The Predictive Validity of Scores Obtained in First Semester Examination on Performance in Introduction to Programming Systems

BY

DAVID MUTAMBARA

2017

The Predictive Validity of Scores Obtained in First Semester Examination on Performance in Introduction to Programming Systems

BY

DAVID MUTAMBARA

201640095

A mini-dissertation submitted to the

Faculty of Education

in partial fulfillment of the requirements for the degree of

Master of Education (Research Methodology) in the

Department of Educational Psychology and Special Education

at the University of Zululand.

SUPERVISOR:	**PROFESSOR D.R. Nzima**
CO- SUPERVISOR(S):	**Dr D. Kent**
	Mr A. Chibisa

Date Submitted: December 2017

DECLARATION

I, **David Mutambara** (Student Number **201640095**), declare that this mini-dissertation, which is submitted in partial fulfilment of the requirements of the degree of Master of Education to the University of Zululand, is my own work in design and execution and has not been previously submitted by me for a degree at any university, and that all sources I have used have been indicated and acknowledged by means of a complete reference.

----------------------------- **-------------------------------**

SIGNATURE **DATE**

ACKNOWLEDGEMENT

This study would not have been possible without the help of many people. I really thank you all.

Firstly, I would like to express my appreciation for the support and encouragement I received from my supervisor Professor D.R. Nzima. His experience and critical insight has been of great value in completing this dissertation.

Dr Kent's (co-supervisor) support and encouragement has taken me this far. I am grateful for the job well done.

I acknowledge my advisor, Mr A. Chibisa, for his guidance throughout this research. With a feeling of humility, I appreciate his unfailing encouragement and all the time that he invested in this research. I thank him.

I am deeply indebted to the late Prof P.T. Sibaya for suggesting this topic. He will be forever missed.

My classmates, Mrs T. Chihambakwe and Mrs Ntshangase, thank you for the support.

I extend my sincere gratitude to my family members. My late father, Festo, my mother, Chipo, and Charity, their encouragement has taken me this far.

Constance and Nenyasha - thank you for encouraging me and giving time to study. God bless.

To God be glory.

DEDICATION

This study is dedicated to my wife Farai and our son Nenyasha for their continued support and encouragement. May this work be a source of inspiration throughout their lives.

ABSTRACT

Introduction to Programming Systems is considered to be very difficult and has a very high average failure rate of between 30% and 40%. Some researchers have studied the characteristics of students who pass Introduction to Programming Systems without struggling and used those characteristics as predictors of success in Introduction to Programming Systems. This research studied the relationship between selected predictors (Calculus, Discrete Mathematics, Classic Mechanics and General Chemistry) and Introduction to Programming Systems. The study adapted a case study and correlation research design. A sample size of 399 was selected using a non-probability sampling method called convenient sampling. Data from only one university were used. SPSS's Pearson correlation and multiple regression was used to analyse the collected data. The results showed that there is a positive correlation between the criterion (Introduction to Programming Systems) and the predictors. Multiple regression results showed that the ordinal strength of predictor was as follows: Calculus, Discrete Mathematics, Classic Mechanics and General Chemistry. Only General Chemistry had an insignificant effect on the criterion. The variation was 34%.

Table of Contents

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION TO STUDY

At tertiary level, Computer Programming courses are among the most essential parts of the curriculum to be studied in fields like Science, Mathematics, Engineering, and Commerce. Important as it is, Programming is considered to be very difficult and complex. Programming involves activities such as analysis, developing understanding, generating algorithms, verification of requirements of algorithms, including their correctness and resources consumption, and coding. The corner-stone of Programming is coding, which is the writing, testing, de-bugging/troubleshooting, and maintaining the source of code of computer programs. When a person is Programming, he or she needs to analyse and understand the problem, create an algorithm to solve the problem, and write executable statements that the computer can understand, using the syntax and semantics of Programming language. If the problem is too big to be solved by one algorithm, an individual has to subdivide the problem into smaller and more manageable problems. This makes Programming complex and difficult, and as a result it has a high failure rate of 30%-50% at universities worldwide (Norwawi, Hibadullah, & Osman, 2005; Shukur, Alias, Hanawi, & Arshad, 2003).

