Lecturers' perspectives of using problem solving guidelines during the teaching of computer programming

K Montoeli

orcid.org 0000-0002-1832-5175

Dissertation submitted in fulfilment of the requirements for the degree *Masters of Science in Computer Science* at the North West University

Supervisor: Prof DB Jordaan

Graduation Ceremony July 2018

Student number: 24036552

DECLARATION

I, Kgarebe Montoeli declare that **Lecturers' perspectives of using problem solving guidelines during the teaching of computer programming** is my own work and that all the sources I have used or quoted have been indicated and acknowledged by means of complete references.

Signature: [illegible]

Date: July 2018

CTrans
CENTRE FOR TRANSLATION AND PROFESSIONAL LANGUAGE SERVICES

NWU®
NORTH-WEST UNIVERSITY
NOORDWES-UNIVERSITEIT
UNIBESITI YA BOKONE-BOPHIRIMA

PO Box 1174, Vanderbijlpark, 1900

Tel: (016) 910-3485
Fax: (016) 910 3463
Web: www.nwu.ac.za

Mrs Wendy Barrow
email: wendy.barrow@nwu.ac.za
Ms Menitza Botha
email: menitza.botha@nwu.ac.za

13 March 2018

This letter serves to confirm that this document has been edited by the *Centre for Translation and Professional Language Services* (CTrans). CTrans is a registered corporate member of the South African Translators' Institute (SATI) that makes use of qualified and experienced language practitioners to provide professional translation and language editing services.

CTrans hereby acknowledges that the dissertation with the title: **Lecturers' Perspectives of Using Problem-Solving Guidelines During the Teaching of Computer Programming** has undergone a proper and professional language edit (including the checking of spelling, grammar, register and punctuation). The onus rests on the client to work through the proposed changes after the edit and accept or reject these changes.

Yours sincerely

[illegible]

Mrs Wendy Barrow

CTrans Coordinator

DEDICATION

To my children Olorato, Tshegofatso, Keabetswe, and Kefentse Jnr. The amount of time spent away from you cannot be replaced by words.

ACKNOWLEDGEMENTS

In general, the road of completing this dissertation has not been easy. The stumbling blocks that I had to overcome were overwhelming. This has been an intense period of learning in both this field and at a personal level. The support and words of encouragement I have received through this journey kept me in this race. Hence, I would like to acknowledge the people who have supported me throughout this period:

- God Almighty, without His grace, love, and insight all of this could not have been possible.
- My supervisor, Professor Dawid Jordaan; the patience, support, and guidance that you have provided me has been highly appreciated. This journey was not easy for either of us, but your time and knowledge that I have received is not measurable.
- My husband, Kefentse Makgere Snr, your love, support and words of encouragement gave me the strength to continue.
- My VUT family, Dr Hans Brits and my colleagues, thank you for the words of encouragement and for listening to me when I wanted to throw in the towel.
- My family, the Mphahlele’s, looking back and where I am now, it is only by your love and guidance.
- Lastly, my Vaal University of Technology (VUT) lecturers who granted me the opportunity to conduct my study with them.

ABSTRACT

The purpose of this study was to discover what the views and experiences of the lecturers are when presenting programming, using problem-solving guidelines. The study was prompted by the “digital natives” students who are computer literate because of being exposed to computers, the Internet, and social media. Therefore, learning and thinking of these students is different to other generations. The fast-paced digital environment that the students find themselves in, requires an individual to possess various skills in order to adapt and fit in. Amongst many fundamental skills students have to master, and to become skilled in computer programming, problem solving skills were identified. The requirements and expectation of digital native students calls for the higher education systems to revise and adapt to these students’ kind of learning.

Different theories guided this study in answering the main research question: *How do lecturers perceive the use of problem-solving guidelines during the teaching of computer programming?* This study followed interpretivist methodology as the appropriate research strategy. Data was collected using semi-structured interviews. Lecturers who presented different programming subjects were the participants of the study. The findings of the study revealed lecturers’ experiences of students’ ability to solve problems; lecturers’ perceptions of students’ attitude when confronted with problem solving situations; different approaches used by lecturers when teaching programming; and lecturers’ perceptions of problem-solving guidelines.

Recommendations in this study suggest that lecturers should be cognisant of principles and strategies of good problem solving, provide assistance, and make provision for students to maximize their problem-solving knowledge.

Keywords: problem solving, guidelines, computer programming, critical thinking, creative thinking, strategies.

Contents

CONTENTS 1

LIST OF FIGURES 7

LIST OF TABLES 8

1 CHAPTER ONE: INTRODUCTION 9

1.1 INTRODUCTION 9

1.2 BACKGROUND 10

1.2.1 Problems, problem solving and thinking 10

1.2.2 Types of problems 11

1.2.3 Teaching for problem solving 12

1.2.4 Teaching of problem solving 13

1.2.5 Problem solving skills and programming 13

1.3 PROBLEM STATEMENT 14

1.4 RESEARCH STRATEGY 15

1.5 OBJECTIVE OF THE STUDY 15

1.5.1 Primary objective 15

1.5.2 Research question 15

1.6 SIGNIFICANCE OF THE STUDY 16

1.7 RESEARCH SCOPE AND LIMITATIONS 16

1.7.1 Research scope 16

1.7.2 Limitations 16

1.8 ETHICAL CONSIDERATIONS 17

1.9 STRUCTURE OF THE DISSERTATION 17

1.10 SUMMARY 18

2 CHAPTER TWO: LITERATURE REVIEW 19

2.1 INTRODUCTION 19

2.2 FUNDAMENTALS AND CONCEPTS OF PROBLEM SOLVING 19

2.2.1 Problems' nature and structure 19

2.2.2 Types of problems 20

2.3 CRITICAL AND CREATIVE THINKING 22

2.3.1 Critical thinking 22

2.3.2 Obstacles of critical thinking 24

2.3.3 Monitoring students through the critical thinking process 24

2.3.4 Concluding remarks on the critical thinking process 27

2.3.5 Creative thinking 28

2.4 LEARNING STYLES 31

2.4.1 Sensing and intuitive learners 32

2.4.2 Visual and auditory learners 33

2.4.3 Inductive and deductive learners 33

2.4.4 Active and reflective learners 34

2.4.5 Sequential and global learners 34

2.5 TEACHING FOR PROBLEM SOLVING 35

2.6 APPROACHES TO SOLVE PROBLEMS 38

2.6.1 Explicit teaching 40

2.6.2 Command style teaching 40

2.6.3 Teaching by task 41

2.6.4 Problem-solving teaching 42

2.6.5 Pre-recorded lectures 43

2.6.6 Pair/group programming 43

2.6.7 Peer tutoring 44

2.6.8 Concluding remarks on teaching methods 44

2.6.9 Dual Common Model for problem solving 45

2.6.10 Problem-Based Learning (PBL) 46

2.6.11 Puzzle-Based Learning (PZBL) 46

2.6.12 Game-themed programming 47

2.6.13 Concluding remarks on methods used in programming 48

2.7 GUIDELINES USED IN TEACHING PROBLEM SOLVING 48

2.7.1 The problem-solving activity framework 48

2.7.2 Flowcharts 52

2.7.3 Algorithms 53

2.7.4 Pseudo code 54

2.8 SUMMARY 56

3 CHAPTER THREE: RESEARCH DESIGN AND METHODOLOGY 57

3.1 INTRODUCTION 57

3.2 THEORETICAL FRAMEWORK FOR THE RESEARCH AND DESIGN METHODOLOGY 57

3.3 RESEARCH PARADIGMS 59

3.3.1 Interpretivism 62

3.4 RESEARCH APPROACH 66

3.4.1 Research design: Qualitative research approach 67

3.4.2 Rationale for a qualitative study 69

3.4.3 Deductive and inductive reasoning 71

3.4.4 Grounded theory as qualitative approach 72

3.4.5 Time horizons 78

3.4.6 Unit of analysis 79

3.4.7 Selection of research participants 79

3.5 DATA COLLECTION STRATEGY: INTERVIEWS 79

3.5.1 Semi-structured interviews 81

3.6 DATA ANALYSIS 81

3.6.1 Trustworthiness 82

3.7 ETHICAL CONSIDERATIONS APPLICABLE TO THE CURRENT STUDY 83

3.8 CONCLUSION 84

4 CHAPTER FOUR: DATA COLLECTION AND ANALYSIS 86

4.1 INTRODUCTION 86

4.2 RESEARCH APPROACH 86

4.2.1 Selection of participants 86

4.2.2 Data collection 87

4.2.3 Data analysis 88

4.3 DISCUSSION OF THE RESULTS 92

4.3.1 Lecturers' experience of students' ability to solve problems 92

4.3.2 Lecturers' perceptions of students' attitude when confronted with problem solving 93

4.3.3 Approaches used by lecturers when teaching programming 95

4.3.4 Lecturers' perspectives of using problem-solving guideline 96

4.3.5 Lecturers' perceptions of the usefulness of guidelines 96

4.3.6 Methods used to present programming problems 97

4.3.7 Additional methods used for different learning styles 98

4.4 CONCLUSION 100

5 CHAPTER FIVE: CONCLUSIONS, REFLECTIONS AND RECOMMENDATIONS 101

5.1 INTRODUCTION 101

5.2 SUMMARY OF THE RESEARCH 101

5.2.1 Research objective: To explore lecturers' perspectives of using problem-solving guidelines during the teaching of computer programming 104

5.3 SYNTHESIS 104

5.3.1 Summary of findings 105

5.4 DISCUSSION AND REFLECTION..106

5.4.1 Methodology reflection..106

5.4.2 Substantive reflection..108

5.4.3 Limitations of the study..108

5.4.4 Scientific reflection..108

5.5 RECOMMENDATIONS 109

5.5.1 Recommendations for policy and practice..109

5.5.2 Recommendations for future research 110

5.6 SUMMARY 111

REFERENCES 112

ADDENDUM A 128

LIST OF FIGURES

Figure 1: 5-Step model to move students towards critical thinking* 25

Figure 2: Steps in teaching students how to solve problems 37

Figure 3: The logical flow of a process of ordering of a burger meal* 53

Figure 4: Research onion framework (Sauders *et al.*, 2003) 58

Figure 5: Research paradigms used in the analysis of social research. Adapted from Burrell and Morgan (1979) 61

Figure 6: Interrelated aspects in information systems research 65

Figure 7: Induction, deduction and falsification of theories adapted from Welman and Kruger (2001:29) 72

LIST OF TABLES

Table 1: Dimensions of Learning and Teaching Styles* 32

Table 2: Teaching methods and strategies* 40

Table 3: The problem-solving activities framework* 49

Table 4: Interpretive characteristics of the present study 66

Table 5: Summary of qualitative characteristics * 69

Table 6: Advantages and disadvantages of interviews * 80

Table 7: Themes and codes emerged from the Atlas.ti™ Version 8 analyses 91

CHAPTER ONE: INTRODUCTION

1.1 Introduction

A principal goal of education is to provide students with ways that will enable them to solve problems in new situations by using what they have learned. This implies that educators aim to improve students' problem-solving abilities. "Central to programming is the ability to comprehend a computer program, so to establish a valid mental representation of the problem solved by the program. Because of the lack of knowledge and experience, novice programmers have problems with constructing the viable models of problems" (Bednarix, Moreno, & Myller, 2006).

Programming is part of almost all computer science curricula and there is a perception that programming is a difficult skill to master. This perception stands out as a major problem when reasons for the decline in interest in studying computer science are addressed (Khaleel *et al.*, 2015; Peters & Pears, 2012). The programming environment requires higher order thinking skills, which include problem-solving abilities (Kotovsky, 2003) and today's digital environment requires from students the ability and fundamental skills to solve problems (Nag, Katz, & Saenz-Otero, 2013). Stanescu, Stefan, and Roceanu (2011) argue that the determination to attempt to solve problems improves when students are motivated and interactively involved in what they are doing. When problem solving and programming problems are presented in a context that students can relate to, they are more motivated and have a better understanding of what proper solutions should entail (Tan & Rahaman, 2009). Algorithms (or a set of rules or problem solving guidelines) play an essential role when solutions to problems are developed in a programming environment. Shabanah and Chen (2009) argue that problem solving, or algorithms should be presented in ways that make sense to students.

1.2 Background

1.2.1 Problems, problem solving and thinking

According to the Gestalt psychologist, Karl Duncker, “a problem arises when a living creature has a goal but does not know how this goal is to be reached. Whenever one cannot go from the given situation to the desired situation simply by action, then there has to be recourse to thinking. Such thinking has the task of devising some action, which may mediate between the existing and desired situations” (Duncker, 1945).

Problem solving occurs when the problem solver wants the problem in a goal state but no obvious way of changing the problem from its initial state to the goal state is known (Mayer, 2012). Problem solving is a process or an activity in which the known is used to discover what is unknown. Mayer and Wittrock (2006) define problem-solving as “cognitive processing directed at achieving a goal when no solution method is obvious to the problem solver.” This definition of Mayer and Wittrock (2006) has four elements: (1) problem solving is cognitive, that is, problem solving occurs within the problem solver's cognitive system and can only be inferred from the problem solver's behavior; (2) problem solving is a process, that is, problem solving involves applying cognitive processes to cognitive representations in the problem solver's cognitive system; (3) problem solving is directed, that is, problem solving is guided by the problem solver's goals; and (4) problem solving is personal, that is, problem solving depends on the knowledge and skill of the problem solver. From these four elements, problem solving is summarised as a reasoning or thinking process, focused on transforming the problem from one state (the given state) to another state (the goal state) in the absence of an immediate solution method to the problem solver. For example, problem solving occurs when a student understands how the heart works by reading a biology textbook or solves a complex mathematics word problem.

Problem solving is related to terms such as thinking, reasoning, decision making, critical thinking, and creative thinking. Thinking refers to a problem solver's cognitive processing, but it includes both directed thinking (which is problem solving) and undirected thinking (such as daydreaming). Thus, thinking is a broader term that includes problem solving as a subset of thinking (i.e., a kind of thinking, i.e., directed thinking) (Mayer & Wittrock, 2006). Reasoning, decision making, critical thinking, and creative thinking are subsets of problem solving – these concepts are discussed in detail in Chapter 2.

1.2.2 Types of problems

Problems can be categorised as well-defined (well-structured), ill-defined (ill-structured), routine, or non-routine problems. Well-defined problems have a clearly specified given state, a clearly specified goal state, and a clearly specified set of allowable operations; while ill-defined problems lack one or more of these variables (Hong, 1998).

A well-defined problem yields a right answer through the application of an appropriate algorithm. Most textbook problems set in mathematics, science, engineering, or business, feature well-structured problems that have right answers. For example, converting a unit of measure between its English and metric equivalents. In contrast, an ill-defined problem does not yield a particular, certain answer. Ill-structured problems mirror real-world problems with missing, conflicting, or inclusive data, where disputants disagree about appropriate assumptions or theories, or where values conflict. An example of an ill-defined problem is to predict how to safely dispose of nuclear waste.

When a problem solver knows exactly how to go about solving a problem, the problem is routine. The problem in non-routine if the problem solver initially does not know how to go about solving a problem (Mayer & Hegarty, 1996). It is significant to understand that a problem can be routine or non-routine subject to the solver's knowledge, and the same problem can be routine for one person and non-routine for another.

1.2.3 Teaching for problem solving

The distinction between learning by rote and learning by understanding was documented years ago by Wertheimer (1959). For example, in teaching students to learn how to compute the area of a parallelogram by a rote method, students are shown how to measure the height, how to measure the base, and how to multiply height times base using the formula: *area = height x base*. According to Wertheimer (1959), this rote method of instruction leads to good performance on retention tests (solving similar problems) and poor performance on transfer tests (solving new problems). In contrast, learning by understanding involves helping students see that if they can cut off the triangle from one end of the parallelogram and place it on the other side to form a rectangle; then, they can put 1 x 1 squares over the surface of the rectangle to determine how many squares form the area. This meaningful method of instruction leads to good retention and good transfer performance. Wertheimer (1959) claims that rote instruction creates reproductive thinking—applying already learned procedures to a problem—whereas meaningful instruction leads to productive thinking—adapting what was learned to new kinds of problems.

Mayer and Wittrock (2006) identify instructional methods that are intended to promote meaningful learning, such as providing advance organizers that prime appropriate prior knowledge during learning; asking learners to explain aloud a text they are reading; presenting worked out examples along with commentary; or providing hints and guidance as students work on an example problem. A major goal of meaningful methods of instruction is to promote problem-solving transfer, that is, the ability to use what was learned in new situations.

1.2.4 Teaching of problem solving

To become better problem solvers, students need problem solving knowledge and skills. In this regard, Mayer (2008) identifies four issues that are involved in designing a problem-solving course: 1) what to teach; 2) how to teach; 3) where to teach; and 4) when to teach. In Chapter 2, a few classic problem-solving cases will be described that meet these four criteria.

1.2.5 Problem solving skills and programming

According to Estivill-Castro (2010) IT graduates must possess problem-solving skills by expressing methods and solutions in the language that defines the operation of automation. The latter is challenging for graduates to learn.

Estivill-Castro (2010) argues that problem solving skills are more crucial than learning programming languages. He further deliberates that the current technologies are constantly changing. Computers are becoming faster and more powerful, technologies used to program vary, but the computers remain to be state-transition machines. Learning programming is a bearable and a temporary skill (Estivill-Castro, 2010) yet proven to be difficult. Therefore, describing solutions to problems algorithmically, in addition to teaching this skill, will continue to be problematic for many (Oddie, Hazlewood, Blakeway, & Whitfield, 2010). The outlined difficulties of programming are influenced by the lack of confidence, as well as the lack of conceptual and abstract thinking.

The traditional way of teaching and learning programming includes the identification and understanding of syntax and the structural elements of the chosen language have been noted to be fruitful. The success can be notable if students have experience of problem solving and symbolic reasoning.

1.3 Problem statement

Students today live in the fast-paced digital world: They are used to digital ways of communicating, learning, and acquiring knowledge via various technologies such as the Internet and social media (Nag et al., 2013:146). The stereotypical old fashion technology that students are confronted with inside classrooms discourage them as they are used to modern, up-to-date technologies outside the classroom (Husain, 2011:1). In comparison to previous generations, students today grow up in an advanced digital environment and they think and learn differently (Prensky and Berry (2001:3). According to Maravić Čisar, Radosav, Pinter, and Čisar (2011) programming is a difficult skill to learn. Students are often demotivated by the time and effort it takes to become skilled in computer programming as they are used to a fast-paced digital environment (Zeeman, 2014:21). As a result, they find it boring to learn how to program (Ali & Smith, 2014:62). Conneely, Girvan, and Tangney (2012:2) highlight the fact that requirements, in terms of learning in the twenty-first century, have changed while education systems generally remain the same.

A change is needed in education so as to adapt to the changing environment, improve student engagement in the classroom and actively learning, in order to remain relevant to the requirements of the changing world (Kaiser & Wisniewski, 2012:138). Students should be active participants rather than passive observers in today's digital world (Prensky, 2001:11). Houghton (2004:1) reports that students at higher education institutions (HEI) have indicated that there have been insufficient preparations in the process of problem solving. Cai and Lester (2010:1) suggest that for students to become effective problem solvers at all levels, educators must know how to incorporate problem solving meaningfully into their curriculums.

This discussion led to the formulation of the problem statement for this research: guidelines are required to teach problem solving in the computer programming class to meet the problem-solving needs of the new generation of digitally oriented students.

1.4 Research strategy

A qualitative approach was used in this study. The aim of the study was to determine lecturers' views concerning the use of problem-solving guidelines during the lecturing of computer programming. An interpretivist paradigm was applied throughout the study. Data were gathered and interpreted to draw conclusions to address the objective of the study.

1.5 Objective of the study

1.5.1 Primary objective

The main objective of the study was to explore lecturers' perspectives of using problem-solving guidelines during the teaching of computer programming.

1.5.2 Research question

To achieve the primary objective, the main research question for the dissertation is:

How do lecturers perceive the use of problem-solving guidelines during the teaching of computer programming?

1.6 Significance of the study

The importance of the study is to understand how Information and Communication Technology (ICT) lecturers perceive the use of problem-solving guidelines. The findings of this study may benefit both ICT lecturers and students. Through the study's recommendations, the ICT lecturers will be capable of enhancing and incorporating their teaching strategies and methods when presenting programming.

1.7 Research scope and limitations

1.7.1 Research scope

To write good computer programs, students need to have good problem-solving skills. The use of tried and tested problem-solving guidelines can assist in providing students with good problem-solving skills. This research will focus on how lecturers perceive the use of problem-solving guidelines or methods to improve the problem-solving skills of students. This study will focus on lecturers who present computer programming classes at tertiary institutions.

1.7.2 Limitations

The semi-structured interviews with lecturers will be limited to two tertiary institutions and will not include all academic institutions that offer computer programming in South Africa. Only the perceptions of lecturers who offer computer programming courses will be obtained and analysed.

1.8 Ethical considerations

This research study was executed in such a manner that it complies with the ethical standards of academic research. Participants were not requested to disclose any information that might identify them or others. All information was handled confidentially, used for research purposes and in an accumulated form. Participation was completely voluntarily, and any participant could withdrew at any time. Ethical clearance was obtained from the Ethics Committee of the North West University (ECONIT-2016-004).

1.9 Structure of the dissertation

Chapter 1: Introduction

This chapter included the introduction and background to the study, problem statement, and objective of the study. This chapter also provided the research strategy, scope and limitations of the study.

Chapter 2: Literature review

Chapter 2 explores the literature and provides a detailed discussion on the fundamental and concepts of problems, problem-solving and the role of problem-solving in the teaching of computer programming.

Chapter 3: Research design and methodology

Research design, methodologies, and the data collection method used in this study is discussed in Chapter 3.

Chapter 4: Data collection and analysis

Chapter 4 provides the collection of data, analysis and interpretation of the empirical findings, and the design of guidelines for the teaching of computer programming.

Chapter 5: Conclusions, reflections and recommendations

The final chapter concludes the study with an overview, conclusions, reflections, recommendations, and suggestions for further studies.

1.10 Summary

This chapter introduced the study, discussed the background of the research, the purpose of the study, research approach, and outlined the structure of the dissertation.

2 CHAPTER TWO: LITERATURE REVIEW

2.1 Introduction

Chapter 1 highlighted that there is a high demand and a need for an adapted environment where computer science students can be kept engaged in the classroom. With programming as a primary unit in computer science curricula, appropriate pedagogical approaches should be adopted.

This chapter reviews the literature associated with problem solving, the quest towards the development of guidelines, and techniques in computer science. The chapter will begin with problem solving and is followed by a taxonomy of problem solving, as well as the teaching for and of problem solving. Many educators are concerned with developing students into good problem solvers (Hardin, 2003:1). The chapter also addresses the current practices of problem-solving in the computer programming class.

2.2 Fundamentals and concepts of problem solving

2.2.1 Problems' nature and structure

A problem is a concept that can be translated in different ways. The Merriam-Webster Dictionary (Merriman-Webster, 2017) defines a problem as something that must be solved. The development of a problem is defined by the solution from which the question is presented. Jonassen (2010:1) links a problem with social, cultural, and academic principles. For example, a mathematician sees a given problem, examines it and then resolves it with mathematical methods (Schoenfeld, 2010:337). In a lecture room, a lecturer sees a problem with specific conditions, which a student has not laid their eyes on (Lampert, 1985:180). An information technologist regards a problem by applying a set of data to develop a computer model (Hrabovsky, no date).

The attempt to solve a problem can be observed from; what is given, what fulfills the constraints, and ends with a goal (Jonassen, 2010:2). *Given* is described as data or a fact that occurs when one starts with a problem. *Goals* are the outcomes that are developed from a solved problem. However, Breuker (1994:3) disagrees that problems are not goal related. The problem concept becomes questionable if it covers the expected conditions. The third part of what constitutes a problem is the constraints, which are achieved through reaching a goal.

Problems are articulated in four principles: domain (models, guidelines, and ethics); type (combination of models, guidelines, and standards); problem-solving process (determined by the solver's understanding and interpretation of the problem type); and a solution (signifies the goal of the solver). These principles are imperative for study, with a strong focus on addressing expectation of a student when solving problems. The following section gives an overview of different types of problems.

2.2.2 Types of problems

People deal with problems and resolve them in the manner that they are presented to them (Hardin, 2003:227). These problems are known as well-defined and ill-defined problems. Following is the discussion of the types mentioned above and how they have an impact in the educational space.

2.2.2.1 Well-defined problems

Well-defined problems are problems that have goals leading to solutions with an available method that can be used to solve them (Eysenck & Keane, 2000:503). Well-defined problems can be broken into small problems (Davidson & Sternberg, 2003:4) with only one correct answer to their solution. Such problems are meant to encourage, for example, programming students to learn, understand syntax, and code the program (Mendonça, de Oliveira, Guerrero, & Costa, 2009:1).

The process of solving well-defined problems begins with a solver attempting to solve the given problem (Jonassen, 1997:12). The solver must first try to understand the problem statement and search for solutions. In case the solver fails to implement the solution, the problem must be re-defined and different methods should be applied. Jonassen (1997:78) recommends that students should be guided by different processes to improve their problem-solving abilities. Comparative discussions on ill-defined problems follow next.

2.2.2.2 Ill-defined problems

Students experience ill-defined problems on a regular basis (Park & Jang, 2010:32; Yampinij & Chaijaroen, 2010), where these problems are perceived to be complicated (Hong, 1998:12). The presentation of ill-defined problems in classrooms, such as case studies and scenarios, (Peter, 2012:41) have goals that are not clear and have information that is not complete (Park & Jang, 2010:28). The latter allows a student to have a different view and understanding of the nature of the problem. The solutions to the ill-defined problems that the students present may be accepted even though the solutions are not correct. The students' reflection on the ill-defined problems becomes different; they become flexible, they concentrate, memorize, and have an understanding of what ill-defined problems are (Park & Jang, 2010:28). Schorr and Amit (2005:137) also discovered that this reflection helps students to review their ideas in the context of the problem, and therefore, improve and study their solutions carefully.

