($A^-$) | | | FPIS($A^+$) | | |
| 0,11 | 0,11 | 0,11 | 9,00 | 9,00 | 9,00 |
| 1,00 | 1,00 | 1,00 | 9,00 | 9,00 | 9,00 |
| 0,11 | 0,33 | 0,33 | 9,00 | 9,00 | 9,00 |
| 2,78 | 2,78 | 2,78 | 9,00 | 9,00 | 9,00 |

**Table 4.12 Fuzzy FNIS and FPIS values**

**Step 6: Calculate the distance of each alternative from FNIS ($d^-$) and FPIS ($d^+$)**

The distance of each weighted alternative from the FNIS and the FPIS is computed using Equations 4.10 and 4.11 in Section 4.5 on the Weighted Normalised App Rankings Fuzzy Matrix shown in Table 4.11 and using the FNIS and FPIS values shown in Table 4.12. The result is illustrated in Table 4.13.

| | $d^-$ | | | | $d^+$ | | | |
|---|---|---|---|---|---|---|---|---|
| | A1 | A2 | A3 | A4 | A1 | A2 | A3 | A4 |
| C1 | 6,07926 | 5,95539 | 5,96092 | 6,49394 | 5,46317 | 5,54904 | 5,31262 | 5,01176 |
| C2 | 5,08083 | 5,41004 | 5,08083 | 5,65476 | 5,25287 | 4,61178 | 5,25287 | 4,45289 |
| C3 | 5,23521 | 5,43915 | 5,28102 | 5,43915 | 6,19172 | 5,66594 | 6,00482 | 5,66594 |
| C4 | 4,52928 | 4,92255 | 4,70030 | 4,52928 | 3,68793 | 2,96499 | 3,63599 | 3,68793 |

**Table 4.13 Distance of each weighted alternative from the FNIA and FPIS values**

**Step 7: Calculate the Closeness Coefficient**

The closeness coefficient represents the distances to fuzzy positive ideal solution and the fuzzy negative ideal solution simultaneously. The closeness coefficient of each alternative is calculated using the formulae depicted in Equation 4.12 and represented in Table 4.14.

| Closeness Coefficient (CCi ) | | | | |
|---|---|---|---|---|
| | A1 | A2 | A3 | A4 |
| $d^-$ | 20,92458 | 21,72713 | 21,02307 | 22,11713 |
| $d^+$ | 20,59569 | 18,79175 | 20,20629 | 18,81852 |
| CCi | 0,50396 | 0,53622 | 0,50991 | 0,54029 |

**Table 4.14 Closeness Coefficient of each alternative**

**Step 8: Rank the order of the Alternatives**

The alternative with highest closeness coefficient represents the best alternative and is closest to the FPIS and farthest from the FNIS. Therefore, the ranking of the alternatives is determined by arranging the alternatives in descending order of its closeness coefficient value as illustrated in Table 4.15.

| Ranking | Learning Apps |
|---|---|
| **A4** | MIT App Inventor |
| **A2** | Scratch |
| **A3** | Blockly |
| **A1** | Alice |

**Table 4.15 Ranking of Alternatives**

## 4.7 Chapter Summary

This chapter provided the foundation for this research study and introduced Fuzzy TOPSIS as the technique of multi-criteria decision analysis for the research problem. An analysis and comparison of other MCDM methods has also been presented. The rationale for choosing TOPSIS in a fuzzy environment was also discussed after considering conventional TOPSIS as an option. The chapter included an overview of fuzzy set theory and explained the step-by-step process behind the Fuzzy TOPSIS technique. The other sections of the chapter presented the detailed application of the technique using an example with real data and the designed the algorithm for the implementation of the method in MatLab R2020a. The next chapter presents an analysis of the data from the survey questionnaire and a discussion of the results from the Fuzzy TOPSIS method implemented on MATLAB.

# CHAPTER FIVE

# PRESENTATION OF RESULTS AND DISCUSSION

## 5.1 Introduction

This chapter comprises two sections, namely Section A and Section B. Results from the statistical analysis performed on the data are presented in Section A (Objective 3); the outcome of the Fuzzy TOPSIS algorithm performed on the data collected from the questionnaire and implemented in MatLab R2020a is presented and analysed in Section B. The purpose of Section A is to illustrate reliability and significance test results of the data used in the application of the Fuzzy TOPSIS method in MatLab R2020a highlighted in Section B to determine the ranking of the four alternatives used in the study (Objective 4).

## Section A: Statistical Analysis of Survey

### 5.2 Introduction to the Survey Results

The questionnaire was the primary tool distributed to 217 students to collect data. SPSS version 26.0 was used to analyse the collected data. The results of the analysis are presented as descriptive statistics using tables and graphs. Cross tabulations are used to analyse the criteria across various independent variables, where appropriate. Correlations within the collected data together with the chi square test values were used as an inferential technique to interpret the results reported in this section. This study uses the traditional approach to reporting, wherein it is required that a statement of statistical significance be made. Accordingly, such significant results are indicated in this report as "$p < 0.05$".

### 5.3 The Sample

The population represents the 500 first year programming students, of which a sample of 217 students (in accordance with Table 3.2) was randomly offered the opportunity to participate on the survey. In total, 217 questionnaires were despatched and 175 were returned, which represents an 81% response rate. In general, many researchers would employ various strategies to pursue a high response rate from participants as this would enable them greater confidence to generalise the research results from the sample to the population under study (Creswell 2012).

The researcher used the online survey where cell phones were permissible, anonymity was preserved, students were provided with all of the resources needed to empower and motivate

them around the survey study, and the survey was circulated on multiple platforms for easy and convenient accessibility. According to Standish *et al.* (2018), increasing response rates reduces non-response bias when students with lower response propensities respond in higher numbers and have substantively different views than high response propensity students.

According to Table 5.1 presented by Chapman and Joines (2017) and adapted from Nulty (2008), an eighty percent response rate under stringent conditions covers most requirements for survey validity for a class size of 200. Notwithstanding, Creswell and Creswell (2017) advise that although response rates are important, response bias is a bigger concern and should therefore not be compromised despite acquiring a high response rate.

*Suggested Minimum Response Rates Required for Validity of Data (Adapted from Nulty, 2008)*

| Class Size | Recommended Rates under Liberal Conditions* | Recommended Rates under Stringent Conditions** |
|---|---|---|
| 10 | 75% | 100% |
| 30 | 48% | 96% |
| 50 | 35% | 93% |
| 70 | 28% | 91% |
| 100 | 21% | 87% |
| 200 | 12% | 77% |
| 300 | 8% | 70% |
| 500 | 5% | 58% |

*10% sampling error; 80% confidence level; **3% sampling error; 95% confidence level

**Table 5.1 Suggested Minimum Response Rates**
Source: Extracted from Chapman and Joines (2017)

## 5.4 The Research Instrument

The purpose of the research instrument was to acquire feedback on the impact of each of 4 learning Apps on various critical thinking skills. The survey questionnaire was divided into Section A, which comprised 5 biographical questions, and Section B, which comprised 25 questions designed to measure 14 critical thinking themes, as illustrated in Table 5.2.

| | |
|---|---|
| B1 - B2 | Feedback |
| B3 | Interactivity |
| B4 | Problem-solving |
| B5 - B7 | Collaboration |
| B8 | Metacognition |
| B9 | Logic |
| B10 | Logic and reasoning |
| B11 - B13 | Evaluation |
| B14 | Alternate solutions |
| B15 | Synthesis |
| B16 | Application |
| B17 | Metacognitive monitoring |
| B18 | Multimedia |
| B19 | Simulation programming |
| B20 - B22 | Creativity |
| B23 | Analysis |
| B24 | Complexity of problem |
| B25 | Disposition |

**Table 5.2 Critical Thinking Themes by Question Number**
Source: Researcher's own construction (2021)

The survey questionnaire used the Likert scale to enable complex phenomena to be unpacked using linguistic language variables to record the perceptions of the respondents. The 5-point Likert scale allowed for deeper and detailed insights and the closed-ended questions allowed the data to be collected quickly and easily from the large sample of respondents. The research instrument consisted of 113 items, with a level of measurement at a nominal or an ordinal level.

## 5.5 Reliability Statistics

The two most important aspects of precision are reliability and validity. Reliability is computed by taking several measurements on the same subjects. A reliability coefficient of 0.60 or higher is considered as "acceptable" for a newly developed construct (Taherdoost 2016b).

Table 5.3 reflects the Cronbach's alpha score for all the items that constituted the questionnaire.

| | Section | Number of Items | Cronbach's Alpha |
|---|---|---|---|
| B1 - B2 | Feedback | 8 | 0.877 |
| B3 | Interactivity | 4 | 0.796 |
| B4 | Problem-solving | 4 | 0.827 |
| B5 - B7 | Collaboration | 12 | 0.924 |
| B8 | Metacognition | 4 | 0.829 |
| B9 | Logic | 4 | 0.814 |
| B10 | Logic and reasoning | 4 | 0.849 |
| B11 - B13 | Evaluation | 12 | 0.923 |
| B14 | Alternate solutions | 4 | 0.838 |
| B15 | Synthesis | 4 | 0.833 |
| B16 | Application | 4 | 0.850 |
| B17 | Metacognitive monitoring | 4 | 0.837 |
| B18 | Multimedia | 4 | 0.830 |
| B19 | Simulation programming | 4 | 0.828 |
| B20 - B22 | Creativity | 12 | 0.927 |
| B23 | Analysis | 4 | 0.885 |
| B24 | Complexity of problem | 4 | 0.816 |
| B25 | Disposition | 4 | 0.820 |

**Table 5.3 Cronbach Alpha Scores by Critical Thinking Themes**
Source: Researcher's own construction (2021)

The reliability scores for all sections exceeded the recommended Cronbach's alpha value. All scores exceeded 0.8 with the exception of one (0.796), which illustrates a high degree of reliability and internal consistency on the data (Hinton, McMurray and Brownlow 2014; Larkin 2015). This confirms the degree of acceptable, consistent scoring for these sections of the research.

## 5.6 Biographical Data Analysis

This section summarises the biographical characteristics of the respondents and sought information on their age, gender, device, internet access and programming experience. The overall gender distribution by age is described in Table 5.4.

| Age group (years) | | Gender | | | Total |
|---|---|---|---|---|---|
| | | Male | Female | Prefer not to say | |
| < 18 | Count | 2 | 3 | 1 | 6 |
| | % within Age group (years) | 33.3% | 50.0% | 16.7% | 100.0% |
| | % within Gender | 2.1% | 3.8% | 50.0% | 3.4% |
| | % of Total | 1.1% | 1.7% | 0.6% | 3.4% |
| 18 - 24 | Count | 88 | 69 | 1 | 158 |
| | % within Age group (years) | 55.7% | 43.7% | 0.6% | 100.0% |
| | % within Gender | 93.6% | 87.3% | 50.0% | 90.3% |
| | % of Total | 50.3% | 39.4% | 0.6% | 90.3% |
| > 24 | Count | 4 | 7 | 0 | 11 |
| | % within Age group (years) | 36.4% | 63.6% | 0.0% | 100.0% |
| | % within Gender | 4.3% | 8.9% | 0.0% | 6.3% |
| | % of Total | 2.3% | 4.0% | 0.0% | 6.3% |
| Total | Count | 94 | 79 | 2 | 175 |
| | % within Age group (years) | 53.7% | 45.1% | 1.1% | 100.0% |
| | % within Gender | 100.0% | 100.0% | 100.0% | 100.0% |
| | % of Total | 53.7% | 45.1% | 1.1% | 100.0% |

**Table 5.4 Respondents Gender Distribution by Age**

The analysis of biographical data reflects that the age distributions are not similar ($p < 0.001$). A significantly larger number of survey respondents were in the age category of 18-24 (90.3%), compared to other age groups of under 18 (3.4%) and over 24 (6.3%). Overall, the ratio of males to females is approximately 5:4 (53.7% : 45.1%) ($p < 0.001$). Within the age category of 18 to 24 years, 55.7% were male. Within the category of males (only), 93.6% were between the ages of 18 to 24 years. This category of males between the ages of 18 to 24 years formed 50.3% of the total sample.

Figure 5.1 indicates the device that respondents have been using for the programming module with a Pearson chi-square value of under 0.001 on the categories of laptop, desktop and tablet, but a chi-square value of 0.005 for the category of cell phone, demonstrating a high confidence level that findings can be generalised to the population.

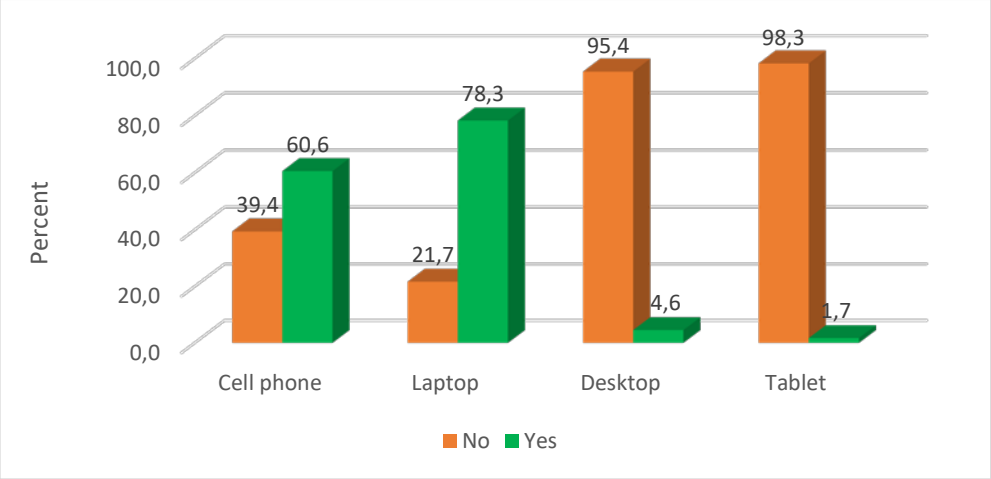


**Figure 5.1 Device used for Programming**

There were significantly more respondents who used cell phones (60.6%) and laptops (78.3%), with significantly fewer using desktops (4.6%) or tablets (1.7%). This is congruent with the current status at the university, where laboratories have been out of bounds for large student numbers due to the current pandemic.

Figure 5.2 indicates the type of internet connectivity that respondents have been using with a Parson chi-square value of under 0.001 on the home Wi-Fi and mobile data options and a chi-square value of 0.705 on the university Wi-Fi option.

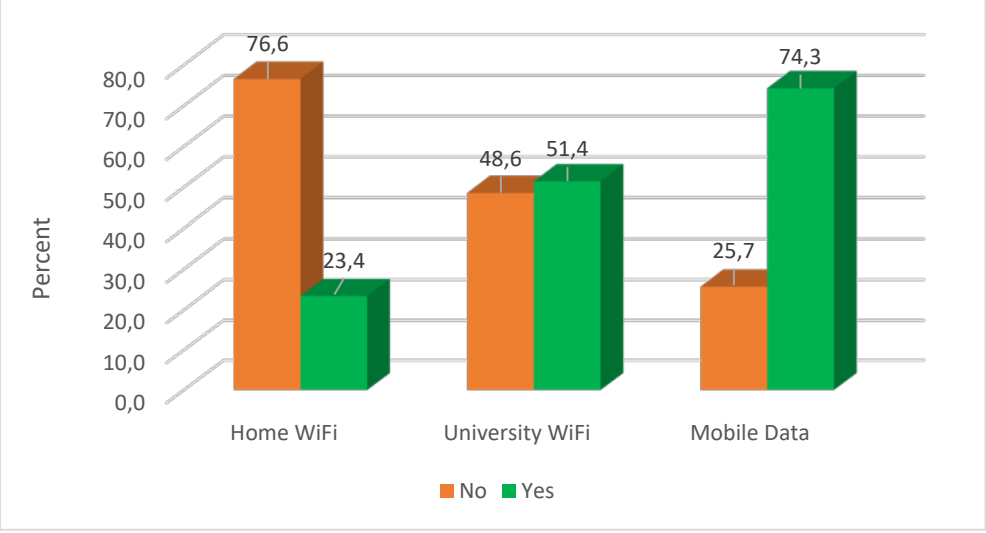


**Figure 5.2 Type of Internet Connectivity**

The majority of respondents (74.3%) used mobile data, with a significantly smaller number using home Wi-Fi (23.4%). There were a similar number of respondents who used the university Wi-Fi to those who did not.

Table 5.5 indicates whether respondents had been exposed to programming before.

| | Frequency | Percent |
|---|---|---|
| No | 91 | 52.0 |
| Yes | 84 | 48.0 |
| Total | 175 | 100.0 |

**Table 5.5 Exposure to Prior Knowledge of Programming**

There were as many respondents who had some experience of programming as there were of those who had not (p = 0.597).

## 5.7 Section Analysis

The section that follows analyses the scoring patterns of the respondents per variable per section. The results are first presented using summarised percentages for the variables that constitute each section. Results are then further analysed according to the importance of the statements.

### 5.7.1 Feedback: The App helped me to correct my errors while I was coding

Table 5.6 represents the results for question B1 of the survey questionnaire.

| | Not sure | Very poor | Poor | Satisfactory | Good | Very good |
|---|---|---|---|---|---|---|
| B1_Alice_Feedback | 6,9 | 1,1 | 4,0 | 34,9 | 37,1 | 16,0 |
| B1_Scratch_Feedback | 3,4 | 1,7 | 3,4 | 18,9 | 37,1 | 35,4 |
| B1_Blockly_Feedback | 6,9 | 3,4 | 5,7 | 32,6 | 35,4 | 16,0 |
| B1_MIT App Inventor_Feedback | 8,0 | 1,1 | 6,3 | 24,6 | 39,4 | 20,6 |

**Table 5.6 Feedback to correct errors**

The results in Table 5.6 show that Scratch is the most favourable learning App providing feedback for students to correct errors, with 73% of the respondents rating the App as ‘good’ or ‘very good’. The least favourable App regarding feedback for students to correct errors is Blockly, with a combined rating of 9.1% of the respondents indicating a rating of ‘poor’ or ‘very poor’.

This evidence is in keeping with a study by Kaya and Yildiz (2019), which reported that the nature of the IDE of Scratch and MIT App Inventor minimises the instances of errors through its interlocking block feature.

### 5.7.2 Feedback: The App immediately alerted me to incorrect code

Table 5.7 represents the results for question B2 of the survey questionnaire.

| | Not sure | Very poor | Poor | Satisfactory | Good | Very good |
|---|---|---|---|---|---|---|
| B2_Alice_Feedback | 5,1 | 2,3 | 8,0 | 29,7 | 38,9 | 16,0 |
| B2_Scratch_Feedback | 5,1 | 1,7 | 8,6 | 18,9 | 36,6 | 29,1 |
| B2_Blockly_Feedback | 8,0 | 1,7 | 6,3 | 33,7 | 38,3 | 12,0 |
| B2_MIT App Inventor_Feedback | 9,1 | 0,6 | 6,3 | 26,3 | 37,7 | 20,0 |

**Table 5.7 Feedback alert on incorrect code**

The results in Table 5.7 show that each of all Apps scored almost the same (86% on combined score from satisfactory to very good) on this critical thinking criterion, while respondents scored Scratch slightly better (66% on a combined score of good to very good) compared to the remaining Apps. Alice and Scratch scored least favourable on this criterion, with 10% of respondents rating these Apps as poor or very poor on this criterion.

The findings indicated in Section 5.7.1 and Section 5.7.2 is synonymous with the observations made in an article by Bau at al. (2017), where the author highlights how the instant locking action of block shapes in Scratch helps novice learners quickly comprehend the legality or error in the design construct, forcing them to reflect on the immediate feedback received.