A lot of research has been conducted on Programming in education, especially on predictors of success in computer Programming. Generally, is it possible to foresee which students are likely to perform well in Programming. The goal is that if it could be predicted which students are most likely to do well in Programming, the current high failure rates could be reduced. This could be achieved by setting up appropriate selection criteria at entry level, or by giving students modules that could help them to increase their chances for being successful in Programming, even before they begin the Programming course. If a student fails these predictive modules, they would be advised not to take Programming because they would most likely fail Programming.

Some of the predictors that researchers have identified include SAT Mathematics score (Leeper & Silver, 1982); SAT verbal score (Leeper & Silver, 1982); high school rank (Leeper & Silver, 1982; Ramalingam, LaBelle, & Wiedenbeck, 2004.); age and a strong mathematics background (Leeper & Silver, 1982); cognitive development (Cafolla, 1988); number of hours playing computer games prior to the course (Evans & Simkin, 1989); a constructivist learning environment (Gibbs, 2000); foreign language units (Hagan & Markham, 2000); prior knowledge of Programming languages before taking a Programming course (Hagan & Markham, 2000); learning styles (Gibbs, 2000); comfort level (Wilson, 2002); mental model (Ramalingam et al., 2004.); Programming self-esteem (Ramalingam et al., 2004.); self-efficacy and personality type (Ramalingam et al., 2004.) and science modules taken prior to taking the course (Golding & McNamarah, 2005).

A strong Mathematics background, coupled with strong performance in Science modules, are the key indicators of success in Programming. This was stressed by Bohlmann and Pretorius (2008, p. 43) in the statement, "…the conceptual complexity and problem-solving nature of Mathematics makes extensive demands on the reasoning, interpretive and strategic skills of learners." These skills are very useful in Computer Programming.

Many researchers have focused on the positive relationship between high school (university entry level) Mathematics and success in Programming (Byrne & Lyons, 2001; Chowdhury, Nelson, Fuelling, & McCormick, 1987; Owolabi, Olanipekun, & Iwerima, 2014b; Van Der Westhuizen & Barlow-Jones, 2015b; Wirth, 2002). These indicators are now being incorporated by universities like the University of Zululand in their admission policy for the Computer Science program and in their curriculum by offering learners Mathematics and Science modules before Introduction to Programming Systems.

At the University of Zululand (UniZulu), the failure rate of Introduction to Programming Systems ranges from 30% to 45% (University of Zululand, 2014), which is in line with findings of other researchers (Bennedsen & Caspersen, 2007; Watson & Li, 2014). This is a cause of concern to the university because the students who fail will have to repeat the course. Repeating the course makes it difficult for University of Zululand to enroll large numbers of new students because of limited resources. This also means that the number of students not graduating at their expected time will increase, since Introduction to Programming Systems is a prerequisite for other Programming courses in the second and third years of study. When students do not finish their degrees in time, it is a loss to the government and students' parents because they will have to pay fees for students to repeat the same level. When students fail Programming, they choose other professions which are not computer related, which affects the number of computer related professionals. The demand for computer system solutions is increasing, while professionals joining the field are decreasing. This is causing South Africa to have to import people with computer system solutions skills from other countries. This affects the country's global competitive edge.

One way of reducing this high failure rate at universities like University of Zululand is by testing indicators that can be used to predict the student's performance in Programming. This will assist them in advising students who are most likely to succeed in Introduction to Programming Systems, and who may not have taken this course because of its difficult nature. It will also be possible to advise those students who are most likely not to succeed in Introduction to Programming Systems, not to take this course.

The reason for testing indicators is to see if they will work in a South African context and setting, which differs from settings and contexts under which the original observations were made. Settings may differ in many ways. The module arrangement, that is, the number of laboratory and lecture room lectures may be different. Availability of resources, such as lecturer-student ratio, availability of tutors and student-computer ratio, may be different. Another contextual difference might be in the form of teaching methods and assessment methods of the module. All these differences make it difficult to generalise, and use these indicators, and observe their positive impacts at University of Zululand.