Mendonça *et al.* (2009:1) emphasize that the adoption of ill-defined problems into programming requires “non-trivial” skills for students to use beyond coding a program. The teaching strategies for ill-defined problems should be different from the traditional ones. The latter makes it easier for programming students to engage in any difficulties they may encounter when new strategies are adopted (Mendonça *et al.*, 2009:1). Researchers have this perception that the two problem types (well-defined and ill-defined problems) are not different to each other, but the difficulty might be in choosing the appropriate methodology and technology to use in solving them (Le, Loll, & Pinkwart, 2013:258).

2.3 Critical and creative thinking

Critical and creative skills are metacognitive skills fundamental in education. A consensus was reached that curricula should be reviewed to assist students in thinking thoroughly and to think for themselves (Pithers & Soden, 2000:238; Swartz & Perkins, 2016). Thinking of the highest-level targets both skills. The ability to have both skills proves the kind of intelligence one can possess. However, this is a gray area in education (Thuraisingam *et al.*, 2014:137). Each type will be discussed separately along with their respective challenges and limitations.

2.3.1 Critical thinking

According to Duron et al. (2006:160) critical thinking has many definitions. For his own understanding, critical thinking is the “ability to analyse and evaluate information.” Paul and Elder (2004:2) refer to critical thinking as the skill of examining and evaluating thinking with an understanding to improve it.

Moreover, Peter (2012:41) stresses that students are not born with the ability to think critically. They are not taught to reason, and they cannot take charge of their thinking (Snyder & Snyder, 2008:93). For the same reason, this makes students become inquisitive. Critical thinkers ask essential questions and problems; know how to express the problems; collect and evaluate related information; and think open-minded (Duron, Limbach, & Waugh, 2006:160).

Critical thinking is continuously turning into a challenge in teaching and learning for many educators (Broadbear, 2012:2). There are no clear directions for lecturers on how they should help students to develop such skills (Pithers & Soden, 2000:239). The lack of clarity on the nature of critical thinking creates confusion, and the blame is shifted to the teaching methods and strategies for problem solving (Pithers & Soden, 2000:239). The majority of the educational programmes do not emphasize and encourage good thinking, where students can criticize and analyze other features of thinking (Pithers & Soden, 2000:245). One of the recommended teaching methods that fosters critical thinking and inspires students about the content of the course, is problem-based learning (§ 2.6.10).

Once critical thinking is incorporated and promoted, students are competent to engage in higher order thinking and be responsible for their performance in their assessments. Teaching critical thinking to a knowledgeable and experienced critical thinker channels the thinker to do introspection. Self-assessment or introspection is a skill developed in the real world. The idea of this development is, therefore, valuable in creating intellectual traits (such as humility, courage, empathy, perseverance, intellectual, faith in reason, and fair-mindedness) that contributes to the knowledge and dimensions of solving future problems.

2.3.2 Obstacles of critical thinking

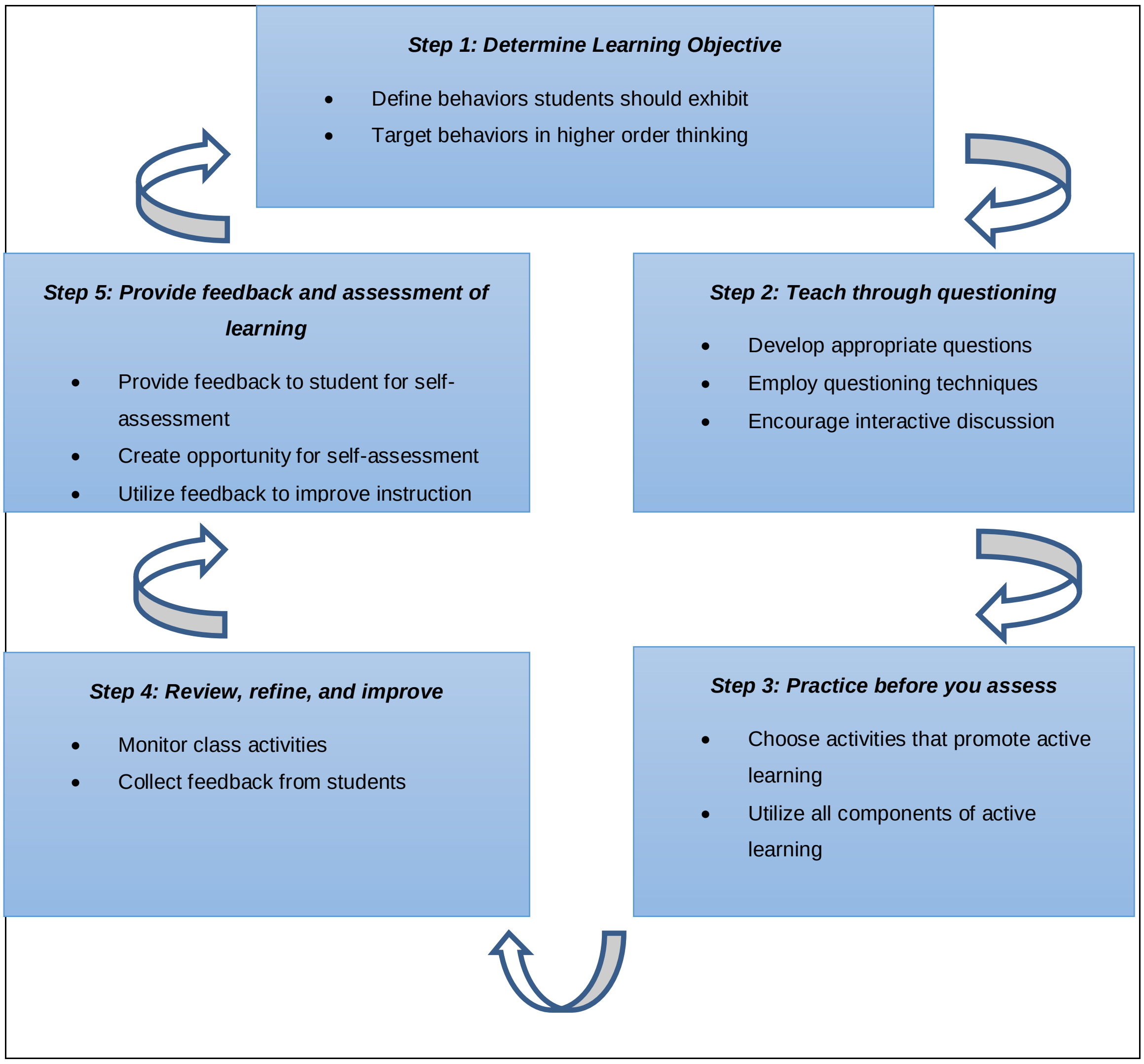
There are obstacles to critical thinking that complicate teaching. These obstacles are found in areas such as training, information, biased preconceptions, and time constraints (Snyder & Snyder, 2008:92). This enormous impact is evident if lecturers do not receive sufficient and relevant training in critical thinking, which affects both the students and the lecturer (Broadbear, 2012; Scriven & Paul, 2007). If training is not received, students are not open-minded and not curious about the content.

Another mentioned barrier is time. Time forces lecturers to present their course material in a limited time or short period. Hence, lecturers are forced lecture, which is the fastest and easiest teaching style. This study supports the notion that lecturing is not the best method of instruction (Broadbear, 2012; Brodie & Irving, 2007). It is crucial to monitor students through the critical thinking process – the discussion of the latter follows in the next paragraph.

2.3.3 Monitoring students through the critical thinking process

It might be problematic for students to participate in active learning that requires critical thinking skills. Active learning creates an exciting environment for both students and lecturers (Duron et al., 2006:160). For this to transpire, students should not be accustomed to learning by rote (Snyder & Snyder, 2008:96). Duron et al. (2006:160) agree that students gain some experience by learning by rote. Students are likely to understand what they are doing until they actively engage in the learning process. This will allow and direct the lecturers to develop and create a learning environment where students feel relaxed when they think about, analyze, and devise a solution.

It is the responsibility of the students to be analytical, and change the mindset from being recipients of information to users of information (Snyder and Snyder (2008:97). In other words, the shift from the lecturers' mindset of how students receive information can assist lecturers to support and develop critical skills. Duron *et al.* (2006) assist the lecturers in developing a model that might be useful and effective in a method that can uplift and increase students' critical thinking skills. Demonstrated in a 5 Step Model in Figure 1.



Step 1: Determine Learning Objective
Define behaviors students should exhibit
Target behaviors in higher order thinking
Step 2: Teach through questioning
Develop appropriate questions
Employ questioning techniques
Encourage interactive discussion
Step 3: Practice before you assess
Choose activities that promote active learning
Utilize all components of active learning
Step 4: Review, refine, and improve
Monitor class activities
Collect feedback from students
Step 5: Provide feedback and assessment of learning
Provide feedback to student for self-assessment
Create opportunity for self-assessment
Utilize feedback to improve instruction

Figure 1: 5-Step model to move students towards critical thinking*

**Source: Adapted from Duron et al. (2006)*

From Figure 1, the steps are summarised as follows (Duron *et al.*, 2006):

Step 1: Determine learning objectives. The learning objectives of each lesson can be identified in lesson plans where the performances of a student are demonstrated at the end of each lesson. A well-written and structured lesson plan should be aimed at a particular behavior (Duron *et al.*, 2006).

Step 2: Teaching through questioning. The questioning technique as part of teaching and learning promotes students' thinking skills. These techniques allow the lecturer to ask what is known to the student and expand to the newly acquired knowledge. Students' level of thinking should be related to the questions asked (Duron *et al.*, 2006). When students fail to ask and seek questions, they automatically consider the lesson or content as not important (Paul & Elder, 2004). It is recommended that during the planning of each lesson, lecturers must think of the purpose of each question (to be presented) and what they want to accomplish. This will accelerate students' learning, participation, and engagement in critical thinking.

Step 3: Practice before you assess. Active learning has been advocated as an area were lecturers should actively involve students to learn and retain knowledge (Duron *et al.* (2006). Active learning is a method that involves students during their learning process (Prince, 2004). More activities that support active learning are encouraged to be included in the learning process such as dialogs, ideas and information, and experiences.

Step 4: Review, refine, and improve. Every lesson should be continuously and collectively reviewed with the relevant teaching strategies to support and develop students' critical thinking skills (Duron *et al.*, 2006). Students' attendance, participation, and feedback are practically helpful in revising the lessons. Various tools and methods can be used to facilitate and ensure students' attendance, and feedback is a success.

Step 5: Provide feedback and assessment of learning. The purpose of providing feedback improves the value of students' learning and performance. Feedback creates a channel of communication for both student and lecturer where they can engage in what works and what does not work. Different platforms, methods, and tools can be used to provide feedback. Such as online, consultations, groups, peers, and many others. Eventually, the feedback will assist the lecturer in improving and modifying his/her lessons if necessary.

The adoption of Duron et al.'s (2006) framework calls for full participation and commitment from both students and lecturers. However, there are identified limitations to Duron et al.'s (2006) framework, which can be overcome by planning and being creative. Limitations such as class size and time might discourage lecturers to motivate and foster critical thinking in students. In this framework, Duron et al. recommend that the content can be adjusted throughout the lectures by integrating active learning techniques (Duron *et al.*, 2006:6). Such adjustments can benefit and help students engage without putting in much energy when they stumble across new and challenging situations that test their thinking modes (Kong, 2014:161).

Duron *et al.* (2006) acknowledge that the framework can be applied in any discipline to promote critical thinking skills. A shift in teaching strategies and techniques should be acknowledged, especially from traditional lecture-based methods to other instructional methods (§ 2.6.1-2.6.12).

2.3.4 Concluding remarks on the critical thinking process

Measuring the impact of critical thinking can thus be evaluated in a particular learning context and cannot be generalized (Brookhart & Nitko, 2011:236). According to Paul and Elder (2004:34), those who teach students to reflect within the logic of the subject must focus on these two areas. Firstly, educators must be clear what critical thinking is, and secondly, consider simplifying the learning of critical thinking for students to understand.

Ultimately, students will learn and be able to reflect on the meaning of what they are doing (Duron *et al.*, 2006:1; Ikayanti, Suratno, & Wahyuni, 2017).

2.3.5 Creative thinking

Creativity is an elusive idea, and differently understood within different disciplines (Ayoufu, Afshari, & Ghavifekr, 2012:2). One definition is that creativity is a method for detecting a problem, looking for solutions, and conveying solutions to others (Torrance, 1969). Torrance's definition is criticized and is regarded as old-fashioned, but it is significant and adds value. Sternberg and Lubart (1995) define creativity as creating new and unique ideas. The term '*new*' means original, unique, unusual, varied, and breaking from existing patterns (Ayoufu *et al.*, 2012:2). New ideas do not exist, but they are merely the reconfiguration of current ideas (Ayoufu *et al.*, 2012; Razeghi, 2008:3)

Craft (2001:13) divides creativity into two categories. *High creativity* is the ability of an individual to produce new ideas, reconstruct, and design something that is acknowledged by specialists or experts. *Ordinary creativity* is when average people think in familiar ways when they encounter real-world problems. The theory of creativity does not restrict individuals who are not extraordinary in demonstrating their talent in any field. Ordinary creativity supports the student's abilities by forming the best possible learning situations (Craft, 2001:13).

The teaching and learning environment should be structured in a manner that students can develop their innovations using applicable teaching methods and strategies (Ersoy, 2014:1). Cheng (2010:1) established three approaches that foster creative thinking into the curriculum. The methods include creative thinking through the lesson's content, processes, and scenarios. In the lesson's *content,* the usages of analogies add value and guide an individual to promote new ideas and imaginations (Cheng, 2010:4). In the lesson's *process,* educators must integrate teaching creative thinking in selected open-inquiry processes. Open-inquiry is the primary method to promote creativity in education – predominantly science (Craft, 2001). Lastly, from the *scenario's* approach, educators must incorporate creative problem-solving tasks aiming to present an opportunity to students to deal with open-ended problems and creative solutions. Craft's study concluded with lessons being restructured with the aim of teaching thinking skills and processes.

Cheng (2010:18) cannot recommend which approach is best to implement as each of the approaches have their own limitations and constraints such as original "content-curriculum, time constraints, student interests and abilities, and the discrepancies between student and teacher expectations" Cheng (2010:18). Cheng (2010) expresses that educators must develop teaching methods and learning activities for every approach and observe their strengths. The methods can be adopted and incorporated but not taught as a separate subject.

The lack of teaching methods to promote creative thinking discourage lecturers (Ryhammar & Brolin, 1999:266, 296). The latter calls for universities to transform and refine the curriculum to challenge students to be artistic. This transformation will enable students to understand, investigate, and apply knowledge to the new circumstances (Awang & Ramly, 2008; Ayoufu *et al.*, 2012; Newton, 2011).

Mayer (2008) proposes four issues for thinking skills programs: 1) what to teach (e.g., thinking as one ability); 2) how to teach (e.g., concentrating on the process); 3) where to teach (e.g., domain-independent course, or particular course); and 4) when to teach (e.g., after or before core competencies are learned). Mayer’s (2008) study proved that teaching thinking skills could be advantageous when:

- the syllabus concentrates on one or more skills, like decoding a problem into sections;
- the teaching method concentrates on problem-solving processes, like having specialists demonstrating problem-solving steps;
- there is an expectation to address the problems in a similar domain of instruction; and
- skills are demonstrated before students have programmed the primary ability.

Mayer’s (2008) study proved that before students could solve problems, they were taught to “think aloud.” In the process, students were forced to listen to other good solvers during the disentanglement of the problems. Students were given a chance to describe their methods then later document the differences in their methods and the good solvers’ methods. The results were compared to the students who did not receive the training; the outcome of the study shows that students who thought aloud were more successful in their problem-solving abilities.

In conclusion, critical and creative thinking skills are valuable skills, and neither is superior to the other. The development of both skills is interdependent and promote student-centered learning. The above discussion confirms that if one skill is overlooked in the process, problem solving does not become effective. The combination of the two skills might intimidate students, but these skills will gradually develop over time.

2.4 Learning styles

Research has shown that lecturers must consider students' learning styles and realign their teaching and learning strategies (Hwang *et al.*, 2012:626). Learning styles are techniques used for people to remember, learn, and use information (Franzoni, Assar, Defude, & Rojas, 2008:779; Tie & Umar, 2010). One individual can understand the information differently, while another learns by seeing and listening; reflecting and acting; thinking and intuitively remembering and visualizing; and sketching and building models (Franzoni *et al.*, 2008:779; Schmeck, 2013). When the teaching style of an educator is not compatible with the learning style of a student, the students becomes discouraged. The student becomes bored, irritated, disengaged in class, fails tests, and ultimately, drops the course (Felder & Spurlin, 2005:103).

The more the subject becomes complex, the harder the learning process is for the learner (Jenkins & Keefe, 2001:73). The process of complex learning is developed over a period. (Jenkins & Keefe, 2001) further stress that the way lecturers present the content, impacts student learning and highlight that lecturers mistakenly present their preferred teaching style, trusting that is what their students prefer. This forces students to learn in a particular style, which is expected of them (Jenkins & Keefe, 2001:75)

Felder and Silverman (1988:674) developed a model that presents learning styles to articulate the teaching approach to address the needs of engineering students. The model classifies students in five dimensions as shown in Table 1.

Table 1: Dimensions of Learning and Teaching Styles*

Preferred Learning Style		**Corresponding Teaching Style**	
Sensory / Intuitive	*Perception*	Concrete / Abstract	*Content*
Visual / Auditory	*Input*	Visual / Verbal	*Presentation*
Inductive / Deductive	*Organization*	Inductive / Deductive	*Organization*
Active / Reflective	*Processing*	Active / Passive	*Student participation*
Sequential / Global	*Understanding*	Sequential / Global	*Perspective*

**Source:* *Adapted from* Felder and Silverman (1988)

2.4.1 Sensing and intuitive learners

The term sensing is observing, collecting data through senses. Sensing learners would rather have information based on facts. Although these learners are cautious and sluggish, they are good at remembering information at a slow pace. They prefer to solve problems with methods or algorithms. Intuition is to understand instantly (Merriam-Webster Dictionary, 2015). Intuitors are better performers in comparison to sensors. They like concepts and are good with new theories. They are creative but dislike repetition; they become bored with details but like problems (Felder & Silverman, 1988:676). Both sensors and intuitors have their strengths and weakness. To be able to reach both learners, educators should blend two teaching styles called concrete (facts, data) and abstract (principles, theories).

2.4.2 Visual and auditory learners

Information is received in several ways, referred to as modalities; visual (images, diagrams); auditory (sounds, words); and kinesthetic (touching, smelling). Majority of learners learn using one or more modality. A visual learner understands and learns when he sees diagrams, charts, or films. Traditional lectures are one of the suitable teaching methods for these learners because of the nature of their learning (Deek & Espinosa, 2005:413). Should it happen that a different way is used, the learner will, most likely, forget. Auditory learners learn from what they hear using discussions. Felder and Silverman (1988:678) remark that in colleges, the teaching styles used are mostly verbal (lecturing) or visual (words) which does not match other learner’s preferred learning style. To accommodate both learners, educators ought to modify what they present by using visual material and live demonstrations.

2.4.3 Inductive and deductive learners

Induction is defined as making broad generalizations from specific observations. Inductive learners are motivated by what they see before they understand and appreciate the theory. Deductive starts with a general statement and progresses to examining the options to reach a specific, logical conclusion. Deductive presentations often mislead learners. Learners think that their educator formulates a perfect explanation of a complicated method. This perception makes a student to have perceptions and doubt themselves concerning their abilities. To reach both inductive and deductive learners, educators must follow the scientific method by using induction (presenting theoretical material) followed by deduction (rules or principles that describe the observed phenomena) (Felder & Silverman, 1988:676).

2.4.4 Active and reflective learners

An active learner feels comfortable with ongoing experiments, beyond listening and watching. Active experiments are described as “doing something in the external world with the information or testing it in some way” (Felder & Silverman, 1988:678). Active learners prefer working in groups, evaluate, design ideas and carry out experiments. Reflective observation is manipulating and checking information. Reflective learners prefer to work alone, like theories and define problems including proposing solutions. To accommodate both active and reflective learners, educators should alternate lectures by pausing to give reflectors an opportunity to reflect and give active learners the chance for brief discussion (splitting them into groups to have a discussion).

2.4.5 Sequential and global learners

A sequential learner absorbs the content presented step-by-step. These students work their way to the solution of the problem one step at a time. Global learners understand the overall picture of the content but are fuzzy on the details. At times, global learners see the solution, however, struggle to figure out the steps to get to the solution. To reach both sequential and global learners, educators are advised to explain the goal of the lesson before presentation. It gives learners an opportunity to devise their methods to problem solve. Any technique used by educators should be sufficient to meet the needs of all mentioned learners. This could be overwhelming to some educators to accommodate all learners. Learners are therefore encouraged to adopt a style that is suitable to the subject learned (Jenkins, 2002:72).

2.5 Teaching for problem solving

Problem solving is transferred through knowledge in many ways. The transfer is rare (Rebello *et al.*, 2007:1) and not much research could be found (Eseryel *et al.*, 2014:1; Frerejean, van Strien, Kirschner, & Brand-Gruwel, 2016). Many students have challenges in identifying the relationship between learning context and transfer context (Rebello *et al.*, 2007:1). These challenges limit students to solve problems, regardless of the teachings they receive.

Teaching problem solving is criticized by many educators (Jonassen, 2010:243). One critic draws attention to objective tests (such as multiple-choice questions) which promote learning by rote. To strengthen the principles and teachings of problem solving, educators need to develop methods to produce good solvers who can face any problems. A successful, yet frustrating way to good problem solvers is reported by Lewis (1991). The method involves two students; one playing a solver and the other the receiver. The solver reads a problem to the receiver. The problem is read out loud until the solver resolves the problem. The receiver is only permitted to identify the errors. However, the receiver must not mention the correct answers (Whimbey, Lochhead, & Narode, 2013). The frustration of a receiver not able to indicate the answers slows down this process for the other student. Lewis (1991) reveals that a poor problem solver read the problem and then stopped talking; this posed a challenge to both solvers (poor and good solvers). A good solver remembers the construction of the problem while a poor solver only remembers some aspects of the problem statement. Problem solvers require the skill to detect the structure when the problem solution is presented. Good solvers are expected to access information from the first occurrence of problem solving. Sutton (2003) explains that good solvers accesses the solution of the problem apart from the directed problem. Lewis's method does not only teach both solvers how to problem solve but teaches them how to use their skills and express themselves effectively.

The contribution of teaching problem solving to the sciences, advocates on the process of solving problems by looking at three factors: how the problem is represented; the solver's experiences; and the understanding of a problem (Sutton, 2003).

Representation of the problem symbolizes how the solver can best solve or process the presented problem statement. The solver’s intellectual interpretation will be dependent on their background experience and previous knowledge (Sutton, 2003).

From the solver's experiences, a solver is given a problem statement and unpacks in a way it can be understood. Not all solvers can succeed in unpacking the problem statement (Sutton, 2003).

Understanding of the problem is the combination of the experience of a solver and a problem representation. Based on a solver’s experiences, a solver can create an intellectual interpretation using different views of a problem. The focal point of the process symbolizes the learning transfer. The transfer begins when a solver understands the problem statement – its fundamental structure. Gomes and Mendes (2007:2) admit that students often misinterpret problem statements, while others do not read and become anxious to write the solution.

The conceptualization of how to teach problem solving and utilizing suitable problem solving methods, is proposed by Malouff (2011) to prepare students at any level. Ismail, Ngah, and Umar (2010:126) point out that students must understand how to interpret a problem statement before they can use dedicated tools and giving solutions. Figure 2 illustrates these steps (Malouff, 2011:3).

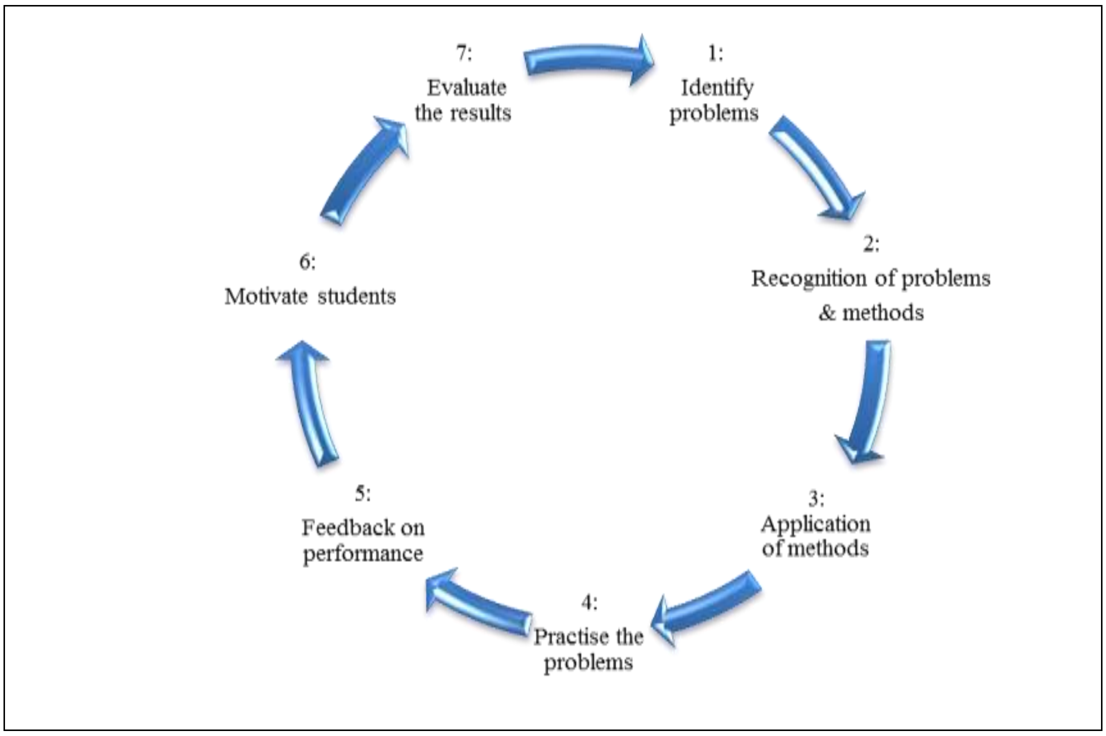


Figure 2: Steps in teaching students how to solve problems

Source: Adapted from Malouff (2011)

The steps from Figure 2 are explained as follows:

- Deciding on the **types of problems and problem-solving methods** is influenced by what problems are to be covered. This is based on the discretion of the instructor.