### 5.7.3 Interactivity: The App tells me exactly where my error lies

Table 5.8 represents the results for question B3 of the survey questionnaire.

| | Not sure | Very poor | Poor | Satisfactory | Good | Very good |
|---|---|---|---|---|---|---|
| B3_Alice_Interactivity | 4,6 | 2,3 | 5,1 | 31,4 | 42,3 | 14,3 |
| B3_Scratch_Interactivity | 5,1 | 1,1 | 5,7 | 28,0 | 33,7 | 26,3 |
| B3_Blockly_Interactivity | 9,1 | 2,9 | 8,6 | 31,4 | 32,0 | 16,0 |
| B3_MIT App Inventor_Interactivity | 8,6 | 1,7 | 6,9 | 26,9 | 34,3 | 21,7 |

**Table 5.8 Interactivity of the App in detecting errors**

On the criterion of interactivity of the App to detect errors, Scratch scored highest (60% on combined score of good to very good), with Alice (57%) and MIT App Inventor (56%) not far behind, while Blockly scored lowest (11% on combined score of very poor to poor).

This is congruent with the literature that highlights the online live environment of MIT App Inventor making for more robust interaction by converting actions to quick visual representations (Bau *et al.* 2017). A study by Erol and Kurt (2017) found that the group interaction offered by Scratch improved the programming achievement scores of the participants. João *et al.* (2019) analysed 26 VPEs which highlighted the interactive and stimulating environment of Alice for 3D game-based programming.

### 5.7.4 Problem-solving: The App allows me to solve large, complex, real-world, authentic problem scenarios

Table 5.9 represents the results for question B4 of the survey questionnaire.

| | Not sure | Very poor | Poor | Satisfactory | Good | Very good |
|---|---|---|---|---|---|---|
| B4_Alice_Problemsolving | 7,4 | 0,0 | 2,9 | 26,3 | 43,4 | 20,0 |
| B4_Scratch_Problemsolving | 5,1 | 0,0 | 1,7 | 18,9 | 38,9 | 35,4 |
| B4_Blockly_Problemsolving | 8,6 | 1,7 | 4,6 | 38,3 | 34,3 | 12,6 |
| B4_MIT App Inventor_Problemsolving | 8,0 | 1,7 | 2,9 | 22,3 | 30,3 | 34,9 |

**Table 5.9 Capacity for large complex problem-solving**

The results in Table 5.9 illustrate that Scratch scored most favourably (74%) on the combined score of good to very good) on its capacity to solve large complex real-world problem scenarios when compared with MIT App Inventor (65%) and Alice (63%), while Blocky was rated least favourable, with 6.3% of the respondents rating it as poor or very poor.

This finding is not supported in a study conducted by Bala and Alacapinar (2021) albeit with younger children, where the researchers found that there was no significant impact of exposing learners to Scratch on their problem-solving ability. However, the authors also refer to other studies with contrasting outcomes. Sharing in the perspective that teaching with Scratch slightly improves the problem-solving confidence levels of the learners, is a study by Kalelioglu and Gülbahar (2014).

The results from this critical thinking criterion are also validated in a study by Xu *et al.* (2019), which identified coding tools such as Scratch, Alice and MIT App Inventor as supporting the novice learner understand complex programming concepts, as well as a study by and Kaya and Yildiz (2019), which highlights the potential of Scratch to tackle complex problems. In contrast, the findings on this criterion are not in line with a study by Weintrop and Wilensky

(2015), where students reported that VPEs such as Scratch lack the power and the authenticity that is needed to tackle large sophisticated project scenarios. According to Sandoval-Reyes (2011), it is not only cumbersome to code complex problems in MIT App Inventor, but the tool also requires deeper knowledge to succeed in complex problem-solving.

### 5.7.5 Collaboration: The App allows me to work on a shared program with my friends

Table 5.10 represents the results for question B5 of the survey questionnaire.

| | Not sure | Very poor | Poor | Satisfactory | Good | Very good |
|---|---|---|---|---|---|---|
| B5_Alice_Collaboration | 11,4 | 0,0 | 5,7 | 29,1 | 35,4 | 18,3 |
| B5_Scratch_Collaboration | 9,1 | 1,7 | 4,6 | 22,9 | 33,1 | 28,6 |
| B5_Blockly_Collaboration | 12,6 | 1,7 | 6,9 | 30,3 | 37,1 | 11,4 |
| B5_MIT App Inventor_Collaboration | 12,0 | 0,0 | 3,4 | 22,9 | 42,3 | 19,4 |

**Table 5.10 Enabling code sharing**

The results in Table 5.10 show that both Scratch and MIT App Inventor are the most favourable learning Apps enabling collaboration through sharing of code, with 61.7% of the respondents rating the App as good or very good. The least favourable App regarding collaboration through sharing of code is Blockly with a combined rating of 8.6% of the respondents indicating a rating of poor or very poor.

### 5.7.6 Collaboration: The App allows me to reuse my code or re-use my peers' solutions

Table 5.11 represents the results for question B6 of the survey questionnaire.

| | Not sure | Very poor | Poor | Satisfactory | Good | Very good |
|---|---|---|---|---|---|---|
| B6_Alice_Collaboration | 9,1 | 0,0 | 4,6 | 21,1 | 42,3 | 22,9 |
| B6_Scratch_Collaboration | 6,9 | 1,1 | 2,3 | 15,4 | 42,3 | 32,0 |
| B6_Blockly_Collaboration | 10,9 | 1,1 | 7,4 | 25,7 | 37,7 | 17,1 |
| B6_MIT App Inventor_Collaboration | 9,1 | 0,6 | 4,6 | 27,4 | 39,4 | 18,9 |

**Table 5.11 Facilitating collaboration through re-use of code**

On the criterion of the App's potential to enable collaboration through re-use of code, Table 5.11 shows Scratch as most favourable (74.3%), followed by Alice (65.1%) and MIT App Inventor (58.3%), while Blockly is rated least favourable (54.9%) in the combined rating of very poor to poor.

This is in line with a study by Kalelioglu and Gülbahar (2014), which reported the potential of Scratch to facilitate the sharing and re-use of code that can easily be integrated and adapted. The authors describe the benefits of Scratch as an IDE where users can create projects using downloaded or web-based software and then share their projects to the Scratch web site for effective collaboration.

### 5.7.7 Collaboration: The App allows me to communicate with my friend about my questions and queries about our projects

Table 5.12 represents the results for question B7 of the survey questionnaire.

| | Not sure | Very poor | Poor | Satisfactory | Good | Very good |
|---|---|---|---|---|---|---|
| B7_Alice_Collaboration | 10,9 | 1,1 | 4,6 | 25,1 | 39,4 | 18,9 |
| B7_Scratch_Collaboration | 9,7 | 1,7 | 4,0 | 20,6 | 37,7 | 26,3 |
| B7_Blockly_Collaboration | 13,1 | 1,1 | 5,7 | 33,7 | 34,3 | 12,0 |
| B7_MIT App Inventor_Collaboration | 14,3 | 1,7 | 4,0 | 26,9 | 34,3 | 18,9 |

**Table 5.12 Engagement between peers about the project**

The results in Table 5.12 show Scratch as the most favourable App (64%) that enables collaboration between peers on solution design compared to the remaining Apps. Blockly is rated as the least favourable (6.9% on a combined rating of very poor to poor) on enabling collaboration through questions and queries.

This finding is synonymous with the results from a comparative study between Alice and Scratch by Durak (2020), which reflected a greater impact of Scratch on the engagement and reflective analysis skills of the learners for problem-solving.

### 5.7.8 Metacognition: The App allows me to easily and repeatedly make changes to my solution

Table 5.13 represents the results for question B8 of the survey questionnaire.

| | Not sure | Very poor | Poor | Satisfactory | Good | Very good |
|---|---|---|---|---|---|---|
| B8_Alice_Metacognition | 5,7 | 1,1 | 1,7 | 17,7 | 37,1 | 36,6 |
| B8_Scratch_Metacognition | 4,0 | 0,6 | 1,1 | 12,0 | 29,1 | 53,1 |
| B8_Blockly_Metacognition | 8,0 | 0,0 | 2,9 | 22,9 | 40,0 | 26,3 |
| B8_MIT App Inventor_Metacognition | 8,0 | 1,1 | 1,7 | 21,7 | 36,6 | 30,9 |

**Table 5.13 Ease of making changes to solutions**

The results in Table 5.13 shows that each of all Apps was generally rated favourably on this criterion, with a combined rating of 89% and above from satisfactory to very good. Notwithstanding, Scratch was rated as the most favourable App, with 82.3% of respondents rating it as the best App to facilitate changes and revisions to code design compared to the remaining Apps, with Alice at 73.7%, MIT App Inventor at67.4% and Blockly at 66.3%.

### 5.7.9 Logic: The App helped me to improve my logic skills

Table 5.14 represents the results for question B9 of the survey questionnaire.

| | Not sure | Very poor | Poor | Satisfactory | Good | Very good |
|---|---|---|---|---|---|---|
| B9_Alice_Logic | 4,6 | 0,6 | 2,3 | 25,1 | 37,1 | 30,3 |
| B9_Scratch_Logic | 3,4 | 0,0 | 1,1 | 15,4 | 27,4 | 52,6 |
| B9_Blockly_Logic | 7,4 | 0,0 | 4,0 | 24,0 | 34,3 | 30,3 |
| B9_MIT App Inventor_Logic | 7,4 | 0,0 | 2,9 | 28,6 | 26,3 | 34,9 |

**Table 5.14 Improving logic skills**

The results in Table 5.14 show that Scratch is the most favourable learning App to improve logic skills, with 80% of the respondents rating the App as good or very good. More specifically, more than 50% of the respondents rated this criterion as very good for Scratch.

According to a study by Erol and Kurt (2017) which employed two instruction methods, learning with Scratch and problem-solving with flowcharts, it was determined that the experience of visualizing the algorithm process in Scratch helps learners understand logic more easily. Based on study results, the authors argued that learners were able to learn logic skills as a result of the experiences obtained during the Scratch process, and then transferred this experience to the learning of a new programming language.

### 5.7.10 Logic and reasoning: The App quickly alerts me when the sequencing of steps in my solution is logically incorrect

Table 5.15 represents the results for question B10 of the survey questionnaire.

| | Not sure | Very poor | Poor | Satisfactory | Good | Very good |
|---|---|---|---|---|---|---|
| B10_Alice_Logicandreasoning | 6,3 | 1,7 | 8,0 | 26,3 | 43,4 | 14,3 |
| B10_Scratch_Logicandreasoning | 6,3 | 1,7 | 8,6 | 17,7 | 32,0 | 33,7 |
| B10_Blockly_Logicandreasoning | 9,1 | 1,7 | 9,1 | 26,3 | 33,7 | 20,0 |
| B10_MIT App Inventor_Logicandreasoning | 8,6 | 1,1 | 5,7 | 20,0 | 40,6 | 24,0 |

**Table 5.15 Quick alert to erroneous logic**

Scratch was rated highest (65.7%) by the respondents on the criterion of facilitating logic and reasoning, with MIT App Inventor following closely behind (64.6%). Blockly and Alice were rated least favourable with 10.9% and 10.3% respectively on the combined rating of poor to very poor.

This finding concurs with the comparative study conducted by Kaya and Yildiz (2019), which identified the salient characteristics of Scratch, Alice and MIT App Inventor, the last identified as having a powerful library and providing an enabling environment to teach computational thinking. The literature review in Chapter 2 defined critical thinking as encompassing abstraction, problem-solving and logic and reasoning in the main. This is also congruent with the study by João *et al.* (2019), which found that designing with blocks assists novice learners to focus their attention on the meaning and reason for the design, thereby promoting logic and reasoning skills.

### 5.7.11 Evaluation: The App helps me to make my code more efficient

Table 5.16 represents the results for question B11 of the survey questionnaire.

| | Not sure | Very poor | Poor | Satisfactory | Good | Very good |
|---|---|---|---|---|---|---|
| B11_Alice_Evaluation | 5,7 | 1,1 | 3,4 | 27,4 | 37,7 | 24,6 |
| B11_Scratch_Evaluation | 4,6 | 1,1 | 0,6 | 18,3 | 31,4 | 44,0 |
| B11_Blockly_Evaluation | 9,1 | 0,0 | 5,1 | 26,9 | 36,0 | 22,9 |
| B11_MIT App Inventor_Evaluation | 8,6 | 1,1 | 2,9 | 21,7 | 28,6 | 37,1 |

**Table 5.16 Enabling efficiency of code design**

The results in Table 5.16 indicate that the respondents rated Scratch the highest (75.4% on the combined rating of good and very good) on the criteria of evaluation to promote more efficiency in code design when compared to the other Apps of MIT App Inventor (65.7%), Alice (62.3%) and Blockly (58.9%).

### 5.7.12 Evaluation: The App is able to evaluate my work

Table 5.17 represents the results for question B12 of the survey questionnaire.

| | Not sure | Very poor | Poor | Satisfactory | Good | Very good |
|---|---|---|---|---|---|---|
| B12_Alice_Evaluation | 6,9 | 1,1 | 2,3 | 17,1 | 40,0 | 32,6 |
| B12_Scratch_Evaluation | 5,7 | 0,0 | 2,3 | 12,6 | 34,3 | 45,1 |
| B12_Blockly_Evaluation | 7,4 | 0,0 | 1,7 | 21,7 | 42,3 | 26,9 |
| B12_MIT App Inventor_Evaluation | 8,0 | 0,0 | 3,4 | 15,4 | 40,6 | 32,6 |

**Table 5.17 Evaluating code design**

The results in Table 5.17 identify Scratch as the most favourable App (79.4%) to evaluate code design, compared to MIT App Inventor (73.1%), Alice (72.6%) and Blockly (69.1%). Notwithstanding this, each of all Apps was generally rated satisfactory and above (a combined score of 88% and above from satisfactory to very good) with the highest combined score of poor to very poor not exceeding 3.4% on this criterion.

### 5.7.13 Evaluation: The App allows me to compare my solution against my peers

Table 5.18 represents the results for question B13 of the survey questionnaire.

| | Not sure | Very poor | Poor | Satisfactory | Good | Very good |
|---|---|---|---|---|---|---|
| B13_Alice_ Evaluation | 9,7 | 0,0 | 4,0 | 28,0 | 39,4 | 18,9 |
| B13_Scratch_ Evaluation | 9,7 | 1,1 | 3,4 | 18,3 | 30,3 | 37,1 |
| B13_Blockly_ Evaluation | 10,3 | 1,1 | 6,3 | 26,3 | 37,1 | 18,9 |
| B13_MIT App Inventor_ Evaluation | 12,0 | 0,6 | 3,4 | 25,1 | 37,7 | 21,1 |

**Table 5.18 Enabling comparison of solutions between peers**

The results in Table 5.18 show that Scratch is the most favourable learning App to enabling evaluation through comparisons of code between peers, with 67.4% of the respondents rating the App as good or very good. The least favourable App that facilitates code comparisons between peers is Blockly with a combined rating of 7.4% of the respondents indicating a rating of poor or very poor.

### 5.7.14 Alternate solutions: The App helps me to think about solving the problem in different ways

Table 5.19 represents the results for question B14 of the survey questionnaire.

| | Not sure | Very poor | Poor | Satisfactory | Good | Very good |
|---|---|---|---|---|---|---|
| B14_Alice_Alternatesolutions | 4,6 | 0,6 | 2,3 | 22,9 | 37,1 | 32,6 |
| B14_Scratch_Alternatesolutions | 4,0 | 1,1 | 3,4 | 13,7 | 28,6 | 49,1 |
| B14_Blockly_Alternatesolutions | 6,9 | 2,3 | 6,9 | 22,3 | 34,9 | 26,9 |
| B14_MIT App Inventor_Alternatesolutions | 8,0 | 1,7 | 4,6 | 16,6 | 33,1 | 36,0 |

**Table 5.19 Facilitating alternate solutions**

On the criterion of problem-solving that encourages alternate solutions, the respondents rated Scratch as best (77.7% on a combined rating of good to very good) followed by Alice (69.7%), MIT App Inventor (69.1%) and Blockly (61.7%). Close to 50% of respondents scored Scratch as very good (49.1%), followed by Alice as good (37.1%), MIT App Inventor as very good (36%) and Blockly as good (34.9%).

In a study by Hutchison, Nadolny and Estapa (2016), the authors highlight the benefit of learning Apps as students identifying, analyzing and implementing possible solutions with the goal of attaining the most efficient and effective arrangement of steps. The results on this criterion are indicative of an exploratory quantitative study of 49 primary school learners by Kalelioglu and Gülbahar (2014), where it was reported that the Scratch participants tried to solve their problems in different ways.

### 5.7.15 Synthesis: The App allows me to solve problems where I have to draw my knowledge from different programming topics

Table 5.20 represents the results for question B15 of the survey questionnaire.

| | Not sure | Very poor | Poor | Satisfactory | Good | Very good |
|---|---|---|---|---|---|---|
| B15_Alice_Synthesis | 6,3 | 0,6 | 2,9 | 27,4 | 38,9 | 24,0 |
| B15_Scratch_Synthesis | 6,9 | 0,6 | 2,9 | 17,7 | 34,9 | 37,1 |
| B15_Blockly_Synthesis | 10,3 | 1,7 | 4,6 | 30,3 | 36,0 | 17,1 |
| B15_MIT App Inventor_Synthesis | 9,7 | 1,1 | 2,9 | 22,9 | 32,0 | 31,4 |

**Table 5.20 Enabling synthesis of the knowledge**

The results in Table 5.20 show Scratch as most favourable, with 72% of the respondents rating the App as good to very good for enforcing synthesis of the knowledge during code design, compared to just over 50% of the respondents rating Blockly as good or very good (53.1%) on the criterion of drawing knowledge from different sections of the programming syllabus during code design.

### 5.7.16 Application: The App allows me to apply the concepts of sequence, selection and iteration

Table 5.21 represents the results for question B16 of the survey questionnaire.

| | Not sure | Very poor | Poor | Satisfactory | Good | Very good |
|---|---|---|---|---|---|---|
| B16_Alice_Application | 4,6 | 0,0 | 2,3 | 22,3 | 46,9 | 24,0 |
| B16_Scratch_Application | 4,6 | 0,0 | 2,9 | 17,7 | 34,9 | 40,0 |
| B16_Blockly_Application | 7,4 | 0,6 | 3,4 | 25,7 | 45,7 | 17,1 |
| B16_MIT App Inventor_Application | 8,6 | 0,6 | 3,4 | 22,9 | 35,4 | 29,1 |

**Table 5.21 Enabling application of knowledge**

The results in Table 5.21 illustrate an over 70% combined rating of good to very good for Scratch (74.9%) and Alice (70.9%), and an over 60% combined rating for MIT App Inventor (64.6%) and Blockly (62.9%) on the criterion of enabling application of the concepts of sequence selection and iteration. The combined rating of poor to very poor on each App is indicated as under 5% of the respondents.

The evidence suggests that the students responded favourably for all Apps on this criterion, which is supported in a study by Tsai (2019) who used MIT App Inventor as the VPE and found increased participation levels and a reinforced foundational knowledge of sequence and loops and a slightly lesser impact on conditions.