1.2 PRELIMINARY LITERATURE REVIEW

Programming is a cognitively demanding task, and mastering it can be a challenging undertaking. If students find it difficult to learn a concept, it means that there is something about that concept that makes it difficult to learn (Jenkins, 2002). Programming is considered to be difficult in universities world-wide (Watson and Li (2014), and hence it has a high failure rate. In the survey of failure rates for Programming, Bennedsen and Caspersen (2007) found that, in universities in the United States, it averages at 33% compared with 41% for universities outside the US. With these high failure rates, a lot of research has been carried out. Some of the results that were brought forward are predictors in Computer Programming.

Many researchers studied the characteristics possessed by students who learn Programming without struggling, and those who struggle to learn Programming (Bergin & Reilly, 2005; Cafolla, 1988; Vihavainen, Kurhila, Paksula, & Luukkainen, 2011; Wiedenbeck, 2005). Those characteristics are now used as predictors of success in Introduction to Programming Systems. Many researchers like Bergin and Reilly (2005); Leeper and Silver (1982); Nowaczyk (1983) found that a good mathematical background and high self-esteem have a positive correlation with success in Programming. A good Programming background also has a positive relationship with success in Programming (Wiedenbeck, 2005). Percent laboratory usage, comfort level, and SAT Mathematics score accounted for 53% of the variance (Ventura (2003), but some of the predictors like number of hours playing computer games prior to the course were found to have a negative correlation with success in Programming (Wilson, 2002).

The one finding that seems to be most consistent across various investigations, although not strong, is the correlation between Mathematics score in high school and performance in Introduction to Programming Systems by Caspersen (2007), but even this result is questionable in a South African context. The research that was conducted by Van Der Westhuizen and Barlow-Jones (2015) indicated that the school Mathematics marks correlate only marginally, and that correlations were not significant with performance in the two programming courses.

Most researchers such as Bergin and Reilly (2005); Bohlmann and Pretorius (2008); Byrne and Lyons (2001); Chowdhury et al. (1987); Gomes and Mendes (2008); Leeper and Silver (1982); Owolabi et al. (2014) believe that high school Mathematics is a good predictor of success in Introduction to Programming Systems, but the findings of Van Der Westhuizen and Barlow-Jones (2015) have proved otherwise in a South African context. The current researcher seeks to find out the relationship between success in Introduction to Programming Systems and university level one Mathematics (Discrete Mathematics and Calculus), and Science modules (Classic Mechanics and General Chemistry) completed before taking Programming at University of Zululand. This study has not been conducted before in South Africa. Previously, researchers like ; Owolabi et al. (2014); Van Der Westhuizen and Barlow-Jones (2015) examined the relationship between high school Mathematics and Introduction to Programming Systems, but no one explored the relationship between university Mathematics and Introduction to Programming Systems.

1.3 PROBLEM STATEMENT

The problem statement for this research was to determine the relationship between selected indicators and success in Introduction to Programming Systems. The selected predictors were Discrete Mathematics, Calculus, Classic Mechanics and General Chemistry. These are university modules that are taken in the first year and the first semester. The students' scores were used to test for the predictive validity of Introduction to Programming Systems in a first year, second semester module.

If this research is not carried out, students will keep on failing Programming. The results of this research will help to advise students at risk to choose other modules, or if changing courses is not an option, then lecturers will be able to identify those students and may help them to succeed. This can also help to encourage capable students who may not have taken Programming, because of its difficult nature, to take it, and hence increase the number of students who will succeed. This will change Programming's image that has been tarnished by high failure rates.

1.3.1 Research questions

1.3.1.1 Do selected predictors have an effect on performance in Introduction to Programming Systems?

1.3.1.2 Which of the selected predictors have a high predictive validity?

1.3.1.3 How can universities use these predictors as criteria for success in Introduction to Programming Systems?

1.3.2 Aim of the study

The general aim of this study is to find out whether performance in selected first semester modules examinations can serve as predictors of achievement in subsequent studies in programming.

1.3.3 Research objectives

1.3.3.1 To establish the predictive validity of selected predictors on performance in Introduction to Programming Systems.

1.3.3.2 To establish an ordinal strength of selected predictors on success in Introduction to Programming Systems.

1.3.3.3 To find out if universities can rely on selected predictors as predictors of success in Introduction to Programming Systems.