- The **appropriate methods of different types of methods** provide an instructor with varying ways of presenting the problem-solving including textbooks, supplementary readings, and videos.

- The **application of methods** demonstrates to students the value of problem solving in action using videos, presentations, and textbooks, which might provide models of problems solving. The techniques shown through these different platforms are ways to engage students in identifying positive and negative elements used by professionals.

- During **practicing of problems,** students can be formed into a group while they do problem solving. Malouff (2011:3) echoes that the more students practice, the more they learn. As soon as students are given realistic problems to practice, it becomes easier for them to generalize and relate to real-life problems.

- Students can obtain **feedback** on their work done or their knowledge of problem-solving in many ways. The feedback can be done online, in classrooms, by asking questions, written feedback, getting their marks on assessment and oral feedback by the instructor.

- **Various motivational methods** can be used for supporting students' learning such as one-one consultation, group tasks, and role-playing.

- The **evaluations of results** are subject to the assessment methods conducted on problem solving. Various assessment methods can be used to test the impact of teaching problem solving. Graded assessment can form part of the process and observe the success of certain types of problems achieved. Comments written on the assessment sheet can motivate and bring attention to students on the significance of what they have learned.

As seen in Figure 2, steps can be applied to help produce students who can problem solve. The emphasis in teaching problem solving in general is to understand how to develop questions; present problem solving as a problem; and use various problem-solving methods (Malouff, 2011:3). This will improve students' confidence and competence in problem-solving skills (Malouff, 2011; Thompson, Barnes, & Fincher, 1997).

2.6 Approaches to solve problems

Programming is a form of art that calls for a student to have an ability to interpret problems into solutions (Sarpong, Arthur, and Amoako (2013:27). Those individuals who learn this form of art should possess the skill to problem solve. The language promotes problem solving through top-down and bottom-up approaches. Top-down refers to a sophisticated program divided into smaller pieces. Making the program to be efficient and easy to understand. The bottom-up approach focuses on the syntax and distinct programming language (Mohorovičić & Strčić, 2011:3).

The teaching methods in other courses can be structured, but in programming it becomes different and complicated (Mohorovičić & Strčić, 2011:2). Hence, lecturers are found applying their preferred blended methods. The blended teaching method is referred to implementing more than one teaching method or strategy (Sadeghi, SEDAGHAT, & AHMADI, 2014:146). A comparison study by Vihavainen, Airaksinen, and Watson (2014:19) proved that those who moved from traditional lectures and lab-based approached, leaned towards pair programming (§ 2.6.6), game-based learning (§ 2.6.12), and extreme apprenticeship.

One highlighted difficulty students face is the application of knowledge. Ismail *et al.* (2010:126) are convinced that there are possibilities that students understand concepts, except that they might not be able to apply them. They may understand how to solve the problem manually but have difficulties in translating the problem into a computer program, thus making it challenging for lecturers to teach programming. The latter are stated to have a negative impact on the instructional process. For example, “lack of skills in analyzing problems, ineffective use of problem representation techniques for problem solving, inefficient use of teaching strategies for problem-solving and coding and difficulty in mastering programming syntaxes and functions“ Ismail *et al.* (2010:126). Ismail *et al.* (2010:126) recommends alternative teaching strategies be investigated.

Sarpong *et al.* (2013:27) reported on eight teaching methods and strategies and conclude that students prefer more than one approach during their learning. The study was undertaken at the Valley View University (VVU) in Ghana. Table 2 lists teaching methods and strategies discussed in the next section.

Table 2: Teaching methods and strategies*

Method	**Strategy**
Lectures	Explicit teaching
Laboratory	Command style teaching
Projects	Teaching by task
e-Learning	Problem-solving teaching
Seminars and tutorials	Pre-recorded lectures
Field trips	Puzzled-based learning
Continuous assessment and examinations	Pair/group programming
Problem-based teaching	Peer tutoring

**Source: Adapted from Sarpong et al. (2013)*

2.6.1 Explicit teaching

Explicit teaching provides students with clear instructions, affirming learning goals before the lesson (Edwards-Groves, 2003). The content is then fragmented into small portions and taught separately. This will include detail explanation (what to do), presentation (how to do), and exercises (practice). This process guides a student until independence is reached.

The content is taught in sequence by the teacher. However, not all aspects of programming can be demonstrated explicitly (Ismail *et al.*, 2010:126).

2.6.2 Command style teaching

Command style teaching places a student in a “closed” environment, positioning a lecturer as the sole authoritarian in a lecture room (Riley, Campbell, & Farrows, 2004) where students are restricted and act by the lecturer's instructions. Throughout the lesson, a lecturer introduces a learning structure, and students follow and obey all restrictions and guidelines. The style produces students who are likely to struggle with reasoning and learning by themselves. Students turn out to have low self-esteem, lack motivation and their social interaction is reduced. The impact of this style is negative, but there are some advantages to it (Riley *et al.*, 2004):

- Command style teaching gives the lecturer an upper hand to have total control of his class
- A lesson can be executed as planned by the lecturer
- The tasks are completed on time

Eventually, a lecturer is entitled to decisions while a student has a minimal contribution (Mosston & Ashworth, 2002:79). Decisions such as subject matter, location, start and end time, posture, duration and feedback are solely the lecturer's decision. (Sarpong *et al.*, 2013:31) revealed that command style is not suitable for programming, therefore, it should not be encouraged for the simple reason that the student disengages in the learning process.

2.6.3 Teaching by task

Task-based teaching embodies what the student must do, to learn as a substitute of acquiring a skill that is mastered (Ellis, 2003:20). The definition of a task in a pedagogical view by Nunan (2006:15) cited (Breen, 1987:23):

> "... *any structured language learning endeavor which has a particular objective, appropriate content, a specified working procedure, and a range of outcomes for those who undertake the task. 'Task' is therefore assumed to refer to a range of work plans which have the overall purposes of facilitating language learning – from the simple and brief exercise type to more complex and lengthy activities such as group problem-solving or simulations and decision-making.*"

Ellis (2003:20) developed a model that involves both student and the lecturer. Both have their roles to play in the strategy, summarized next.

Role of a student: The student takes control and become accountable for any given task and their performance (Oxford, 2006:108). They are perceived to be task-analyzers where they must analyze a task with its requirements. Therefore, this compels a student to be dedicated, driven, confident, and keen to take risks (Oxford, 2006:108). The style a student chooses impacts immensely when selecting suitable strategies to complete a task. The approach will allow a student to be accountable for their learning in various ways. It also helps students to develop at their own pace (Riley *et al.*, 2004).

Role of a lecturer: Lecturers play the role of a task-preparer, selector, strategy instructor, a monitor, and a provider of assistance (Oxford, 2006:108). The lecturer will offer support to students only when needed during the task. At times, a lecturer might not clarify the student's error, as he/she provides indirect assistance to help the student solve the problem without any help.

2.6.4 Problem-solving teaching

Like command style, the lecturer makes all decisions about the content, and formulates questions and answers whereas students decide on the sequence leading to the correct answers (Mosston & Ashworth, 2002:237). The lecturer facilitates the lesson by providing students with feedback rather than correct answers turning the lecturer into a facilitator (Riley *et al.*, 2004). The limitation of this strategy is when students are not able to get correct answers, and consequently, their motivation levels decrease. Teaching this style can take a long time because students need adequate time to learn in an environment where they can work out correct solutions (Harrison & Blakemore, 1989:329).

Lecturers must be fully prepared for the each lesson to achieve the outcome desired.

2.6.5 Pre-recorded lectures

Pre-recorded lectures allow students to understand programming concepts to a satisfactory level (Mohorovičić & Strčić, 2011:4). The method supplements traditional teaching methods calling for lecturers to record and keep their notes on a multimedia platform. This includes well-prepared lecture notes, videos, slides, assignments, and much more.

This strategy bridges a gap for students who are challenged by some issues such as failing to engage and attend classes (Mohorovičić & Strčić, 2011:4). Such methods help students who did not understand some concepts and those who skipped classes can watch the recorded material.

2.6.6 Pair/group programming

Pair programming represents two programmers working with each other on the same program or statement to achieve the results. The collaboration between the two programmers is seen as one taking the leading role of a “driver” and one of a “navigator” (Teague, 2011:41). The driver writes the codes, while the navigator looks out and checks for errors. A conversation will take place while the two programmers are playing these roles to clarify ideas. Roles will be swapped during programming to benefit both individuals.

The disadvantage of the method is a problem when both programmers do not think alike, affecting their collaboration (Shirsath, 2014:117). The method can be irritating and exhausting for both programmers (Mohorovičić & Strčić, 2011:4). Kafilongo (2016:98) proposes that students must be taught how to use pair-programming. (Kafilongo, 2016) concludes that pair-programming is not about sharing work equally, but each student should concentrate on their dedicated work. His study recommends the following should be taught to students when paired:

- Working together at the same workstation.
- Switch roles occasionally.
- Each student must actively participate.
- Discussions and collaboration between the pairs are advised.

Students who are paired show fewer signs of stress and anxiety. They become eager to learn and have a positive attitude towards programming (Kafilongo, 2016:107).

2.6.7 Peer tutoring

Peer tutoring is collaborative learning supported inside and outside the classroom by students teaching other students (Colvin, 2007:165). One student plays the role of a tutor (teacher or expert) guiding those who play the role of a tutee (learner or novice) (Carberry, 2008). The strategy does not only give educational support but closes the social gap for those students who struggle in both aspects (Baleni, Malatji, & Wadesango, 2016:127). Full participation and being hands-on is vital from a tutee to understand the subject content. Baleni *et al.* (2016:127) explain that the weakness of peer tutoring for students who do not attend tutorial programs regularly or when scheduled are likely to lose out on useful information.

2.6.8 Concluding remarks on teaching methods

Sarpong *et al.* (2013) make assumptions that the teaching strategies lecturers choose may confuse students and may result in failure. For example; if the lecturing method is the only method adopted, the course turns out to be challenging. It is the responsibility of lecturers to select an appropriate method that suits their teaching environment. Sarpong *et al.* (2013:32) recommend that lecturers should adapt peer tutoring, pair programming and problem solving to provide a better interaction between students and lecturers. Not only will these methods encourage students but will motivate them to become skilled programmers.

The next section discusses the Dual Common Model (DCM) which incorporates problem solving built on existing methodologies of other researchers.

2.6.9 Dual Common Model for problem solving

The Dual Common Model (DCM) developed by Deek and McHugh (2003:266), serves as a foundation for problem solving and was customized to the area of program development. The purpose was to take the qualities of problem-solving methods and to provide a framework to define the problem, plan how to solve the problem, design and implement the plan, and to verify the results.

The model addresses the challenge of writing an algorithm, which is beyond learning the syntax of any programming languages. A valuable lesson from the study is to reduce a complex system to individual parts through decomposition and debugging exercises. When possible, solutions are detected, and the program can be tested to check if the algorithm works. If these changes are considered, mastering a problem becomes better and easier, restoring hope to students to solve problems.

Another prominent and preferred model, known as Problem-Based Learning (PBL), is discussed next.

2.6.10 Problem-Based Learning (PBL)

Problem-Based Learning is learning through complex real-world problems and is associated with ill-defined problems (§ 2.2.2.2). According to Nuutila, Törmä, and Malmi (2005), real-world problems in the context of programming, are referred as something that might occur as a design task for a professional software engineer. As the approach supports student's engagement learning, analytical thinking, general thinking, creativity, and cultural awareness, it is effective in teaching and learning.

Kay *et al.* (2000:109) implemented PBL for investigating the challenges faced by computer science students and lecturers such as huge classes with administrative loads for lecturers. The impact of these challenges caused negativity within students' experiences leading to the course to be redesigned. The redesigning of the course was notable with students showing a positive attitude and felt optimistic about the improved learning method. Another documented benefit was a commitment of students, where the drop-out rate showed a significant decline. Group work was also highlighted and proved that students learned how to plan their job. Communication skills and reasoning skills were among other benefits.

Another model designed for problem-solving methods that benefit programming is Puzzle-Based Learning (PZBL).

2.6.11 Puzzle-Based Learning (PZBL)

The use of puzzles has been a way of making programming courses more attractive. Puzzle-Based Learning (PZBL) is developed to inspire computer science students to structure and answer problems that are not found at the end of the textbook (Falkner, Sooriamurthi, & Michalewicz, 2010:20). Puzzles are used by distributing problem statements into small sections and arranging each line of code in the correct order (Shirsath, 2014:117).

PZBL development focuses on student engagement, independent learning, satisfaction, and motivation. Merrick (2010:1) proved that PZBL intensifies students’ participation and they become active while problem-solving skills are improved. Another benefit of PZBL is that it equips students with critical thinking and problem-solving skills (Mohorovičić & Strčić, 2011). Solving the puzzles in programming should be presented to students with guidance by the lecturer or an automated system. The puzzling problem is presented within the context, and the problem solution is divided into pieces of the puzzle. This will be subject to the level of difficulty. A student will challenge himself to rebuild the program by choosing the correct program pieces in the precise order (Mohorovičić & Strčić, 2011:2). Falkner *et al.* (2010:21) add a valuable point that puzzles can help to retain, motivate, and attract students into computer science programs.

2.6.12 Game-themed programming

Game-themed programming is an approach that teaches students to construct computer games. Students learn abstract programming concepts through game applications. Abstract programming is a technique used for arranging difficulties of computer systems. Mohorovičić and Strčić (2011:3) disclose that game-themed programming makes students understand games. The time spent by students on these programs inspires and improves students’ passion for programming. The incorporation of the game-themed approach into teaching, promotes learning, improves cognitive performance, and increases motivation and concentration (Hwang *et al.*, 2012:122).

2.6.13 Concluding remarks on methods used in programming

The above mentioned methods discussed prove to be more efficient, enhancing students' confidence and their ability to problem solve (Falkner *et al.*, 2010; Hwang *et al.*, 2012). These methodologies also enhance the quality of programming courses and make it easy to understand (Mohorovičić & Strčić, 2011:5). This will eliminate students' frustration and challenges they face during the course. In line with this, continuous staff development is also critical for refining teaching methods.

2.7 Guidelines used in teaching problem solving

Several guidelines or tools are available to assist students in systematically solving problems. This section will briefly introduce algorithms and is followed by a brief description of the problem-solving activity framework, flowcharts, and pseudo code.

2.7.1 The problem-solving activity framework

Govender *et al.* (2014:2) discovered approaches used for Grade 10 Information and Technology (IT) and Computer Systems (CS) teachers when presenting problem-solving activities. They discovered approaches used by teachers where problems for learners were created where the learners had to apply their knowledge in a computer program. Approaches such as "breaking down a problem into parts, algorithms, question analysis, problem analysis, analyze examples, scenarios, IPO tables (input, processing and output tables), creative programming, problem-based learning, cooperative learning, and questions and answers" (Govender *et al.*, 2014:1). Approaches were based on the teacher's intuition in which the outcome was not effective. Programming at a high school level empowers learners to be able to employ techniques and methods to solve problems. As a result, Govender *et al.* (2014:2) designed a problem-solving guideline (Table 3) as part of an intervention and giving direction to the teachers in order to benefit the learners.

Table 3: The problem-solving activities framework*

Main problem-solving activities
1. *Write down the main ideas and requirements of the problem*. • Read the problem and underline the key concepts to understand and interpret the question. • Determine what you do not understand.
2. *Represent the problem.* • Use a diagram, table, flow chart, description or any other method to indicate how you understand the problem.
3. Plan the detailed steps. • Determine the purpose of each method. • Plan the input, processing, and output. • Go back to Step 1 and check whether you are on the right track.
4. Code your planning in a programming language. • Determine which code/constructs you will use to input the data. • Which statements will you use to process or calculate the data? • Which statements will you use to display the output? • Compile the program and correct the programming errors
5. *Reflect on how well you have solved the problem*. • Use test data and ensure that the extreme cases of test data are included. • How did you choose the experimental data and extreme values? • Explain if you could correct any programming errors. • Did you use resources to support your programming process? • Are you satisfied with your solution? Explain.

**Source: Adapted from Problem-solving Activities: SANPAD Project, Govender, 2011*

Prior to the implementation of the framework, (Govender *et al.*, 2014) state that learners lacked interest in learning how to program. The framework served as a tool to influence and force learners to plan before touching a computer. Govender et al.'s study proved that learners' confidence increased, and the level of enjoyment was pleasing. The educators were motivated to use this strategy and were able to persist when challenges arose (Govender *et al.*, 2014:196).

Barnes, Fincher, and Thompson (1997:1) adopted Polya's (Polya, 2014) methods to help solve problems and address one of the most common questions in problem solving, "*Where do I begin*?". Barnes *et al.* (1997) examined students who had average grades but lacked necessary problem-solving skills. The lack of these essential skills progressed with students from level to level (e.g., from first year to second year). Barnes *et al.* (1997) model is structured in four phases: understanding, designing, writing, and reviewing. For a student to clarify what the problem should do; a computer program name or function is written. Once a program is named, the inputs (feeding some data into a computer program) and outputs (display some data on a computer program) of the program are identified; a student can then reflect and understand the problem. However, when a student fails to name the program at an early stage, this should be a concern (Barnes *et al.*, 1997:4). When designing a program, students must find ways to solve the problem. The student then reflects on what is achieved. The phases are summarized below:

Understanding the problem refers to acknowledging the problem. A student is given a problem to be solved. That problem statement must be read over and over until it is well understood (Barnes *et al.*, 1997:37; Thompson, 1997). Furinghetti and Morselli (2009:73) use signs, words, images, drawings, and patterns to make students understand the problem statement. According to Barnes *et al.* (1997) the questions can be asked to assist students in understanding the problem statement. When such questions are asked to students, writing down a computer program and drawing diagrams, can be easier. Questions such as:

- *What are the inputs and outputs?*
- *What are the requirements of the problem?*
- *Can the requirements be fulfilled?*

- *Are there any conditions on the inputs and outputs?*
- *Can the problem be divided into sections?*

During the d*esigning of a program,* there should be relationship between the inputs and outputs. Like the first phase, below are the questions students can ask:

- *Have I seen this before, maybe in a different format?*
- *Do I know any associated problem?*
- *Can I apply any function that can be suitable?*

The designing phase strengthens the link between theory and practice, mainly when assignments or classwork are used. Lecturers need to rethink and design their materials in order to elicit the best out of their students (Thompson *et al.*, 1997).

Writing a program is a struggle for a lot of programming students (Perkins *et al.*, 1986:257), specifically novice programmers who do not plan before they write their programs (Robins, Rountree, & Rountree, 2003:252). Students are encouraged to first write their ideas of a program manually before typing it. This phase assists in correcting syntax and understanding of what the program does. Again, students become motivated and gain confidence, which enables them to produce results.

The last phase *review of a program* takes place when a student is able to observe and recognize what is learned. During instruction time, students are given not enough time to reflect back in detail on their completed work. Barnes *et al.* (1997:38) believe the latter leads to students forgetting what they learned, and the value of the exercise is lost.

Barnes *et al.* (1997) approach assures vulnerable students to continue using different methods to solve difficult problems. It also equips students with tools allowing them to write complicated programs using a well-organized technique. The impact of the technique is practical and indicates that students who had problems, succeeded in passing examinations.

2.7.2 Flowcharts

A flowchart is a diagram demonstrating decisions and outcomes drawn out using shapes and arrows (Smartdraw, 2018). A flowchart provides step-by-step illustrations for a complicated situation or problems that must be solved. Every step is indicated by the connected links (lines and arrows) allowing an individual to understand the process from start to finish.

Flowcharts are applied from complicated process to the most straightforward methods such as a travelling process or the processing of an invoice. These processes use basic symbols to write a flowchart. The rounded end symbols represent the start and end of the process. The rectangular shape is used for interim steps, and arrows are used to control the flow of the flowchart.

2.7.2.1 An example of a flowchart

The flowchart illustrated in Figure 3, shows the process of ordering a burger meal in a logical flow. The method allows the cashier to process the order according to the needs of a customer until the order is complete.

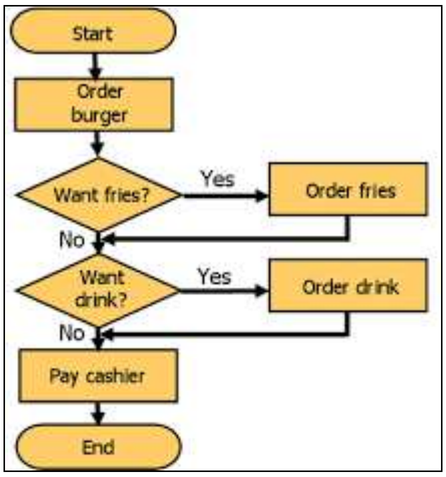


Figure 3: The logical flow of a process of ordering of a burger meal*

**Source: Adapted from (Smartdraw, 2018)*

2.7.3 Algorithms

An algorithm is a sequence of instructions followed to solve problems or to reach a goal. Cormen, Leiserson, and Rivest (2009:6) describes an algorithm as a well-defined computer procedure that accepts an input value and produces it as an output value. A typical example of where algorithms are used are food recipes. Food recipes are used to describe how to prepare a meal. In programming, a recipe is translated into a procedure; the ingredients are called inputs, and the final product will be the output.

Algorithms describe how the task must be performed; the computer performs that task in that exact order it has been outlined similar to the food recipe. Algorithms are not computer code. Algorithms are written in plain and simple English, in a way that a computer program speaks. Algorithms have a start, middle, and an end. They can be written in a list of steps using text, pictures, shapes, and arrows called flowcharts (§ 2.7.2) and with pseudo code (§ 2.7.4).

2.7.3.1 An example of an algorithm

This example is an algorithm that shows the user entering two numbers and displaying the sum.

```
Step 1: Start
Step 2: Declare variables num1, num2 and sum.
Step 3: Read values num1 and num2.
Step 4: Add num1 and num2 and assign the result to sum.
        sum←num1+num2
Step 5: Display sum
Step 6: Stop
```

The example has been numbered using plain English, expressing what the procedure must do.

In Step 1, shows that the procedure has a start. Algorithms have a start, middle and the end. The first step must be labelled ‘Start’ and the last step must have the ‘End’.

In Step 2, the variables are declared where the numbers must be stored in the computer.

In Step 3, the variables are read and validated by the computer.

In Step 4, the two values are added together.

In Step 5, the sum of the two values are displayed.

In Step 6, the procedure is ended.

2.7.4 Pseudo code

Pseudo code is a clear, descriptive, and the simplest way of writing programming code in English. It is an artificial and casual language that assists to develop algorithms. Pseudo code uses small phrases to write a computer code for programs before it switches over to a specific programming language.

2.7.4.1 Understanding pseudo code

Pseudo code is not an actual program code, but summaries what a program must do step by step. Hence, there is no syntax that is prescribed to be used for pseudo code. It is recommended that it is good practice to use pseudo code structures that can be understood by other programmers.

Programmers use pseudo code to write a rough draft of a program to understand the requirements and align the program code. This will serve as a guide and a tool for programmers to think around the problems and ideas for a specific program.

2.7.4.2 An example of pseudo code

A simple example of a standard process of ordering a burger using a pseudo code is as follows:

```
Approach counter
Order burger
If you want fries, Then
        Order Fries
Else, go to the next step
If you want drink, Then
        Order Drink
Else, go to the next step
Pay cashier
```

The advantage to using pseudo code is that it can be written and be revised at a later stage. This mechanism can assist programmers to think through their problem one step at a time.

2.8 Summary

This chapter outlined the difficulties that both the student and lecturer face when problem solving. The need for approaches to teach programming has been observed as crucial to teaching and learning. Following this, it is imperative that lecturers understand the needs and learning styles of every student to address and enhance their learning. Therefore, several methods and approaches for teaching and learning programming discussed in this chapter can be adapted and implemented.

3 CHAPTER THREE: RESEARCH DESIGN AND METHODOLOGY

3.1 Introduction

Any research study should clearly formulate the essence of the research design and methodology that were followed. According to Mason (2002:13), a clearly formulated set of research objectives and questions form an intellectual puzzle that has to be solved. This chapter explains the root of the research question supported by the objectives of this study as the intellectual puzzle that needs to be addressed. The research philosophy, approach, methods, strategies, participant selection, and instruments used to collect data is inter alia linked to the research objectives, which are substantiated by the application of the research process. The purpose of this research was to determine lecturers' perspectives of using problem-solving guidelines during the teaching of computer programming; therefore, the research question aim to address:

How do lecturers perceive the use of problem-solving guidelines during the teaching of computer programming?

To determine lecturers' perspectives of using problem-solving guidelines during the teaching of computer programming, trustworthy qualitative data were obtained, analyzed, and described.