### 5.7.17 Metacognitive monitoring: The App interface is designed in a way that encourages me to improve my solution

Table 5.22 represents the results for question B17 of the survey questionnaire.

| | Not sure | Very poor | Poor | Satisfactory | Good | Very good |
|---|---|---|---|---|---|---|
| B17_Alice_Metacognitivemonitoring | 5,7 | 1,7 | 2,9 | 20,6 | 38,9 | 30,3 |
| B17_Scratch_Metacognitivemonitoring | 5,1 | 1,1 | 2,3 | 14,3 | 26,9 | 50,3 |
| B17_Blockly_Metacognitivemonitoring | 7,4 | 2,3 | 4,0 | 22,9 | 37,7 | 25,7 |
| B17_MIT App Inventor_Metacognitivemonitoring | 9,1 | 0,0 | 4,0 | 20,6 | 33,1 | 33,1 |

**Table 5.22 Interface design enabling solution improvement**

The results in Table 5.22 show that Scratch, Alice, MIT App Inventor and Blockly have all been rated satisfactory (> 80%) on an interface design that promotes solution improvement compared to the remaining Apps.

The results from Table 5.13 and Table 5.22 on the criterion of metacognition is supported in the study by Tsai (2019), who reported that the instant feedback and continuous trial and error reflective analysis process using MIT App Inventor forced learners to learn from their mistakes and make improvements to their design.

### 5.7.18 Multimedia: The interface is visually rich in multimedia and includes sound, colour, graphics and animation

Table 5.23 represents the results for question B18 of the survey questionnaire.

| | Not sure | Very poor | Poor | Satisfactory | Good | Very good |
|---|---|---|---|---|---|---|
| B18_Alice_Multimedia | 6,3 | 0,0 | 4,0 | 22,9 | 30,3 | 36,6 |
| B18_Scratch_Multimedia | 4,6 | 1,7 | 1,1 | 13,7 | 28,6 | 50,3 |
| B18_Blockly_Multimedia | 9,7 | 1,7 | 5,7 | 21,7 | 38,9 | 22,3 |
| B18_MIT App Inventor_Multimedia | 12,0 | 0,0 | 4,6 | 20,6 | 36,6 | 26,3 |

**Table 5.23 Enabling use of rich multimedia**

The results in Table 5.23 show that over 80% of the respondents rated all the Apps as satisfactory and above for encompassing rich multimedia. More specifically, Scratch was rated highest (a combined rating of 78.9% from good to very good) compared to the remaining Apps, with 7.4% of the respondents rating Blockly as least favourable with a combined rating of poor to very poor on this criterion.

According to Bau *et al.* (2017), the rich colour blocks, sound palette of Scratch and voice palettes of MIT App Inventor assist learners to quickly and easily select components of the IDE by manipulating recognition over recall.

### 5.7.19 Simulation programming: The App uses simple statements to mimic a high-level programming language in an active learning environment

Table 5.24 represents the results for question B19 of the survey questionnaire.

| | Not sure | Very poor | Poor | Satisfactory | Good | Very good |
|---|---|---|---|---|---|---|
| B19_Alice_Simulationprogramming | 6,3 | 0,6 | 3,4 | 19,4 | 40,0 | 30,3 |
| B19_Scratch_Simulationprogramming | 5,1 | 1,7 | 1,7 | 16,6 | 30,3 | 44,6 |
| B19_Blockly_Simulationprogramming | 8,6 | 1,1 | 5,7 | 23,4 | 37,7 | 23,4 |
| B19_MIT App Inventor_Simulationprogramming | 9,1 | 1,1 | 5,1 | 24,0 | 32,0 | 28,6 |

**Table 5.24 Simulates high-level programming language environment**

The results in Table 5.24 indicate Scratch as most favourable (a combined rating of 74.9% on good to very good) for simulating a high-level programming language environment compared to Alice (70.3%), Blockly (61.1%) and MIT App Inventor (66.6%). Blockly is rated least favourable, with 6.9% of the respondents scoring it a combined rating of poor to very poor.

Although Scratch has been identified as the most popular App on this criterion, the literature describes MIT App Inventor as a live development environment that allows code designers to instantly see their App as it is being coded and as they experiment with blocks and components. Furthermore, a doctoral study by Raddad (2019) revealed the positive impact of MIT App Inventor on computational skills encompassing abstraction and complex problem-solving.

### 5.7.20 Creativity: The App allows me to create useful and original applications like games and movies

Table 5.25 represents the results for question B20 of the survey questionnaire.

| | Not sure | Very poor | Poor | Satisfactory | Good | Very good |
|---|---|---|---|---|---|---|
| B20_Alice_Creativity | 6,9 | 1,1 | 2,9 | 21,1 | 37,1 | 30,9 |
| B20_Scratch_Creativity | 7,4 | 1,7 | 4,6 | 21,7 | 33,7 | 30,9 |
| B20_Blockly_Creativity | 10,9 | 3,4 | 5,7 | 27,4 | 27,4 | 25,1 |
| B20_MIT App Inventor_Creativity | 10,3 | 2,3 | 5,7 | 20,0 | 37,7 | 24,0 |

**Table 5.25 Creativity to design games and movies**

The results in Table 5.25 shows that Alice is the most favourable App (a combined rating of 68% from good to very good) in its capacity to create games and animations compared to the remaining Apps. Blockly was rated least favourable with a combined rating of 52.6% from poor to very poor on this criterion.

These results are congruent with a study by Noone and Mooney (2018), which reported that learners were more interested and excited on creating and building the world rather than on programming concepts.

### 5.7.21 Creativity: The App has various features that enable me to use my creative skills

Table 5.26 represents the results for question B21 of the survey questionnaire.

| | Not sure | Very poor | Poor | Satisfactory | Good | Very good |
|---|---|---|---|---|---|---|
| B21_Alice_Creativity | 5,7 | 0,0 | 0,6 | 22,3 | 30,9 | 40,6 |
| B21_Scratch_Creativity | 5,7 | 0,0 | 1,7 | 13,7 | 32,6 | 46,3 |
| B21_Blockly_Creativity | 6,9 | 1,7 | 5,1 | 24,6 | 38,3 | 23,4 |
| B21_MIT App Inventor_Creativity | 9,1 | 0,0 | 4,0 | 18,9 | 30,9 | 37,1 |

**Table 5.26 Promoting creative skills through GUI**

The results in Table 5.26 show that the GUI in Scratch promotes the users' creative skills the best (a combined rating of 78.9% on good to very good) compared to the remaining Apps, with Alice rated second best by 71.4% of the respondents on this criterion. The least favourable App regarding the promotion of creative skills is Blockly, with a combined rating of 6.9% of the respondents indicating a rating of poor or very poor.

### 5.7.22 Creativity: The App supports the simulation of many creative ideas when solving the problem

Table 5.27 represents the results for question B22 of the survey questionnaire.

| | Not sure | Very poor | Poor | Satisfactory | Good | Very good |
|---|---|---|---|---|---|---|
| B22_Alice_Creativity | 5,1 | 0,0 | 2,9 | 19,4 | 38,9 | 33,7 |
| B22_Scratch_Creativity | 6,3 | 0,0 | 2,3 | 14,3 | 38,3 | 38,9 |
| B22_Blockly_Creativity | 7,4 | 1,1 | 3,4 | 24,0 | 43,4 | 20,6 |
| B22_MIT App Inventor_Creativity | 8,6 | 1,1 | 4,6 | 18,9 | 33,7 | 33,1 |

**Table 5.27 Supporting simulation of creative ideas**

The results in Table 5.27 indicate that over 90% of the respondents rated Alice and Scratch as satisfactory and above to support simulation of creative ideas, with Scratch rated most favourable (on a combined rating of good to very good) on this criterion. Kalelioglu and Gülbahar (2014) described Scratch as a VPE that allows learners to think creatively through their personal creation of stories, games, art, music, animations, and much more.

### 5.7.23 Analysis: When interpreting the problem statement, the App interface gives me clues on how to solve the problem

Table 5.28 represents the results for question B23 of the survey questionnaire.

| | Not sure | Very poor | Poor | Satisfactory | Good | Very good |
|---|---|---|---|---|---|---|
| B23_Alice_Analysis | 8,6 | 2,3 | 6,3 | 29,1 | 36,6 | 17,1 |
| B23_Scratch_Analysis | 6,9 | 2,9 | 6,9 | 23,4 | 37,1 | 22,9 |
| B23_Blockly_Analysis | 9,1 | 2,3 | 6,3 | 30,9 | 33,7 | 17,7 |
| B23_MIT App Inventor_Analysis | 10,9 | 2,9 | 6,3 | 28,6 | 34,3 | 17,1 |

**Table 5.28 Promoting problem statement analysis through GUI**

The results in Table 5.28 reveal that the highest number of respondents rated each of all Apps good for promoting problem analysis through its graphical user interface (GUI). More specifically, 60% of the respondents rated Scratch as most favourable on this criterion compared to the remaining Apps, while close to 10% of the respondents also rated Scratch as least favourable (9.7%).

### 5.7.24 Complexity of problem: The App allows me the flexibility to elaborate or build on my idea

Table 5.29 represents the results for question B24 of the survey questionnaire.

| | Not sure | Very poor | Poor | Satisfactory | Good | Very good |
|---|---|---|---|---|---|---|
| B24_Alice_Complexityofproblem | 6,9 | 0,6 | 2,3 | 20,0 | 37,7 | 32,6 |
| B24_Scratch_Complexityofproblem | 5,1 | 0,6 | 2,9 | 16,6 | 33,7 | 41,1 |
| B24_Blockly_Complexityofproblem | 8,6 | 2,3 | 5,1 | 22,9 | 37,7 | 23,4 |
| B24_MIT App Inventor_Complexityofproblem | 8,6 | 0,6 | 5,1 | 21,1 | 32,6 | 32,0 |

**Table 5.29 Flexibility to build on project design**

The results in Table 5.29 indicates that over 70% of the respondents rated Scratch (74.9%) and Alice (70.3%) as more favourable to enable flexibility to build on code design, while over 60% of the respondents rated MIT App Inventor (64.6%) and Blockly (61.1%) on this criterion.

This is synonymous with the study by João *et al.* (2019), who analysed 26 VPEs and identified Scratch as the most appropriate and flexible environments to learn programming concepts.

### 5.7.25 Disposition: The App forces me to have an enquiring mind

Table 5.30 represents the results for question B25 of the survey questionnaire.

| | Not sure | Very poor | Poor | Satisfactory | Good | Very good |
|---|---|---|---|---|---|---|
| B25_Alice_Disposition | 5,1 | 0,6 | 4,0 | 23,4 | 38,3 | 28,6 |
| B25_Scratch_Disposition | 4,0 | 0,6 | 2,3 | 18,3 | 30,3 | 44,6 |
| B25_Blockly_Disposition | 6,3 | 0,6 | 2,3 | 25,7 | 34,3 | 30,9 |
| B25_MIT App Inventor_Disposition | 6,9 | 0,6 | 4,0 | 18,9 | 29,7 | 40,0 |

**Table 5.30 Promotes enquiry**

The results in Table 5.30 show that Scratch is the most favourable learning App that promotes enquiry with 74.9% of the respondents rating the App as 'good' or 'very good' compared to MIT App Inventor (69.7%), Alice (66.9%) and Blockly (65.1%). Notwithstanding, Table 5.30 also reveals a combined rating of 88.6% and above, from satisfactory to very good on each of all Apps on this criterion. A rating of poor to very poor represented under 5% of the scoring on each App.

These findings are supported in the study by Tsai (2019), which highlights the learner-centred characteristic of VPEs providing motivation, excitement and a sense of accomplishment to the learner.

## 5.8 Cross Tabulations

A series of cross tabulations were conducted in terms of Laptop users and feedback provided by the four Learning Apps, namely, Scratch, Alice, MIT App Inventor and Blockly. Feedback relates to the learning App helping the user to correct errors while they were coding. According to Table 5.31, Laptops are significant as the majority of the respondents (78.3%) used this device for programming.

Table 5.31 show the results of the cross tabulations between *Feedback Vs Lap Top* users for the Scratch learning application.

| **B1_Scratch_Feedback * Laptop** | | | Laptop | | Total |
|---|---|---|---|---|---|
| | | | No | Yes | |
| | Not sure | % within Laptop | 7,9% | 2,2% | 3,4% |
| | Very poor | % within Laptop | 2,6% | 1,5% | 1,7% |
| | Poor | % within Laptop | 0,0% | 4,4% | 3,4% |
| | Satisfactory | % within Laptop | 26,3% | 16,8% | 18,9% |
| | Good | % within Laptop | 28,9% | 39,4% | 37,1% |
| | Very good | % within Laptop | 34,2% | 35,8% | 35,4% |
| **Total** | | % within Laptop | 100,0% | 100,0% | 100,0% |

**Table 5.31 Cross Tabulation Scratch Feedback vs Laptop**

Approximately 75% of the respondents who use a laptop believed that Scratch was helpful in providing feedback. Only around 2% was not sure. This result concurs with the result in Table 5.6 that shows Scratch as the most favorable App for all device users when it came to helping users to correct their errors while coding. Table 5.32 shows the results of the cross tabulations between *Feedback vs Laptop* users for the Alice learning application.

| **B1_Alice_Feedback * Laptop** | | | Laptop | | Total |
|---|---|---|---|---|---|
| | | | No | Yes | |
| | Not sure | % within Laptop | 10,5% | 5,8% | 6,9% |
| | Very poor | % within Laptop | 0,0% | 1,5% | 1,1% |
| | Poor | % within Laptop | 0,0% | 5,1% | 4,0% |
| | Satisfactory | % within Laptop | 31,6% | 35,8% | 34,9% |
| | Good | % within Laptop | 39,5% | 36,5% | 37,1% |
| | Very good | % within Laptop | 18,4% | 15,3% | 16,0% |
| **Total** | | % within Laptop | 100,0% | 100,0% | 100,0% |

**Table 5.32 Cross Tabulation Alice Feedback vs Laptop**

Approximately 51% of the respondents who use a laptop believed that Alice provided important feedback. This means that Scratch is more favorable for providing feedback than Alice among Laptop users.

Table 5.33 shows the results of the cross tabulations between *Feedback vs Laptop* users for the MIT App Inventor learning application.

| B1_MIT App Inventor_Feedback * Laptop | | | Laptop | | Total |
|---|---|---|---|---|---|
| | | | No | Yes | |
| | Not sure | % within Laptop | 10,5% | 7,3% | 8,0% |
| | Very poor | % within Laptop | 0,0% | 1,5% | 1,1% |
| | Poor | % within Laptop | 5,3% | 6,6% | 6,3% |
| | Satisfactory | % within Laptop | 31,6% | 22,6% | 24,6% |
| | Good | % within Laptop | 39,5% | 39,4% | 39,4% |
| | Very good | % within Laptop | 13,2% | 22,6% | 20,6% |
| **Total** | | % within Laptop | 100,0% | 100,0% | 100,0% |

**Table 5.33 Cross Tabulation MIT App Inventor Feedback vs Laptop**

Approximately 62% of Laptop users believed that MIT App Inventor provided sufficient feedback for coding. This percentage is greater than the 51% of Alice and less than the 75% of the Scratch application.

Table 5.34 shows the results of the cross tabulations between *Feedback Vs Laptop* users for the Blockly learning application.

| B2_Blockly_Feedback * Laptop | | | Laptop | | Total |
|---|---|---|---|---|---|
| | | | No | Yes | |
| | Not sure | % within Laptop | 13,2% | 6,6% | 8,0% |
| | Very poor | % within Laptop | 2,6% | 1,5% | 1,7% |
| | Poor | % within Laptop | 2,6% | 7,3% | 6,3% |
| | Satisfactory | % within Laptop | 28,9% | 35,0% | 33,7% |
| | Good | % within Laptop | 44,7% | 36,5% | 38,3% |
| | Very good | % within Laptop | 7,9% | 13,1% | 12,0% |
| **Total** | | % within Laptop | 100,0% | 100,0% | 100,0% |

**Table 5.34: Cross Tabulation Blockly Feedback vs Laptop**

Approximately 49% of Blocky laptop users perceived that the App was favourable for providing feedback. Blockly is least preferred for laptop users in terms of providing feedback as the above results show that Scratch is 75%, Alice is 51% and MIT App Inventor is 62%. The cross tabulation results from Table 5.31, Table 5.32, Table 5.33 and Table 5.34 show that

the most preferred learning App by novice first year programming Laptop users is Scratch and the least preferred by students is Blockly.

## 5.9 Pearson's Chi-square Tests for Hypothesis testing

The Chi-square test is used to show statistical significance between two or more variables. There is a significant relationship between the two variables if the p-value is less than 0.05. Chi-square tests were conducted to test the relationship between each of the four learning applications on the attributes, namely, problem-solving, collaboration, logic and reasoning, alternate solutions, creativity and complexity of problem.

The results of the Chi-square tests are shown in Table 5.35 and helped the researcher discover trends in the relationships between the learning Apps and the attributes.

| | Chi-Square | df | Asymp. Sig. |
|---|---|---|---|
| B4_Alice_Problemsolving | 91,029 | 4 | < 0.001 |
| B4_Scratch_Problemsolving | 100,629 | 4 | < 0.001 |
| B4_Blockly_Problemsolving | 129,149 | 5 | < 0.001 |
| B4_MIT App Inventor_Problemsolving | 108,92 | 5 | < 0.001 |
| B5_Alice_Collaboration | 52,686 | 4 | < 0.001 |
| B5_Scratch_Collaboration | 92,189 | 5 | < 0.001 |
| B5_Blockly_Collaboration | 101,72 | 5 | < 0.001 |
| B5_MIT App Inventor_Collaboration | 73,829 | 4 | < 0.001 |
| B10_Alice_Logicandreasoning | 128,189 | 5 | < 0.001 |
| B10_Scratch_Logicandreasoning | 96,989 | 5 | < 0.001 |
| B10_Blockly_Logicandreasoning | 76,76 | 5 | < 0.001 |
| B10_MIT AppInventor_Logicandreasoning | 111,594 | 5 | < 0.001 |
| B14_Alice_Alternatesolutions | 138,886 | 5 | < 0.001 |
| B14_Scratch_Alternatesolutions | 187,091 | 5 | < 0.001 |
| B14_Blockly_Alternatesolutions | 90,886 | 5 | < 0.001 |
| B14_MIT App Inventor_Alternatesolutions | 114,474 | 5 | < 0.001 |
| B20_Alice_Creativity | 122,703 | 5 | < 0.001 |
| B20_Scratch_Creativity | 102,131 | 5 | < 0.001 |
| B20_Blockly_Creativity | 66,406 | 5 | < 0.001 |
| B20_MIT App Inventor_Creativity | 91,914 | 5 | < 0.001 |
| B24_Alice_Complexityofproblem | 133,263 | 5 | < 0.001 |
| B24_Scratch_Complexityofproblem | 154,589 | 5 | < 0.001 |
| B24_Blockly_Complexityofproblem | 97,88 | 5 | < 0.001 |
| B24_MIT App Inventor_Complexityofproblem | 101,377 | 5 | < 0.001 |

**Table 5.35 Chi-square Significance Testing**

Since the p-value is less than 0.05 in the table above for attributes of problem-solving, collaboration, logic and reasoning, alternate solutions, creativity and complexity of problem. Table 5.35 shows there is a significant relationship between Alice, Scratch, Blockly, and the MIT App and the attributes problem-solving, collaboration, logic and reasoning, alternate solutions, creativity and complexity of problem. We can conclude from these tests that the four learning Apps play a significant role in developing problem-solving ability, improving collaboration, expanding logic and reasoning potential, providing alternate solutions, improving creative ability and providing better skills in dealing with complex problems.