1.4 INTENDED CONTRIBUTION TO THE BODY OF KNOWLEDGE

The findings of this research can be used to advice students who are at risk not to take Programming, and encourage capable students who may not have taken Programming to take it. Findings are also going to be used to create a positive image of Programming otherwise tarnished by the negative image of the high failure rate. The study also has implications for the University of Zululand admissions' policy. The results should help University of Zululand to identify an optimal set of Introduction to Programming Systems prerequisite modules, which have the potential of predicting students' performance.

1.5 DEFINITION OF TERMS

Programming*:* First year computing module. The module is being taught in Java programming language at the University of Zululand.

Predictor/Indicator: an independent variable used to forecast a criterion variable. In this study, the predictors/indicators are Discrete Mathematics, Calculus, Classic Mechanics and General Chemistry. All these modules are completed before taking Introduction to Programming Systems.

1.6 THE RESEARCH METHODOLOGY

1.6.1 Paradigm

This study is embedded in the transfer of learning paradigm. Transfer learning is the improvement of learning in a new task through the transfer of knowledge from a related task that has already been learned (Torrey & Shavlik, 2010). The theory was originally introduced by Thorndike and Woodworth (1901). They explored how individuals would transfer learning in one context to another similar context, or how "improvement in one mental function" could influence a related one (Thorndike & Woodworth, 1901). Their theory implied that transfer of learning depends on how similar the learning task and transfer tasks are, or whether "identical elements are concerned in the influencing and influenced function", now known as the identical element theory (Thorndike & Woodworth, 1901). Transfer of knowledge goes far beyond simply repeating memorised material, but rather concerns being able to take old knowledge and experiences, and apply this old knowledge to a new concept, and being able to use both the new and old knowledge to solve a problem that has never been encountered before (Torrey & Shavlik, 2010). This research explored how Mathematics and Science modules completed prior to taking Introduction to Programming Systems affected performance in Introduction to Programming Systems. The researcher sought to establish if students can transfer old knowledge (Mathematics and Science), to solve new problems (Programming).

1.6.2 Tradition

A quantitative approach was used in this research. Quantitative research is defined as research that explains phenomena by collecting numerical data that are analysed using mathematically based methods, in particular, statistics (Cohen and Manion, 1980).

1.6.3 Research design

A research design is the conceptual structure within which research is conducted. It constitutes the blueprint for the collection, measurement and analysis of data (Kothari, 2004). According to Kumar (2011), research design is a procedural plan or blueprint for how a research study is to be completed, operationalising variables so they can be measured, selecting a sample of interest to study, collecting data to be used as a basis for testing hypotheses and analysing the results. A research design has two main functions. The first one relates to the identification and development of procedures and logical arrangement required to undertake a study, and the second emphasises the importance of quality in these procedures to ensure their validity, objectivity and accuracy (Kothari, 2004).

The research used two research designs which are case study and correlational research designs. These research designs were used by many researchers who studied predictors, including (Barlow-Jones, Van der Westhuizen, & Coetzee, 2015;

Bergin & Reilly, 2005; Golding & McNamarah, 2005; Jegede, 2009) Van Der Westhuizen and Barlow-Jones (2015). In correlational research designs, investigators use the correlation statistical test to describe and measure the degree of association (or relationship) between two or more variables or sets of scores (Creswell, 2012). In this design, the researcher did not try to control or influence the variables as in an experiment. Instead, the researcher related, using the correlation statistic, two or more scores for each person (Creswell, 2012). The main purpose of a correlational study is to determine relationships between variables, and if a relationship exists, to determine a regression equation that could be used to make predictions for a population (Kothari, 2004). There are two primary correlation designs, namely explanation and prediction.

An explanatory research design is a correlational design in which the researcher is interested in the extent to which two variables (or more) co-vary. That is, where changes in one variable are reflected in changes in the other (Kothari, 2004). In this research, the researcher, like other studies on predictors (Ayaya (1996); Barlow-Jones et al. (2015), was interested in the extent to which predictors predict the criterion. The researcher wanted to explain the association between predictors and criterion.

In a prediction design, researchers seek to anticipate outcomes by using certain variables as predictors (Creswell, 2012). The purpose of a prediction research design is to identify variables that will predict an outcome or criterion (Creswell, 2012). In the current study Mathematics and Science modules were used to predict success in Introduction to Programming Systems. The collective and the individual contributions of all the predictor variables were estimated using multiple regression.