3.2 Theoretical framework for the research and design methodology

The research design is a strategic framework which identifies actions that bridge the gap between research questions and execution, as well as the implementation of the research (Babbie, 2015; Blanche, Durrheim, & Painter, 2006:195; Burns & Grove, 2010). Parahoo (2014:142) describes research design as a logical process or strategy describing how the study should be conducted, how data should be collected, and how the data should be analyzed. Sauders, Lewis, and Thornhill (2003) provide a framework that guides the researcher to choose and incorporate different elements of the study. Figure 4 represents the research onion framework (Sauders *et al.*, 2003) adopted by starting from the outer layer into the inner layer.

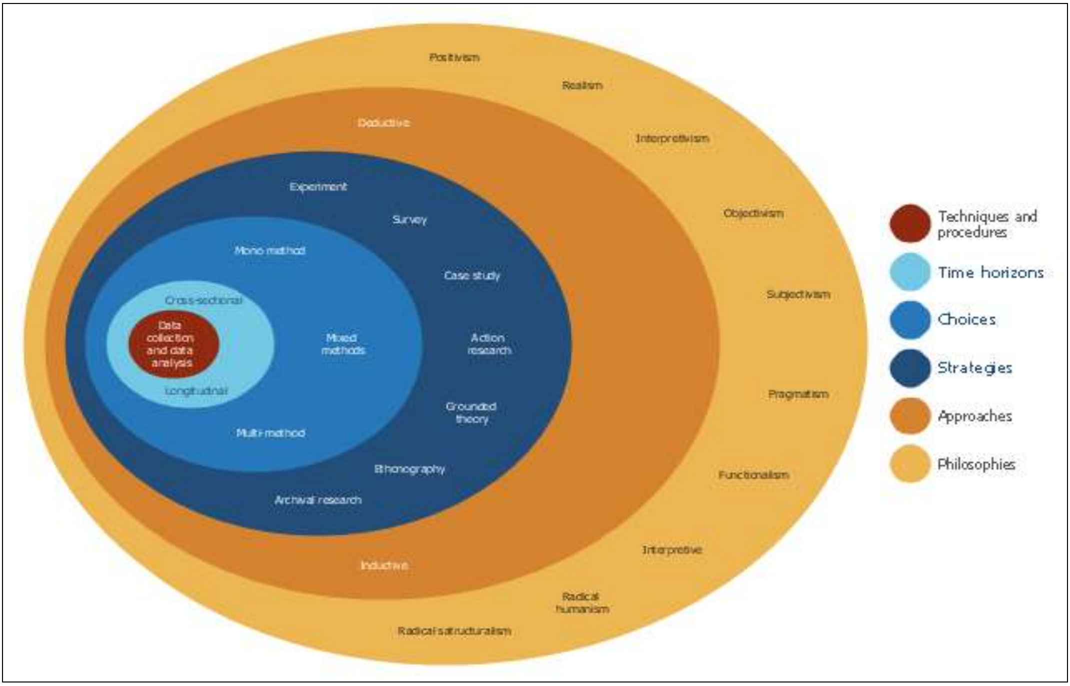


Figure 4: Research onion framework **(Sauders *et al.*, 2003)**

According to Patton et al. (2016), the research process defines the research strategy in detail. The research process aligns three dimensions, namely ontology, epistemology, and methodology (Blanche *et al.*, 2006:6; Patton, Renn, Guide, & Quaye, 2016). The study should follow a research paradigm, which is an all-inclusive system of interrelated thinking and practice that define the nature of enquiry (Blanche *et al.*, 2006). According to Patton et al. (2016), the researcher should understand and have knowledge of related philosophies that support different principles of the study. The three dimensions aligning the research process (ontology, epistemology, and methodology) are summarized in the next section.

3.3 Research paradigms

A paradigm is defined by many authors (Dills & Romiszowski, 1997; Guba, 1990; Kuhn, 1970; Salter, 1990; Schwandt, 2001; Swartz, 2000). Kuhn (1970:175) defines a paradigm as, “the set of common beliefs and agreements shared between scientists about how problems should be understood and addressed.” When research is based on common paradigms, equivalent rules and standards for scientific practice dictate the process. A paradigm serves as a guide to steer the research by direct modeling and through abstracted regulations. Paradigms do not govern subject matter but worldview of a group of practitioners (Kuhn, 1970). According to Salter (1990) a research paradigm is a set of questions, opinions, and models while Guba (1990) defines a paradigm as an interpretative framework that is directed by the beliefs and views about how the world should be understood. Creswell (2013:5) explains a paradigm as being a general philosophical positioning about the world and the nature of research that guides the researcher. Philosophies or paradigms are ways of observing the world and make certain assumptions about the world (Creswell, 2013:5; Lichtman, 2010:7; Saunders, Lewis, & Thornhill, 2009:108) which will ultimately lead the researcher to embrace a qualitative, quantitative, or mixed method approach.

Different research paradigms have distinctive characteristics, which can be clarified by linking research and philosophical schools of thinking. Assumptions between different research paradigms act as a map that directs thinking and action. A research paradigm includes three levels; philosophical, social, and technical. The philosophical level talks about to the fundamental beliefs about the world. At a philosophical level, theories contrast five sets of assumptions: ontological (subjective or objective), epistemological, axiological, methodological, and assumptions about human nature. At the social level, guidelines exist as to how a researcher should conduct his/her research activities and finally, the technical level includes methods and techniques applied when conducting research.

According to Strydom, Fouché, and Delport (2002:266) a paradigm guides the researcher where to look for answers instead of addressing research questions. Kuhn (1970) draws attention to the role of paradigms in the natural sciences, while social scientists develop paradigms for the use of understanding social behavior. Four paradigms; positivism, post-positivism, critical theory, and constructivism are suggested by Guba and Lincoln (1994:108-111). Burrell and Morgan (1979:25-35) present an additional four paradigms; functionalist, interpretive, humanist, and structuralist as depicted in Figure 5. Expectations regarding the nature of social science are classified by Burrell and Morgan (1979:25-35) as the assumption of ontological nature, epistemological nature, methodological nature, and human nature. From Figure 5, four paradigms: radical humanist, radical structuralist, interpretive and functionalist paradigm are presented, which guide investigations of social theory that emerge from the divide by the subjective-objective (horizontal) dimension and the regulation-radical change (vertical) dimension.

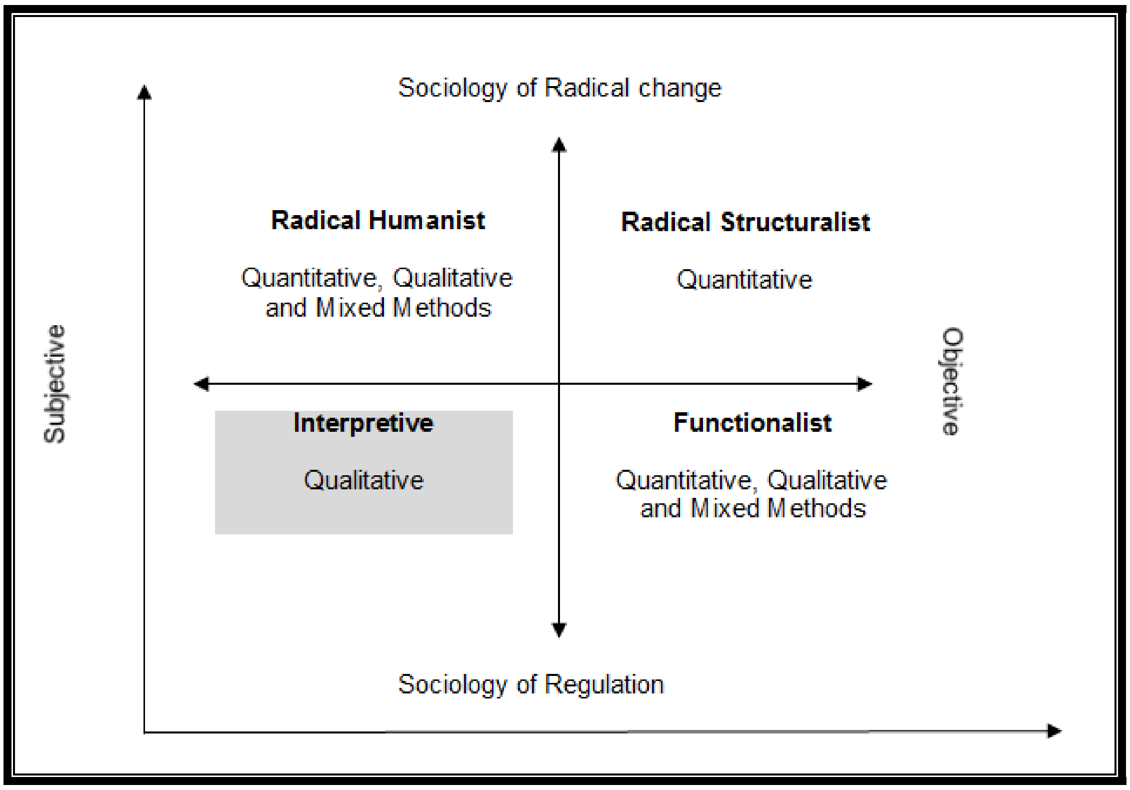


Figure 5: Research paradigms used in the analysis of social research. Adapted from Burrell and Morgan (1979)

The problem under investigation in this research study could have more than one reality and this is accepted by the researcher. This study does not involve testing theories or to empirically verify or falsify a theory, which is the objective of the positivist paradigm. The main objective of the study is to explore lecturers' perspectives of using problem-solving guidelines during the teaching of computer programming. The positivist paradigm will, therefore, not be a suitable choice as research method for this study. Instead, the current study aims to understand the use of problem-solving guidelines during the teaching of computer programming. The critical paradigm is also not an appropriate research method for this study as the purpose of critical paradigm is to emancipate people, which is not the aim of this research study. The interpretive paradigm forms the basis of the current study and is discussed in the next section.

3.3.1 Interpretivism

The interpretive paradigm is also referred to as the constructivist paradigm because individuals create their own constructs of reality (Creswell, 2013:8). The interpretive paradigm was developed as a reaction to the positivism paradigm (Mack, 2011:7). The interpretive paradigm advocate that people create and associate their own subjective and intersubjective meanings as they interact with the world they live in (Al Riyami, 2015:413; Orlikwoski & Baroudi, 1991:5). Through in-depth inspection of the phenomenon of interest, interpretivists develop theories from the field of interest (Orlikowski & Baroudi, 1991:5; Walsham, 1995b:76).

Scientific procedures and substantiation thereof are abandoned by the interpretive paradigm (Sarantakos, 2012:42). The subjective nature is the main shortcoming associated with interpretivism as argued by Dudovskiy (2016). The generalization of generated data is a problem as the data is impacted by the researcher’s personal viewpoints and values, which undermine reliability and representativeness. Results are isolated and applicable to a specific situation. The value of interpretive research is, therefore, questioned by positivists. The goal of theory created by interpretivism lays in the creation of local theories rather than to generalize findings (Mack, 2011:8). Subjective, Information and Communication Technology (ICT), specific procedures applicable to specific situations will be adopted by this research study to understand the use of problem-solving guidelines in the computer programming class.

Interpretivism is not a single paradigm, but rather a combination of various paradigms (Burrell & Morgan, 1979:28). The interpretive and positivistic philosophies consist inter alia of hermeneutics and phenomenology (Boland & Richard, 1986; Mack, 2011:7; McKay, Marshall, & Hirschheim, 2016:12; Saunders, Lewis, & Thornhill, 2011:115).

The researchers who follows the positivistic paradigm uses experimental methods to generate quantitative data. The emphasis is on what is noticeable and available with the primary focus on questions and areas that adhere to empirical methods of inquiry. Emphasis on these methods is continuing, but the use of qualitative research methodologies has grown. In the search for truth, significant questions in the human realm within the requirements of empirical methods arose. A variety of methodologies have grown in popularity from this environment (Laverty, 2003:21).

Hermeneutics is a process of investigation (Saldaña, 2014:54) and is a critical method in both scientific and interpretive approaches (Maxwell (2012:53). To make sense of data and understand discrepant data, theories are developed or borrowed and repeatedly tested. This process is described as hermeneutics by Saldaña (2014:54) and Myers (1997). To search for meaning, the text is interpreted in this cyclic process of understanding (Butler, 1998:290). Saunders *et al.* (2011:116) support this view and refer to hermeneutics as a "symbolic interaction" – a continuous process of interpreting the social world. Interpretive studies in Information Systems (IS) are performed with hermeneutics and phenomenology as the philosophical base (Myers, 1997). The importance of hermeneutics in Information Systems research was stressed by explaining that research in this field aims to produce an understanding of the context of the Information Systems and the process whereby Information Systems influence and is influenced by the context. The relationship between the organization, people, and information technology forms a unit and hermeneutic analysis and aims to make sense of the whole.

Phenomenology is a phenomenon described by participants (Creswell, 2013:14). The description of the participants determines the essence of the experiences by those who have experienced the phenomenon. Therefore, phenomenology in research has been referred to as, "a description of lived experiences, the essences and essentials of experiential states, natures of being and personally significant meanings of concepts" (Saldaña, 2014:73). When a phenomenon is studied, the analysis of the meanings, description of the phenomenon, and the experience and understanding is referred to as interpretative phenomenological analysis (Medico, 2005). The current study can be positioned in phenomenology as a subfield of this qualitative investigation. Many different methods can be used in phenomenological research, for example interviews, action research, participant observation, focus meetings, conversations, and analysis of personal texts (Lester, 1999:2). In this research study, the phenomenon to be studied is the use of problem-solving guidelines in the computer programming class. During the interviews, it is expected from the ICT lecturers to convey their opinions regarding their experiences as ICT lecturers.

Interpretive methods are grounded on the assumption that human actors form a social construction in which knowledge of the reality of human actions is positioned (Walsham, 1995a:376). For this reason, no objective reality can be discovered by researchers and replicated by others. This contrasts with positivistic assumptions but guide the ontological assumptions of interpretivism. The interpretivist accepts that social reality is perceived differently by various people, their understanding, and interpretation of events that influence their views. The main ontological assumptions of the interpretive paradigm can be summarized as follows:

- Relativism, thus knowledge is subjectively constructed by peoples' thoughts and actions (Al Riyami, 2015:412; Morgan, 1980:608; Scotland, 2012:11; Taylor & Medina, 2013).
- Single realities do not exist, instead multiple socially constructed realities exist; in particular situations knowledge arises that is not reducible to simplistic interpretations (Hill, 2012:24).
- Knowledge is gained inductively to create a theory (Mack, 2011:8).
- Knowledge is generated through individual experience (Morgan, 1980:608; Scotland, 2012:11).

- Causation in social sciences is determined by interpreted meaning and symbols (Mack, 2011:8).
- Aims to acquire an perceptive understanding, within context of its complexity about the phenomenon studied (Al Riyami, 2015:413).

The interpretive paradigm aims to produce an understanding of the context of the information system. Human interpretations and meanings associated with computer systems will be considered in an interpretive information system (Walsham, 1995a:75). Philosophical assumptions, information systems methodology, and practice that focus on research activities in information systems, form the basis of this research study. Philosophical assumptions, methodology and practice should be considered as fundamentally interlinked and mutually supportive (Midgley, 2000:1) as illustrated in Figure 6.

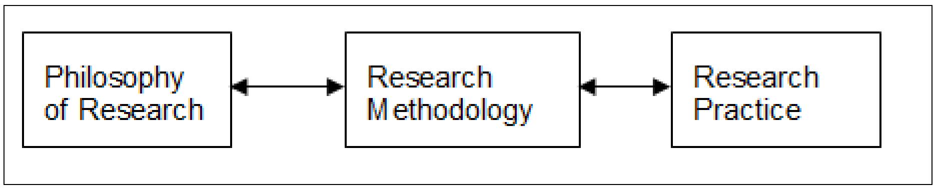


Figure 6: Interrelated aspects in information systems research

To gain a holistic understanding of what students require to solve problems, this research aims to get an understanding of how lecturers perceive the use of problem-solving guidelines in the computer programming class. This aim roots the current study in the interpretive paradigm. Lecturers will be investigated, and grounded theory will be used as platform to build a theory and conclusions. Additionally, the constant comparative method (CCM) of structural coding was used during the analysis process of this study, as a number of participants took part in the study (Boeije, 2002).

As the aim is to understand the perspectives of lecturers teaching computer programming, this study was categorized as interpretive. Table 4 summarizes the theoretical frameworks that underpin the basis of this study. These include ontology, epistemology, methodology, axiology, and human nature.

Table 4: Interpretive characteristics of the present study

Feature	Description
Purpose of research	To understand the perspectives of lecturers towards using problem-solving guidelines when presenting programming classes.
Paradigm	Interpretive
Ontology	• It is recognized that a variety of realities could exist. • Realities could be explored and created through collaborations and meaningful actions. • Find out how ICT lecturers make sense of their social worlds in their workplace settings by means of their daily routines and communications with other lecturers. • As ICT lecturers' knowledge, views, interpretations and experiences vary, multiple social realities exist.
Epistemology	• Records are understood through the processes of interpretation that is influenced by the interaction with social context. • Those active in the research (researcher and ICT lecturers) socially construct knowledge by interpreting social settings. • A personal interactive mode of data collection was done.
Methodology	• Processes of data collected by interviews with lecturers. • Research is a product of the interpretations of the researcher.
Axiology	• The values and lived realities of the researcher cannot be removed from the research process that affects the interpretation of the participants' (ICT lecturers) lived experiences.
Human Nature	• Employs determinist and voluntarist views as essential elements in social-scientific theories to define the nature of the relationships between man and the society in which (s) he lives.

3.4 Research approach

A research method is a strategy of investigation based on assumptions made concerning research design and data collection (Myers, 2009). Qualitative and quantitative methods are two research approaches. Their differences are (a) the nature of knowledge: how one understands the world and the ultimate purpose of the study; and (b) research methods: the way in which data are collected and analyzed, the type of generalizations and representations derived from the data.

Research methodologies and methods exist in both quantitative and qualitative research causing confusion when referring to “methodology” and “methods”. Bryman (1984:76) suggests that “methodology” refers to the epistemological assumptions (research methodologies or research methods) made by the researcher that guide the questions to be explored as well as the choice of techniques used, while “methods” are used to indicate the ways of collecting and analyzing data (methods and techniques).

3.4.1 Research design: Qualitative research approach

According to Lincoln, Lynham, and Denzin (2011:10) ‘qualitative’ emphasizes the qualities of entities and processes and meanings that are not experimentally examined or measured. Qualitative research is applied to expose and recognize the motives behind the phenomenon under study and concentrates on artefacts and human behavior. The researcher plays the key role when data is collected and analyzed (Janesick, 2003). Questions arise, and researchers seek answers to give meaning when they focus on how social experiences are created. Qualitative research involves the use of qualitative data such as pictures, documents, observations, and interviews and could be used in a variety of disciplines and fields including information systems, despite the fact that information systems was previously dominated by quantitative research (Myers, 1997). Any research where the results are not obtained by the application of statistical procedures or other means of quantification is identified as qualitative research (Strauss and Corbin (1998:10-11). Strauss and Corbin (1998:10-11) identify qualitative research as research which reflects peoples’ lives, lived experiences, behaviors, emotions, and feelings about organizational functioning, social movements, and cultural phenomena.

The qualitative research methodology aims to interpret and understand social interactions and studies the “whole” rather than specific variables, and the emphasis is on words rather than numbers. The data is collected in an orderly process and analyzed through coding, using computer programs (Lichtman, 2006:6). The researcher is critical in data collection and analysis (Merriman, 2001:7). In this way, the researcher can determine what people say, do and create in their natural settings to discover the world the way in which people themselves perceive and experience it. Various data collection instruments are available, which include participant observation, case studies, focus groups, and participatory research (Welman & Kruger, 2001:182-190). Research could be qualitative; and could be positivist, interpretive or critical and Myers (1997) stresses that qualitative research does not imply the research to be interpretive. The underlying philosophical assumptions, such as ontology and epistemology, guides the research whether it is quantitative or qualitative research (Myers, 2009:23). The underlying philosophical assumptions is followed by a specific qualitative research method, for example, case study or action research, which could be positivist, interpretive, or critical methods.

Several researchers (Chenail, 2011; Guba & Lincoln, 1994; Johnson & Onwuegbuzie, 2004) have addressed concerns with qualitative research. The main consideration when choosing a research sample is to select one that fits the research question. Generalizability remains a significant critique against qualitative research sampling (Gelo, Braakmann, & Benetka, 2008:287; Johnson & Onwuegbuzie, 2004:20), although qualitative researchers do intend to apply their findings beyond their samples. This is referred to as transferability (Hill, 2012:72). It is believed that results cannot be generalized because quantitative studies have larger populations when compared to qualitative studies that have small samples. The effectiveness of qualitative research is also questioned as researchers believe that the lack of generalizability and objectivity (Gelo *et al.*, 2008:286) are the weaknesses of qualitative research. Quantitative researchers believe that this could lead to subjectivity in the inferences and deductions of qualitative studies. Table 5 is a summary of the main characteristics of the qualitative approach.

Table 5: Summary of qualitative characteristics *

Orientation	Qualitative
Assumption about the world	Various realities.
Research purpose	Understanding a social situation from participants' perspectives.
Research methods and processes	• Flexible, changing strategies; design emerges as data are collected. • A hypothesis is not needed to begin research. • Inductive in nature.
Researcher's role	• The researcher takes part and becomes absorbed in the research/social setting.
Generalizability	• Detailed context-based generalizations.
Strengths	• Rich, deep information can be used to generate the outcome. • Useful for describing complex phenomena. • Useful for studying a limited number of cases in-depth. • Offers individual case information. • Can be used to study naturally occurring real-life situations. • Can be used to study the lived experience of people. • Describes rich detailed phenomena in specific context and settings. • Enables cross case comparisons and analysis. • Can study dynamic processes. • Researcher can be responsive to changes during the conduct of the study.
Weaknesses	• May be difficult to test hypotheses and theories. • Data collection and data analysis take more time compared to quantitative approaches. • Results produced may not generalize to other contexts and settings. • Results may be easily influenced by the researcher's personal biases.

* Adapted from Johnson and Onwuegbuzie (2004) and Thomas (2010)

3.4.2 Rationale for a qualitative study

The study aimed to investigate the use of problem-solving guidelines from an interpretive paradigm's viewpoint. When selecting a research methodology, "it is proper to select that paradigm whose assumptions are best met by phenomenon being investigated" (Guba, 1981:76). A qualitative approach is to investigate an appropriate method for this research that will be applied to answer the research question for this study. Merriam (1998:5) indicates that qualitative research is an idea covering numerous forms of investigation to "help us understand and explain the meaning of social phenomena with as little disruption to the natural setting as possible" (Merriam (1998:5). Qualitative studies require an in-depth look at a narrow subject and the current study followed a qualitative approach to answer the research question. The purpose of the study was to investigate, and gain insights of ICT lecturers' perspectives related to their lived experiences on problem-solving guidelines. The focus was on the multiple opinions of ICT lecturers and the researcher's understanding of these:

- Qualitative approaches can better account for the complexity of "meanings" from different participants. It was important to "unpack" how lecturers perceive problem-solving guidelines and to reveal interrelationships among different perception stated by different lecturers. This approach allowed for 'thick narrative descriptions' of the phenomena investigated and provided the researcher an opportunity to consider the views of the lecturers.

- The nature of the study suggested an inductive research strategy as most suitable although theories were created and not tested. The qualitative approach allowed the researcher to find a rich description of lecturers' lived experiences. To enhance the possibility of objectivity– that would have been lost if quantitative approach was used–an inductive analysis of data was applied. In line with qualitative research methods, a purposeful sample was applied to select and interview lecturers.

- The nature of the study implied that a significantly enriched description was the required approach. Words rather than numbers and statistics best conveyed the ICT lecturers' thoughts and perceptions.

- It was critical for the study design to be evolving, flexible and responsive to different ICT environments. Questions and explanations are built on one another as the tentative codes and theories emerged.

3.4.3 Deductive and inductive reasoning

Supplementing deductive and inductive, the two major reasoning approaches, is a third approach whereby the research process is dedicated to an explanation of incomplete observations, surprising facts, or puzzles specified at the beginning of a study (Mouton, 2005:114). The current study intended to create a theory not to be tested and, therefore, an inductive reasoning approach is followed.

Theory building research uses the inductive reasoning process to draw a theory from observations, starting with particular observations (De Vaus, 2014); and the broader generalization of specific observations (Burney & Mahmood, 2006) informally referred to as a bottom-up approach. Initially, specific observations are used to identify and measure patterns and regularities to generate a tentative theory that may be explored to lead to the development of general conclusions or theories (Welman & Kruger, 2001).

Deductive approaches aim to test theories while inductive approaches use emerging data to generate new theories. Figure 7 graphically illustrates distinctions amongst deductive and inductive approaches. An inductive approach was followed for this study as data were collected and a theory was developed following data analysis. The researcher was the main instrument for data collection and analysis. Information gathering included interviewing ICT lecturers in their work environment. Following the inductive research strategy, a theory was developed from interviews and understandings gained from the field.