## Section B: Analysis of Fuzzy TOPSIS Results

### 5.10 Results generated from Fuzzy TOPSIS App developed on MatLab R2020a

In this section, the results produced from the Fuzzy TOPSIS software application implemented on MatLab R2020a is presented with explanations in steps according to input and outputs of the application. This application utilised complex mathematical equations on the backend which corresponds with the Fuzzy TOPSIS method presented in Section 4.5. The results are based on preferences shown by decision-makers (first year programming students) on choice of visual programming learning applications based on set critical thinking criteria.

Table 5.36 shows the fuzzy rating scale where linguistic terms, namely, Not Important (N), Slightly Important (SI), Moderately Important (MI), Important (I) and a Very Important (VI) are expressed as a fuzzy triple using integers in the range 1 to 9. These linguistic terms are used to rate the criteria.

| Linguistic term | Membership function |
|---|---|
| Not Important (N) | (1,1,3) |
| Slightly Important (SI) | (1,3,5) |
| Moderately Important (MI) | (3,5,7) |
| Important (I) | (5,7,9) |
| Very Important (VI) | (7,9,9) |

**Table 5.36 Linguistic and Fuzzy Triple for rating Criteria**

Table 5.37 presents the measures for critical thinking synthesized from the extant literature. Symbols C1 to C25 are used to represent the critical thinking criteria for selection of programming learning applications for first year university students.

| No. | Critical Thinking Criteria |
|---|---|
| C1 | Feedback: The App helped me correct my errors while I was coding |
| C2 | Feedback: The App alerted me to incorrect code |
| C3 | Interactivity: The App tells me where my error lies |
| C4 | Problem-solving: The App allows me to solve large, complex, real world, authentic problem scenarios |
| C5 | Collaboration: The App allows me to work on a shared program with my friends |
| C6 | Collaboration: The App allows me to re-use my code or re-use my peers' solutions |
| C7 | Collaboration: The App allows me to communicate with my friend about my questions and queries about our projects |
| C8 | Metacognition: The App allows me to easily and repeatedly make changes to my solution |
| C9 | Logic: The App helped me to improve my logic skills |
| C10 | Logic and reasoning: The App quickly alerts me when the sequencing of steps in my solution is logically incorrect |
| C11 | Evaluation: The App helps me to make my code more efficient |
| C12 | Evaluation: The App is able to evaluate my work |
| C13 | Evaluation: The App allows me to compare my solution against my peers |
| C14 | Alternate solutions: The App helps me to think about solving the problem in different ways |
| C15 | Synthesis: The App allows me to solve problems where I have to draw my knowledge from different programming topics |
| C16 | Application: The App allows me to apply the concepts of sequence, selection and iteration |
| C17 | Metacognitive monitoring: The App interface is designed in a way that encourages me to improve my solution |
| C18 | Multimedia: The interface is visually rich in multimedia and includes sound, color, graphics and animation |
| C19 | Simulation programming: The App uses simple statements to mimic a high-level programming language in an active learning environment |
| C20 | Creativity: The App allows me to create useful and original applications like games and movies |
| C21 | Creativity: The App has various features that enable me to use my creative skills |
| C22 | Creativity: The App supports the simulation of many creative ideas when solving the problem |
| C23 | Analysis: When interpreting the problem statement, the App interface gives me clues on how to solve the problem |
| C24 | Complexity of problem: The App allows me the flexibility to elaborate or build on my idea |
| C25 | Disposition: The App forces me to have an enquiring mind |

**Table 5.37 Measures for Critical Thinking (Criteria)**

Table 5.38 shows the linguistic rating of the 25 criteria by 10 Academic Experts teaching programming at university level. Their input values are represented by the symbols D1 to D10 in the table below.

| | CRITERIA | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| LINGUISTIC VALUES | C1 | C2 | C3 | C4 | C5 | C6 | C7 | C8 | C9 | C10 | C11 | C12 | C13 | C14 | C15 | C16 | C17 | C18 | C19 | C20 | C21 | C22 | C23 | C24 | C25 |
| D1 | I | VI | MI | VI | MI | MI | I | I | VI | VI | VI | I | MI | VI | VI | VI | MI | SI | SI | MI | I | I | VI | I | I |
| D2 | I | VI | I | MI | I | SI | I | MI | VI | VI | VI | I | MI | I | I | VI | I | MI | MI | I | MI | MI | MI | I | I |
| D3 | VI | VI | I | VI | I | I | MI | I | I | VI | VI | MI | MI | I | I | I | MI | SI | SI | SI | SI | SI | I | I | I |
| D4 | I | I | I | I | I | I | MI | VI | VI | VI | VI | I | MI | VI | VI | VI | MI | MI | MI | MI | MI | MI | I | I | VI |
| D5 | VI | VI | I | MI | VI | VI | I | VI | VI | VI | I | I | MI | I | I | VI | I | MI | I | VI | I | VI | VI | VI | VI |
| D6 | VI | VI | VI | MI | MI | MI | I | VI | VI | VI | VI | I | I | VI | MI | VI | I | I | I | MI | MI | MI | I | I | VI |
| D7 | SI | VI | I | I | MI | SI | SI | I | I | VI | SI | SI | SI | VI | I | I | I | SI | SI | MI | SI | SI | MI | VI | VI |
| D8 | I | I | I | VI | MI | MI | MI | VI | VI | VI | I | I | MI | I | VI | VI | I | I | I | I | MI | I | VI | VI | VI |
| D9 | VI | MI | I | MI | I | VI | VI | VI | VI | VI | VI | I | I | VI | VI | VI | MI | MI | MI | VI | MI | I | VI | I | I |
| D10 | VI | SI | VI | I | SI | SI | I | VI | VI | VI | VI | VI | VI | VI | VI | VI | I | SI | I | VI | SI | I | I | MI | I |

**Table 5.38 Linguistic and Fuzzy Triple for the Critical Thinking Attributes (Criteria)**

The Best Non-fuzzy Performance value (BNP) for a criterion weighting j, can be calculated using the following equation (Safari, Faghih and Fathi 2012):

$$BNP_{wj} = [(Upper\ bound_{wj} - lower\ bound_{wj}) + (Middle\ bound_{wj} - lower\ bound_{wj})]/3 + lower\ bound_{wj} \quad (5.1)$$

The BNP values give an indication of the relative importance of the criteria. Table 5.39 shows the aggregated score for each criterion. Table 5.39 also shows the calculated BNP value for each criteria using Equation 5.1. Linguistic Weights were assigned based on the BNP values according to descriptions given in Table 5.36.

| Criteria | Aggregated Fuzzy Score | BNP Value | Linguistic Weight |
|---|---|---|---|
| C1 | 1, 7.4, 9 | 5,8 | I |
| C2 | 1, 7.2, 9 | 5,733333 | I |
| C3 | 1, 7, 9 | 5,666667 | I |
| C4 | 1, 6, 9 | 5,333333 | MI |
| C5 | 1, 5, 9 | 5 | MI |
| C6 | 1, 4.4, 9 | 4,8 | SI |
| C7 | 1, 5.4, 9 | 5,133333 | MI |
| C8 | 1, 7.8, 9 | 5,933333 | I |
| C9 | 5, 8.6, 9 | 7,533333 | VI |
| C10 | 7, 9, 9 | 8,333333 | VI |
| C11 | 1, 7.8, 9 | 5,933333 | I |
| C12 | 1, 6.2, 9 | 5,4 | MI |
| C13 | 1, 4.2, 9 | 4,733333 | SI |
| C14 | 5, 8.2, 9 | 7,4 | VI |
| C15 | 1, 7.6, 9 | 5,866667 | I |
| C16 | 5, 8.6, 9 | 7,533333 | VI |
| C17 | 1, 5.4, 9 | 5,133333 | MI |
| C18 | 1, 3, 9 | 4,333333 | SI |
| C19 | 1, 4, 9 | 4,666667 | SI |
| C20 | 1, 5.4, 9 | 5,133333 | MI |
| C21 | 1, 3.2, 9 | 4,4 | SI |
| C22 | 1, 4.8, 9 | 4,933333 | SI |
| C23 | 1, 7, 9 | 5,666667 | I |
| C24 | 1, 7.2, 9 | 5,733333 | I |
| C25 | 5, 8, 9 | 7,333333 | VI |

**Table 5.39 Aggregated Scores and BNP values for Critical Thinking Attributes (Criteria)**

Table 5.40 shows the ranking of the criteria based on the BNP values indicated in Table 5.39.

| Ranked Criteria | | | | | |
|---|---|---|---|---|---|
| 1. | C10 | 11. | C24 | 21. | C6 |
| 2. | C9 | 12. | C3 | 22. | C13 |
| 3. | C16 | 13. | C23 | 23. | C19 |
| 4. | C14 | 14. | C12 | 24. | C21 |
| 5. | C25 | 15. | C4 | 25. | C18 |
| 6. | C8 | 16. | C7 | | |
| 7. | C11 | 17. | C17 | | |
| 8. | C15 | 18. | C20 | | |
| 9. | C1 | 19. | C5 | | |
| 10. | C2 | 20. | C22 | | |

**Table 5.40 Ranked Criteria based on BNP values**

Table 5.40 shows that C10 (Logic and reasoning: The App quickly alerts me when the sequencing of steps in my solution is logically incorrect) was the most highly ranked criteria by the expert decision-makers. The second most highly recommended criterion is C9 (Logic: The App helped me to improve my logic skills), followed by C16 (Application: The App allows me to apply the concepts of sequence, selection and iteration) and C14 (Alternate solutions: The App helps me to think about solving the problem in different ways). The least important criterion to decision-makers is criterion C18 (Multimedia: The interface is visually rich in multimedia and includes sound, color, graphics and animation).

Table 5.41 shows the fuzzy rating scale for weighing critical thinking criteria. Linguistic terms such as Very Poor (VP), Poor (P), Satisfactory (S), Good (G) and Very Good (VG) are expressed as a fuzzy triple with integers in the range 1 to 9. The Not Sure (N) option expressed as (0, 0, 0) is included in the application to cater for students who are not sure what rating to assign to criteria.

| Linguistic term | Membership function |
|---|---|
| Not Sure (N) | (0,0,0) |
| Very Poor (VP) | (1,1,3) |
| Poor (P) | (1,3,5) |
| Satisfactory (S) | (3,5,7) |
| Good (G) | (5,7,9) |
| Very Good (VG) | (7,9,9) |

**Table 5.41 Linguistic and Fuzzy Triple for the Critical Thinking Alternatives (Learning Apps)**

Table 5.42 shows the numeric labels for Learning Apps (Alternatives).

| Programming Learning Apps | Alternatives Label |
|---|---|
| Alice | 1 |
| Scratch | 2 |
| Blockly | 3 |
| MIT App Inventor | 4 |

**Table 5.42 Numeric Labels for the Programming Learning Apps (Alternatives)**

Table 5.43 presents a snapshot of the assigned ratings by 175 student decision-makers to assess Scratch using the 25 critical thinking criteria in linguistic terms as shown in Table 5.41.

| Criteria | C1 | C2 | C3 | C4 | C5 | C6 | C7 | … | C25 |
|---|---|---|---|---|---|---|---|---|---|
| DM1 | G | G | VG | G | VG | VG | G | … | VG |
| DM2 | P | S | S | G | G | S | G | … | G |
| DM3 | G | VG | G | VG | G | VG | VG | … | VG |
| DM4 | VG | VG | VG | G | VG | VG | VG | … | VG |
| DM5 | VG | VG | VG | VG | G | VG | VG | … | VG |
| DM6 | VG | G | VG | VG | VG | VG | VG | … | VG |
| DM7 | G | G | G | VG | G | S | G | … | S |
| DM8 | G | P | S | VG | G | G | S | … | VG |
| DM9 | S | G | S | S | G | S | S | … | VG |
| DM10 | P | P | VP | G | VP | VP | VP | … | VG |
| … | … | … | … | … | … | … | … | … | … |
| … | … | … | … | … | … | … | … | … | … |
| DM173 | G | G | S | S | G | VG | VG | … | VG |
| DM174 | VG | S | VG | VG | S | VG | VG | … | G |
| DM175 | VG | S | G | G | G | G | VG | … | G |

**Table 5.43 Assigned Rating by Decision-makers for Scratch**

Table 5.44 presents the results of the Normalised Fuzzy Decision Matrix from the MatLab R2020a application. This matrix was generated using Equation 4.4.

| | Alice (A1) | Scratch (A2) | Blockly (A3) | MIT Inventor (A4) |
|---|---|---|---|---|
| C1 | [0,0.657142857142857,1] | [0,0.761269841269841,1] | [0,0.639365079365079,1] | [0,0.671111111111111,1] |
| C2 | [0,0.656507936507937,1] | [0,0.711111111111111,1] | [0,0.627936507936508,1] | [0,0.660952380952381,1] |
| C3 | [0,0.666031746031746,1] | [0,0.700952380952381,1] | [0,0.615238095238095,1] | [0,0.657777777777778,1] |
| C4 | [0,0.693333333333333,1] | [0,0.766984126984127,1] | [0,0.622222222222222,1] | [0,0.719365079365079,1] |
| C5 | [0,0.639365079365079,1] | [0,0.687619047619048,1] | [0,0.596190476190476,1] | [0,0.661587301587302,1] |
| C6 | [0,0.690158730158730,1] | [0,0.743492063492063,1] | [0,0.633650793650794,1] | [0,0.663492063492064,1] |
| C7 | [0,0.651428571428572,1] | [0,0.685714285714286,1] | [0,0.594285714285714,1] | [0,0.619682539682540,1] |

| | | | | |
|---|---|---|---|---|
| C8 | [0,0.760000000000000,1] | [0,0.829206349206349,1] | [0,0.710476190476191,1] | [0,0.720634920634921,1] |
| C9 | [0,0.739682539682540,1] | [0,0.828571428571429,1] | [0,0.716190476190476,1] | [0,0.721269841269841,1] |
| C10 | [0,0.655238095238095,1] | [0,0.714920634920635,1] | [0,0.640634920634921,1] | [0,0.686984126984127,1] |
| C11 | [0,0.704126984126984,1] | [0,0.789206349206349,1] | [0,0.674920634920635,1] | [0,0.725079365079365,1] |
| C12 | [0,0.740952380952381,1] | [0,0.795555555555556,1] | [0,0.723809523809524,1] | [0,0.738412698412698,1] |
| C13 | [0,0.664126984126984,1] | [0,0.721269841269841,1] | [0,0.645714285714286,1] | [0,0.656507936507937,1] |
| C14 | [0,0.749841269841270,1] | [0,0.802539682539682,1] | [0,0.688888888888889,1] | [0,0.726984126984127,1] |
| C15 | [0,0.704761904761905,1] | [0,0.751111111111111,1] | [0,0.636825396825397,1] | [0,0.700952380952381,1] |
| C16 | [0,0.735873015873016,1] | [0,0.779047619047619,1] | [0,0.681904761904762,1] | [0,0.706031746031746,1] |
| C17 | [0,0.730793650793651,1] | [0,0.800000000000000,1] | [0,0.693333333333333,1] | [0,0.716825396825397,1] |
| C18 | [0,0.741587301587302,1] | [0,0.806984126984127,1] | [0,0.666666666666667,1] | [0,0.676825396825397,1] |
| C19 | [0,0.733968253968254,1] | [0,0.780952380952381,1] | [0,0.678095238095238,1] | [0,0.686349206349206,1] |
| C20 | [0,0.725714285714286,1] | [0,0.708571428571429,1] | [0,0.640000000000000,1] | [0,0.666031746031746,1] |
| C21 | [0,0.771428571428572,1] | [0,0.798095238095238,1] | [0,0.687619047619048,1] | [0,0.729523809523810,1] |
| C22 | [0,0.756825396825397,1] | [0,0.773333333333333,1] | [0,0.689523809523810,1] | [0,0.714920634920635,1] |
| C23 | [0,0.641269841269841,1] | [0,0.673650793650794,1] | [0,0.634285714285714,1] | [0,0.620952380952381,1] |
| C24 | [0,0.738412698412698,1] | [0,0.775873015873016,1] | [0,0.674285714285714,1] | [0,0.708571428571429,1] |
| C25 | [0,0.727619047619048,1] | [0,0.791111111111111,1] | [0,0.726349206349206,1] | [0,0.749841269841270,1] |

**Table 5.44 Normalized Fuzzy Decision Matrix**

Table 5.45 shows the results for FNIS and FPIS generated from the MatLab R2020a Application using Equations 4.9 and 4.8 respectively.

| FNIS (A-) | FPIS (A+) |
|---|---|
| [0,4.47555555555556,9] | [0,5.32888888888889,9] |
| [0,4.39555555555556,9] | [0,4.97777777777778,9] |
| [0,4.30666666666667,9] | [0,4.90666666666667,9] |
| [0,3.11111111111111,7] | [0,3.83492063492064,7] |
| [0,2.98095238095238,7] | [0,3.43809523809524,7] |
| [0,1.90095238095238,5] | [0,2.23047619047619,5] |
| [0,2.97142857142857,7] | [0,3.42857142857143,7] |
| [0,4.97333333333333,9] | [0,5.80444444444444,9] |
| [0,6.44571428571429,9] | [0,7.45714285714286,9] |
| [0,5.76571428571429,9] | [0,6.43428571428571,9] |
| [0,4.72444444444444,9] | [0,5.52444444444445,9] |
| [0,3.61904761904762,7] | [0,3.97777777777778,7] |
| [0,1.93714285714286,5] | [0,2.16380952380952,5] |
| [0,6.20000000000000,9] | [0,7.22285714285714,9] |
| [0,4.45777777777778,9] | [0,5.25777777777778,9] |
| [0,6.13714285714286,9] | [0,7.01142857142857,9] |
| [0,3.46666666666667,7] | [0,4,7] |
| [0,2,5] | [0,2.42095238095238,5] |
| [0,2.03428571428571,5] | [0,2.34285714285714,5] |
| [0,3.20000000000000,7] | [0,3.62857142857143,7] |
| [0,2.06285714285714,5] | [0,2.39428571428571,5] |

| | |
|---|---|
| [0,2.06857142857143,5] | [0,2.32000000000000,5] |
| [0,4.34666666666667,9] | [0,4.71555555555556,9] |
| [0,4.72000000000000,9] | [0,5.43111111111111,9] |
| [0,6.53714285714286,9] | [0,7.12000000000000,9] |

**Table 5.45 Normalized Fuzzy Decision Matrix**

Table 5.46 gives the results of the Closeness Coefficient values generated by the MatLab application. These values are generated by the MatLab App using Equation 4.12.

| Rank Order | Programming Learning App | CCi |
|---|---|---|
| **2** | Scratch | 0.9941 |
| **1** | Alice | 0.4345 |
| **4** | MIT inventor | 0.3258 |
| **3** | Blockly | 0.0064 |

**Table 5.46 Closeness Coefficient**

The CCi value is a ratio scale rating showing the ranking of learning App by 175 decision-makers from highest to lowest, CCi (Scratch) > CCi (Alice)> CCi (MIT Inventor) > CCi (Blockly). The results show that Scratch is the most preferred application for learning programming as it had the highest CCi value. The App with the lowest CCi value is Blockly, which therefore is the least preferred application.

## 5.11 Chapter Summary

This chapter presented the data collected through the research instrument and analyzed the scoring patterns of the respondents before discussing the findings of the study. It discussed the reliability and validity of these results, then concluded with applying the data to the Fuzzy TOPSIS method implemented on the MatLab App to determine the ranking of the alternatives in this multi-criteria, decision-making problem. The ranking derived from the study has highlighted Scratch as the optimal learning App to promote critical thinking among novice programming students. This is congruent with various studies that compared visual programming environments under different contexts and found Scratch to be the most preferred learning tool (Erol and Kurt 2017; João *et al*. 2019; Ropii, Hardyanto and Ellianawati 2019). The final chapter which follows, will summarize the study, state its main limitations, discuss its contributions and present recommendations for further research.