1.6.4 Sampling

The population of the study was students who took all the five university modules, namely, Introduction to Programming Systems, Discrete Mathematics, Calculus, Classic Mechanics and General Chemistry in South Africa. Sampling is the process of selecting a subset of population elements to represent the whole population (Polit & Beck, 2010). In this study, the sampling technique that was used is convenient sampling. In this study, the sample size was the scores of all 399 students who took Introduction to Programming Systems and selected predictors at University of Zululand between 2011 and 2016.

1.6.5 Instruments

The research comprised a desktop research which made use of secondary data collected from the examination results of students. The researcher did not interfere with the examinations. Data was collected from the student records from the University of Zululand database. Permission was solicited from the university to access and use their records. The instrument used was a data extraction form. The researcher designed a data extraction form and used it to collect data.

1.6.6 Scoring and analysis of data

The examinations had already been marked, and the results for individual students were used as scores.

The data on the predictive validity of selected predictors to performance in Programming was analysed using SPSS's Pearson product–moment correlation coefficient. Each of the predictor module's scores was correlated with the examination scores for Introduction to Programming Systems.

For the aim of finding the ordinal strength of selected predictors, regression was used to analyse the data. The students' scores for each predictor were considered against their respective Introduction to Programming Systems scores and an ordinal strength was established.

The results of research objectives one and two were used to analyse and answer research objective three.

1.7 ETHICAL AND SAFETY ISSUES

The researcher has read and understood the university's ethical policy. The researcher has complied with all ethical issues. This research was designed, reviewed and undertaken to ensure adherence to the highest standards of quality, integrity, ethical propriety and governance, and legal compliance. The researcher ensured that other people's work is well acknowledged to avoid plagiarism. The results of the findings of this research was discussed without giving false information.

Anonymity

All the names, student numbers, genders, or anything that could be used to identify the identities of students, was deleted. Names were replaced by numbers. Only marks and numbers were analysed and stored as data. The names and identifications of people who provided this research with data remain anonymous.

Confidentiality

In this research, the data gathered was kept secure, and access to it was limited to lawful users.

Consent

Permission was asked to access the marks of Computer Science students. The researcher informed and gave sufficient information about the research project, its aims and outcomes to the assistant registrar, so that permission could be given to access the results of students. No consent was gained from students since their marks are considered as data within the public realm.

1.8 SUMMARY

Chapter one introduced the high failure rate of Introduction to Programming Systems world-wide. It included a preliminary literature review which discussed the predictors of Introduction to Programming Systems and the gap that this research seeks to close. Problem statement, three research questions, aim and three objectives were also discussed in this chapter. The research methodology which consisted of the transfer of learning paradigm, case study and correlational research designs, convenient sampling, the data extraction form as an instrument and ethical considerations were also discussed in this chapter.

1.9 ORGANISATION OF THE STUDY

The research is going to be divided into five chapters as follows:

Chapter 1: Includes the study background, aim of the study, problem statement and research question, and significance of the study.

Chapter 2: This chapter deals with the related literature review. The two main parts of this review are theoretical framework and empirical evidence.

Chapter 3: Contains methodology and research design, the study sample, variables and data sources.

Chapter 4: This chapter discusses and interprets the results of the data analysis.

Chapter 5: The last chapter contains conclusions, limitations and recommendations.

CHAPTER 2

LITERATURE REVIEW

2.1 INTRODUCTION.

Predicting the success of prospective students in Programming at university has been a significant problem for the undergraduate admitting committees for years. The problem being, the predictors that can be used to separate Programming capable students from students who are at risk of failing Programming are unclear. This chapter provides a review of existing literature that is related to the area of the study. The importance of this chapter is two-fold; firstly, to provide a theoretical background to the study and, secondly, to enable the researcher to contextualise the findings in relation to the existing body of knowledge (Kumar, 2011). In this chapter a selection of research articles and books are reviewed to get an in-depth understanding of what selection and prediction entails, what are the failure rates in Programming and the predictors of success in Programming. Moreover, among the predictors in success, the predictors which are widely accepted as good predictors, their ordinal strength and the applicability of world-wide findings to the South African context will also be reviewed.