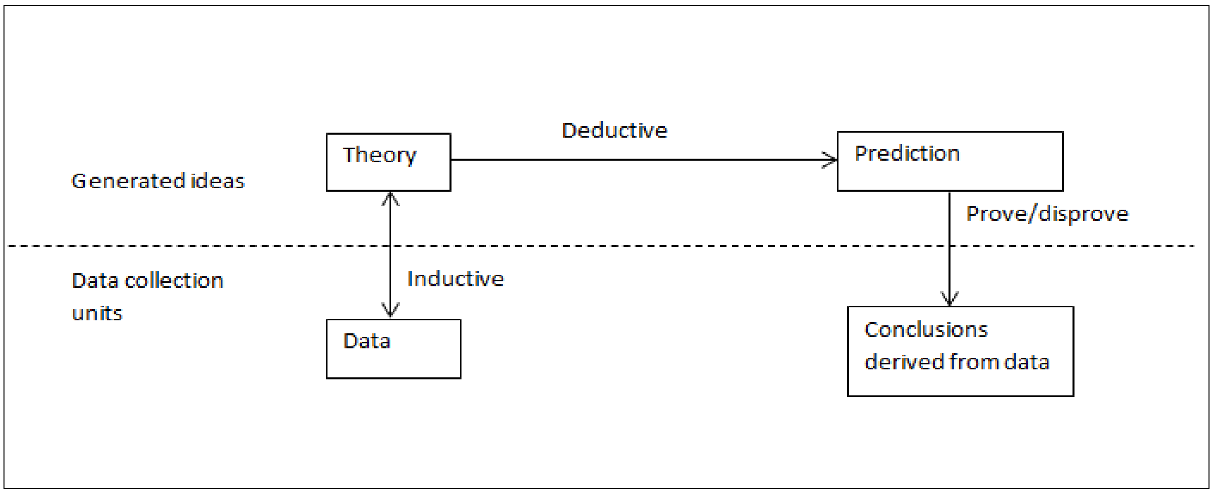


Figure 7: Induction, deduction and falsification of theories adapted from Welman and Kruger (2001:29)

At the essence of qualitative analysis is the constant comparison method (CCM) (Boeije (2002). Grounded in the data a theory is developed through an iterative process of data analysis and theoretical sampling that allows for a cycle of comparison involving the following five steps. Themes are compared within: (a) single interviews; (b) archival course analyses; (c) across multiple interviews; (d) course analyses; and (e) between the cases that make up the embedded single-case study. A well-structured plan is necessary to stipulate what is to be compared with what as it is not necessary to compare everything with everything else. In this five-step approach, the kind of material involved dictates the number of steps to follow as it is neither always necessary to follow all steps nor is it necessary to execute them sequentially (Boeije, 2002:302).

3.4.4 Grounded theory as qualitative approach

A descriptive, exploratory or explanatory research question directs the purpose of research (Saunders *et al.*, 2011:138). Tesch (2013) identifies 28 research approaches across diverse disciplines. Grounded Theory, one of the approaches identified by Tesch (2013), provides insights to explain a phenomenon by exploring new topics developed from data obtained. An overview of Grounded Theory is presented in this section to justify Grounded Theory as the suitable choice for this research study.

Grounded Theory was developed by Glaser and Strauss in 1967 (Corbin & Strauss, 2015:6; Merriam, 2002:142). This study implemented Grounded Theory to determine lecturers' perceptions of using problem-solving guidelines in the computer programming class. According to Glaser and Strauss (1967) Grounded Theory is a discovery process and fundamentally includes the systematic collection of data, a constant comparative method of qualitative analysis, and the generation of a theory. This systematic discovery process is closely linked to data that cannot be altered and reformulated (Glaser & Strauss, 1967:4). This suits the thoroughness of qualitative studies where data and conclusions from data are important as opposed to hypothesis testing and predetermined theories. In their first study, Glaser and Strauss used a continually comparative analysis of data rather than hypothesis testing (Ke & Wenglensky, 2010). The theory developed by this method is grounded in the data, which is responsible for the term grounded theory (Corbin & Strauss, 2015:6; Glaser & Strauss, 1967:1; Merriam, 2002:142). Traditional research approaches start with an existing theory or hypothesis whereas Grounded Theory aims to develop a theory from data, thus supporting the inductive reasoning approach. Theories are, therefore, derived inductively from the phenomenon represented (Charmaz, 2011:359; Merriam, 2002:142).

In building a theory to explain associations within a phenomenon using Grounded Theory, data may be collected from a variety of sources. The researcher contributes in analyzing the data and developing concepts, which is referred to as patterns in data by Glaser (2002). Categories, associated with the main theme of the research outcomes, may also be identified in the data (Glaser & Strauss, 1967). A concept is studied with the core category, which describes the main problem of the population, and associated sub-categories (Pham, 2016:171). The core category defines the significance of the concept connecting all other concepts, including completion of the open coding stage (Bryant & Charmaz, 2007:279; Hood, 2007:64). The open coding process breaks data down into separate parts such as words, phrases, sentences or paragraphs (Whiting & Sines, 2012:23). The researcher can now produce an explanatory theory by conceptualizing categories and relationships between them. This theory is grounded in the collected data and expands the knowledge base about the phenomenon.

It is very important that strategies used in conducting research should be consistent with research methods. The use of Grounded Theory can be flexible according to Charmaz (2006:15), as well as Glaser and Strauss (2009:242). Charmaz (2006:9) suggests that Grounded Theory is not bound by rules, recipes or requirements, but rather by a set of principles and flexible guidelines. McCann and Clark (2003:7-18) explain that Grounded Theory is defined by categories such as theoretical sensitivity, theoretical sampling, constant comparative analysis, coding and categorizing data, theoretical memos and diagrams, literature as data sources, and theory integration.

Theoretical sensitivity is an essential feature in Grounded Theory, which refers to a personal quality of the researcher to understand the meaning of data (Glaser & Strauss, 1967:46; Glaser, 1978:93). The researcher should be sensitive to the data and should be investigated without any predetermined viewpoint (Glaser, 1978:3). Theoretical sensitivity is developed by personal experiences, data immersion, and literature reviews.

Theoretical sampling is defined by Glaser and Strauss (1967:45) as “the process of data collection for generating theory whereby the analyst jointly collects codes and analyses his data and decides what data to collect next and where to find them, in order to develop his theory as it emerges." During the theoretical sampling process, participants are selected, research questions are formulated, and research sites are identified. From this, the researcher's understanding of themes and categories that could emerge from the data is improved. As a general methodology, Grounded Theory could be used for qualitative or quantitative approaches, which allows for several data collection methods and data analyses methods. Theoretical sampling is a vital principle of Grounded Theory (Breckenridge & Jones, 2009:113). Samples are selected to provide the best possible response to questions as they arise during analysis. Logical relevance guides the initial sample, and during the data collection and analysis process, gaps in data and theories are identified. Theoretical sampling strives to: (a) improve theory and explanation; and (b) describe emerging categories by obtaining data that confirm or disprove original categories (Charmaz, 2006:96). Theoretical sampling does not focus on increasing generalization of the results or to create a representative sample, as Charmaz (2006:101) emphasizes. To start somewhere, sampling begins at one site and, after a period of data collection and analysis, will then extend to other sites. In this way, theoretical comparison is possible. This process of "site spreading" is subsequently determined by the emerging theory (Glaser, 2001:181).

The "pivotal link between collecting data and developing an emergent theory to explain these data" is defined as coding Charmaz (2014:113). An analytical frame is developed when data is analyzed using coding strategies. Coding provides the tools for questioning, sorting and analyzing interviews, field notes, and documents (Charmaz, 2014:113). Grounded Theory requires two types of coding, namely substantive and theoretical coding. Substantive coding is divided into open and selective coding and works with raw data directly. Theoretical coding seeks interrelationships between concepts. Substantive coding follows open coding (Bryant & Charmaz, 2007:265) of the data collected first. All occurrences of the data are coded and split into comparable sections. Coded occurrences of the data are constantly compared with each other and comparisons are documented in memos that begin to highlight underlying consistencies in the data (Glaser, 1978). This process yields conceptual categories, which represent patterns of activities and thus cannot be constructed upon one coded occurrence by itself. During open coding, groupings are temporarily labelled for convenience and through the process, continually modified and tailored as other occurrences are continuously compared (Glaser, 1978:75-82).

Gaps in the evolving theory are identified, which call for additional explanations to direct theoretical sampling and the on-going process of data collection, analysis, coding, and decisions on the next set of data to be collected (Glaser & Strauss, 1967:45; Glaser & Holton, 2007). In this case, research analysis is interconnected with data collection (Corbin & Strauss, 2015:7). A variety of data collection techniques could be used but interviews and observations are most often used (Corbin & Strauss, 2015:7; Merriam, 2002:142). Some of the available data collection techniques include videos, journals, dairies, memos, and Internet postings (Corbin & Strauss, 2015:7).

As the process continues, new occurrences of coded data are compared to existing categories and no longer to one another. The researcher determines the limits of each category and highlight gaps in the emerging theory. Based on the theoretical purpose of the research, this will direct the researcher where to sample next (Glaser, 1978). Samples seek theoretical similarity and differences in the data to expand the properties of each category. This process tries to saturate all categories until the appearance of a core category that surfaces frequently, relates easily with many of the other categories, and accounts for the variation in the data. When a core category is identified the researcher switches from open coding to selective coding. The focus is now on data that are sufficiently relevant to the core category. The purpose of open coding is to run the data to completeness, while selective coding demarcates the emerging theory as it is focused on the core category. At this point, data analysis and memo-writing gradually become conceptual. Repeated comparison of incident to category and category to category improves the developing theory. This continues until a point of saturation is reached whereby new incidents in the data merely provide indicators of the same categories and their properties (Glaser, 1978). Theoretical sampling ceases when the core category is saturated.

When theoretical sampling ceases, theoretical sorting begins. The theory is conceptually reintegrated when theoretical codes are used to sort memos (Glaser & Holton, 2007). Memos are ideas that relate to evolving categories and are documented by the researcher (Khan, 2014:227; Lawrence & Tar, 2013:32). During the process of coding, constant comparison and theoretical sampling memos are written. As data is collected, a collection of memos, documents, and conceptual framework of the theory is accumulated (Holton, 2007). Memos are written sequentially (Charmaz, 2014) and reflect the researcher's analytic ideas (Bryant & Charmaz, 2007:249) through the comparison of incident to incident and incident to category. As the theory become demarcated, memos start to mature. As the process moves from substantive coding through to theoretical saturation, memos become gradually conceptually abstract (Holton, 2007:266). When data collection no longer generates new leads and categories, and their properties are considered sufficiently dense, theoretical saturation is reached and sampling stops (Glaser & Strauss, 1967:61). Memos are manually sorted to produce an emerging theoretical outline, which is based on theoretical codes. There is no fixed method to sort the memos. Memos are simply compared to find relationships between concepts and are selected where each idea fits within the emerging theoretical framework (Glaser, 1978). Relevant codes could emerge during this sorting for a theoretical outline. Functional codes can form a hypotheses conceptualized by the abstract model of the integrated theory (Glaser & Holton, 2005). Theoretical sorting provides a framework for the final theoretical document. From here, the developing theoretical framework is compared with existing literature to support, modify and nest the theory for publication (Glaser & Holton, 2007).

3.4.5 Time horizons

Research could be performed in different time horizons labelled cross-sectional and longitudinal. Significant types of time orientation in research design are cross-sectional, longitudinal, and retrospective (Babbie, 2015:105-110; Neuman, 2013:44). Observations in a specific moment are known as cross-sectional research. When data is collected at numerous points in time a longitudinal time horizon research is implicated. Research is called retrospective when data is generated and examined over different periods in the past.

Exploratory and/or descriptive research make use of the cross-sectional time horizon to obtain an initial understanding of the research problem. Frequently, these studies are also known for descriptions across a large population sample at a given time, as well as for understanding immediate causes in explanatory research. A drawback of a cross-sectional study is that it does not capture changes over time to describe why the observed patterns exist (Easterby-Smith, Thorpe, & Jackson, 2012:67). Exploratory or descriptive research usually aims to determine the existence of a phenomenon, or problem, by taking a cross-section of the population to obtain an overall picture at the time of the study. Data collection methods used in cross-sectional research include questionnaires and survey techniques that are usually appropriate to positivist studies (Easterby-Smith *et al.*, 2012:67). Where cross-sectional designs aim to investigate a problem at a specific period, longitudinal designs are generally implemented to understand a process of change over time. The current study is categorized as a cross-sectional study.

3.4.6 Unit of analysis

The unit of analysis is an important step in research design and is referred to as the descriptions of the level at which research is conducted and that objects are researched. According to Blumberg, Cooper, and Schindler (2008:224) the unit of analysis is not the same as the kind of participant the researcher interviews to get data. For example, in a study where the general manager of a company might be interviewed, the unit of analysis is the company and not the general manager. Trochim (2006) emphases the importance of understanding “who” or “what” the researcher is analyzing for the study. The unit of analysis could be individuals, groups, artefacts, geographical units, and social interactions. Duncan and Humphreys (1989) states that the sample must be the illustration of the universe from which it is drawn.

3.4.7 Selection of research participants

When collecting data, the qualitative researcher is required to set the boundaries for the study, collect data through appropriate methods, and establish conventions for recording information. Creswell (2013:189) suggests that sites or individuals should be purposefully selected. This should be done to assist the researcher in understanding the problem and the research question.

3.5 Data collection strategy: interviews

Interviews are a prominent approach for collecting data in qualitative research. Methods such as observations, diaries, the generation of visual images, or other forms of text are also used (King & Horrocks, 2010:6). Interviews may be conducted in many ways with diverse intentions and principles. Kvale (2008:11) states that in qualitative research, “interviews are the interviewee’s lived everyday world.” Interviews collect the subjects’ experiences and lived meanings making it a very sensitive yet powerful method for collecting data as noted by Kvale (2008:11).

In social sciences, qualitative research interviews are widely used (Kvale, 2008:5). Research questions should not be confused with interview questions (Maxwell, 2008:230). Issues to be understood are identified by research questions while the data necessary to understand the issues are generated using interview questions. The lived experience of people (Patton & Cochran, 2002:4) and the meaning they make of an experience is captured by interviewing people (Seidman, 2013:9). Conducting interviews provided the researcher with the opportunity to acquire more information on the subject matter (Bowling, 2014). Brynard and Hanekom (2006:37) stress that interviews should be conducted in a conducive environment with little or no interruptions. Advantages and disadvantages of interviews are summarized in Table 6.

Table 6: Advantages and disadvantages of interviews *

Advantages	**Disadvantages**
• Useful when participants cannot be directly observed. • Participants can provide historical information. • Allows researcher control of the line of questioning. • Allows questioning over a long period of time. • Encourages open exchanges. • Cost and time efficiency. • Interviews yield data in quantity quickly. • Immediate follow-up and clarification are possible.	• Provides indirect information filtered through the views of interviewees. • Provides information in a designated place rather than the natural field setting. • Researcher's presence may bias responses. • Not all people are equally articulate and perceptive. • Privacy may be an issue with online contact. • Technical skills needed for online data collection. • Interviews are often intimate encounters that depends on trust. • Interviewees might not be able to find words to convey their thoughts. • Time consuming to analyze data obtained.

* Adapted from (Creswell, 2015:112; Marshall & Rossman, 2014:150)

Structured interviews take away control from the participants and for this reason, structured interviews are not a good data generation method when the end purpose of the research is to build a theory. (Corbin & Strauss, 2015:39). Semi-structured interviews consist of features of structured and unstructured interviews and the next section explains the use of semi-structured interviews as data collection method.

3.5.1 Semi-structured interviews

During semi-structured interviews, the interviewer asks open-ended questions, which is followed by the expressions and opinions of the interviewees. According to (Fitchat, 2016:72) both interviewer and the interviewee are comfortable as it is a discussion or a thinking process on the given topic and usually goes into significant depth. The direction of the interview is not predetermined but determined by both parties. Researchers such as Corbin and Strauss (2015:38) found that the semi-structured interviews provide the richest source of data for theory building. The researcher enters the interview with only a bare minimum number of questions, usually with the theme or topic and subsequent questions arise during the conversation with the interviewee (Kendall, 2008:133). The semi-structured interview method wants to determine the importance of the phenomenon under study from the point of view of the interviewees (Van Teijlingen, 2014:17).

3.6 Data Analysis

According to Saldaña (2015:4) qualitative data analysis refers to the ability to produce different aspects from what the researcher has observed and reconstruct to form new interpretations. The objective of qualitative data analysis is to get a fuller picture using data to explain the phenomenon and what it implies. It also relates to a specific type of analysis where the study was designed from the responses of the semi-structured interviews according to content analysis. The aim of using content analysis was to construct a model to define the phenomenon under study in a theoretical way. Content analysis is used for analyzing transcribed, vocal or visual messages (Elo & Kyngäs, 2008). This permits the researcher to examine theoretical issues to improve the understanding of the collected data.

Two approaches that are employed in qualitative research to define the purpose of the study are inductive and deductive analysis. The deductive content analysis is “when previous knowledge operationalizes the structure of analysis, and the purpose of the study is theory testing” (Elo & Kyngäs, 2008:108). The approach is used when the researcher wants to re-assess the existing data in a new perspective.

Data was analyzed using thematic and hermeneutics principles. The term hermeneutics refers to “interpretation.” Ricoeur (1992) explains hermeneutics as the “theory of the operations of understanding in their relation to the interpretation of texts.” Hermeneutics’ purpose is to disclose what is not known and reflect on the lived meaning of people’s experiences.

The inductive content analysis is recommended when there is inadequate knowledge about the phenomenon. Given that, the inductive content analysis served as a framework for the study.

3.6.1 Trustworthiness

The purpose of qualitative research is to capture the lived experiences of the people as authentically as possible (Onwuegbuzie & Collins, 2007:49). Denzin and Lincoln (2005:19) note that “such experience, it is argued, is created in the social text written by the researcher. This is the representational problem.

It confronts the inescapable problem of representation but does so within a framework that makes the direct link between experience and text problematic." For this qualitative research, the trustworthiness of this study was guaranteed by employing *credibility, authenticity, dependability, and confirmability.*

Credibility: The researcher engaged with the collected data from the IT and CS lecturers (such as notes, transcripts, audio tapes) rigorously to represent the participant's responses and worldview (Polit & Beck, 2010:492).

Authenticity: The interview questions were developed based on the theoretical basis as described in Chapter 2. The questions were pre-tested to ensure that there are no flaws, unbiased data and other weakness (discussed in § 4.2.3).

Dependability: The research process of the study (discussed in § 4.3) was followed to enable prospect researchers to reproduce the study despite the fact that the outcomes of that study might not yield same results (Fitchat, 2016:77).

Confirmability: The researcher demonstrated the raw data attached as Appendix A that reflects the participants' viewpoints (Lincoln & Guba, 1985:320).

3.6.1.1 Software used to analyze the data

The computer-assisted qualitative data analysis software (CAQDAS) Atlas.ti™ Version 8 was used to analyze the data.

3.7 Ethical considerations applicable to the current study

Following the primary research, research ethics were employed to ensure that research results are believed to be robust and relevant. Research ethics principles used included consent, harm and risk, honesty and trust, privacy, confidentiality and anonymity, and voluntary participation. The next section summaries ethical issues that were attended to.

Informed consent: All participants were informed about the rationale of the study, nature, and data collection methods. The researcher obtained informed consent from the participants.

Harm and risk: Participants were not exposed to any risk of stress, humiliation, or loss of self-esteem. The researcher guaranteed the participants that they were not exposed to any harm that might impact their participation in the current study.

Honesty and trust: The researcher followed all ethical requirements to describe the outcomes and findings of the study in a trustworthy manner.

Privacy, confidentiality, and anonymity: The right to privacy and confidentiality was ensured during the interviews. Participants were assured that no names would be mentioned in the study to protect their identity.

Voluntary participation: Participants were guaranteed that the study was mainly for academic purpose and their contribution was voluntary.

3.8 Conclusion

This chapter summarizes the research philosophy, paradigm, approach, data collection strategy, analysis, and interpretation of data along with ethical issues that explored the perception of the lecturers concerning the use of problem-solving guidelines with computer science students.

The research philosophy was aligned with the ontological tenet based on the subjectivism and also based on the epistemological tenet based on the interpretive approach.

The research paradigm underpinned the interpretive approach and followed the inductive approach. The study applied mono-method using one type of method – qualitative method. Semi-structured interviews were used for collecting data from the lecturers.

The unit of sampling was problem-solving guidelines from the lecturers who present programming subjects. The hermeneutical approach was employed to analyze the lecturer’s perceptions.

4 CHAPTER FOUR: DATA COLLECTION AND ANALYSIS

4.1 Introduction

In Chapter 3, the research methodology was presented. The aim of this chapter is to use the data collected through unstructured interviews and put it into context. Thus, the purpose of Chapter 4 aligns with the primary objective of this study (§ 1.5) to explore lecturers' perspectives of using problem-solving guidelines during the teaching of computer programming.

The discussion will be based on data analysis (lecturers' perspectives of using problem-solving guidelines during the teaching of computer programming) and the foregoing literature.

4.2 Research approach

The use of problem-solving guidelines from the IT and CS lecturers' viewpoint was the main objective of this research. Consistent with the interpretivism research philosophy, the qualitative method was followed and applied in the study. The inductive research approach was accomplished using semi-structured interviews, which were the basis for interaction with the participants (§ 3.4).

4.2.1 Selection of participants

Participants in the study were individuals who teach computer programming. They were selected for their knowledge of teaching computer programming, perceptions of problem-solving methods and guidelines, and experiences using problem-solving guidelines as a strategy in teaching programming. They were purposefully (Ravitch & Riggan, 2016:61) selected to assist in understanding the research problem and questions (Bowling, 2014:198; Creswell, 2013) from two of the tertiary institution in the Gauteng province of South Africa (§ 3.4.7). Their programming teaching experience includes programming languages such as Procedural Language/Structured Query Language (PL/SQL), Structured Query Language (SQL), C-Sharp (C#), Java etcetera.

4.2.2 Data collection

As motivated in Chapter 3, interviews were conducted to collect data for this study (§ 3.5). According to King and Horrocks (2010) the goal of interviews is to ensure that the researcher understands the interviewees' perceptions of the research topic. To allow the researcher to ask the same questions to all participants and have the flexibility to ask additional questions if individual responses are not clear (Gillham, 2005), semi-structured interviews were conducted for this study. The researcher should be well-prepared to conduct semi-structured interviews as the responses from the interviewees cannot be predicted (Wengraf, 2001). Therefore, short, clearly worded, open-ended questions were formulated. Open-ended questions gave the participants the opportunity to provide free-form responses allowing the researcher to find more than anticipated.

To identify and detect possible problems during the data collection process, the open-ended questions were pre-tested for possible refining and improving the validity of the findings of the study (Collins, 2003). During the pre-testing process, attention was given to the following aspects, as suggested by Hurst *et al.* (2015).

- Assessing the language appropriateness and validity of data collection.
- Calculate the time length of the entire interview.
- Exploiting the methodological skills for the data collection.

- Evaluating the feasibility and reliability of translation and transcript methods in preparation for the interview for analysis.

Interviews were conducted with two lecturers who are responsible for teaching information technology. The intention was, firstly, to clarify if the questions were appropriate to address the research objectives. Secondly, the pre-test was conducted to check whether the responses can be used as variables for further analyses. Responses were recorded, and the semi-structured questionnaires were improved accordingly. The pre-test process was repeated a second time, and no adjustments were necessary thereafter. Consequently, the semi-structured questionnaire was implemented as the data collection tool for this study.

4.2.2.1 Interview process

After the purpose of the study was explained to participants, ethical issues were clarified (§ 3.7). With the knowledge of what the study entitled and the ethical issues agreed on, the interviews commenced. Interviews took approximately 30 to 60 minutes per interview and were conducted in the lecturers' offices – a space where they could feel comfortable. After each interview the responses were abstracted, categorized and coded using Atlas.ti™ Version 8, as described in the next section. Interviews were conducted until data saturation was reached. Data saturation was reached after five interviews.

4.2.3 Data analysis

Responses from the participants during the semi-structured interviews were tape recorded (§ 3.6). The collected recorded responses were later transcribed. The researcher took notes during the interviews, which assisted with the accuracy and transcription of the data. Atlas.ti™ Version 8 was used as data analysis tool to make the data manageable and sort it into a systematic order (Elo & Kyngäs, 2008). The documents that originated from the interviews with the lecturers were used as input data. The aim was to analyze the data in order to recognize and create patterns in the data. In order to achieve this, transcribed interviews were imported into Atlas.ti™ Version 8 as a hermeneutic unit named: “How do lecturers perceive the use of problem solving guidelines during the teaching of computer programming?” The hermeneutic unit was used as primary data source to be analyzed to answer the research question for this research.

As mentioned above, Atlas.ti™ Version 8 allows the transcribed data to be abstracted, categorized and coded. Using this tool, the transcribed data were categorized and coded into themes and codes. The following steps were followed:

- **Step 1:** A lecturer was interviewed. The interview was recorded and transcribed.
- **Step 2:** Transcribed interview data were documented in Microsoft Word. The interview transcript was carefully read and re-read to categorize data obtained from the interview. This Microsoft Word document was imported into Atlas.ti™ Version 8.
- **Step 3:** Using the imported data, phrases, words, concepts, and sentences relevant to the topic were labeled. The researcher tried to be open-minded, creative and unbiased during this process.
- **Step 4:** This coding process was repeatedly executed (steps 1, 2, 3 and 4), as phrases, words, concepts, and sentences were repeated by several participants. Steps 1 to 4 were repeated until data saturation was reached. Data saturation was reached after five interviews and the process of abstraction, categorization, and coding terminated and steps 5 to 7 were executed next.
- **Step 5:** The codes were categorized by combining similar codes. Created codes were checked and rechecked several times for

accuracy. Where necessary, new codes were created by combining more codes. The created codes were classified, and themes emerged.

- **Step 6:** Related emerged themes were categorized.
- **Step 7:** Lastly, themes were summarized and aligned to the objective and research question of the study.

These steps (Steps 1 to 7 above) guided the researcher to analyze the data collected from the participants during the interview process. Collected data provided a rich explanation of lecturers' experiences when teaching computer programming. The process of data analysis advises that data must first grouped then reduced (Elo & Kyngäs, 2008). Data reduction started with coding, where themes emerged serving as a form of important groups helping with data analysis. Codes and categories were compared to decide on the common link. The emerged themes were summarized and validated to write the findings. The themes created during the data analysis process described above are as follows:

- Lecturers' experience of students' shortcomings with problem solving.
- Lecturers' perception of students' attitude when confronted with problem-solving.
- Approaches used by lecturers when teaching programming.
- Lecturers' perspectives of using problem-solving guidelines.
- Lecturers' perception of the usefulness of guidelines.
- Methods used to present programming problems.
- Alternative methods used for different learning styles.