# CHAPTER SIX

# SUMMARY, CONCLUSIONS AND RECOMMENDATIONS

## 6.1 Introduction

Chapter Six provides a summary of the study and elucidates how the research aim and objectives were achieved. In Section 6.4 on contribution and implications of the study, the importance of the study to policy, theory, practice and subsequent research are discussed. Further, how the findings will impact and benefit society is outlined. Some limitations of the study are also identified. The chapter concludes with recommendations for future research. A snap summary of the research study is found in Table 6.1. This chapter represents the culmination of this study.

## 6.2 Summary of Study

Chapter One described the context of the study. Furthermore, the aim, objectives and scope of the study were defined. The aim of the study was to apply an intelligent decision support system using the Fuzzy TOPSIS multi-criteria decision-making algorithm to assist higher education institutions explicate preferences for learning applications to promote critical thinking in first year programming students. Five research objectives were garnered to meet the aim of the study as presented in Table 6.1. The problem statement expatiated with evidence the need for this study. In Chapter Two, the importance of critical thinking was discussed to fill in the lacuna of our understanding on critical thinking as related to first year programming students. The measures for critical thinking were extracted from the extant literature.

The theoretical framework and research methodology were described in Chapter Three. Decision Theory and its application to the study was discussed in Section 3.2. Another framework used to underpin the critical thinking aspect of this dissertation was also presented, namely, the Diane Halpern's 4-Part Model on Critical Thinking. This study used a quantitative research methodology to meet the aims and objectives. The study was set at the Durban University of Technology and investigated first year students' preference for visual applications to learn programming concepts. A survey questionnaire was the primary research instrument. The data collected was analysed statistically using SPSS software. Furthermore, Fuzzy TOPSIS, a scientific method, was used to analyse students' preferences amongst four visual learning applications, namely, Scratch, Alice, MIT App Inventor and Blockly.

Chapter Four began with a literature review on the evaluation of selection techniques and motivation for Fuzzy TOPSIS as a precursor to mathematical definitions of the Fuzzy TOPSIS method. Each step was elaborated at great length using mathematical equations. In Chapter Five, Section A presented a statistical analysis of the survey questionnaire. Data analysis was completed using both descriptive and inferential statistics. In Section B, the results of the Fuzzy TOPSIS method as implemented on MatLab R2020a was presented. This represented the presentation of results using an intelligent decision support system. The results show that Scratch was the most preferred learning App, while Blockly is the least preferred learning App for first year programming students. Finally, Chapter Six will outline the findings and show how each research objective of the study was met.

Table 6.1 shows the alignment of the aims and objectives of this study and how each research objective is aligned to the Theoretical Framework, Data Collection Methods and Data Analysis.

| **Aim:** To propose a decision support system (DSS) that uses Fuzzy TOPSIS for multi-criteria decision-making to assist higher education institutions explicate preferences for learning applications to promote critical thinking in first year programming students. | | | |
|---|---|---|---|
| **Research Objectives** | **Theoretical Framework** | **Data Collection Methods/Data Sources** | **Data Analysis/Results** |
| [RO1]: synthesize the criteria that are used to promote critical thinking skills in students. | - | Literature Review<br>*(Chapter Two)* | Presented in Chapter Five-Table 5.32 |
| [RO2]: assess the impact of visual programming on critical thinking. | Diane Halpern's 4-Part Model<br>*(Chapter Three)* | Survey Questionnaire<br>*(Chapter Three)* | Survey Results<br>*(Chapter Five)* |
| [RO3]: select the most efficient Apps using Multi-Criteria Decision-Making for novice programming students. | Decision Theory<br>*(Chapter Three)* | Fuzzy TOPSIS Method<br>*(Chapter Four)* | MatLab R2020a Fuzzy TOPSIS App<br>*(Chapter Five)* |
| [RO4]: Investigate with live data an automated intelligent decision support prototype using Fuzzy TOPSIS. | Decision Theory<br>Diane Halpern's 4-Part Model<br>*(Chapter Three)* | Fuzzy TOPSIS Method | MatLab R2020a Fuzzy TOPSIS App; Entity Relationship Diagram (ERD)-blueprint<br>*(Chapter Six)* |

**Table 6.1 Alignment of the research**

## 6.3 Conclusions of Study

This section examines the extent to which each research objective was achieved while also highlighting the conclusions that were reached in each case as follows:

### 6.3.1 [RO1]: Synthesize the criteria that are used to promote critical thinking skills in students

An extensive review of the literature in Chapter 2 has uncovered various definitions of critical thinking in the realm of psychology, education and philosophy; however, the focus of the researcher was to unpack the one definition that best described critical thinking in the broad field of Computer Science and more specifically, in the discipline of programming. Diane Halpern's 4-part model for critical thinking discussed in Chapter 3 was used as a basis to elucidate the various measures for critical thinking that the literature concomitantly supported as skills promoted by learning Apps. The list of criteria that has been synthesised is shown in Table 5.32.

The researcher also reviewed research studies that purported to improve critical thinking skills through programming as these were used as a conduit to extricate the criteria that promoted critical thinking skills among students learning programming through a learning tool. Therefore, the extensive review of the literature was useful in identifying the measures for critical thinking synthesized in Section 2.3. The feature questionnaire (Annexure G) was used to assess criteria weightings. The calculations for the criteria weightings are shown in Table 5.33. These were integrated into the Fuzzy TOPSIS calculations to determine the most preferred leaning App.

### 6.3.2 [RO2]: Assess the impact of visual programming on critical thinking

This study used a survey method for data collection to obtain feedback from first year programming students on the impact of visual programming on critical thinking. The critical thinking criteria shown in Table 5.32 was used in the survey questionnaire (Annexure F). The statistical analysis of the data from the questionnaire has shown reliability scores in excess of 0.7, indicating a high degree of reliability and internal consistency. The percentage of respondents who have rated each learning App on the Likert scale of poor/very poor per criteria,

is illustrated in Table 6.2, with 100% of the response ratings falling below 12% on the category of poor/very poor.

| Criteria # | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Alice | 5 | 10 | 7 | 3 | 6 | 5 | 6 | 3 | 3 | 10 | 5 | 3 | 4 | 3 | 3 | 2 | 5 | 4 | 4 | 4 | 1 | 3 | 9 | 3 | 5 |
| Scratch | 5 | 10 | 7 | 2 | 6 | 3 | 6 | 2 | 1 | 10 | 2 | 2 | 5 | 5 | 3 | 3 | 3 | 3 | 3 | 6 | 2 | 2 | 10 | 3 | 3 |
| Blockly | 9 | 8 | 11 | 6 | 9 | 9 | 7 | 3 | 4 | 11 | 5 | 2 | 7 | 9 | 6 | 4 | 6 | 7 | 7 | 9 | 7 | 5 | 9 | 7 | 3 |
| MIT App Inventor | 7 | 7 | 9 | 5 | 3 | 5 | 6 | 3 | 3 | 7 | 4 | 3 | 4 | 6 | 4 | 4 | 4 | 5 | 6 | 8 | 4 | 6 | 9 | 6 | 5 |

**Table 6.2 Distribution of response percentages per criteria on poor/very poor rating**

A relatively low number of respondents rated each of the learning Apps on all 25 critical thinking criteria as poor/very poor. The highest percentage response on a poor/ very poor rating on all 25 criteria is 11%. Of all the response percentages within this rating category, 89% of the poor/very poor scorings were below 5%.

The percentage of respondents who have rated each learning App on the Likert scale of good/very good per criteria is illustrated in Table 6.3, with all response ratings exceeding 50% on the category of good/very good.

| Criteria # | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Alice | 53 | 55 | 57 | 63 | 54 | 65 | 58 | 74 | 67 | 58 | 62 | 73 | 58 | 70 | 63 | 71 | 69 | 67 | 70 | 68 | 71 | 73 | 54 | 70 | 67 |
| Scratch | 73 | 66 | 60 | 74 | 62 | 74 | 64 | 82 | 80 | 66 | 75 | 79 | 67 | 78 | 72 | 75 | 77 | 79 | 75 | 65 | 79 | 77 | 60 | 75 | 75 |
| Blockly | 51 | 50 | 48 | 47 | 49 | 55 | 46 | 66 | 65 | 54 | 59 | 69 | 56 | 62 | 53 | 63 | 63 | 61 | 61 | 53 | 62 | 64 | 51 | 61 | 65 |
| MIT App Inventor | 60 | 58 | 56 | 65 | 62 | 58 | 53 | 67 | 61 | 65 | 66 | 73 | 59 | 69 | 63 | 65 | 66 | 63 | 61 | 62 | 68 | 67 | 51 | 65 | 70 |

**Table 6.3 Distribution of response percentages per criteria on good/very good rating**

A relatively high number of respondents rated each of the learning Apps on all 25 critical thinking criteria as good/very good. The lowest percentage response on a good/very good rating on all 25 criteria for Scratch is 60% and the highest is 82%. The tables above reveal the good impact of visual programming on critical thinking by the 175 respondents on this study.

### 6.3.3 [RO3]: Select the most efficient Apps using Multi-Criteria Decision-Making for novice programming students

The literature review of decision theory in Chapter 3 revealed the multi-criteria decision-making model as the most appropriate decision model to use for the decision problem in the study. The literature also revealed a multitude of learning Apps discussed in Section 2.4 used across diverse learning platforms to serve a variety of needs. The literature was used to select the four popular learning Apps that also aligned well to the DUT context. Fuzzy TOPSIS was selected and defended as the technique to be employed from several MCDA methods within

the field of decision theory, as highlighted in Sections 4.2 and 4.3 to determine the optimal learning App that promotes critical thinking among novice programming students. Each mathematical step of the Fuzzy TOPSIS method was explained in Chapter 4 and carefully applied to the data analysis in Chapter 5.

### 6.3.4 [RO4]: Investigate with live data an automated intelligent decision support prototype using Fuzzy TOPSIS

The study used a population of 500 and a sample population of 217 decision-makers to rate four learning Apps against each of 25 critical thinking criteria. Applying the Fuzzy TOPSIS MCDM method to the decision problem would have made handling the matrices both inconvenient and cumbersome. The researcher created code using MatLab to automate the process. The implementation of the coding is illustrated in Section B of Chapter 5. The Fuzzy TOPSIS MCDM method was coded on MatLab R2020a from the pseudocode design. This resulted in the ranking of four learning Apps, from the App that promoted critical thinking the best to the one that promoted it least among 1$^{st}$ year programming learners; these Apps were Scratch, Alice, MIT App Inventor and Blockly.

## 6.4 Contributions and Implications of Study

The study is significant for lecturers teaching introductory programming modules to novice learners across educational institutions, especially during the global Covid-19 pandemic, when online learning has become commonplace and its teaching tools have become a requirement. Knowledge acquired in this study can potentially assist policy-makers and academic staff to make informed decisions about the types of learning Apps to consider which have been supported by research and guided by sound academic reasoning and principles.

The study highlights 25 critical thinking criteria, referenced in Table 5.32, that are effective to support the understanding of programming by novice learners. These may provide lecturers teaching programming with an opportunity to further reinforce their teaching plans and preparations for introductory programming modules. Some of these measures may be useful for the formulation of teaching, learning and assessment (TLA) policies and practices, especially when universities worldwide recognise the importance of critical thinking as a graduate attribute. The list of weighted critical thinking criteria shown in Table 5.35 represents the expert opinions of programming academics, which may provide further food for thought as

lecturers search for innovative ways to improve the performances of their novice programming learners.

The study draws attention to how learning Apps may be analytically selected in the presence of a theoretical framework. Ordinarily, lecturers are known to select Apps as tools for TLA based on their likes, dislikes and limited experience. This study introduces the lecturer to seriously consider the view of an important stakeholder such as the learner and recognise the importance of critical thinking for programming when selecting a learning App as a supplementary tool for teaching programming. It is posited that such decisions made within the framework of critical thinking introduced by this study will result in robust decision-making. Specifically, it may also benefit new computer science educators who wish to support their decisions for the best tool to teach programming to novice learners, while needing to choose from a different set of learning Apps. Alternatively, it can be used by other professionals and institutions with an interest in this field.

The knowledge gained from the implementation of the Fuzzy TOPSIS method in the selection of learning Apps can be used in MCDM contexts within the education and other domain disciplines. The study highlights the theory and rationale for its choice from among other MCDM methods, which is an important aspect to consider in decision problems. Multi-criteria decision problems are widespread in educational settings, hence this study provides a framework to implement a sound research-supported resolution.

Another contribution of this study is depicted in Figure 6.1, which displays the blueprint in the form of an Entity Relationship Diagram (ERD) for an Intelligent Decision Support System. The blueprint shows a conceptual ERD and foundational model for an Intelligent decision support system. The illustrated ERD represents a structural diagram of the new system and can be implemented in any programming language to support decision-making by senior management from educational institutions on the choice of learning applications for first year or novice programming students. The ERD is represented by 9 entities and their respective attributes, showing the relationships between them. Attributes are represented with primary keys and foreign keys. Figure 6.1 uses connecting lines with crow's foot notation to show how one entity relates to another. The relationships represented in the ERD show one-to-one, one-to-many and many-to-many cardinality.

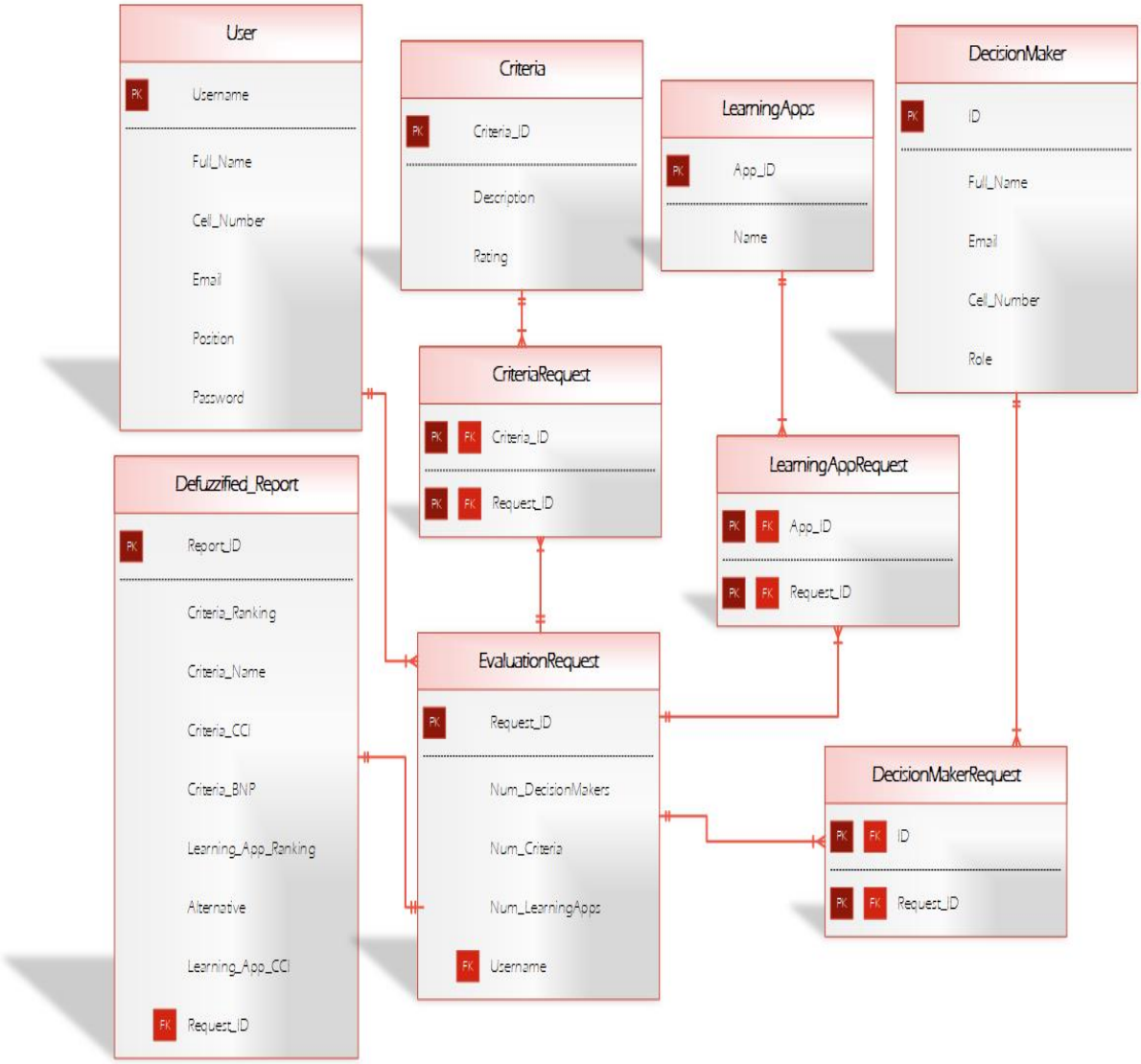


**Figure 6.1 Entity Relationship Diagram (ERD) for an Intelligent Decision Support System**

Source: Researcher's own construction (2021)

## 6.5 Limitations of Study

Although the study achieved its aim and objectives, the research has some limitations. The survey respondents were restricted to the first year programming students in the Department of Information Technology (IT) at the Durban University of Technology (UoT). Therefore, the outcome of the Fuzzy TOPSIS method depended on the opinions of 175 first year student respondents in the IT department. Generally, there also exists a high level of homogeneity among students at South African UoTs due primarily to the admission requirements having a strong reliance on academic performance. In addition, the socio-economic and demographic characteristics of the participants in this study may have had an impact on the participants' ability to engage with and interpret the impact of the learning Apps. Increasing the

representativeness of the sample to a larger, more diverse sample size that includes first year programming students in the Engineering Faculty of DUT and other Universities of Technology in the region, may be more desirable to detect other significant effects.

The opinions derived from the academic experts within the IT department to determine the weightings of the 25 critical thinking criteria may have been restrictive. Sourcing the expertise of programming educators from other UoTs and traditional universities across the country may increase the generalisability of the resultant weights used in the Fuzzy TOPSIS application.

## 6.6 Future Research

The development of this research provided a framework for key decision-making around the choice of a learning App that best promotes critical thinking among first year programming students. Although the research suggests the positive impact of the visual programming environment on one's critical thinking skills, further research is required to investigate its impact in a text-based programming environment. As future work, it will be pertinent to examine how this framework will be applied in similar decision-making contexts. Furthermore, future research could focus on participants from more diverse backgrounds enrolled for introductory programming modules across a range of institutions and the framework of this study could be replicated to confirm these effects.

# REFERENCES


Abrami, P. C., Bernard, R. M., Borokhovski, E., Waddington, D. I., Wade, C. A. and Persson, T. 2015. Strategies for teaching students to think critically: a meta-analysis. *Review of Educational Research*, 85 (2): 275-314.

Adwok, J. 2014. Application of Brim's and Simon's sequential decision theories in healthcare administration. *Journal of Biology, Agriculture and Healthcare*, 4 (14): 23-31.

Agbo, F., Oyelere, S., Suhonen, J. and Adewumi, S. 2019. A systematic review of computational thinking approach for programming education in higher education institutions. In: Proceedings *of the 19th Koli Calling International Conference on Computing Education Research*. 2019. 1-10.