2.2 THEORETICAL FRAMEWORK.

The study is informed by Anderson and Krathwohl's Revision of Bloom's Taxonomy which was published in 2001. Anderson was a former student of Bloom. The theory was first published in Bloom's Taxonomy. Bloom used six major categories, namely, knowledge, comprehension, application, analysis, synthesis and evaluation. These nouns were then changed by Anderson to verbs which are remembering, understanding, applying, analysing, evaluating and creating (Anderson et al., 2001).

According to Anderson et al. (2001), the cognitive process dimension from low to high order thinking skills is as follows:

- Remembering- is to retrieve applicable knowledge from long-term memory.
- Understanding- is to make meaning from instructional messages.
- Applying – is to use method in a given situation.
- Analysing- breaking material into its constituent parts and determining how these parts relate to one another and to an overall structure or purpose.
- Evaluating- make judgement based on criteria or standards.
- Creating- put elements together to form a coherent whole.

When students learn Introduction to Programming Systems, all the cognitive levels of thinking are used. The low order thinking skills are remembering, understanding and applying. The high order thinking skills are analysing, evaluating and creating. When students are given programming problems, the low order thinking skills are needed for remembering the syntax of the programming language, understanding the problem and applying some programming concepts to solve the problem. The high order thinking skills are also used for analysing the problem, evaluating possible solutions and creating a computer programme.

2.3 STUDIES ON PREDICTIVE VALIDITY AND PSYCHOLOGICAL TESTS (PREDICTION AND SELECTION)

Previous studies have shown that testing as a method of selection and appointment started around 2200BC (Anastasi & Urbana, 1997; Gregory, 1996). Testing of mental abilities began in Europe and USA in the late 1800's (Anastasi & Urbana, 1997; Foxcroft & Roodt, 2001; Gregory, 1996). This method of selection and appointment became popular for the past two centuries. In the middle of the 20th century, testing for educational purposes began (Gregory, 1996). Tools like the Graduate Record Exam (GRE) and Scholastic Aptitude Test (SAT) became well accepted for selection and appointment (Gregory, 1996). The traditional admission and the most used criteria into South African institutions of higher learning are Matriculation results.

Secondary school level performance is widely accepted as a predictor for academic performance at institutions of higher learning (Ayaya, 1996; Dawes, Yeld, & Smith, 1999; Faulkner, 2002; Greyling, 2000; Huysamen & Roozendaal, 1999.; Lindblom-Ylanne, Lonka, & Leskinen, 1999; Louw, Meyer, & van Schalkwyk, 1998). As a result, Matriculation results are being used to separate sheep (top academic students at secondary level) from goats (below average academic students at secondary level). However, many recent studies are now showing that the so-called sheep do not always translate into academic success at institutions of higher learning (Ayaya, 1996; Cavanagh, 2003; Huysamen & Roozendaal, 1999.; Lindblom-Ylanne et al., 1999; Miller & Bradbury, 1999). The predictive validity of secondary school level performance as a predictor of success at institutions of higher learning is now questioned. In South

Africa there is a high demand for access to institutions of higher learning, and hence selection and prediction has become a critical issue (Greyling, 2000).

Prediction refers to the accuracy of the selection process in terms of identifying correctly capable Programming students from students at risk of failing. On the other hand, selection is described as a process where students with potential to succeed in specified degree programs at tertiary level are identified. This process should be based on sound psychological theory and practice (Lindblom-Ylanne et al., 1999; Miller & Bradbury, 1999). The process should be fair, effective and efficient (Zaaiman, Van der Flier, & Thijs, 1998). This will result in a high percentage of academically successful students. This will be achieved by restricting students who are in danger of failing and admitting capable students only. Since new research is questioning the selection method, there is a need to find an alternative. In South Africa the validity of Grade 12 examination results are being questioned by the public, educational experts and by universities (Van Der Westhuizen & Barlow-Jones, 2015). Firstly, the matric results do not reflect the actual performance of learners because in some instances their marks may be adjusted upwards or downwards by as much as 10% in any subject to align it with historical performance trends (Parliament, 2014). Secondly, the local press has also reported cheating during examinations (SAnews, 2015).