Table 7 illustrates the emerged themes and associated codes from the Atlas.ti™ Version 8 analyses.

Table 7: **Themes and codes emerged from the Atlas.ti™ Version 8 analyses**

Lecturers' experience of students' shortcomings with problem solving	
	Students lack logical thinking skills. Students make assumptions. Students have difficulty in understanding specific concepts or to make it applicable. Students do not apply prior learning. Students do not understand the problem. Students lack programming logic. Students lack programming skills. Students cannot think out of the box.
Lecturers' perception of students' attitude when confronted with problem-solving	
	Students leave the work until it is too late. Students copy each other's work. Students seek solutions on the Internet. Students do not want to spend time to solve a problem. Students rely on the Internet for solutions.
Approaches used by lecturers when teaching programming	
	Lecturers divide students into small groups. Lecturers use real-life scenarios. Lecturers combine different approaches. Lecturers apply explicit teaching. Lecturers apply task-based teaching. Lecturers apply peer-tutoring.
Lecturers perspectives of using problem-solving guidelines	
	Lecturers claim that there are no guidelines to use. Lecturers use a combination of different guidelines. Lecturers amend guidelines to suit students' needs. Lecturers do not use any guidelines. Lecturers claim that there are no proper guide lines and approaches available. Lecturers develop their own guidelines based on their experience.
Lecturers' perception of the usefulness of guidelines	
	Guidelines are useful and increased students' performance. Guidelines help students to solve problems independently. Guidelines are useful in smaller groups.
Methods used to present programming problems	
	Step-by-step explanation using examples related to that topic. Expect students to be prepared and compile short tests. Start with real-life examples. Textbook examples. Make use of tutors.

Alternative methods used for different learning styles	
	Use LMS (Learning Management System). Online study material. Websites showing tutorials, videos, PowerPoint slides. YouTube videos. Google.

The following section aims to describe the themes and associated codes that were identified during the Atlas.ti™ Version 8 analyses process as shown in Table 7.

4.3 Discussion of the results

4.3.1 Lecturers' experience of students' ability to solve problems

- Students lack logical thinking skills.
- Students make assumptions.
- Students have difficulty to understand specific concepts or to make it applicable.
- Students do not understand the problem.
- Students cannot think out of the box.
- Students do not apply prior learning.
- Students lack programming logic.
- Students lack programming skills.

While logical thinking or reasoning skills are essential for today's job market, lecturers observe that students lack logical thinking skills. This concern is underlined by Selingo (2015) when he cited a recent survey of 31,652 college seniors at 169 institutions, which indicate that only sixty percent of the students surveyed left college with adequate reasoning skills.

Albrecht (2009) argues that logical thinking is based on sequential thought where one think in steps. The important facts, ideas, and conclusions given in a problem are arranged in a chain-like progression that makes meaning in itself. He, (Albrecht, 2009) further states that logical thinking is not something magical but must be mentally learned. People can be trained to become “smarter” in logical thinking processes (Dettmer, 2007). This allows students to dismiss quick responses, such as "I don't know," or "this is too difficult," by helping them to concentrate on their thinking processes and to better understand the methods used to arrive at possible solution. The lecturer should try to put himself in the shoes of the student and try to read the student’s mind or think like the student and predict possible questions that the student might have (Polya, 2014). It is possible that lecturers, who are the experts in a specific field, are so fluent in solving problems from that field that they can find it difficult to articulate the problem-solving principles and strategies they use to teach students in their field because these principles and strategies are second nature to the lecturer (§ 2.6).

Recommendation: When teaching problem-solving skills to students, a lecturer should take cognizance of the principles and strategies of good problem-solving in their discipline**.**

4.3.2 Lecturers’ perceptions of students’ attitude when confronted with problem solving

- Students leave the work until it is too late.
- Students copy each other’s work.
- Students seek solutions on the Internet.
- Students do not want to spend time to solve a problem.
- Students rely on the Internet for solutions.

In order for students to solve problems and learn in the process, they must experience some struggle (§ 2.7). Hiebert and Grouws (2007) use the term productive struggle to refer to “the effort to make sense of a problem and to figure something out that is not immediately obvious.” Additionally, destructive struggle can lead to an overwhelming situation, whereas productive struggle can be valuable and beneficial when solving problems (Jackson & Lambert, 2010).

Destructive struggle leads to frustration, learning goals feel hazy and out of reach, feel fruitless, and lead students to feel abandoned and on their own and creates a sense of inadequacy. On the contrary, productive struggle leads to understanding, makes learning goals achievable and effort seem worthwhile, yields results, leads the student to feelings of empowerment, and creates a sense of hope. According to Warshauer (2015) struggle may or may not be visible and can be expressed in several ways. If a problem is easy enough that the student can solve it without difficulty, struggle may not be observed. On the contrary, a student may choose not to attempt to solve a difficult problem and, in this case, there is also no struggle.

There is a fine line between the when and how help should be offered to a student to support him or her towards productive struggle. There is also a fine line between doing the problem with or for the student and supporting productive struggle. Students will not experience productive struggle if they are not allowed to deal with challenging problems. But if the lecturer does not provide them the right assistance, at the right time, then their experience can become that of destructive struggle.

Recommendation: Lecturers should provide students the right assistance at the right time to make their experience a productive struggle.

4.3.3 Approaches used by lecturers when teaching programming

- Lecturers divide students into small groups.
- Lecturers use real-life scenarios.
- Lecturers combine different approaches.
- Lecturers apply explicit teaching.
- Lecturers apply task-based teaching.
- Lecturers apply peer-tutoring.

Keefe's (1979) meaning on learning styles emphasizes how students learn, with regards to cognitive, affective, and psychological behavior, which have an impact on how they understand, work together, and react to their environment (§ 2.4). Literature (Jenkins, 2002; Lahtinen, Ala-Mutka, & Järvinen, 2005) reveals that students at universities around the world experience problems in learning to program (§ 2.7.1). Different strategies and tools are developed to try and address this problem (Cooper, Dann, & Pausch, 2003; Moritz, Wei, Parvez, & Blank, 2005). Cândida, Silva Carmo, and Mendes (2007) argue that the problem still exist today as high dropout and failure rates (Zeeman, 2014) and low enrolment numbers (Kirlidog, van der Vyver, Zeeman, & Coetzee, 2016) continue to be reported.

The ideal situation is for lecturers to seek new strategies that may prove more effective in supporting programming learning. Cândida *et al.* (2007) emphasize the importance of designing learning activities that will take each student's learning style into consideration, which may help the student to learn. This is not always practical or feasible because of constraints like class size and available time, and lecturers are forced to use the same learning materials and strategies to all students. Different learning styles and approaches have been discussed in Chapter 2 (§ 2.4). The data obtained from the interviews revealed that lecturers try to accommodate students by using different teaching strategies.

Recommendation: Lecturers must create an environment where students feel comfortable, and the lecturer must use a teaching/learning style that suits most students.

4.3.4 Lecturers' perspectives of using problem-solving guideline

- Lecturers claim that there are no guidelines to use.
- Lecturers use a combination of different guidelines.
- Lecturers amend guidelines to suit students' needs.
- Lecturers do not use any guidelines.
- Lecturers claim that there are no proper guidelines and approaches available.
- Lecturers develop their own guidelines based on their experience.

Problem-solving guidelines help students to be articulate and think logically (§ 2.7). When they start breaking down what is happening, they can start predicting what is going to happen. Programming is about getting the computer to solve a problem and for that, one needs to design an algorithm or a set of instructions. The ability to do that is a very valuable skill. The data from the interviews show that lecturers are not using problem-solving guidelines consistently.

Recommendation: Lecturers should be aware of existing problem-solving guidelines and use them in the design of the lecturing material and during the presentation of the lecture.

4.3.5 Lecturers' perceptions of the usefulness of guidelines

- Guidelines are useful and increased students' performance.
- Guidelines help students to solve problems independently.
- Guidelines are useful in smaller groups.

Problem solving is a vital skill that students need to write computer programs and problem-solving fills much of what people do. Often solutions must be found in a short time causing stressful situations. In response to new problems, many people react with a solution that seemed to work before. With this approach, it is easy to get stuck in a circle of solving the same problem or using the same problem-solving technique over and over again (Lahiri, 2016). An organized approach can be very useful to solve a problem. Different problem-solving methods have been presented in Chapter Two (§ 2.7). As one gains experience with different approaches, it will become second nature in so much that one can deepen and enrich the approaches to suit one's personal needs and nature (Lahiri, 2016).

Recommendation: Lecturers should be familiar with many problem-solving techniques and expose students to as many as possible. A positive attitude towards problem-solving guidelines should encourage and improve students' problem-solving skills.

4.3.6 Methods used to present programming problems

- Step-by-step explanation using examples related to that topic.
- Expect students to be prepared and compile short tests.
- Start with real life examples.
- Textbook examples.
- Make use of tutors.

In general, lecturers present problems to students beginning with an idea. The heart of problem-solving is the connection between this idea (or information) and the solution. Different strategies can be utilized to prepare a solution. First, there is a need to know what is required. When, why, and where questions should be asked until the task is completely understood. Typical questions could be: What should I know about the problem? What does the solution look like? Are there any sort of special cases? How will I recognize that I have found the solution?

In the computer science domain, there are numerous algorithms available to solve problems. For example, sorting and searching are two important types. Other examples are sequential or linear search (Wikibooks, 2017).

Recommendation: Lecturers should expose students to various problem-solving techniques and different approaches, until it becomes second nature and deepens and enriches students' problem-solving skills.

4.3.7 Additional methods used for different learning styles

- Use LMS (Learning Management System).
- Online study material.
- Websites presenting tutorials, videos, PowerPoint slides.
- YouTube videos.
- Google.

There are many alternative teaching methods available that lecturers can implement. Teaching methods are outside the scope of this study and therefore only a few will be mentioned but not discussed (Owens, 2016:1).

- **The flipped classroom**. In this method, students take responsibility by watching or reading lessons at home and, during class, complete hands-on experiences. The homework becomes the lesson, and the lesson becomes the action.

- **The use of technology**. Students use technology, which could include interactive websites, webquests, videos, and other activities. Quick, formative assessments and class dialogue can also utilize technology.
- **Makerspace**. Students work in teams to create and innovate using digital tools. This could include basic coding, robotics, or 3-D printing.
- **Problem-Based Learning (PBL)**. Rather than expecting students to memorize items or facts and show mastery by writing tests, problem-based learning allows students to show mastery of content through problems.
- **Experiential Learning**. Learning take place through experience and process rather that a lecture. In experiential learning, students solve problems, but this time there is an added element of reflection. This essential piece strengthens student understanding with longer-lasting effects compared to traditional lectures.
- **Game-Based Learning**. Technology savvy students love playing games, which makes game-based learning an attractive alternative teaching method. There are many educational games available. Finding the most suitable educational game might be a challenge. Games will not only help students learn content, but also further develop their 21st century skills (for example critical thinking, problem solving, reasoning, analysis, interpretation, synthesizing information, etc.).

Recommendation: With many alternative teaching methods available, lecturers should consider exposing students to different teaching and learning environments and let students take more responsibility for their learning. Methods with emphasis on critical thinking, problem solving, reasoning, analysis, interpretation, synthesizing information, for example, should take preference.

4.4 Conclusion

In this chapter, recommendations aimed at assisting lecturers who are presenting programming modules were proposed as the primary objective of the study. Based on the interview data collected for the purpose of this study, recommendations were proposed to address the problem that initiated this study by addressing the research question.

CHAPTER FIVE: CONCLUSIONS, REFLECTIONS AND RECOMMENDATIONS

4.5 Introduction

This final chapter gives an overview of the study. This includes an overview of the purpose, objectives, research methodology, findings and ends with recommendations and possible future research. The recognition that many students find programming difficult while educators aim to improve students' problem-solving abilities, initiated this study. This alerted the researcher to investigate lecturers' perspectives of using problem solving guidelines during the teaching of computer programming. This study focused on the following research question:

> How do lecturers perceive the use of problem-solving guidelines during the teaching of computer programming?

The need to investigate lecturers' perspectives of using problem solving guidelines in class is the basic building block of this research and providing results for this can be regarded as the closing of the feedback loop.

To assist in answering the research question the main objective (§ 1.5.1) was addressed. The next sections provide an overview of the study from conceptualisation to the end results.

4.6 Summary of the research

Almost every computer science curriculum includes computer programming. Studies indicate that there is a perception that programming is a difficult skill to master. This perception is one of the major reasons why the interest in studying computer science is declining. Computer programming requires higher order thinking skills, which include problem-solving abilities and today’s digital milieu requires from students the ability and essential skills to solve problems. Interactive involvement not only motivates students but make them more determined to attempt to solve problems. Tan and Rahaman (2009) argues that students are more motivated if problem solving and programming problems are presented a context that they can relate to.

In today’s fast-paced digital world students are used to digital ways of learning, communicating and getting information via various technologies such as social media and the Internet. They use modern up-to-date technologies but are many times confronted with stereotypical old fashion technology in classrooms and that discourage them. Students think and learn differently because they grow up in an advanced digital environment. The fast-paced digital environment discourages students to become skilled in computer programming because of the time and effort it takes to master programming, which is often seen as a difficult skill to master. Requirements, in terms of learning in the twenty-first century, have changed while education systems generally remain the same.

Education needs to transform to reflect changes in the environment, improve student engagement in the classroom and actively learning, to remain relevant to the requirements of the changing world. Students should be active participants instead of passive observers. According to Houghton (2004:1) students at higher education institutions (HEI) have indicated that there have been insufficient preparations in the process of problem solving. Cai and Lester (2010:1) suggest that for students to become effective problem solvers at all levels, educators must know how to incorporate problem solving meaningfully into their curriculums. This observation led to the initiation of this study: guidelines are required to teach problem solving in the computer programming class to meet the problem-solving needs of the new generation of digitally oriented students.

Consequently, this study was undertaken with a dual purpose: firstly, to gain an understanding of the experiences and viewpoints of lecturers using problem-solving guidelines during the teaching of computer programming. Secondly, to develop recommendations to assist lecturers when they prepare for and teach problem solving in computer programming classes. To achieve this purpose the main objective was (§ 1.5.1):

> To explore lecturers' perspectives of using problem-solving guidelines during the teaching of computer programming.

The main research question for the dissertation was (§ 1.5.2):

> How do lecturers perceive the use of problem-solving guidelines during the teaching of computer programming?

The structure and layout of the thesis was presented in § 1.9. The rest of this section provides a summary of the preceding chapters.

In **Chapter 1** the motivation of this study was explained. The background that gave rise to the research was explained and the reality of the research was emphasized. A brief glance at problem-solving skills and programming, supported by the teaching for and the teaching of problem-solving were presented. The research problem was formulated in this chapter and the purpose and method of the research were set out.

Chapter 2 presented a brief overview of the fundamentals and concepts of problem-solving followed by a brief discussion of critical and creative thinking.

Different learning styles were documented followed by a brief look at guidelines or tools that are used in teaching problem solving.

In **Chapter 3**, the theoretical framework, paradigm and research approach that was followed were outlined. Furthermore, this chapter also presented the process followed for data collection and data analysis.

In **Chapter 4**, the primary objective of the study was addressed. The research data was analyzed in this chapter. From the insight gained from the reviewed literature and the results of the interview data, recommendations were proposed and explained.

4.6.1 Research objective: To explore lecturers' perspectives of using problem-solving guidelines during the teaching of computer programming.

To accomplish this objective, a descriptive phenomenological design was used to discover and describe the experiences of ICT lecturers who are currently lecturing computer programming. Data was collected applying interviews (§ 4.2.2) from five lecturers. Data analysis was conducted using the Constant Comparison Method (CCM).

4.7 Synthesis

This section summarises and interprets the findings of the research objective (§ 1.5).

4.7.1 Summary of findings

The objective of this study deals with lecturers' perspectives of using problem-solving guidelines during the teaching of computer programming. When data were analysed, the following themes emerged:

- Lecturers' experience of students' shortcomings with problem solving (§ 4.3.1). It is very important that lecturers should apply good problem-solving strategies when teaching problem-solving skills to students.
- Lecturers' perception of students' attitude when confronted with problem-solving (§ 4.3.2). Students need relevant assistance at the right time to make their problem-solving experience productive.
- Approaches used by lecturers when teaching programming (§ 4.3.3). A comfortable environment where the lecturer applies teaching/learning styles that suits most students is essential.
- Lecturers perspectives of using problem-solving guidelines (§ 4.3.4). Lecturers should integrate appropriate problem-solving guidelines in the design of the lecturing material and during the presentation of the lecture.
- Lecturers' perception of the usefulness of guidelines (§ 4.3.5). To encourage a positive attitude and improve students' problem-solving skills, students should be exposed to a variety of problem-solving guidelines implying that lecturers are familiar with a wide range of such guidelines.
- Methods used to present programming problems (§ 4.3.6). Exposure to various programming techniques, until it becomes second nature will deepens and enriches students' problem-solving skills.
- Alternative methods used for different learning styles (§ 4.3.7). Different teaching and learning environments should be considered while students should be encouraged to take more responsibility for their learning. The emphasis should be on critical thinking, problem solving,

reasoning, analysis, interpretation, synthesizing information, for example, should take preference.

4.8 Discussion and reflection

Reflection after completing a study is a way of reviewing lessons learnt from the research. This section reflects on the lessons learnt in terms of methodological, substantive and scientific aspects of the study.

4.8.1 Methodology reflection

Research is based on a methodology. Reflection on the methodology used and its appropriateness thereof for this study is the aim of this section. Limitations of this study are also emphasized. A detailed explanation for choices of the methodology and its limitations was discussed in Chapter Three, together with trustworthiness (§ 3.6.1) and ethical considerations (§ 3.7) applied for this study.

The literature review revealed that computer programming is viewed by many students as a difficult skill to master. Programming involves or is preceded by problem-solving. According to Mayer (2012) problem-solving is an act to transform a problem from an initial state to a goal state but no obvious way of changing the problem is known. Estivill-Castro (2010) agrees that IT graduates must possess problem-solving skills by writing methods and solutions in the language that will transform into computer programs. Students should be actively engaged in the learning process which implies that the learning environment should be transformed to be suitable to accommodate today’s digital savvy students. On the other hand, educators must also adapt their teaching methods and they must know how to incorporate problem solving meaningfully into their curriculums. The exploration and focus on these issues instigated the researcher’s curiosity to investigate how problem-solving guidelines are used in teaching problem solving in the computer programming class.

To answer the research question (§ 1.5.2) this study aimed to deliver a descriptive analysis and understanding of the views and perspectives of lecturers who present computer programming classes. To accomplish this objective the interpretivist, subjective, qualitative research paradigm was considered suitable, and a qualitative research approach was adopted. To understand the views and perspectives of lecturers an inductive research approach (§ 3.4.3) was applied. This process involve gathering data and make sense of it by grouping data segments into codes, followed by themes and finally larger viewpoints (Marshall & Rossman, 2014:222). By following this path, theories are formulated implying Grounded Theory as research strategy for this study (§ 3.4.4). Purposeful sampling (Donoso-Maluf, 2014) was used for the identification of computer science lecturers as possible interviewees.

To enable individuals to provide personal perspectives to the interviewer, semi-structured interviews (§ 3.5.1), a qualitative data collection technique, were used to collect data. Lecturers from two institutions of higher education were considered a convenient and purposeful sample. Five semi-structured interviews were conducted when data saturation (§ 4.2.2) was reached.

The grounded theoretical approach (§ 3.4.4) was used to analyse the interview transcripts (§ ADDENDUM A) according to the constant comparative method (CCM) (§ 3.4.4), which allows for the identification of patterns that could provide an explanation and understanding of the perspectives of lecturers lecturing computer programming, with respect to the use of problem solving guidelines. Atlas.ti™ Version 8 was used to support the data analysis process.

4.8.2 Substantive reflection

The focus of substantive reflection is to compare the results found in this research to other research on the same topic. No direct comparative study on this topic could be found in the literature. Despite this, an attempt was made to authenticate the findings and resulted recommendations (§ 4.3.1–4.3.7) with evidence found in the literature.

4.8.3 Limitations of the study

A limitation of this study is that the number of participants was very limited and selected from only two of the more than twenty institutions of higher education. Although the number of lecturers available for the study was limited, interviews were conducted until data saturation was achieved. Due to the mentioned limitations, the results of this study cannot be generalized to other HEIs in South Africa.

4.8.4 Scientific reflection

Scientific reflection focuses on the contribution of this study to the 'scientific body of knowledge'.

The view that computer programming is a difficult skill to master with the higher order thinking skills associated with problem solving initiated this study. The purpose of the recommendations that followed data analysis is to contribute to the existing body of knowledge on using problem solving methods to teach computer programming to produce more qualified programmers in South Africa. Apart from contributing to the existing body of knowledge, this study provides insights on the perspectives that lecturers have about the use of problem solving guidelines in the computer programming class.

Given the insights as discussed above, recommendations regarding policy and practice and further research are presented in the following sections.

4.9 Recommendations

This section presents recommendations that have arose from the findings of this study. Recommendations in terms of policy and practice and further and possible new research are discussed in the next sections.

4.9.1 Recommendations for policy and practice

In this section recommendations for policy and practice are made based on the findings of this study.

From a policy point of view, the research findings should challenge and prompt Information and Communication Technology departments at HEIs to identify gaps in their current computer science offerings. Lecturers are expected by virtue to be experts in their different field of knowledge. However, the outcomes of the study point out that the same lecturers have limitations. In the context of the study, amongst other limitations, this echoes with problem-solving skills and its underlying principles and strategies. The nature of pedagogy places problem-solving skills as students' responsibilities. The problem-solving skills and strategies are seen by lecturers as a secondary mandate to their teaching responsibilities and expectations. The lecturer in this case is a carrier, guardian and a determining factor of what might be appropriate or inappropriate for students.

From a practical viewpoint, the focus should be on addressing the different learning styles of students and on the demand for emotionally intelligent computer science graduates as well as the high demand for qualified students equipped with sufficient problem-solving skills. This could shape students' self-confidence and could contribute to emotionally self-aware students, who are also able to develop emotional awareness in others.

Based on findings of this study, recommendations were proposed to guide lecturers when preparing and presenting computer programming classes. The recommendations direct lecturers to be aware of principles and strategies of good problem solving in the discipline; to provide students with the right assistance at the right time to maximize their problem-solving experience; to ensure that students are comfortable in the learning environment and with the teaching style; to use existing problem solving guidelines during presentations; to support and guide students with no prior or limited knowledge; to expose students to various problem-solving tools or techniques; to encourage positive attitude towards problem-solving guidelines; and to encourage students to take responsibility for their learning.

4.9.2 Recommendations for future research

This research process continuously reminded the researcher into new areas of exciting potential future research topics:

- This research was executed from a lecturer's viewpoint. It should be interesting to do the same research from the student's point of view.
- The limited number of participants raise the question whether the results would be significantly different if more lecturers were involved.

- A valuable study could be to determine whether the curriculum embraces problem solving. *Are lecturers trained to place themselves in the mode of students thinking (how they perceive and understand knowledge)? Are lecturers trained to understand the flexibility of students' mind concerning problem solving?* There is not much evidence in literature to answer these questions. A lecturer that cannot think like a student, can become a barrier to students' understanding and creativity in term of problem solving. This calls for further research.

4.10 Summary

In this chapter a summary of the research is provided in chapter order and recommendations were made in response to findings from the literature and from the results obtained from the current investigation. Finally, recommendations were made for future research.

REFERENCES

Al Riyami, T. 2015. Main approaches to educational research. *International Journal of Innovation and Research in Educational Sciences*, 2(5):2349–5219.

Albrecht, K. 2009. Brain power: Learn to improve your thinking skills. http://www.simonandschuster.com/books/Brain-Power-Learn-to-Improve-Your-Thinking-Skills/Karl-Albrecht/9781439188644 Date of access: 30 January 2018.

Ali, A. & Smith, D. 2014. Teaching an introductory programming language in a general education course. *Journal of Information Technology Education: Innovations in Practice*, 13:57-66.

Awang, H. & Ramly, I. 2008. Creative thinking skill approach through problem-based learning: Pedagogy and practice in the engineering classroom. *International journal of human and social sciences*, 3(1):18-23.

Ayoufu, W., Afshari, M. & Ghavifekr, S. 2012. Factors contributing students' creativity.

Babbie, E.R. 2015. The practice of social research. Belmont, CA: Nelson Education. 608 p.

Baleni, L.S., Malatji, K.S. & Wadesango, N. 2016. The Influence of Peer Tutoring on Students' Performance in a South African University.

Barnes, D.J., Fincher, S. & Thompson, S. 1997. Introductory problem solving in computer science. (*In* 5th Annual Conference on the Teaching of Computing organised by.) Retrieved.

Bednarix, R., Moreno, A. & Myller, N. 2006. Various Utilizations of an Open-Source Program Visualization Tool, Jeliot 3. *Informatics in Education*, 5(2):195-206.

Blanche, M.T., Durrheim, K. & Painter, D. 2006. Research in practice: Applied methods for the social sciences. Cape Town: Juta and Company Ltd. 565 p.

Blumberg, B., Cooper, D. & Schindler, P. 2008. Business research methods. 2nd ed. Maidenhead: McGraw-Hill Higher Education. 768 p.

Boeije, H. 2002. A purposeful approach to the constant comparative method in the analysis of qualitative interviews. *Quality & Quantity*, 36(4):391-409.

Boland, J. & Richard, J. 1986. Phenomenology: A preferred approach to research on information systems. Paper presented at the Trends in information systems. Retrieved from http://dl.acm.org/.