Ahmad, R., Sarlan, A., Hashim, A. S. and Hassan, M. F. 2017. Relationship between hands-on and written coursework assessments with critical thinking skills in structured programming course. In: Proceedings of *7th World Engineering Education Forum (WEEF)*. 2017. 231-235.

Ajoodha, R., Jadhav, A. and Dukhan, S. 2020. Forecasting learner attrition for student success at a South African university. In: Proceedings of *Conference of the South African Institute of Computer Scientists and Information Technologists 2020*. 19-28.

Al-Mubaid, H., Abukmail, A. and Bettayeb, S. 2016. Empowering deep thinking to support critical thinking in teaching and learning. In: Proceedings of *the 2016 ACM SIGMIS Conference on Computers and People Research*. 69-75.

Almerich, G., Suárez-Rodríguez, J., Díaz-García, I. and Cebrián-Cifuentes, S. 2019. 21st-century competences: The relation of ICT competences with higher-order thinking capacities and teamwork competences in university students. *Journal of Computer Assisted Learning*, 36 (4): 468-479.

Anyango, J. T. and Suleman, H. 2018. Teaching Programming in Kenya and South Africa: What is difficult and is it universal? In: Proceedings *of the 18th Koli Calling International Conference on Computing Education Research*. 1-2.

Apuke, O. D. 2017. Quantitative research methods: A synopsis approach. *Kuwait Chapter of Arabian Journal of Business and Management Review*, 33 (5471): 1-8.

Arawjo, I., Wang, C.-Y., Myers, A. C., Andersen, E. and Guimbretière, F. 2017. Teaching programming with gamified semantics. In: Proceedings of *the 2017 CHI Conference on Human Factors in Computing Systems*. 4911-4923.

Aristodemou, L. and Tietze, F. 2019. Technology Strategic Decision Making (SDM): An overview of decision theories, processes and methods. Available: https://www.repository.cam.ac.uk/bitstream/handle/1810/288412/19_05_Aristodemou_Tietze.pdf?sequence=3&isAllowed=y (Accessed 15 February 2021)

Atmatzidou, S., Atmatzidou, S., Demetriadis, S., Demetriadis, S., Nika, P. and Nika, P. 2018. How does the degree of guidance support students' metacognitive and problem solving skills in educational robotics? *Journal of Science Education and Technology*, 27 (1): 70-85.

Bala, R. B. and Alacapinar, F. G. 2021. Scratch in teaching programming: effect on problem solving skill and attitude. *International Journal of Quality in Education*, 5 (2): 63-81.

Balioti, V., Tzimopoulos, C. and Evangelides, C. 2018. Multi-criteria decision making using TOPSIS method under fuzzy environment. Application in spillway selection. In: Proceedings of *Multidisciplinary Digital Publishing Institute Proceedings*. 637.

Ball, M., Mönig, J., Romagosa, B. and Harvey, B. 2019. Snap! A Look at 5 Years, 250,000 Users and 2 Million Projects. In: Proceedings of the 50th *ACM Technical Symposium on Computer Science Education*. 1279-1279.

Başaran, S. and Haruna, Y. 2017. Integrating FAHP and TOPSIS to evaluate mobile learning applications for mathematics. *Procedia Computer Science*, 120: 91-98.

Bau, D., Gray, J., Kelleher, C., Sheldon, J. and Turbak, F. 2017. Learnable programming: blocks and beyond. *Communications of the ACM*, 60 (6): 72-80.

Berland, M., Martin, T., Benton, T., Petrick Smith, C. and Davis, D. 2013. Using learning analytics to understand the learning pathways of novice programmers. *Journal of the Learning Sciences*, 22 (4): 564-599.

Bernstein, A. G. and Isaac, C. 2018. Critical thinking criteria for evaluating online discussion. *International Journal for the Scholarship of Teaching and Learning*, 12 (2): 1-8.

Binkley, M., Erstad, O., Herman, J., Raizen, S., Ripley, M., Miller-Ricci, M. and Rumble, M. 2012. Defining twenty-first century skills. *Assessment and Teaching of 21st Century Skills*. Springer, 17-66.

Bradley, R. 2014. Decision theory: A formal philosophical introduction. In: Hansson, S.O. and Hendricks, V.F. eds. *Introduction to formal philosophy*. Springer, 611-655.

Buitrago Flórez, F., Casallas, R., Hernández, M., Reyes, A., Restrepo, S. and Danies, G. 2017. Changing a generation's way of thinking: teaching computational thinking through programming. *Review of Educational Research*, 87 (4): 834-860.

Butler, H. A. 2012. Halpern Critical Thinking Assessment predicts real-world outcomes of critical thinking. *Applied Cognitive Psychology*, 26 (5): 721-729.

Chapman, D. D. and Joines, J. A. 2017. Strategies for increasing response rates for online end-of-course evaluations. *International Journal of Teaching and Learning in Higher Education*, 29 (1): 47-60.

Chen, C.-T. 2000. Extensions of the TOPSIS for group decision making under fuzzy environment. *Fuzzy Sets and Systems*, 114 (1): 1-9.

Chen, T., Hsu, H.-M., Stamm, S. W. and Yeh, R. 2019. Creating an instrument for evaluating critical thinking apps for college students. *E-Learning and Digital Media*, 16 (6): 433-454.

Chis, A. E., Moldovan, A.-N., Murphy, L., Pathak, P. and Muntean, C. H. 2018. Investigating flipped classroom and problem-based learning in a programming module for computing conversion course. *Educational Technology & Society*, 21 (4): 232-247.

Creswell, J. W. 2012. *Educational research: Planning, conducting, and evaluating quantitative and qualitative research*. 4th ed. Boston: Pearson.

Creswell, J. W. and Creswell, J. D. 2017. *Research design: Qualitative, quantitative, and mixed methods approaches*. 3rd ed. California: Sage publications.

Davenport, C. E. 2018. Evolution in student perceptions of a flipped classroom in a computer programming course. *Journal of College Science Teaching*, 47 (4): 30-35.

Davies, M. 2015. A model of critical thinking in higher education. In: Paulsen M.B. ed. *Higher Education: Handbook of theory and research*. Springer, 41-92.

Dekhan, S., Xu, X. and Tsoi, M. Y. 2013. Mobile app development to increase student engagement and problem solving skills. *Journal of Information Systems Education*, 24 (4): 299-308.

Delice, A. 2010. The sampling issues in quantitative research. *Educational Sciences: Theory and Practice*, 10 (4): 2001-2018.

Dewi, R. K., Wardani, S., Wijayati, N. and Sumarni, W. 2019. Demand of ICT-based chemistry learning media in the disruptive era. *International Journal of Evaluation and Research in Education*, 8 (2): 265-270.

Durak, H. Y. 2020. The effects of using different tools in programming teaching of secondary school students on engagement, computational thinking and reflective thinking skills for problem solving. *Technology, Knowledge and Learning*, 25 (1): 179-195.

Duran, M. and Sendag, S. 2012. A preliminary investigation into critical thinking skills of urban high school students: Role of an IT/STEM program. *Creative Education*, 3 (02): 241.

Ece, O. and Uludag, A. S. 2017. Applicability of fuzzy TOPSIS method in optimal portfolio selection and an application in BIST. *International Journal of Economics and Finance*, 9 (10): 107-127.

Ellis, J. B., Deutsch, J. C. and Legret, M. 2006. The DayWater decision support approach to the selection of sustainable drainage systems: A multi-criteria methodology for BMP decision makers. *Water Practice and Technology*, 1 (1): wpt2006002.

Elshiekh, R. and Butgerit, L. 2017. Using gamification to teach students programming concepts. *Open Access Library Journal*, 4 (8): 1-7.

Ennis, R. H. 1985. A logical basis for measuring critical thinking skills. *Educational Leadership*, 43 (2): 44-48.

Erol, O. and Kurt, A. A. 2017. The effects of teaching programming with scratch on pre-service information technology teachers' motivation and achievement. *Computers in Human Behavior*, 77: 11-18.

Facione, P. A. 2011. Critical thinking: What it is and why it counts. *Insight Assessment*, 2007 (1): 1-23. Available: https://www.measuredreasons.com/index_htm_files/what&why2015.pdf (Accessed 3 October 2020)

Figueiredo, J. and García-Peñalvo, F. J. 2019. Teaching and learning strategies of programming for university courses. In: Proceedings *of the Seventh International Conference on Technological Ecosystems for Enhancing Multiculturality*. 1020-1027.

Fincher, S., Cooper, S., Kölling, M. and Maloney, J. 2010. Comparing Alice, Greenfoot & Scratch. In: Proceedings *of the 41st ACM Technical Symposium on Computer Science Education*. 192-193.

González-González, I. and Jiménez-Zarco, A. I. 2015. Using learning methodologies and resources in the development of critical thinking competency: An exploratory study in a virtual learning environment. *Computers in Human Behavior*, 51: 1359-1366.

Greyling, J., Cilliers, C. and Calitz, A. 2006. B#: The development and assessment of an iconic programming tool for novice programmers. In: Proceedings *of 2006 7th International Conference on Information Technology Based Higher Education and Training*. IEEE, 367-375.

Grover, S. and Pea, R. 2013. Computational Thinking in K-12: A review of the state of the field. *Educational Researcher*, 42 (1): 38-43.

Halpern, D. F. 1998. Teaching critical thinking for transfer across domains: Disposition, skills, structure training, and metacognitive monitoring. *American Psychologist*, 53 (4): 449.

Halpern, D. F. 1999. Teaching for critical thinking: Helping college students develop the skills and dispositions of a critical thinker. *New Directions for Teaching and Learning*, 1999 (80): 69-74.

Halpern, D. F. 2013. Thought and knowledge: An introduction to critical thinking. *Water Practice and Technology*, 1 (1).

Han, H. and Trimi, S. 2018. A fuzzy TOPSIS method for performance evaluation of reverse logistics in social commerce platforms. *Expert Systems with Applications*, 103: 133-145.

Hansson, S. O. 2011. Decision theory: An overview. In: Lovric, M. ed. *International Encyclopedia of Statistical Science*. Berlin, Heidelberg: Springer Berlin Heidelberg, 349-355. Available: https://doi.org/10.1007/978-3-642-04898-2_22 (Accessed 21 February 2021)

Harrison, T. R. and Lee, H. S. 2018. iPads in the mathematics classroom: Developing criteria for selecting appropriate learning apps. *International Journal of Education in Mathematics, Science and Technology*, 6 (2): 155-172.

Hellmann, M. 2001. Fuzzy logic introduction, *Universite de Rennes*, 18 (1): 319-328.

Hinton, P. R., McMurray, I. and Brownlow, C. 2014. *SPSS explained*. 2nd ed. London: Routledge.

Howe, D. C. 2009. Creativity support for computational literature. PhD, New York University. Available: https://cs.nyu.edu/media/publications/howe_daniel.pdf (Accessed 10 January 2021)

Hutchison, A., Nadolny, L. and Estapa, A. 2016. Using coding apps to support literacy instruction and develop coding literacy. *The Reading Teacher*, 69 (5): 493-503.

Hyytinen, H. and Toom, A. 2019. Developing a performance assessment task in the Finnish higher education context: Conceptual and empirical insights. *British Journal of Educational Psychology*, 89 (3): 551-563.

Irwanto, I., Rohaeti, E. and Prodjosantoso, A. 2018. A survey analysis of pre-service chemistry teachers'critical thinking skills. *MIER Journal of Educational Studies, Trends and Practices,* 8 (1): 57-73.

Ivanović, M., Ivanović, M., Xinogalos, S., Xinogalos, S., Pitner, T., Pitner, T., Savić, M. and Savić, M. 2017. Technology enhanced learning in programming courses – international perspective. *Education and Information Technologies,* 22 (6): 2981-3003.

Jahnke, I. and Liebscher, J. 2020. Three types of integrated course designs for using mobile technologies to support creativity in higher education. *Computers & Education,* 146: 103782.

João, Nuno, Fábio and Ana. 2019. A cross-analysis of block-based and visual programming apps with computer science student-teachers. *Education Sciences,* 9 (3): 181.

Johanson, G. A. and Brooks, G. P. 2010. Initial scale development: Sample size for pilot studies. *Educational and Psychological Measurement,* 70 (3): 394-400.

Junior, F. R. L., Osiro, L. and Carpinetti, L. C. R. 2014. A comparison between fuzzy AHP and fuzzy TOPSIS methods to supplier selection. *Applied Soft Computing,* 21: 194-209.

Kahraman, C., Ateş, N. Y., Çevik, S., Gülbay, M. and Erdoğan, S. A. 2007. Hierarchical fuzzy TOPSIS model for selection among logistics information technologies. *Journal of Enterprise Information Management,* 20 (2): 143-168.

Kalelioglu, F. and Gülbahar, Y. 2014. The effects of teaching programming via scratch on problem solving skills: A discussion from learners' perspective. *Informatics in Education,* 13 (1): 33-50.

Kaya, K. Y. and Yildiz, İ. 2019. Comparing three free to use visual programming environments for novice programmers. *Kastamonu Eğitim Dergisi,* 27 (6): 2701-2712.

Kelleher, C. and Pausch, R. 2005. Lowering the barriers to programming: A taxonomy of programming environments and languages for novice programmers. *ACM Computing Surveys (CSUR),* 37 (2): 83-137.

Khaldi, K. 2017. Quantitative, qualitative or mixed research: Which research paradigm to use? *Journal of Educational and Social Research,* 7 (2): 15.

Knoesel, J. 2017. Effect of implementation of simulation on critical thinking skills in undergraduate baccalaureate nursing students. PhD, City of New York University. Available: https://academicworks.cuny.edu/cgi/viewcontent.cgi?article=3065&context=gc_etds (Accessed 14 December 2020)

Kong, S. C. and Wang, Y. Q. 2020. Formation of computational identity through computational thinking perspectives development in programming learning: A mediation analysis among primary school students. *Computers in Human Behavior,* 106: 106230.

Kules, B. 2016. Computational thinking is critical thinking: Connecting to university discourse, goals, and learning outcomes. In: Proceedings *of the Association for Information Science and Technology,* 53 (1): 1-6.

Kumar, R. 2018. *Research methodology: A step-by-step guide for beginners*. 3rd ed. Thousand Oaks: Sage.

Kurilovas, E. 2014. Research on tablets application for mobile learning activities. *Journal of Mobile Multimedia*, 10 (3): 182-193.

Larkin, K. 2015. “An app! An app! My kingdom for an app”: An 18-month quest to determine whether apps support mathematical knowledge building. In: *Digital Games and Mathematics Learning*. Springer, 251-276.

Li, Q. 2013. A novel likert scale based on fuzzy sets theory. *Expert Systems with Applications*, 40 (5): 1609-1618.

Liu, J., Wimmer, H. and Rada, R. 2016. " Hour of code”: Can it change students’ attitudes toward programming? *Journal of Information Technology Education: Innovations in Practice*, 15: 53.

Liu, O. L., Mao, L., Frankel, L. and Xu, J. 2016. Assessing critical thinking in higher education: the HEIghten™ approach and preliminary validity evidence. *Assessment and Evaluation in Higher Education*, 41 (5): 677-694.

Martín-Gutiérrez, J., Fabiani, P., Benesova, W., Meneses, M. D. and Mora, C. E. 2015. Augmented reality to promote collaborative and autonomous learning in higher education. *Computers in Human Behavior*, 51: 752-761.

Mathew, R., Malik, S. I. and Tawafak, R. M. 2019. Teaching problem solving skills using an educational game in a computer programming course. *Informatics in Education*, 18 (2): 359-373.

McIver, L. 2002. Evaluating languages and environments for novice programmers. In: Proceedings *of* the 14th Annual Workshop of the *Psychology of Programming Interest Group*. 100-110.

McMahon, G. 2009. Critical thinking and ict integration in a western australian secondary school. *Educational Technology & Society*, 12 (4): 269-281.

Medeiros, R. P., Ramalho, G. L. and Falcão, T. P. 2018. A systematic literature review on teaching and learning introductory programming in higher education. *IEEE Transactions on Education*, 62 (2): 77-90.

Memari, A., Dargi, A., Jokar, M. R. A., Ahmad, R. and Rahim, A. R. A. 2019. Sustainable supplier selection: A multi-criteria intuitionistic fuzzy TOPSIS method. *Journal of Manufacturing Systems*, 50: 9-24.

Mintzberg, H., Raisinghani, D. and Theoret, A. 1976. The structure of" unstructured" decision processes. *Administrative Science Quarterly*: 246-275.

Mohammed, H. J., Kasim, M. M. and Shaharanee, I. N. 2018. Evaluation of e-Learning approaches using AHP-TOPSIS technique. *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, 10 (1-10): 7-10.

Muijs, D. 2010. *Doing quantitative research in education with SPSS*. 2nd ed. London: Sage.

Ng'ambi, D. and Johnston, K. 2006. An ICT-mediated constructivist approach for increasing academic support and teaching critical thinking skills. *Educational Technology & Society*, 9 (3): 244-253.

Nguyen, L., Barton, S. M. and Nguyen, L. T. 2015. iPads in higher education—Hype and hope. *British Journal of Educational Technology*, 46 (1): 190-203.

Noone, M. and Mooney, A. 2018. Visual and textual programming languages: a systematic review of the literature. *Journal of Computers in Education*, 5 (2): 149-174.

Nulty, D. D. 2008. The adequacy of response rates to online and paper surveys: what can be done? *Assessment & Evaluation in Higher Education*, 33 (3): 301-314.

Nursal, A. T., Omar, M. F. and Nawi, M. N. M. 2018. The application of Fuzzy TOPSIS to the selection of building information modeling software. *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, 10 (1-10): 1-5.

Palczewski, K. and Sałabun, W. 2019. The fuzzy TOPSIS applications in the last decade. *Procedia Computer Science*, 159: 2294-2303.

Panhwar, A. H., Ansari, S. and Shah, A. A. 2017. Post-positivism: An effective paradigm for social and educational research. *International Research Journal of Arts & Humanities (IRJAH)*, 45 (45).

Papadakis, S. and Kalogiannakis, M. 2017. Using gamification for supporting an introductory programming course: The case of classcraft in a secondary education classroom. In: *Interactivity, game creation, design, learning, and innovation*. Springer, 366-375.

Papadakis, S., Kalogiannakis, M., Orfanakis, V. and Zaranis, N. 2014. Novice programming environments: Scratch & app inventor: a first comparison. In: Proceedings *of the 2014 Workshop on Interaction Design in Educational Environments.* ACM, 1-7.

Papadakis, S. and Orfanakis, V. 2016. The combined use of Lego Mindstorms NXT and App Inventor for teaching novice programmers. In: Proceedings *of International Conference EduRobotics*. 193-204.

Papadakis, S. and Orfanakis, V. 2018. Comparing novice programing environments for use in secondary education: App Inventor for Android vs. Alice. *International Journal of Technology Enhanced Learning*, 10 (1-2): 44-72.

Papert, S. 1980. *Mindstorms: Children, computers & powerful ideas.* New York: Basic Books.

Pasternak, E., Fenichel, R. and Marshall, A. N. 2017. Tips for creating a block language with blockly. In: Proceedings *of 2017 IEEE Blocks and Beyond Workshop (B&B).* IEEE, 21-24.

Pierce, T. S. 2011. Introductory computer programming courses used as a catalyst to critical thinking development. PhD, Ball State University. https://bsu.summon.serialssolutions.com/#!/search?ho=t&l=en&q=1660953-01bsu_inst (Accessed 20 December 2020)

Pinto-Llorente, A. M., Casillas-Martín, S., Cabezas-González, M. and García-Peñalvo, F. J. 2018. Building, coding and programming 3D models via a visual programming environment. *Quality & Quantity*, 52 (6): 2455-2468.