Umalusi's moderation processes identified 'group copying' in Mathematics, Economics and Business Studies. It was also found that there had been 'evidence of possible assistance by an invigilator or exams official' in a Mathematics paper, which was written by 174 candidates (Smith, 2015). Finally, it is also believed that matriculation results are politically manipulated to show an overall improved performance of the school education system, and especially so, since democratisation in 1994 (Van Der Westhuizen & Barlow-Jones, 2015b).

2.4 WHAT IS PREDICTIVE VALIDITY?

Predictive validity is the degree to which test scores predict performance on some future criterion (Shultz & Whitney, 2005). It is concerned with the relationship or correlation between the predictor and the criterion and can be reported in terms of a correlation coefficient (Cronbach, 1971). An expectancy chart or table can also be used to express predictive validity (Anastasi, 1976). High correlation between the original measure and criterion variable reinforces the conclusion that the tool is a valid predictor of the specified criteria.

2.5 HIGH FAILURE RATE IN INTRODUCTION TO PROGRAMMING.

Programming is a cognitively demanding task and mastering it can be a challenging undertaking. If students find it difficult to learn a concept, it means that there is something about that concept which makes it difficult to learn (Jenkins, 2002). Programming is considered to be difficult in universities world-wide according to Watson and Li (2014) and hence its high failure rate. An internal report on community colleges, (Caspersen, 2007, p. 13) stated, "... one of the universities in the United States' average failure rate over a ten-year period was 90%. Another university with 4000 students, where Computer Science is the second largest major, reported a failure rate of 72%." In a survey of failure rate for Programming, Bennedsen and Caspersen

(2007) found that the failure rate for universities in the United States averages 33% while universities outside the United States averages 41%. The highest failure rate reported was 95% (Bennedsen & Caspersen, 2007). After all these years of research, one expects to see some improvements, but recent findings still show high failure rates in Programming. In 2011, Vihavainen et al. (2011) found that the long-time pass rate average (excluding spring 2010) in the fall semester was 58,7%. This means that the average failure rate was 41,3%. In spring semesters they found that the average pass rate was 43,7%, meaning 56,7% was the average failure rate (Vihavainen et al., 2011). In another study that was carried out in 15 different countries, across 51 universities, Watson and Li (2014) found that the pass rate of Introduction to Programming was 67,7%. This means the average failure rate was 32,3%. Structured Programming which is given to engineering students as an Introduction to Programming has an average failure rate of 70% (Bosquea, Martinez, & Torres, 2015). These high failure rates resulted in a lot of research. some of which generated predictors in Computer Programming.

2.6 PREDICTORS OF SUCCESS IN INTRODUCTION TO PROGRAMMING.

Many researchers have studied the characteristics possessed by both students who learn Programming without struggling and those who struggle to learn Programming (Bergin & Reilly, 2005; Byrne & Lyons, 2001; Cafolla, 1988; Gomes & Mendes, 2008; Vihavainen et al., 2011; Wiedenbeck, 2005). Those characteristics are now used as predictors of success in Introduction to Programming. Table A1 was adapted from Caspersen (2007) and gives an overview of research in Programming predictors.

The correlation between Mathematics and success in Introduction to Programming Systems seems to be consistent across many investigations. However, according to the findings of Van Der Westhuizen and Barlow-Jones (2015b), South African high school Mathematics score was not a good predictor of success in Introduction to Programming Systems. Due to this contradiction between the findings of Van Der Westhuizen and Barlow-Jones (2015b) and many researchers including Bergin and Reilly (2005); Byrne and Lyons (2001); Evans and Simkin (1989); Wilson (2002), the current researcher sought to find out if South African high school Mathematics score is not a good predictor, and determine whether university Mathematics and Science courses done before taking Introduction to Programming Systems would be good predictors.