Bowling, A. 2014. Research methods in health: investigating health and health services. McGraw-Hill Education (UK). 198 p.

Breckenridge, J. & Jones, D. 2009. Demystifying theoretical sampling in grounded theory research. *The grounded theory review*, 8(2):113-126.

Breen, M. 1987. Learner contributions to task design. *Language learning tasks*, 7:23-46.

Breuker, J. 1994. Components of problem solving and types of problems. (*In* International Conference on Knowledge Engineering and Knowledge Management organised by Springer.) Retrieved.

Broadbear, J. 2012. Essential elements of lessons designed to promote critical thinking. *Journal of the Scholarship of Teaching and Learning*, 3(3):1-8.

Brodie, P. & Irving, K. 2007. Assessment in work-based learning: investigating a pedagogical approach to enhance student learning. *Assessment & Evaluation in Higher Education*, 32(1):11-19.

Brookhart, S.M. & Nitko, A.J. 2011. Strategies for constructing assessments of higher order thinking skills. *Assessment of Higher Order Thinking Skills*:327-359.

Bryant, A. & Charmaz, K. 2007. The SAGE handbook of grounded theory. Los Angeles, CA: SAGE 656 p.

Bryman, A. 1984. The debate about quantitative and qualitative research: a question of method or epistemology? *British journal of Sociology*:75-92.

Brynard, P.A. & Hanekom, S.X. 2006. Introduction to research in management related fields. Pretoria: Van Schaik. 37 p.

Burney, S.A. & Mahmood, N. 2006. A brief history of mathematical logic and applications of logic in CS/IT. *Karachi University Journal of Science*, 34(1):61-75.

Burns, N. & Grove, S.K. 2010. Understanding Nursing Research-eBook: Building an Evidence-Based Practice. https://books.google.co.za/books?hl=en&lr=&id=Y9T3QseoHiYC&oi=fnd&pg=PP1&dq=Understanding+Nursing+Research-eBook:+Building+an+Evidence-Based+Practice&ots=_rXdWdzc4I&sig=fyZ2V2LDtUEZJGkodfkJFcfzVhU#v=onepage&q=Understanding%20Nursing%20Research-eBook%3A%20Building%20an%20Evidence-Based%20Practice&f=false Date of access: 12 December 2017.

Burrell, G. & Morgan, G. 1979. Sociological paradigms and organisational analysis. Vol. 248. London: Heinemann. 432 p.

Butler, T. 1998. Towards a hermeneutic method for interpretive research in information systems. *Journal of Information Technology*, 13(4):285-300.

Cai, J. & Lester, F. 2010. Why is teaching with problem solving important to student learning. *Reston, VA: National Council of teachers of Mathematics*:1-2.

Cândida, L., Silva Carmo, M.J.M. & Mendes, A.J. 2007. The impact of learning styles in introductory programming learning.

Carberry, A.R. 2008. Learning by teaching as a pedagogical approach and its implications on engineering education. Unpublished thesis: Tufts University 22 p.

Charmaz, K. 2006. Constructing grounded theory: A practical guide through qualitative analysis. London: SAGE. 208 p.

Charmaz, K. 2011. Grounded theory methods in social justice research. (*In* Denzin, N.K. & Lincoln V., *eds*. The Sage handbook of qualitative research. 4th ed. Thousand Oaks, CA: SAGE p. 766.)

Charmaz, K. 2014. Constructing grounded theory. London: SAGE. 416 p.

Chenail, R.J. 2011. Interviewing the investigator: Strategies for addressing instrumentation and researcher bias concerns in qualitative research. *The Qualitative Report*, 16(1):255.

Cheng, V.M. 2010. Teaching creative thinking in regular science lessons: Potentials and obstacles of three different approaches in an Asian context. (*In* Asia-Pacific Forum on Science Learning & Teaching organised by.) Retrieved.

Collins, D. 2003. Pretesting survey instruments: an overview of cognitive methods. *Quality of life research*, 12(3):229-238.

Colvin, J.W. 2007. Peer tutoring and social dynamics in higher education. *Mentoring & Tutoring*, 15(2):165-181.

Conneely, C., Girvan, C. & Tangney, B. 2012. Case Study Report for the NCCA.

Cooper, S., Dann, W. & Pausch, R. 2003. Teaching objects-first in introductory computer science. (*In* ACM SIGCSE Bulletin organised by ACM.) Retrieved.

Corbin, J. & Strauss, A. 2015. Basics of qualitative research techniques and procedures for developing grounded theory. Thousand Oaks, CA: SAGE. 429 p.

Cormen, T.H., Leiserson, C.E. & Rivest, R.L. 2009. Introduction to algorithms. 3 ed. Massacgusetts: MIT press. 1292 p.

Craft, A. 2001. An analysis of research and literature on creativity in education. *Qualifications and Curriculum Authority*:1-37.

Creswell, J.W. 2013. Research design: Qualitative, quantitative, and mixed methods approaches. Thousand Oaks, CA: SAGE. 273 p.

Creswell, J.W. 2015. 30 essential skills for the qualitative researcher. Thousand Oaks, CA: SAGE. 292 p.

Davidson, J.E. & Sternberg, R.J. 2003. The psychology of problem solving. United Kingdom: Cambridge University Press.

De Vaus, D. 2014. Surveys in social research. 6th ed. Obingdon, Oxon: Routledge. 400 p.

Deek, F.P. & Espinosa, I. 2005. An evolving approach to learning problem solving and program development: The distributed learning model. *International Journal on Elearning*, 4(4):409.

Deek, F.P. & Mchugh, J.A. 2003. Problem solving and cognitive foundations for program development: an integrated model. *Information and communication technology*.

Denzin, N.K. & Lincoln, Y.S. 2005. The dicipline and practice of qualitative research. In N.K. Denzin & Y.S. Lincoln (Eds.), Handbook of qualitative research (3rd ed. pp1-32). CA: Sage: Thousand Oaks.

Dettmer, H.W. 2007. The logical thinking process: A systems approach to complex problem solving. Milwaukee, Wisconsin: ASQ Quality Press. 16 p.

Dills, C.R. & Romiszowski, A.J. 1997. Instructional development paradigms. Educational Technology.

Donoso-Maluf, F. 2014. What is the difference between theoretical sampling and snowball sampling in qualitative research? Available from: https://www.researchgate.net/post/What_is_the_difference_between_theoretical_sampling_and_snowball_sampling_in_qualitative_research Date of access: 12 February 2018.

Dudovskiy, J. 2016. Research Methodology: Interpretivsm (interpretevist) Research Philosophy. Available from: http://research-methodology.net/research-philosophy/interpretivism/ Date of access: 13 November 2016.

Duncan, J. & Humphreys, G.W. 1989. Visual search and stimulus similarity. *Psychological review*, 96(3):433.

Duncker, K. 1945. On problem solving. *Psychological Monographs*, 58(5).

Duron, R., Limbach, B. & Waugh, W. 2006. Critical thinking framework for any discipline. *International Journal of Teaching and Learning in Higher Education*, 17(2):160-166.

Easterby-Smith, M., Thorpe, R. & Jackson, P.R. 2012. Management research. 4th ed. London: SAGE. 392 p.

Edwards-Groves, C. 2003. Building an inclusive classroom through explicit pedagogy: A focus on the language of teaching. *G. Bull & M. Anstey (Eds.), The literacy lexicon*, 2:103-121.

Ellis, R. 2003. Task-based language learning and teaching. Oxford: Oxford University Press. 389 p.

Elo, S. & Kyngäs, H. 2008. The qualitative content analysis process. *Journal of advanced nursing*, 62(1):107-115.

Ersoy, E. 2014. The effects of problem-based learning method in higher education on creative thinking. *Procedia-Social and Behavioral Sciences*, 116:3494-3498.

Eseryel, D., Law, V., Ifenthaler, D., Ge, X. & Miller, R. 2014. An investigation of the interrelationships between motivation, engagement, and complex problem solving in game-based learning. *Journal of Educational Technology & Society*, 17(1).

Estivill-Castro, V. 2010. Concrete programing for problem solving skills. (*In* International Conference on Education and New Learning Technologies (EDULEARN 2010) organised by.) Retrieved.

Eysenck, M.W. & Keane, M.T. 2000. Cognitive psychology: A student's handbook. East Suusex: Taylor & Francis. 503 p.

Falkner, N., Sooriamurthi, R. & Michalewicz, Z. 2010. Puzzle-based learning for engineering and computer science. *Computer*, 43(4):20-28.

Felder, R.M. & Silverman, L.K. 1988. Learning and teaching styles in engineering education. *Engineering education*, 78(7):674-681.

Felder, R.M. & Spurlin, J. 2005. Applications, reliability and validity of the index of learning styles. *International journal of engineering education*, 21(1):103-112.

Fitchat, L. 2016. Characterising HCI principles for evaluating the user experience of a serious game. Vanderbijlpark: North West University: Vaal Triangle Campus. (Masters dissertation) 163 p.

Franzoni, A.L., Assar, S., Defude, B. & Rojas, J. 2008. Student learning styles adaptation method based on teaching strategies and electronic media. (*In* Advanced Learning Technologies, 2008. ICALT'08. Eighth IEEE International Conference on organised by IEEE.) Retrieved.

Frerejean, J., Van Strien, J.L., Kirschner, P.A. & Brand-Gruwel, S. 2016. Completion strategy or emphasis manipulation? Task support for teaching information problem solving. *Computers in Human Behavior*, 62:90-104.

Furinghetti, F. & Morselli, F. 2009. Every unsuccessful problem solver is unsuccessful in his or her own way: affective and cognitive factors in proving. *Educational Studies in Mathematics*, 70(1):71-90.

Gelo, O., Braakmann, D. & Benetka, G. 2008. Quantitative and qualitative research: Beyond the debate. *Integrative Psychological and Behavioral Science*, 42(3):266-290.

Gillham, B. 2005. Research Interviewing: The range of techniques: A practical guide. Berkshire, England: McGraw-Hill Education (UK). 173 p.

Glaser, B. & Strauss, A. 1967. The discovery of grounded theory: Strategies for qualitative research. New York, VA: Aldine Transaction. 262 p.

Glaser, B.G. 1978. Advances in the methodology of grounded theory: Theoretical sensitivity. Mill Valley, CA: Sociology Press. 164 p.

Glaser, B.G. 2001. The grounded theory perspective: Conceptualization contrasted with description. Mill Valley, CA: Sociology Press. 232 p.

Glaser, B.G. 2002. Constructivist grounded theory? [Electronic Version]. 3(3): from http://www.qualitative-research.net/index.php/fqs/article/view/825.

Glaser, B.G. & Holton, J. 2005. Staying open: The use of theoretical codes in grounded theory. *The Grounded Theory Review*, 5(1):1-20.

Glaser, B.G. & Holton, J. 2007. Remodeling grounded theory. *Historical Social Research/Historische Sozialforschung. Supplement*, 2007(19):47-68.

Glaser, B.G. & Strauss, A.L. 2009. The discovery of grounded theory: Strategies for qualitative research. Abingdon, Oxon: Transaction publishers. 271 p.

Gomes, A. & Mendes, A.J. 2007. Learning to program-difficulties and solutions. (*In* International Conference on Engineering Education–ICEE organised by.) Retrieved.

Govender, I., Govender, D.W., Havemga, M., Mentz, E., Breed, B., Dignum, F., et al. 2014. Increasing self-efficacy in learning to program: exploring the benefits of explicit instruction for problem solving. *TD: The Journal for Transdisciplinary Research in Southern Africa*, 10(1):187-200.

Guba, E.G. 1981. Criteria for assessing the trustworthiness of naturalistic inquiries. *Educational Communication and Technology*, 29(2):75.

Guba, E.G. 1990. The paradigm dialog. Newbury Park, California Sage publications. 173 p.

Guba, E.G. & Lincoln, Y.S. 1994. Competing paradigms in qualitative research (*In* Denzin, N.K. & Lincoln Y.S., *eds*. Handbook of qualitative research. Thousand Oaks, CA: SAGE p. 105 - 117.)

Hardin, L.E. 2003. Problem-solving concepts and theories. *Journal of veterinary medical education*, 30(3):226-229.

Harrison, J.M. & Blakemore, C.L. 1989. Instructional strategies for secondary school physical education. McGraw-Hill.

Hiebert, J. & Grouws, D.A. 2007. The effects of classroom mathematics teaching on students' learning. *Second handbook of research on mathematics teaching and learning*, 1:371-404.

Hill, C.E. 2012. Consensual qualitative research: A practical resource for investigating social science phenomena. Washington, DC: American Psychological Association. 329 p.

Holton, J. 2007. The coding process and its challenges. (*In* Bryant, A. & Charmaz K., *eds*. The SAGE handbook of grounded theory. London: SAGE p. 265-289.)

Hong, N.S. 1998. The relationship between well-structured and ill-structured problem solving in multimedia simulation. *The Pennsylvania State University. The Graduate School College of Education*, Unpublished thesis.

Hood, J.C. 2007. Orthodoxy vs. power: The defining traits of grounded theory. London: SAGE. 646 p.

Houghton, W. 2004. Problems and problem solving. Higher Education Academy

Hrabovsky, G.E. no date. The Steps in Mathematical and Scientific Problem Solving.

Hurst, S., Arulogun, O.S., Owolabi, M.O., Akinyemi, R., Uvere, E., Warth, S., et al. 2015. Pretesting qualitative data collection procedures to facilitate methodological adherence and team building in Nigeria. *International journal of qualitative methods*, 14(1):53-64.

Husain, L. 2011. Getting serious about math: serious game design framework & an example of educational game.1-34.

Hwang, G.-J., Sung, H.-Y., Hung, C.-M., Huang, I. & Tsai, C.-C. 2012. Development of a personalized educational computer game based on students' learning styles. *Educational Technology Research and Development*, 60(4):623-638.

Ikayanti, R., Suratno, S. & Wahyuni, D. 2017. Critical Thinking Skill In Science On Junior High School By Problem Based Learning Models. *Pancaran Pendidikan*, 6(3).

Ismail, M.N., Ngah, N.A. & Umar, I.N. 2010. Instructional strategy in the teaching of computer programming: a need assessment analyses. *TOJET: The Turkish Online Journal of Educational Technology*, 9(2).

Jackson, R.R. & Lambert, C. 2010. How to Support Struggling Students. Mastering the Principles of Great Teaching Series. ERIC.

Janesick, V.J. 2003. The choreography of qualitative research design: Minuets, improvisations, and crystallization. (*In* Denzin, N. & Lincoln Y., *eds*. Strategies of Qualitative Inquiry. 2nd ed. Thousand Oaks, CA: SAGE p. 46-79.)

Jenkins, J.M. & Keefe, J.W. 2001. Strategies for personalizing instruction: A typology for improving teaching and learning. *NASSP Bulletin*, 85(629):72-82.

Jenkins, T. 2002. On the difficulty of learning to program. (*In* Proceedings of the 3rd Annual Conference of the LTSN Centre for Information and Computer Sciences organised by.) Retrieved.

Johnson, R.B. & Onwuegbuzie, A.J. 2004. Mixed Methods Research: A Research Paradigm Whose Time Has Come. *Educational Researcher*, 33(7):14-26.

Jonassen, D.H. 1997. Instructional design models for well-structured and III-structured problem-solving learning outcomes. *Educational Technology Research and Development*, 45(1):65-94.

Jonassen, D.H. 2010. Learning to solve problems: A handbook for designing problem-solving learning environments. Routledge.

Kafilongo, K.W.M. 2016. The use of pair-programming to enhance the academic performance of tertiary level software development students. Unpublished Thesis 98 p.

Kaiser, C.M. & Wisniewski, M.A. 2012. Enhancing student learning and engagement using student response systems. *Social Studies Research and Practice*, 7(2):137-149.

Kay, J., Barg, M., Fekete, A., Greening, T., Hollands, O., Kingston, J.H., et al. 2000. Problem-based learning for foundation computer science courses. *Computer Science Education*, 10(2):109-128.

Ke, J. & Wenglensky, S. 2010. Grounded theory. Available from: http://avantgarde-jing.blogspot.co.za/2010/03/grounded-theory.html Date of access: 22 February 2017.

Kendall, L. 2008. The conduct of qualitative interviews. (*In* Coiro, J., Knobel M., Lanshear C. & Leu D.J., *eds*. Handbook of research on new literacies. New York: Routledge p. 133-149.)

Khaleel, F.L., Ashaari, N.S., Meriam, T.S., Wook, T. & Ismail, A. 2015. The study of gamification application architecture for programming language course. (*In* Proceedings of the 9th International Conference on Ubiquitous Information Management and Communication organised by ACM.) Retrieved.

Khan, S.N. 2014. Qualitative research method: Grounded theory. *International Journal of Business and Management*, 9(11):224.

King, N. & Horrocks, C. 2010. Interviews in qualitative research. London: SAGE. 248 p.

Kirlidog, M., Van Der Vyver, C., Zeeman, M. & Coetzee, W. 2016. Unfulfilled need: reasons for insufficient ICT skills In South Africa. *Information Development*:0266666916671984.

Kong, S.C. 2014. Developing information literacy and critical thinking skills through domain knowledge learning in digital classrooms: An experience of practicing flipped classroom strategy. *Computers & Education*, 78:160-173.

Kotovsky, K. 2003. Problem solving–large/small, hard/easy, conscious/nonconscious, problem-space/problem-solver. *The Psychology of Problem Solving*.

Kuhn, T.S. 1970. The structure of scientific revolutions. 2nd ed. Chicago, IL: University of Chicago Press. 210 p.

Kvale, S. 2008. Doing interviews. London: SAGE. 160 p.

Lahiri, S.A. 2016. Guidelines to Problem Solving and Decision Making. Available from: https://www.linkedin.com/pulse/guidelines-problem-solving-decision-making-shankar-achintya-lahiri Date of access: 1 March 2018.

Lahtinen, E., Ala-Mutka, K. & Järvinen, H.-M. 2005. A study of the difficulties of novice programmers. (*In* ACM SIGCSE Bulletin organised by ACM.) Retrieved.

Lampert, M. 1985. How do teachers manage to teach? Perspectives on problems in practice. *Harvard Educational Review*, 55(2):178-195.

Laverty, S.M. 2003. Hermeneutic phenomenology and phenomenology: A comparison of historical and methodological considerations. *International journal of qualitative methods*, 2(3):21-35, 2003/09/01.

Lawrence, J. & Tar, U. 2013. The use of grounded theory technique as a practical tool for qualitative data collection and analysis. *The Electronic Journal of Business Research Methods*, 11(1):29-40.

Le, N.-T., Loll, F. & Pinkwart, N. 2013. Operationalizing the continuum between well-defined and ill-defined problems for educational technology. *IEEE Transactions on Learning Technologies*, 6(3):258-270.

Lester, S. 1999. An introduction to phenomenological research. Available from: http://www.rgs.org/nr/rdonlyres/f50603e0-41af-4b15-9c84-ba7e4de8cb4f/0/seaweedphenomenologyresearch.pdf Date of access: 25 October 2017.

Lichtman, M. 2006. Qualitative research in education: A users guide. London: Sage. 249 p.

Lichtman, M. 2010. Qualitative research in education: A users guide. Thousand Oaks, CA: SAGE. 263 p.

Lincoln, V. & Guba, E. 1985. Naturalistic inquiry. Beverly Hills, CA: SAGE. 416 p.

Lincoln, Y.S., Lynham, S.A. & Denzin, N.K. 2011. The SAGE handbook of qualitative research. (*In* Denzin, N.K. & Lincoln Y.S., *eds*. California: Sage p. 766.)

Mack, L. 2011. The philosophical underpinnings of educational research. *Polyglossia*, 19(February 2011):1-11.

Malouff, J.M. 2011. Teaching Problem Solving to College Students. *Online Submission*.

Maravić Čisar, S, Radosav, D., Pinter, R. & Čisar, P. 2011. Effectiveness of Program Visualization in Learning Java: a Case Study with Jeliot 3. *Int. J. of Computers, Communications & Control*, VI (2011)(4):668-680.

Marshall, C. & Rossman, G.B. 2014. Designing qualitative research. 6th ed. Thousand Oaks, CA: SAGE. 352 p.

Mason, J. 2002. Qualitative researching. 2nd ed. London: SAGE. 223 p.

Maxwell, J.A. 2008. Designing a qualitative study. (*In* Bickman, L. & Debra J.R., *eds*. The SAGE handbook of applied social research methods. 2nd ed. Thousand Oaks, CA: SAGE p. 214 - 253.)

Maxwell, J.A. 2012. Qualitative research design: An interactive approach. Vol. 41. Thousand Oaks, CA: SAGE. 218 p.

Mayer, R.E. 2008. Learning and instruction. *Upper Saddle River, NJ: Pearson Merrill Prentice Hall*.

Mayer, R.E. 2012. Problem solving. *In Encyclopedia of Human Behavior*, 2:181-186.

Mayer, R.E. & Hegarty, M. 1996. The process of understanding mathematical problems. *The nature of mathematical thinking*:29-53.

Mayer, R.E. & Wittrock, M.C. 2006. Problem Solving in P. A. Alexandra and P. H. Winne (eds.). *Handbook of Educational Psychology*, 2:287.

Mccann, T.V. & Clark, E. 2003. Grounded theory in nursing research: part 1– methodology. *Nurse researcher*, 11(2):7-18.

Mckay, J., Marshall, P. & Hirschheim, R. 2016. The design construct in information systems design science (*In* Willcocks, L.P., Sauer C. & Lacity M.C., *eds*. Enacting research methods in Information Systems. Switzerland: Springer International Publishing p. 11-42.)

Medico, D. 2005. Introduction to qualitative analysis of in-depth interviews. Available from: https://www.gfmer.ch/Medical_education_En/PGC_RH_2005/pdf/Qualitative_analysis.pdf Date of access: 25 October 2017.

Mendonça, A., De Oliveira, C., Guerrero, D. & Costa, E. 2009. Difficulties in solving ill-defined problems: A case study with introductory computer programming students. (*In* Frontiers in Education Conference, 2009. FIE'09. 39th IEEE organised by IEEE.) Retrieved.

Merriam, S.B. 1998. Qualitative research and case study applications in education: Revised and expanded from "Case Study Research in Education.". ERIC. 275 p.

Merriam, S.B. 2002. Qualitative research in practice: Examples for discussion and analysis. Jossey-Bass Inc Pub. 464 p.

Merrick, K.E. 2010. An empirical evaluation of puzzle-based learning as an interest approach for teaching introductory computer science. *IEEE Transactions on education*, 53(4):677-680.

Merriman-Webster. 2017. Skill. Available from: https://www.merriam-webster.com/dictionary/skill Date of access: 3 June 2017.

Merriman, S.B. 2001. Qualitative research and case study applications in education. San Francisco, CA: Wiley. 275 p.

Midgley, G. 2000. Systemic intervention: Philosophy, methodology and practice. New York, VA: Plenum. 447 p.

Mohorovičić, S. & Strčić, V. 2011. An overview of computer programming teaching methods. (*In* XXII Central European Conference on Information and Intelligent Systems organised by.) Retrieved.

Morgan, G. 1980. Paradigms, metaphors, and puzzle solving in organization theory. *Administrative Science Quarterly*, 25(4):605-622.

Moritz, S.H., Wei, F., Parvez, S.M. & Blank, G.D. 2005. From objects-first to design-first with multimedia and intelligent tutoring. *ACM SIGCSE Bulletin*, 37(3):99-103.

Mosston, M. & Ashworth, S. 2002. Teaching physical education.

Mouton, J. 2005. How to succeed in your Master's & Doctoral studies. Pretoria: Van Schaik. 280 p.

Myers, M.D. 1997. Qualitative research in information systems. *MIS Quarterly*, 21(2):241-242.

Myers, M.D. 2009. Qualitative research in business & management. London: SAGE. 284 p.

Nag, S., Katz, J.G. & Saenz-Otero, A. 2013. Collaborative gaming and competition for CS-STEM education using SPHERES Zero Robotics. *Acta Astronautica*, 83:145-174.

Neuman, W.L. 2013. Social research methods: Pearson new international edition: Qualitative and quantitative approaches. England: Pearson. 594 p.

Newton, D.P. 2011. Teaching for understanding: What it is and how to do it. Routledge.

Nunan, D. 2006. Task-based language teaching in the Asia context: Defining'task'. *Asian EFL journal*, 8(3).

Nuutila, E., Törmä, S. & Malmi, L. 2005. PBL and computer programming—the seven steps method with adaptations. *Computer Science Education*, 15(2):123-142.

Oddie, A., Hazlewood, P., Blakeway, S. & Whitfield, A. 2010. Introductory problem solving and programming: Robotics versus traditional approaches. *Innovation in Teaching and Learning in Information and Computer Sciences*, 9(2):1-11.

Onwuegbuzie, A.J. & Collins, K.M. 2007. A typology of mixed methods sampling designs in social science research. *The Qualitative Report*, 12(2):281-316.

Orlikowski, W.J. & Baroudi, J.J. 1991. Studying information technology in organisations: Research approaches and assumptions. *Information Systems Research*, 2(1):1-28, Mar.

Orlikwoski, W.J. & Baroudi, J.J. 1991. Studying information technolgy in organisations: Research approaches and assumptions. *Information Systems Reserarch*, 2(1):1-28.

Owens, A.D. 2016. 8 Best Alternative Teaching Methods. Available from: http://edu.stemjobs.com/8-best-alternative-teaching-methods/ Date of access: 1 March 2018.

Oxford, R. 2006. Task-based language teaching and learning: An overview. *Asian EFL journal*, 8(3).

Parahoo, K. 2014. Nursing research: Principles, Process and Issues. 3 ed. United Kingdom: Palgrave Macmillan.

Park, S.I. & Jang, S.-Y. 2010. Critical factors influencing problem-solving ability in online learning environments.