Quinn, S., Hogan, M., Dwyer, C., Finn, P. and Fogarty, E. 2020. Development and validation of the student-educator negotiated critical thinking dispositions scale (SENCTDS). *Thinking Skills and Creativity*, 38: 100710.

Raddad, F. 2019. Teaching young people computational thinking using MIT App Inventor. M.Sc. Kansas State University. Available: https://krex.kstate.edu/dspace/bitstream/handle/2097/39807/RaddadFaqihi2019.pdf?sequence=7 (Accessed 16 February 2021)

Rajak, M. and Shaw, K. 2019. Evaluation and selection of mobile health (mHealth) applications using AHP and fuzzy TOPSIS. *Technology in Society*, 59: 101186.

Ranganath, N., Sarkar, D., Patel, P. and Patel, S. 2020. Application of fuzzy TOPSIS method for risk evaluation in development and implementation of solar park in India. *International Journal of Construction Management*: 1-11.

Romero, M., Lepage, A. and Lille, B. 2017. Computational thinking development through creative programming in higher education. *International Journal of Educational Technology in Higher Education*, 14 (1): 1-15.

Ropii, N., Hardyanto, W. and Ellianawati, E. 2019. Guided inquiry scratch increase students' critical thinking skills on the linear motion concept: Can it be? *JPPPF: Jurnal Penelitian & Pengembangan Pendidikan Fisika*, 5 (1): 63-68.

Rose, S., Habgood, J. and Jay, T. 2018. Pirate Plunder: Game-based computational thinking using scratch blocks. In: Proceedings *of the 12th European Conference on Games Based Learning.* 556-564.

Rouyendegh, B. D., Yildizbasi, A. and Üstünyer, P. 2020. Intuitionistic fuzzy TOPSIS method for green supplier selection problem. *Soft Computing*, 24 (3): 2215-2228.

Rose, S., Habgood, J. and Jay, T. 2018. Pirate Plunder: Game-Based Computational Thinking Using Scratch Blocks. In: Proceedings of *Academic Conferences and Publishing International Limited*, 556-XXIII.

Roy, K., Rousse, W. C. and DeMeritt, D. B. 2012. Comparing the mobile novice programming environments: App Inventor for Android vs. GameSalad. In: Proceedings of *2012 Frontiers in Education Conference Proceedings*. IEEE, 1-6.

Ryan, A. B. 2006. Post-positivist approaches to research. In: *Researching and writing your thesis: A guide for postgraduate students*: 12-26. Available: https://mural.maynoothuniversity.ie/874/ (Accessed 15 March 2021)

Safari, H., Faghih, A. and Fathi, M. R. 2012. Fuzzy multi-criteria decision making method for facility location selection. *African Journal of Business Management*, 6 (1): 206-212.

Salih, M. M., Zaidan, B., Zaidan, A. and Ahmed, M. A. 2019. Survey on fuzzy TOPSIS state-of-the-art between 2007 and 2017. *Computers & Operations Research*, 104: 207-227.

Sandoval-Reyes, S., Galicia-Galicia, P. and Gutierrez-Sanchez, I. 2011. Visual learning environments for computer programming. In: Proceedings *of 2011 IEEE Electronics, Robotics and Automotive Mechanics Conference*. IEEE, 439-444.

Sekaran, U. 2000. *Research methods for business: A skill-building approach*. New York: John Wiley & Sons.

Sekaran, U. and Bougie, R. 2016. *Research methods for business: A skill building approach*. 7th ed. New York: John Wiley & Sons.

Sevkli, M., Zaim, S., Turkyilmaz, A. and Satir, M. 2010. An application of fuzzy Topsis method for supplier selection. In: Proceedings *of International Conference on Fuzzy Systems*. IEEE, 1-7.

Shively, K., Stith, K. M. and Rubenstein, L. D. 2018. Measuring what matters: Assessing creativity, critical thinking, and the design process. *Gifted Child Today*, 41 (3): 149-158.

Šiaulys, T. 2020. *Modeling the system for interactive tasks development: engagement taxonomy for introductory programming tools.* Switzerland: Springer.

Simon, H. A. 1960. *The new science of management decision.* Washington: Harper & Brothers.

Smetanová, V., Drbalová, A. and Vitáková, D. 2015. Implicit Theories of Critical Thinking in Teachers and Future Teachers. *Procedia,Social and Behavioral Sciences*, 171: 724-732.

Sodhi, B. and T V, P. 2012. A simplified description of Fuzzy TOPSIS. Available: http://arxiv.org/abs/1205.5098v1 (Accessed 18 January 2021).

Spicer, K.-L. and Hanks, W. E. 1995. *Multiple measures of critical thinking skills and predisposition in assessment of critical thinking*. Available: https://eric.ed.gov/?id=ED391185 (Accessed 14 Nov 2020)

Standish, T., Joines, J. A., Young, K. R. and Gallagher, V. J. 2018. Improving SET response rates: Synchronous online administration as a tool to improve evaluation quality. *Research in Higher Education*, 59 (6): 812-823.

Straub, D., Boudreau, M.-C. and Gefen, D. 2004. Validation guidelines for IS positivist research. *Communications of the Association for Information Systems*, 13 (1): 24.

Su, H. F. H., Ricci, F. A. and Mnatsakanian, M. 2016. Mathematical teaching strategies: Pathways to critical thinking and metacognition. *International Journal of Research in Education and Science*, 2 (1): 190-200.

Swart, R. 2017. Critical thinking instruction and technology enhanced learning from the student perspective: A mixed methods research study. *Nurse Education in Practice*, 23: 30-39.

Taherdoost, H. 2016a. Sampling methods in research methodology: How to choose a sampling technique for research. *International Journal of Academic Research in Management*, 5 (2): 18-27.

Taherdoost, H. 2016b. Validity and reliability of the research instrument: How to test the validation of a questionnaire/survey in a research. *International Journal of Academic Research in Management*, 5 (3): 28-36.

Tee, K. N., Leong, K. E. and Abdul Rahim, S. S. 2018. The mediating effects of critical thinking skills on motivation factors for mathematical reasoning ability. *The Asia-Pacific Education Researcher,* 27 (5): 373-382.

Thabane, L., Ma, J., Chu, R., Cheng, J., Ismaila, A., Rios, L. P., Robson, R., Thabane, M., Giangregorio, L. and Goldsmith, C. H. 2010. A tutorial on pilot studies: The what, why and how. *BMC Medical Research Methodology,* 10 (1): 1.

Tsai, C.-Y. 2019. Improving students' understanding of basic programming concepts through visual programming language: The role of self-efficacy. *Computers in Human Behavior,* 95: 224-232.

Tuloli, M., Latief, M. and Rohandi, M. 2019. Scratching our own itch: Software to teach software programming. In: Proceedings *of International Conference on Education, Science and Technology*. Redwhite Press, 115-121.

Turker, P. M. and Pala, F. K. 2019. A study on students' computational thinking skills and self-efficacy of block-based programming. *i-Manager's Journal on School Educational Technology,* 15 (3): 18.

Van den Berg, L. 2020. 4th Industrial revolution? Ready, Tech, Go: Reflecting on Sport Students' Digital Literacy Skills when Creating Vlogs for Assessment. In: Proceedings *of EdMedia+ Innovate Learning*. 272-291.

Wang, X. S. 2017. Enhancing students' computer programming performances, critical thinking awareness and attitudes towards programming: An online peer-assessment attempt. *International Forum of Educational Technology & Society,* 20 (4): 58-68.

Weintrop, D. and Wilensky, U. 2015. To block or not to block, that is the question: Students' perceptions of blocks-based programming. In: Proceedings *of the 14th International Conference on Interaction Design and Children*. 199-208.

Weintrop, D. and Wilensky, U. 2017. Comparing block-based and text-based programming in high school computer science classrooms. *ACM Transactions on Computing Education (TOCE),* 18 (1): 1-25.

Wong, G. K.-W. and Cheung, H.-Y. 2020. Exploring children's perceptions of developing twenty-first century skills through computational thinking and programming. *Interactive Learning Environments,* 28 (4): 438-450.

Xinogalos, S., Satratzemi, M. and Malliarakis, C. 2015. Microworlds, games, animations, mobile apps, puzzle editors and more: What is important for an introductory programming environment? *Education and Information Technologies,* 22 (1): 145-176.

Xu, Z., Ritzhaupt, A. D., Tian, F. and Umapathy, K. 2019. Block-based versus text-based programming environments on novice student learning outcomes: A meta-analysis study. *Computer Science Education,* 29 (2-3): 177-204.

Yavuz, M. 2016. Equipment selection by using fuzzy TOPSIS method. In: Proceedings *of IOP Conference Series: Earth and Environmental Science,* 44 (4).

Zadeh, L. A. 1975a. The concept of a linguistic variable and its application to approximate reasoning-III. *Information Sciences,* 9 (1): 43-80.

Zadeh, L. A. 1975b. Fuzzy logic and approximate reasoning. *Synthese*, 30 (3): 407-428.

Zanakis, S. H., Solomon, A., Wishart, N. and Dublish, S. 1998. Multi-attribute decision making: A simulation comparison of select methods. *European Journal of Operational Research*, 107 (3): 507-529.

Zimmermann, H. J. 2010. Fuzzy set theory. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2 (3): 317-332.

Zubaidah, S., Corebima, A. D. and Mahanal, S. 2018. Revealing the relationship between reading interest and critical thinking skills through Remap GI and Remap Jigsaw. *International Journal of Instruction*, 11 (2): 41-56.

## ANNEXURE A: The Research Work Plan

| | Aug 2020 | Oct 2020 | Dec 2020 | Jan 2021 | Feb 2021 | April 2021 | May 2021 | June 2021 | July 2021 | Aug 2021 |
|---|---|---|---|---|---|---|---|---|---|---|
| **Preliminary Literature Review** | | | | | | | | | | |
| **Proposal and Ethical Clearance** | | | | | | | | | | |
| **Literature Review** | | | | | | | | | | |
| **Data Collection** | | | | | | | | | | |
| **Data analysis and synthesis** | | | | | | | | | | |
| **Writing thesis** | | | | | | | | | | |
| **Proof reading by 2 academics** | | | | | | | | | | |
| **Final editing by supervisor** | | | | | | | | | | |
| **Intent to submit** | | | | | | | | | | |
| **Submit thesis for examination** | | | | | | | | | | |
| **Research output: Journal Paper** | | | | | | | | | | |

# ANNEXURE B: Letter of Consent


**CONSENT**

**Full Title of the Study:** Intelligent Decision Support System for Selection of Learning Apps to Promote Critical Thinking in First Year Programming Students.

**Names of Researcher/s:** Kesarie Singh

**Statement of Agreement to Participate in the Research Study:**

- I hereby confirm that I have been informed by the researcher, Kesarie Singh about the nature, conduct, benefits and risks of this study.
- I have also received, read and understood the written information (Participant Letter of Information) regarding the study.
- I am aware that the results of the study, including personal details regarding my sex, age, date of birth, initials and diagnosis will be anonymously processed into a study report.
- In view of the requirements of research, I agree that the data collected during this study can be processed in a computerized system by the researcher.
- I may, at any stage, without prejudice, withdraw my consent and participation in the study.
- I have had sufficient opportunity to ask questions and (of my own free will) declare myself prepared to participate in the study.
- I understand that significant new findings developed during the course of this research which may relate to my participation will be made available to me.

| Full Name of Participant | Date | Time | Signature/ Thumbprint |
|---|---|---|---|

I, Kesarie Singh herewith confirm that the above participant has been fully informed about the nature, conduct and risks of the above study.

| Full Name of Researcher | Date | Signature |
|---|---|---|

| Full Name of Witness (if applicable) | Date | Signature |
|---|---|---|

| Full Name of Legal Guardian (if applicable) | Date | Signature |
|---|---|---|

# ANNEXURE C: Ethical Training Certificate

TRREE

Zertifikat
Certificat
Certificado
Certificate

Promouvoir les plus hauts standards éthiques dans la protection des participants à la recherche biomédicale
Promoting the highest ethical standards in the protection of biomedical research participants

Clinical Trials Centre
The University of Hong Kong

**Certificat de formation - Training Certificate**

Ce document atteste que - this document certifies that

Kesarie Singh

a complété avec succès - has successfully completed

Introduction to Research Ethics

du programme de formation TRREE en évaluation éthique de la recherche
of the TRREE training programme in research ethics evaluation

Release Date: 2021/03/01
CID : nJ6YIO6OYK

Professeur Dominique Sprumont
Coordinateur TRREE Coordinator

FMH Continuing Education Program (5 Credits)
Programme de Formation Continue (5 Crédits)

Foederatio Pharmaceutica Helvetiae FPH Programmes de formation continue
Continuing Education Programs
Programmes de formation continue

Ce programme est soutenu par - This program is supported by :
European and Developing Countries Clinical Trials Partnership (EDCTP) (www.edctp.org) - Swiss National Science Foundation (www.snf.ch) - Canadian Institutes of Health Research (http://www.cihr-irsc.gc.ca/e/2891.html) - Swiss Academy of Medical Science (SAMS/ASSM/SAMW) (www.samw.ch) - Commission for Research Partnerships with Developing Countries (www.kfpe.ch)

[REV : 20170310]

# ANNEXURE D: Ethical Clearance Approval


Faculty Research Office
Durban University of
Technology Date March 14,2021

Student Kesarie Singh
Student Number: 20928764
Degree: Master of ICT
Email: 20928764@dut4life.ac.za
Supervisor: Dr N K Naicker
Supervisor email: naickern@dut.ac.za

**Dear Mrs Singh**

ETHICAL APPROVAL: LEVEL 2

I am pleased to inform you that the Faculty Research Ethics Committee (FREC) following feedback from two reviewers, has granted preliminary permission for you to conduct your research "Intelligent Decision Support System for Selection of Learning Apps to Promote Critical Thinking in First Year Programming Students."

**When ethics approval is granted:**
You are required to present the letter at your research site(s) for permission to gather data. Please also note that your research instruments must be accompanied by the letter of information and the letter of consent for each participant, as per your research proposal.

This ethics clearance is valid from the date of provisional approval on this letter for one year. A student must apply for recertification 3 months before the date of this expiry.

Recertification is required every year until after corrections are made, after examination, and the thesis is submitted to the Faculty Registrar.

A summary of your key research findings must be submitted to the FRC on completion of your studies.

Kindest regards.

Yours sincerely

Dr Mdgiveny Rajkoomar
FREC Chair
Faculty of Accounting and Informatics
Durban University of Technology
Ritson Campus

Durban, South Africa
4001

# ANNEXURE E: Permission to Conduct Research


*Directorate for Research and Postgraduate Support*
*Durban University of Technology*
*Tromso Annexe, Steve Biko Campus*
*P.O. Box 1334, Durban 4000*
*Tel.: 031-3732576/7*
*Fax: 031-3732946*

31st March 2021
Mrs Kesarie Singh
c/o Department of Information Technology
Faculty of Accounting and Informatics
Durban University of Technology

Dear Ms Singh

**PERMISSION TO CONDUCT RESEARCH AT THE DUT**

Your email correspondence in respect of the above refers. I am pleased to inform you that the Institutional Research and Innovation Committee (IRIC) has granted **Gatekeeper Permission** for you to conduct your research "Intelligent Decision Support System for Selection of Learning Apps to Promote Critical Thinking in First Year Programming Students" at the Durban University of Technology. **Kindly note that this letter must be issued to the IREC for approval before you commence data collection.**

The DUT may impose any other condition it deems appropriate in the circumstances having regard to nature and extent of access to and use of information requested.

We would be grateful if a summary of your key research findings would be submitted to the IRIC on completion of your studies.

Kindest regards.
Yours sincerely

_____
DR LINDA ZIKHONA LINGANISO
DIRECTOR: RESEARCH AND POSTGRADUATE SUPPORT DIRECTORATE

# ANNEXURE F: The Survey Questionnaire



## The Impact of Visual Programming Environments on Critical Thinking

Survey Questionnaire

Dear Student

Thank you for agreeing to participate in this research project which is intended for your lecturers to make better and more informed decisions on the software tools to be used for beginner programming students. One of these tools is the visual programming environment which aims to provide support for novice programmers to better understand the key thought processes around logic and reasoning, one of the basic requirements for success in understanding programming.

The information you provide in this survey will assist your lecturers choose the most appropriate visual programming environment for the introductory programming module.

Please note that all responses are confidential, participation on this survey is voluntary and that only aggregate reporting will be used. Your responses cannot be linked back to you, as your details are not stored on the system.

Ethical clearance for this study has been obtained from the Faculty Research Ethics Committee and may be viewed on the link HERE.

You may contact the researcher by emailing kesaries@dut.ac.za, should you require any assistance and/or clarification. The questionnaire should take about 15 - 20 minutes to complete.

Thank you in advance for your participation.
K Singh

* Required

### Section A: Demographic Data

All questions require a response

1. Select your age group *

- ◯ less than 18 years
- ◯ 18-24 years
- ◯ More than 24 Years

2. Select your gender *

- ◯ Female
- ◯ Male
- ◯ Prefer not to say

3. What device have you been using for your programming module? (You may select more than one option) *

- ☐ Cell phone
- ☐ Laptop
- ☐ Desktop
- ☐ Tablet
- ☐ Other: _____

4. What internet connectivity have you been using? (You may select more than one option) *

- ☐ Home WiFi
- ☐ University WiFi
- ☐ Mobile data
- ☐ Other: _____

5. Have you been exposed to programming before? *

- ◯ Yes
- ◯ No

## Section B: Evaluate the Apps

For each question in this section, please provide a response for each app / row. All questions require a response.

1. Feedback: The App helped me to correct my errors while I was coding. *

| | Very good | Good | Satisfactory | Poor | Very poor | Not sure |
|---|---|---|---|---|---|---|
| Alice | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ |
| Scratch | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ |
| Blockly | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ |
| MIT App Inventor | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ |

2. Feedback: The App immediately alerted me to incorrect code. *

| | Very good | Good | Satisfactory | Poor | Very poor | Not sure |
|---|---|---|---|---|---|---|
| Alice | ○ | ○ | ○ | ○ | ○ | ○ |
| Scratch | ○ | ○ | ○ | ○ | ○ | ○ |
| Blockly | ○ | ○ | ○ | ○ | ○ | ○ |
| MIT App Inventor | ○ | ○ | ○ | ○ | ○ | ○ |

3. Interactivity: The App tells me exactly where my error lies. *

| | Very good | Good | Satisfactory | Poor | Very poor | Not sure |
|---|---|---|---|---|---|---|
| Alice | ○ | ○ | ○ | ○ | ○ | ○ |
| Scratch | ○ | ○ | ○ | ○ | ○ | ○ |
| Blockly | ○ | ○ | ○ | ○ | ○ | ○ |
| MIT App Inventor | ○ | ○ | ○ | ○ | ○ | ○ |

4. Problem solving: The App allows me to solve large, complex, real-world, authentic problem scenarios. *

| | Very good | Good | Satisfactory | Poor | Very poor | Not sure |
|---|---|---|---|---|---|---|
| Alice | ○ | ○ | ○ | ○ | ○ | ○ |
| Scratch | ○ | ○ | ○ | ○ | ○ | ○ |
| Blockly | ○ | ○ | ○ | ○ | ○ | ○ |
| MIT App Inventor | ○ | ○ | ○ | ○ | ○ | ○ |

5. Collaboration: The App allows me to work on a shared program with my friends. *

| | Very good | Good | Satisfactory | Poor | Very poor | Not sure |
|---|---|---|---|---|---|---|
| Alice | ○ | ○ | ○ | ○ | ○ | ○ |
| Scratch | ○ | ○ | ○ | ○ | ○ | ○ |
| Blockly | ○ | ○ | ○ | ○ | ○ | ○ |
| MIT App Inventor | ○ | ○ | ○ | ○ | ○ | ○ |

6. Collaboration: The App allows me to reuse my code or reuse my peers' solutions. *

| | Very good | Good | Satisfactory | Poor | Very poor | Not sure |
|---|---|---|---|---|---|---|
| Alice | ○ | ○ | ○ | ○ | ○ | ○ |
| Scratch | ○ | ○ | ○ | ○ | ○ | ○ |
| Blockly | ○ | ○ | ○ | ○ | ○ | ○ |
| MIT App Inventor | ○ | ○ | ○ | ○ | ○ | ○ |

7. Collaboration: The App allows me to communicate with my friend about my questions and queries about our projects. *

| | Very good | Good | Satisfactory | Poor | Very poor | Not sure |
|---|---|---|---|---|---|---|
| Alice | ○ | ○ | ○ | ○ | ○ | ○ |
| Scratch | ○ | ○ | ○ | ○ | ○ | ○ |
| Blockly | ○ | ○ | ○ | ○ | ○ | ○ |
| MIT App Inventor | ○ | ○ | ○ | ○ | ○ | ○ |

8. Metacognition: The App allows me to easily and repeatedly make changes to my solution. *

| | Very good | Good | Satisfactory | Poor | Very poor | Not sure |
|---|---|---|---|---|---|---|
| Alice | ○ | ○ | ○ | ○ | ○ | ○ |
| Scratch | ○ | ○ | ○ | ○ | ○ | ○ |
| Blockly | ○ | ○ | ○ | ○ | ○ | ○ |
| MIT App Inventor | ○ | ○ | ○ | ○ | ○ | ○ |

9. Logic: The App helped me to improve my logic skills. *

| | Very good | Good | Satisfactory | Poor | Very poor | Not sure |
|---|---|---|---|---|---|---|
| Alice | ○ | ○ | ○ | ○ | ○ | ○ |
| Scratch | ○ | ○ | ○ | ○ | ○ | ○ |
| Blockly | ○ | ○ | ○ | ○ | ○ | ○ |
| MIT App Inventor | ○ | ○ | ○ | ○ | ○ | ○ |

10. Logic and reasoning: The App quickly alerts me when the sequencing of steps in my solution is logically incorrect. *

| | Very good | Good | Satisfactory | Poor | Very poor | Not sure |
|---|---|---|---|---|---|---|
| Alice | ○ | ○ | ○ | ○ | ○ | ○ |
| Scratch | ○ | ○ | ○ | ○ | ○ | ○ |
| Blockly | ○ | ○ | ○ | ○ | ○ | ○ |
| MIT App Inventor | ○ | ○ | ○ | ○ | ○ | ○ |

11. Evaluation: The App helps me to make my code more efficient. *

| | Very good | Good | Satisfactory | Poor | Very poor | Not sure |
|---|---|---|---|---|---|---|
| Alice | ○ | ○ | ○ | ○ | ○ | ○ |
| Scratch | ○ | ○ | ○ | ○ | ○ | ○ |
| Blockly | ○ | ○ | ○ | ○ | ○ | ○ |
| MIT App Inventor | ○ | ○ | ○ | ○ | ○ | ○ |

12. Evaluation: The App is able to evaluate my work. *

| | Very good | Good | Satisfactory | Poor | Very poor | Not sure |
|---|---|---|---|---|---|---|
| Alice | ○ | ○ | ○ | ○ | ○ | ○ |
| Scratch | ○ | ○ | ○ | ○ | ○ | ○ |
| Blockly | ○ | ○ | ○ | ○ | ○ | ○ |
| MIT App Inventor | ○ | ○ | ○ | ○ | ○ | ○ |

13. Evaluation: The App allows me to compare my solution against my peers. *

| | Very good | Good | Satisfactory | Poor | Very poor | Not sure |
|---|---|---|---|---|---|---|
| Alice | ○ | ○ | ○ | ○ | ○ | ○ |
| Scratch | ○ | ○ | ○ | ○ | ○ | ○ |
| Blockly | ○ | ○ | ○ | ○ | ○ | ○ |
| MIT App Inventor | ○ | ○ | ○ | ○ | ○ | ○ |

14. Alternate solutions: The App helps me to think about solving the problem in different ways. *

| | Very good | Good | Satisfactory | Poor | Very poor | Not sure |
|---|---|---|---|---|---|---|
| Alice | ○ | ○ | ○ | ○ | ○ | ○ |
| Scratch | ○ | ○ | ○ | ○ | ○ | ○ |
| Blockly | ○ | ○ | ○ | ○ | ○ | ○ |
| MIT App Inventor | ○ | ○ | ○ | ○ | ○ | ○ |

15. Synthesis: The App allows me to solve problems where I have to draw my knowledge from different programming topics. *

| | Very good | Good | Satisfactory | Poor | Very poor | Not sure |
|---|---|---|---|---|---|---|
| Alice | ○ | ○ | ○ | ○ | ○ | ○ |
| Scratch | ○ | ○ | ○ | ○ | ○ | ○ |
| Blockly | ○ | ○ | ○ | ○ | ○ | ○ |
| MIT App Inventor | ○ | ○ | ○ | ○ | ○ | ○ |

16. Application: The App allows me to apply the concepts of sequence, selection and iteration. *

| | Very good | Good | Satisfactory | Poor | Very poor | Not sure |
|---|---|---|---|---|---|---|
| Alice | ○ | ○ | ○ | ○ | ○ | ○ |
| Scratch | ○ | ○ | ○ | ○ | ○ | ○ |
| Blockly | ○ | ○ | ○ | ○ | ○ | ○ |
| MIT App Inventor | ○ | ○ | ○ | ○ | ○ | ○ |

17. Metacognitive monitoring: The App interface is designed in a way that encourages me to improve my solution. *

| | Very good | Good | Satisfactory | Poor | Very poor | Not sure |
|---|---|---|---|---|---|---|
| Alice | ○ | ○ | ○ | ○ | ○ | ○ |
| Scratch | ○ | ○ | ○ | ○ | ○ | ○ |
| Blockly | ○ | ○ | ○ | ○ | ○ | ○ |
| MIT App Inventor | ○ | ○ | ○ | ○ | ○ | ○ |

18. Multimedia: The interface is visually rich in multimedia and includes sound, color, graphics and animation. *

| | Very good | Good | Satisfactory | Poor | Very poor | Not sure |
|---|---|---|---|---|---|---|
| Alice | ○ | ○ | ○ | ○ | ○ | ○ |
| Scratch | ○ | ○ | ○ | ○ | ○ | ○ |
| Blockly | ○ | ○ | ○ | ○ | ○ | ○ |
| MIT App Inventor | ○ | ○ | ○ | ○ | ○ | ○ |

19. Simulation programming: The App uses simple statements to mimic a high-level programming language in an active learning environment. *

| | Very good | Good | Satisfactory | Poor | Very poor | Not sure |
|---|---|---|---|---|---|---|
| Alice | ○ | ○ | ○ | ○ | ○ | ○ |
| Scratch | ○ | ○ | ○ | ○ | ○ | ○ |
| Blockly | ○ | ○ | ○ | ○ | ○ | ○ |
| MIT App Inventor | ○ | ○ | ○ | ○ | ○ | ○ |

20. Creativity: The App allows me to create useful and original applications like games and movies. *

| | Very good | Good | Satisfactory | Poor | Very poor | Not sure |
|---|---|---|---|---|---|---|
| Alice | ○ | ○ | ○ | ○ | ○ | ○ |
| Scratch | ○ | ○ | ○ | ○ | ○ | ○ |
| Blockly | ○ | ○ | ○ | ○ | ○ | ○ |
| MIT App Inventor | ○ | ○ | ○ | ○ | ○ | ○ |

21. Creativity: The App has various features that enable me to use my creative skills. *

| | Very good | Good | Satisfactory | Poor | Very poor | Not sure |
|---|---|---|---|---|---|---|
| Alice | ○ | ○ | ○ | ○ | ○ | ○ |
| Scratch | ○ | ○ | ○ | ○ | ○ | ○ |
| Blockly | ○ | ○ | ○ | ○ | ○ | ○ |
| MIT App Inventor | ○ | ○ | ○ | ○ | ○ | ○ |

22. Creativity: The App supports the simulation of many creative ideas when solving the problem. *

| | Very good | Good | Satisfactory | Poor | Very poor | Not sure |
|---|---|---|---|---|---|---|
| Alice | ○ | ○ | ○ | ○ | ○ | ○ |
| Scratch | ○ | ○ | ○ | ○ | ○ | ○ |
| Blockly | ○ | ○ | ○ | ○ | ○ | ○ |
| MIT App Inventor | ○ | ○ | ○ | ○ | ○ | ○ |

23. Analysis: When interpreting the problem statement, the App interface gives me clues on how to solve the problem. *

| | Very good | Good | Satisfactory | Poor | Very poor | Not sure |
|---|---|---|---|---|---|---|
| Alice | ○ | ○ | ○ | ○ | ○ | ○ |
| Scratch | ○ | ○ | ○ | ○ | ○ | ○ |
| Blockly | ○ | ○ | ○ | ○ | ○ | ○ |
| MIT App Inventor | ○ | ○ | ○ | ○ | ○ | ○ |

24. Complexity of problem: The App allows me the flexibility to elaborate or build on my idea. *

| | Very good | Good | Satisfactory | Poor | Very poor | Not sure |
|---|---|---|---|---|---|---|
| Alice | ○ | ○ | ○ | ○ | ○ | ○ |
| Scratch | ○ | ○ | ○ | ○ | ○ | ○ |
| Blockly | ○ | ○ | ○ | ○ | ○ | ○ |
| MIT App Inventor | ○ | ○ | ○ | ○ | ○ | ○ |

25. Disposition: The App forces me to have an enquiring mind. *

| | Very good | Good | Satisfactory | Poor | Very poor | Not sure |
|---|---|---|---|---|---|---|
| Alice | ○ | ○ | ○ | ○ | ○ | ○ |
| Scratch | ○ | ○ | ○ | ○ | ○ | ○ |
| Blockly | ○ | ○ | ○ | ○ | ○ | ○ |
| MIT App Inventor | ○ | ○ | ○ | ○ | ○ | ○ |

Page 1 of 1

Submit

Never submit passwords through Google Forms.

# ANNEXURE G: The Feature Questionnaire



## Feature Questionnaire

Expert Survey: Rating the Critical Thinking Attributes for programming

### Survey Details

Dear Programming Academic Expert

Thank you for agreeing to participate in this research project.

This survey focusses on critical thinking as a deep higher order mental process with particular emphasis on the cognitive and metacognitive skills that complement analysis, synthesis, evaluation, logical reasoning and computational thinking. If students can learn how to think more critically, they will create more robust solutions to programming problems because critical thinking is fundamentally important in improving problem solving skills.

Thinking of alternate solutions, reflective analysis through debugging and creating innovative ways to solve problems are just some of the cognitive strategies required of programming students. Therefore, critical thinking (logical reasoning, problem solving, making judgements and decisions) and programming (algorithm design, testing and debugging) are closely associated.

The purpose of this feature questionnaire is to gather your score of the relative importance of each of the given critical thinking criteria within the context of computer programming. You are therefore required to use your expert knowledge and experience to rate the importance of each criteria listed below in promoting critical thinking among first year novice programmers.

Please note that all responses are confidential, participation on this survey is voluntary and that only aggregate reporting will be used. Your responses cannot be linked back to you, as your details are not stored on the system.

You may contact the researcher by emailing kesaries@dut.ac.za, should you require any assistance and/or clarification. The questionnaire should take about 10 - 15 minutes to complete.

Thank you in advance for your participation. K Singh.

Page 1 of 2

Next

# Feature Questionnaire

Rate the level of Importance of each of the following criteria to promote critical thinking among 1st year programming students.

1. Feedback: giving others feedback on their code

| | Very Important | Important | Moderately Important | Slightly Important | Not Important |
|---|---|---|---|---|---|
| Select an option | ○ | ○ | ○ | ○ | ○ |

2. Feedback: the compiler displaying error messages

| | Very Important | Important | Moderately Important | Slightly Important | Not Important |
|---|---|---|---|---|---|
| Select an option | ○ | ○ | ○ | ○ | ○ |

3. Interactivity: the degree of human-human interaction and human-machine interaction

| | Very Important | Important | Moderately Important | Slightly Important | Not Important |
|---|---|---|---|---|---|
| Select an option | ○ | ○ | ○ | ○ | ○ |

4. Problem solving: solving large, complex, real-world, authentic problem scenarios

| | Very Important | Important | Moderately Important | Slightly Important | Not Important |
|---|---|---|---|---|---|
| Select an option | ○ | ○ | ○ | ○ | ○ |

5. Collaboration: working on shared computer programs

| | Very Important | Important | Moderately Important | Slightly Important | Not Important |
|---|---|---|---|---|---|
| Select an option | ○ | ○ | ○ | ○ | ○ |

6. Collaboration: reusing ones code or ones peers' code

| | Very Important | Important | Moderately Important | Slightly Important | Not Important |
|---|---|---|---|---|---|
| Select an option | ○ | ○ | ○ | ○ | ○ |

7. Collaboration: communicating with a friend about ones questions and queries about ones work

| | Very Important | Important | Moderately Important | Slightly Important | Not Important |
|---|---|---|---|---|---|
| Select an option | ○ | ○ | ○ | ○ | ○ |

8. Metacognition: reflection and rethinking ones solution

| | Very Important | Important | Moderately Important | Slightly Important | Not Important |
|---|---|---|---|---|---|
| Select an option | ○ | ○ | ○ | ○ | ○ |

9. Logic: arranging statements in logical order

| | Very Important | Important | Moderately Important | Slightly Important | Not Important |
|---|---|---|---|---|---|
| Select an option | ○ | ○ | ○ | ○ | ○ |

10. Logic and reasoning: process of finding logic errors in code

| | Very Important | Important | Moderately Important | Slightly Important | Not Important |
|---|---|---|---|---|---|
| Select an option | ○ | ○ | ○ | ○ | ○ |

11. Evaluation: evaluating the efficiency of ones code

| | Very Important | Important | Moderately Important | Slightly Important | Not Important |
|---|---|---|---|---|---|
| Select an option | ○ | ○ | ○ | ○ | ○ |

12. Evaluation: receiving evaluation comments of ones code from another

| | Very Important | Important | Moderately Important | Slightly Important | Not Important |
|---|---|---|---|---|---|
| Select an option | ○ | ○ | ○ | ○ | ○ |

13. Evaluation: comparing ones solution against ones peers

| | Very Important | Important | Moderately Important | Slightly Important | Not Important |
|---|---|---|---|---|---|
| Select an option | ○ | ○ | ○ | ○ | ○ |

14. Alternate solutions: finding alternate ways to solve the problem

| | Very Important | Important | Moderately Important | Slightly Important | Not Important |
|---|---|---|---|---|---|
| Select an option | ○ | ○ | ○ | ○ | ○ |

15. Synthesis: solving problems where one has to draw knowledge from different programming topics

| | Very Important | Important | Moderately Important | Slightly Important | Not Important |
|---|---|---|---|---|---|
| Select an option | ○ | ○ | ○ | ○ | ○ |

16. Application: applying the concepts of sequence, selection and iteration

| | Very Important | Important | Moderately Important | Slightly Important | Not Important |
|---|---|---|---|---|---|
| Select an option | ○ | ○ | ○ | ○ | ○ |

17. Metacognitive monitoring: interaction with the GUI from the IDE

| | Very Important | Important | Moderately Important | Slightly Important | Not Important |
|---|---|---|---|---|---|
| Select an option | ○ | ○ | ○ | ○ | ○ |

18. Multimedia: a visually rich interface in multimedia that includes sound, color, graphics and animation

| | Very Important | Important | Moderately Important | Slightly Important | Not Important |
|---|---|---|---|---|---|
| Select an option | ○ | ○ | ○ | ○ | ○ |

19. Simulation programming: simulation in an active learning environment

| | Very Important | Important | Moderately Important | Slightly Important | Not Important |
|---|---|---|---|---|---|
| Select an option | ○ | ○ | ○ | ○ | ○ |

20. Creativity: creating useful and original applications like games

| | Very Important | Important | Moderately Important | Slightly Important | Not Important |
|---|---|---|---|---|---|
| Select an option | ○ | ○ | ○ | ○ | ○ |

21. Creativity: utilising the various features of a learning App like animation and storytelling

| | Very Important | Important | Moderately Important | Slightly Important | Not Important |
|---|---|---|---|---|---|
| Select an option | ○ | ○ | ○ | ○ | ○ |

22. Creativity: interacting with an App that supports the simulation of many creative ideas to solve a problem

| | Very Important | Important | Moderately Important | Slightly Important | Not Important |
|---|---|---|---|---|---|
| Select an option | ○ | ○ | ○ | ○ | ○ |

23. Analysis: analysing a complex problem statement

| | Very Important | Important | Moderately Important | Slightly Important | Not Important |
|---|---|---|---|---|---|
| Select an option | ○ | ○ | ○ | ○ | ○ |

24. Complexity of problem: the nature and size of a problem

| | Very Important | Important | Moderately Important | Slightly Important | Not Important |
|---|---|---|---|---|---|
| Select an option | ○ | ○ | ○ | ○ | ○ |

25. Disposition: having an enquiring mind

| | Very Important | Important | Moderately Important | Slightly Important | Not Important |
|---|---|---|---|---|---|
| Select an option | ○ | ○ | ○ | ○ | ○ |

Page 2 of 2

Back Submit

# ANNEXURE H: Turnitin Cover Page

Dissertation

ORIGINALITY REPORT

| 14% | 11% | 6% | 4% |
|---|---|---|---|
| SIMILARITY INDEX | INTERNET SOURCES | PUBLICATIONS | STUDENT PAPERS |

PRIMARY SOURCES

| | | |
|---|---|---|
| 1 | researchspace.ukzn.ac.za<br>Internet Source | 2% |
| 2 | hdl.handle.net<br>Internet Source | 1% |
| 3 | pdfs.semanticscholar.org<br>Internet Source | <1% |
| 4 | researchonline.ljmu.ac.uk<br>Internet Source | <1% |
| 5 | link.springer.com<br>Internet Source | <1% |
| 6 | www.tandfonline.com<br>Internet Source | <1% |
| 7 | Submitted to Universiti Teknologi MARA<br>Student Paper | <1% |
| 8 | uir.unisa.ac.za<br>Internet Source | <1% |
| 9 | Submitted to University of KwaZulu-Natal<br>Student Paper | <1% |

## ANNEXURE I: Language Proficiency Certificate

# THE WRITING STUDIO

*Writing and Editing Practice*

**Certificate 2821**

**TO WHOM IT MAY CONCERN**

28 August 2021

This dissertation, entitled **Intelligent Decision Support System for Selection of Learning Apps to Promote Critical Thinking in First Year Programming Students,** by Kesarie Singh**,** has been edited and reviewed to ensure technically accurate and contextually appropriate use of language for research at this level of study.

**Yours sincerely**

**CM ISRAEL, BA Hons (UDW) MA (UND) MA (US) PhD (UNH)**
**LANGUAGE EDITOR AND WRITING CONSULTANT**
Connieisrael90@gmail.com **Mobile 082 4988166**