Patton, L.D., Renn, K.A., Guide, F., M. & Quaye, S.J. 2016. Student development in college: Theory, research, and practice. Third edition ed. San Francisco: John Wiley & Sons. 560 p.

Patton, M.Q. & Cochran, M. 2002. A guide to using qualitative research methodology. *Medecins Sans Frontiers. Retrieved February*, 14:2014.

Paul, R. & Elder, L. 2004. Critical and creative thinking. *Dillon Beach, CA: The Foundation for Critical Thinking*.

Perkins, D.N., Hancock, C., Hobbs, R., Martin, F. & Simmons, R. 1986. Conditions of learning in novice programmers. *Journal of Educational Computing Research*, 2(1):37-55.

Peter, E.E. 2012. Critical thinking: Essence for teaching mathematics and mathematics problem solving skills. *African Journal of Mathematics and Computer Science Research*, 5(3):39-43.

Peters, A. & Pears, A. 2012. Students' experiences and attitudes towards learning computer science. (*In* Frontiers in Education Conference (FIE), 2012 organised by IEEE.) Retrieved.

Pham, R. 2016. Improving the software testing skills of novices during onboarding through social transparency. Berlin: Logos Verlag Berlin GmbH. 216 p.

Pithers, R.T. & Soden, R. 2000. Critical thinking in education: A review. *Educational research*, 42(3):237-249.

Polit, D.F. & Beck, C.T. 2010. Generalization in quantitative and qualitative research: Myths and strategies. *International journal of nursing studies*, 47(11):1451-1458.

Polya, G. 2014. How to solve it: A new aspect of mathematical method. Princeton University Press.

Prensky, M. 2001. The games generations: How learners have changed. *Digital game-based learning*:1-26.

Prensky, M. & Berry, B.D. 2001. Do they really think differently? *On the horizon*, 9(6):1-9.

Prince, M. 2004. Does active learning work? A review of the research. *Journal of Engineering Education*, 93(3):223-231.

Ravitch, S.M. & Riggan, M. 2016. Reason & Rigor: How conceptual frameworks guide research. Thousand Oaks, CA: SAGE. 264 p.

Razeghi, A. 2008. The riddle: Where ideas come from and how to have better ones. John Wiley & Sons.

Rebello, N.S., Cui, L., Bennett, A.G., Zollman, D.A. & Ozimek, D.J. 2007. Transfer of learning in problem solving in the context of mathematics and physics. *Learning to solve complex scientific problems*:223-246.

Ricoeur, P. 1992. Oneself as another. University of Chicago Press.

Riley, S., Campbell, B. & Farrows, M. 2004. The Flinders website. Available from: http://ehlt.flinders.edu.au/education/DLiT/2004/13DLT/ExplicitTeach.htm Date of access: 11 November 2017.

Robins, A., Rountree, J. & Rountree, N. 2003. Learning and teaching programming: A review and discussion. *Computer Science Education*, 13(2):137-172.

Ryhammar, L. & Brolin, C. 1999. Creativity research: Historical considerations and main lines of development. *Scandinavian journal of educational research*, 43(3):259-273.

Sadeghi, R., Sedaghat, M.M. & Ahmadi, F.S. 2014. Comparison of the effect of lecture and blended teaching methods on students' learning and satisfaction. *Journal of Advances in Medical Education & Professionalism*, 2(4):146.

Saldaña, J. 2014. Thinking qualitatively: Methods of mind. Thousand Oaks, CA: SAGE. 215 p.

Saldaña, J. 2015. The coding manual for qualitative researchers. Sage.

Salter, L. 1990. Managing Technology: Social Science Perspectives. Garamond Press.

Sarantakos, S. 2012. Social research. 4th ed. London: Palgrave Macmillan.

Sarpong, K.a.-M., Arthur, J.K. & Amoako, P.Y.O. 2013. Causes of Failure of Students in Computer Programming Courses: The Teacher-Learner Perspective. *International Journal of Computer Applications*, 77(12).

Sauders, M., Lewis, P. & Thornhill, A. 2003. Research methods for business students. *New Jersey*, 4:100-109.

Saunders, M., Lewis, P. & Thornhill, A. 2009. Research methods for business students. 5th ed. England: Pearson Edcuation. 614 p.

Saunders, M.N., Lewis, A. & Thornhill, A. 2011. Research methods for business students. 5th ed. England: Pearson Education. 614 p.

Schmeck, R.R. 2013. Learning strategies and learning styles. Springer Science & Business Media.

Schoenfeld, A.H. 2010. How we think. *Cuadernos de Investigación y Formación en Educación Matemática*(10).

Schorr, R.Y. & Amit, M. 2005. Analyzing student modeling cycles in the context of a 'real world'problem. *International Group for the Psychology of M athematics Education*:137.

Schwandt, T.A. 2001. Dictionary of qualitative research. *In*: Thousand Oaks, CA: Sage.

Scotland, J. 2012. Exploring the philosophical underpinnings of research: Relating ontology and epistemology to the methodology and methods of the scientific, interpretive, and critical research paradigms. *English Language Teaching*, 5(9):9.

Scriven, M. & Paul, R. 2007. Defining critical thinking. The Critical Thinking Community: Foundation for Critical Thinking. Retrieved January 2, 2008. *In*.

Seidman, I. 2013. Interviewing as qualitative research: A guide for researchers in education and the social sciences. 4th ed. New York, VA: Teachers college press. 178 p.

Selingo, J. 2015. Why are so many college students failing to gain job skills before graduation. *The Washington Post*.

Shabanah, S. & Chen, J.X. 2009. Simplifying algorithm learning using serious games. *Proceedings of the 14th Western Canadian Conference on Computing Education: 2009. ACM*:34-41.

Shirsath, S. 2014. Teaching and Learning System for Programming-A Literature Study to Develop Logic for Programming through Puzzles.

Smartdraw, U. 2018. Flowchart. Available from: https://www.smartdraw.com/flowchart/ Date of access: 4 January 2018.

Snyder, L.G. & Snyder, M.J. 2008. Teaching critical thinking and problem solving skills. *The Journal of Research in Business Education*, 50(2):90.

Stanescu, I.A., Stefan, A. & Roceanu, I. 2011. Interoperability in Serious Games. *eLearning & Software for Education*.

Sternberg, R.J. & Lubart, T.I. 1995. Defying the crowd: Cultivating creativity in a culture of conformity. Free Press.

Strauss, A. & Corbin, J. 1998. Basics of qualitative research: Procedures and techniques for developing grounded theory. Thousand Oaks, CA: SAGE. 312 p.

Strydom, H., Fouché, C.B. & Delport, C.S.L. 2002. Research at grass roots: For the social sciences and human service professions. 2nd ed. Pretoria: Van Schaik. 493 p.

Sutton, M.J. 2003. Problem representation, understanding, and learning transfer implications for technology education.

Swartz, R.J. 2000. Towards developing and implementing a thinking curriculum. (*In* ponencia, First Annual Thinking Qualities Initiative Conference Hong Kong organised by.) Retrieved.

Swartz, R.J. & Perkins, D.N. 2016. Teaching thinking: Issues and approaches. Routledge.

Tan, B.K. & Rahaman, H. 2009. Virtual heritage: Reality and criticism. *CAAD Futures: 2009*:143-156.

Taylor, P.C. & Medina, M.N.D. 2013. Educational research paradigms: From positivism to multiparadigmatic. *The Journal of Meaning-Centered Education*, 1(2):1-16.

Teague, M.M. 2011. Pedagogy of introductory computer programming: a people-first approach. Queensland University of Technologyp.

Tesch, R. 2013. Qualitative research: Analysis types and software. Abingdon, Oxon: Routledge. 344 p.

Thomas, P.Y. 2010. Towards developing a web-based blended learning environment at the University of Botswana. Pretoria: University of South Africa. (Doctoral thesis) 535 p.

Thompson, S. 1997. Where do I begin? A problem solving approach in teaching functional programming. (*In* International Symposium on Programming Language Implementation and Logic Programming organised by Springer.) Retrieved.

Thompson, S., Barnes, D. & Fincher. 1997. Introductory problem solving in computer science. *Introductory Problem Solving in Computer Science*.

Thuraisingam, T.G., Siraj, S., Naimie, Z., Abuzaid, R.A. & Halili, S.H. 2014. The teaching of critical and creative thinking skills in the English Language classroom in Malaysia. *Management and Technology in Knowledge, Service, Tourism & Hospitality*:137.

Tie, H. & Umar, I.N. 2010. The impact of learning styles and instructional methods on students' recall and retention in programming education. (*In* Proceedings of the 18th International Conference on Computers in Education. Putrajaya, Malaysia: Asia-Pacific Society for Computer in Education organised by.) Retrieved.

Torrance, E.P. 1969. Creativity. What Research Says to the Teacher, Series, No. 28.

Trochim, W.M. 2006. Units of analysis.

Van Teijlingen, E. 2014. Semi-structured interviews. Available from: https://intranetsp.bournemouth.ac.uk/documentsrep/PGR%20Workshop%20-%20Interviews%20Dec%202014.pdf Date of access: 26 October 2017.

Vihavainen, A., Airaksinen, J. & Watson, C. 2014. A systematic review of approaches for teaching introductory programming and their influence on success. (*In* Proceedings of the tenth annual conference on International computing education research organised by ACM.) Retrieved.

Walsham, G. 1995a. The emerge of interpretivism in IS research. *Information Systems Research*, 6(4):376-394.

Walsham, G. 1995b. Interpretive case studies in IS research: Nature and method. *European Journal of Information Systems*, 4(2):74-81.

Warshauer, H.K. 2015. Productive struggle in middle school mathematics classrooms. *Journal of Mathematics Teacher Education*, 18(4):375-400.

Welman, J.C. & Kruger, S.J. 2001. Research methdodology for the business and administrative sciences. 2nd edition ed. Cape Town: Oxford University Press. 323 p.

Wengraf, T. 2001. Qualitative research interviewing: Biographic narrative and semi-structured methods. Sage.

Wertheimer, M. 1959. Productive thinking. *New York: Harper & Row*.

Whimbey, A., Lochhead, J. & Narode, R. 2013. Problem solving & comprehension. Routledge.

Whiting, M. & Sines, D. 2012. Mind maps: Establishing 'trustworthiness' in qualitative research. *Nurse researcher*, 20(1):21-27.

Wikibooks. 2017. Computational Thinking, Problem-Solving, and Programming. Available from: https://en.wikibooks.org/wiki/IB/Group_4/Computer_Science/Computational_Thinking,_Problem-Solving,_and_Programming Date of access: 3 March 2018.

Yampinij, S. & Chaijaroen, S. 2010. The development knowledge construction model based on constructivist theories to support ill-structured problem solving process of industrial education and technology students. (*In* Education and Management Technology (ICEMT), 2010 International Conference on organised by IEEE.) Retrieved.

Zeeman, M.J. 2014. Modelling the factors that influence computer science students' attitude towards serious games in class. *Unpublished dissertation, NWU*.

ADDENDUM A

Raw interview data are presented in this addendum.

Question 1	***Tell me about your experience when you teach problem solving in the computer programming class?***
Responses L1	"My experience on teaching problem solving is that students struggle in programming. Sometimes students struggle because there is no proper guide and approach that a lecturer can use to help a student with a problem."
L2	"Students make assumptions instead of thinking logically. They don't understand that they have to instruct machines. Each time they must put up instruction they don't even take further steps to complete the instructions, they assume that you understand. They think the machine is a human being. Assumption is a biggest problem."
L3	"I have various experiences. Getting a specific concept understood and applicable. This ranges between the levels of students I teach."
L4	"I have been involved in 2 programming language PL/SQL and SQL. My experience with the students has been that most of them struggle to get the concept. Particularly logical thinking."
L5	"Students don't know how to solve problems. The reason is that they are not been trained meaning students are feed, they are just being given what to eat. They don't know how to grow their own food. When you ask a simple question, they don't understand the question. Their minds are not open to certain type of questions. Sometimes I need to go back to high school material explain some stuff that they did and come back to our own material**.** To make the link that what they did there, was to be used here. The problem is don't remember as they were trained to cram. They do not have the right training."
Question 2	***What exactly are students struggling with when solving problems in the computer programming class?***
Responses L1	"Students struggle because they don't have programming logic. And from high school they come to university and there is a gap between. I think they should have foundation first then they can go to programming."
L2	"Assumption is a problem as students do not complete the instructions. Logic thinking."
L3	"Students struggle to think out of the box. When they have a problem in real live they are not able to see what tools, theory or concept can be used to solve that problem. More of having eye to end coordination. There is a problem, the moment students

	see a problem, and they panic and tend to rely on solutions that are done on the internet. Instead of building from starch from the knowledge that they have acquired.”
L4	“Students are lazy, most of them don’t want to do the job. They will leave it until late and do rash job and most of them give them to the rest of the students. You will find 5 exact same assignments which tells you that one person only did the work.”
L5	“Programming is logic. Logic comes from mathematics. People say those big formula I will not use it in my life. Which is true. We were trained to use those formula to solve certain problems. There are certain things we can reach because we were trained. The brain is a muscle. If a student doesn’t train his brain through those type of subjects such as mathematics, there is a level that a student cannot reach. Think how to solve a basic a problem without looking for a solution or written solution. If you ask him, sit down and don’t touch a computer and write down the steps how you are going to solve the problem they won’t know.”
Question 3	***What approach or approaches do you use when you teach students to solve problems in the computer programming class?***
Responses L1	“I divide my students into a group of 2 or 3 students. Once they work in team they can help each other. I used a guide that can assist the students in steps to follow and solve problems.”
L2	“It’s not a method as per say, one method is that I took an ATM, try to find out from them what kind of processes they do themselves in order for them to rewrite that program (a reverse engineering). For example, if you go to an ATM as an approach, what is it that the ATM ask you? What kind of a concept is it? I take small things based on their own syllabus. At times, I take concept to student and they get lost completely. But with more practicals that they do every day, out of them I show them which of the concept that they learned or going to learn is in that concept. Basically, I use real live situations.”
L3	“The students are at the center of learning, to help them I make them work for everything. I do not spend time teaching but making sure that students apply the concept that was given. The teaching takes place when the students have tested something by himself. From the 100% of my class time, I spend hardly 50% teaching, and the other 50% making sure that I work with the students the solutions to the problems that they have acquired. The lack of students thinking, performing and working all the processes to the solution is that they are given solutions are all they do is cram them. If the students could not be given the solutions but work them they would remember the processes. In the manner, they would be actually solving problems. If solutions are given during the time that they are learning, then they will be looking for read-made solutions. It is important that during the

	learning process, the students should be involved. When students for example are grouped, other decide to relax and the other is the hard worker. You are going to give marks to the group. You think that that person has learned while they did not participate. The only time you can tell, is when the person is in front of you. From time to time, you spot crucial point in the learning to assess see how the student is performing. You can be able to say there is learning taking place or there is a lot of students who are crashing. It is much more of a quick assessment to see if the basics are there. If the foundation is correct, then the students can go on the internet and do some self-learning. If the foundation is not there, then they will not be able do thing even if he downloads solutions. We focus on the foundations in class, then the advanced concepts can be done a little if there's enough time then go deeper. The grounds should be strong for the students to build up."
L4	"I use a combination of approaches. I use explicit teaching, where I would introduce the concept. I would incorporate task teaching, where I would do an example with them then give them an exercise. Which is similar to the example, just to see if they have understood the concept. Many of them are struggling with logical thinking. Those of them that are still struggling with the concept, I would then group them. Some of them are shy to ask me or tell me that they don't understand. So they would be open to talk to their peers in a group. They get to help each other out. We have tutors who helps, so that will be peer tutoring. If they can't ask me, they can ask the tutor during tutoring."
L5	"When I am in class, first we don't take notes. We discuss the problem. By the time we go to the computer, by the time you have an error I don't fix it. You must figure it out how to fix it. By the time to fix the problem you understand how it works. Before I give them a problem I ask them how to solve them. They need to be able to say it loud. If they can't be able to say it loud how to solve a problem then, they can't be able to solve it. Sometimes I take my paper and pen and give 1% to a person who say it right. If a question is more difficult I give 2%. Every time they come to class they are prepared. I don't allow them to touch a computer unless they know how to solve a problem."
Question 4	***Are you using any problem-solving guidelines when you teach students to solve problems in the computer programming class?***
Responses L1	"I am using different guidelines like Pair Programming adopted by different authors including mine which is effective considering the type of students that I have in my class. I can change the guideline to suit the type of students that I have in my class."
L2	"I don't have any guidelines that I use. I try to come up with more problems for them without any guide. For example, I use

	open-ended question to see how each one of them will approach them. At times I use closed-ended questions, then I know I gave them a guideline of how to solve this. A class consists of various students, people who don't think the same way. A class consists of students from different areas. I try as much as possible on each one of them to bring them to a certain level."
L3	"I have developed the guidelines based on my experience. I have not checked how others have written. From other people's experience with the same type of students. I go around and see how to adapt, and if there is something that is not working. To me it is natural to try another one. My new approach is that by each class, a student must touch and write something on that computer. If that can be achieved first, then you can test if a student has done something meaningful. That gap that the students finished the course and they have not done any practical work. The student must get in touch with the material, secondly then try to assess the quality of the work they are doing. If the student is present in class, make sure they do something. Then can the student be able to achieve something on a singular basis not group basis. If you ask students if they have any problems, then they do have any problems. When they are not attended individually, then you carry on with the attitude of not caring what they are doing unless they are passing. Some students do not do anything as they know to cheat the system. If students are not attended individually they tend to continue the attitude of not caring what they are doing if they are passing."
L4	"Not necessarily, I just use what works. It depends with the group of students that you have. It also depends with the number of the group. How big or how small. The bigger the group it becomes difficult to reach everyone. The smaller it is, not only to manage the class and to reach everyone. I don't have my own approach guideline that I have developed. I will use what is already available. I will group them, if the class is small enough. I will walk around after introducing the concept and doing the example with them. Then I would give them a task to do on their own and walk around to see the progress. I will notice that some of the have a problem I would go through with them and sometimes time them. I would give them a task and at times tell them at this time they must be finished. That's where you realized that most of them have the ability to do it, they just don't want to. if you give them a time frame and also tell them it will count for marks then you will see them putting a little bit of effort than they would any other time."
L5	"It depends on the level of the problem, it depends how to solve it. If the level is high I can give those tips. The whole ideas is that they must be able to think. If I ask them how to make a cup of tea, they must tell me what I need. They must be able to give those type of steps before they sit on the computer and write.

	They know all those programming key words, but they don't know how to use them. But they don't know what they are going to do with them. When you have all these steps, to apply them it becomes easier. When the problem is difficult I give them tips, otherwise they must be able to have the steps how to solve it. I am against using other people's methods because you cannot guarantee me that when you that methods. Those students are in the same conditions as mine, therefore I tested your method where your students have different set up as mine. You should not think that method will work on my students. If the method has been tested on students who had proper mathematics and expect me to test the same method on students who don't have proper mathematics and expect the same results. Whatever the situation is to me, I adapt to it."
Question 5	***Does problem solving guidelines help students to perform better in solving problems in the computer programming class?***
Responses L1	"Yes, the guideline that I am using helped and increased the students' performance. Mostly because I use pair programming, for groups work."
L2	"In terms of performance, when it starts it's always a problem. If I started my students at Level 1 and go up with them. I see improvement as they start to be independent. But when I am receiving students from a lower level to my level with a kind of approach I am using therefore I have a problem because some of them are spoon-feed. Whatever the lecturer is saying its right to them they do not challenge him/her. If you challenge them, you can get anything from them because expectation is "try to show us first." Hence, I don't use that approach that is why I use the open-ended questions. I even test them with wrong answer if they agree then I start showing them. In between this, this is where you see students that becomes involved in a class. I challenge students so that they can be independent."
L3	"Yes, because when I compare students that go through my modules and those that go through other modules. I do not believe the pass rate tells how the class was presented. How I evaluate my modules is when I get students thanking me personally after completing the module, this is when you tell that you did a good job. Student who are in the work field come back and thank me as the work done in my module has helped them. There is two ways to look at why students don't perform well. One is how the concept is presented in class and the other is the student."
L4	"To a certain degree, it has helped. Especially if a group is smaller, then you have a chance and opportunity to help everyone if the struggling the most will have the chance to reach

	them. When I had a smaller group, my pass rate or performance of the students was much better than when I had a larger group. When we have a large group, I tend to group them. In a group, you will find that there is one strong student who understands and then the rest are just riding on his back. So, you are not sure if everyone has participated in the work unless you do presentations. When I do presentations that is where I could pick up who has contributed to the project. If the group is large, even those presentations don't help much. You don't always have the time to go back and help those students who you have picked up that they still have problems."
L5	"Yes, when I started asking them to solve problems in three groups I will get one students from each who can solve the problem. At this stage I can get 3 students per group that can give me solutions. They now know that they need to prepare for the class before they come. They try to make connection on what they need to do and those works. I do class test which are surprise tests which they all failed the 1st test, 2nd test they almost got the test, 3rd time they all got good marks. Because they were now prepared. I ask questions based on what I will be covering that time, they come prepared. It works."
Question 6	***How do you present programming problems in your class?***
Responses L1	"There are many ways, firstly I try to explain the concept first to the students. Then I give the method on how to solve that problem using an example related to that topic. There after I go through them using step- step."
L2	"Every time I come to class and ask them, if there anyone with a problem they do say. I compile a test or exercise for them and pretend that it contributes to their final mark. Then you will realize that 50% of them would be flowing in answering that question. 80% of them would be lost, but 20% will be trying something. Students fail to ask questions and open books." ***Follow up question: Do you compile your questions for coming to class?*** This year I came up with different approach. I designed an application that covers 3 chapters. All I am saying to them is go and read. What they see on the application is for the 3 chapters. The will not see any code but a program doing something. They don't know how it is done. I will then start with the Chapter 1, indicating with an example from a program. Using the tools that are indicated in chapter 1 then I can be able to help them with their errors. My intention is to make the students see each chapter as they move from the program. But then the students complained about the method, then I had to do it in black and white. The students believe in reading it as a case study then starting to tackle it from there."

L3	“There are 2 ways. The first one is real live issues and try to simplify them in the simplest form. It that way it makes it easier. I make assumptions to remove the things that are not fitting the situation. For instance, when we do programming. That’s assume we want to write a program that can do online registration. That alone is a complex thing, but we will limit the scope to registering one student for one subject. At least they know how one is done and later they can expect. Other approach, to take problems that are already made, in the textbooks. Those problems we can try to solve them, but the reason why we don’t sit well with our normal live. For instance, these books are written in the US, UK or Canada and the case study in there not applicable to what we do. We use them as example, but then we try to make the concept understood. We try to make real live situation of our normal live environment and then we give them that to represent.”
L4	“I start by introducing the concept, for instance we are dealing with cursors, functions or procedures. I will start by explaining what a function is, and how is it different to an anonymous block. After explaining the concept, I will give them an example, go through the example with. I give them a task to do on their own. The task would be timed or do the task on the remainder of the lesson. If they are not able to finish on that timed they would take it home. Or ask the tutor to help them during the tutor lessons. It the combination of approaches that I use.”
L5	“I link the problems with real live situations. If there is a concept I want to explain I first bring it as an example from a real live situation and show them how they solve it or use that. People can understand what they can see or touch. I believe you will get better respond from the students.”
Question 7	***How do you cater for different types of learning styles?***
Responses L1	“All types of students I do try to accommodate, visual students we try a method in a program via a software e.g. solving a method using Java if we are using java. So, they can see when we execute the problem. Then other students we use scenarios that are real-time.”
L2	“I do not start with high level explanation first. I simplify for those how don’t know the concept. I come up with 3 examples that defines the concept. Showing the one who doesn’t understand and showing the one who understand in order for them to be in the middle. Then I choose combination of that concept and see if you use it that way, what happens. In the lower lever, middle level and higher level. To check who respond to which level and what. That prepares me to see when I treat this level, I must remember the other level. I observe as much as I give out information. Sometimes when you have moved on from Concept A to B or C, then students realize how Concept A comes to B. they didn’t understand it when making examples.

	Now and then they can take you back. I interconnect concepts.” ***Follow up question: Has the method worked?*** “Sometimes, yes. I have realized that some students understand at the end of the semester, where time is short. For example, in my first class I address 3 concepts which is 3 chapters. The second class I address again other 3 concepts. Then maybe I will be left with one chapter. This gives us time to go in detail of each chapter.”
L3	“To cater these type of students is difficult. What I do in programming I make use of LMS, blackboard. I put up videos, power point slides, all class notes. At the end of the class put that up online, even that student who was not in class, or slow or doesn’t like listening to the lecturer he can still go to the notes and see what was done. I use LMS (Learning Management System) that is the effective way to me. In PLS/SQL I gave them a link to a tutorial website. Those who like tutorial online, there is a link for tutorials, videos, PowerPoint slide, the ones I use on class. Everything is up there to the convenience of different preferred learning mode.”
L4	“Most times, as lecturers we are guilty of teaching the way you learn. If you learn through reading, you will give your students a lot of reading materials. If you learn through visuals, you will give your students visuals. I have started incorporating both traditional way (face-face) of teaching and visuals. I would go online, on YouTube and find videos that explains the concept that we have dealt with in class. I would upload that on the LMS system then they can download the video and watch it. If they didn’t understand something, then can pause, rewind, and listen to it over and over. As opposed to listening to me and when the lesson is over it’s done. Most of the student don’t come during consultations to me until it’s too late and they realized that the ship is sinking. I have not done my own videos yet. I use videos a lot, for those who learn by seeing. I would give them reading materials as well and task to do. Some of them through been actively involved.”
L5	“I will explain, draw on the board. Some go on google and see what they can get. I always want to mix. I try to adapt to all students that I have. Not always possible but whenever is possible I try. Especially the talking and the drawing.”