2.7 THE PREDICTIVE VALIDITY OF MATHEMATICS AND SCIENCE ON PERFORMANCE IN INTRODUCTION TO PROGRAMMING SYSTEMS

Research carried out in Taiwan by Tai-sheng and Yi-ching (2002) used a sample of 940 students. Among the participating subjects, 796 were male (85%) and 144 were female (15%). The researchers used the average scores of 15 Computer Sciences core courses (CS-MAJOR) as college academic performance measurements. These core courses included (1) Calculus, (2) Linear Algebra, (3) Discrete Mathematics, (4) Probability, (5) Numerical Methods, (6) Introduction to Computer Science, (7) Programming, (8) Programming Languages, (9) Data Structures, (10) Assemblers, (11) Introduction to Digital Systems, (12) Electric Circuits, (13) System Programs, (14) Operating Systems, and (15) Computer Structures. Mathematics ability was measured by scores on the College Entrance Examination (CEE) Mathematics component (CEE-MATH), average scores of high school Mathematics courses (HS-MATH) and average scores of college Mathematics courses (C-MATH). The Pearson's product moment correlation coefficients (r) were computed to observe relationships between examined variables. To find the relationship between mathematical ability and college performance, HS-MATH, C-MATH, and CEE-MATH were correlated with scores of Introductory Computer Science courses (CS-INTRO) and CS-MAJOR accordingly.

HS-MATH was found to correlate significantly with CS-INTRO. However, the correlation coefficients achieved were too weak to provide evidence for the existence of a significant relationship between HS-MATH and CS-INTRO. HS-MATH was consistently found to be associated significantly with CS-MAJOR. The study indicated that CEE-MATH and C-MATH were closely correlated to CS-MAJOR.

This study sought to find out the correlation between mathematical ability and Introduction to Programming Systems only, and not with all Computer Science major courses. However, the scores of first semester Computer Science major courses were examined to see if they predicted success in Introduction to Programming Systems. In this study, Pearson product–moment correlation coefficient was used to assess the correlation between scores of individual modules (Discrete Mathematics, Calculus, Classic Mechanics and General Chemistry) to the scores of Introduction to Programming Systems. The results were used to investigate if these modules were predictors of success in Introduction to Programming Systems.

Students' Mathematics scores from high school were also used by Bennedsen and Caspersen (2005) to predict performance in Programming. They used 235 college students in an object-first CS1 course and found that high school Mathematics score was the best predictor of success in Programming and it explained over 15% of the variance in their predictive model. In another similar research study, Bergin and Reilly (2005) investigated students' performance in an introductory Programming course and their performance in the Irish Leaving Certificate examinations in Mathematics and Science subjects. The researchers found that Mathematics and Science both significantly correlated positively with students' performance in Programming. In this study university Mathematics and Science courses were investigated to test their validity as predictors of success in Programming.

A sample size of 837 students was used by White and Sivitanides (2005) to assess the relationship between mathematical ability and success in Introduction to Programming Systems using Visual Basic Programming Language. Freshman College Algebra course, Freshman Mathematics for Business and Economics I and Freshman Mathematics for Business and Economics II courses were used to assess students' mathematical ability. The sequence in taking the courses was ignored.

The researchers concluded that Freshman Mathematics was a good predictor of Programming success, without using any statistical tool to analyse data. The researchers did not show how they came to that conclusion because the order of taking modules was ignored. It might be that Introduction to Programming is a predictor of Freshman Mathematics.

In this research, the order of taking courses is of paramount importance. All the courses that are going to be considered should be taken before taking Introduction to Programming Systems for them to be predictors. In White and Sivitanides (2005)'s research, only the Freshman College Algebra course was a pure Mathematics course while the other two modules were commerce courses. In this research, pure mathematics courses (Discrete Mathematics and Calculus) were used to assess students' mathematical ability. The average scores of pure mathematics courses completed before taking Introduction to Programming Systems was used to establish correlation with scores in Introduction to Programming Systems. Lastly, unlike in White and Sivitanides (2005), Algebra was not used as a predictor of success in Programming.

In a research carried out by Erdogan, Aydin, and Kabaca (2006), C Programming language was used to introduce students to Programming basic concepts. The researchers used 48 students from Profilo Anatolia Technical High School, Istanbul as participants. The sample was composed of 25% females and 75% males. The students' mathematical ability was measured their first semester marks which were obtained from the school administration. A bivariate Pearson's correlation coefficients test was run to determine the degree of relationship between the students' Programming achievement and Mathematics achievement.

The results showed that there was a significant correlation between the students' Programming achievement and Mathematics achievement at the level of 0.01 ($r=0.447$; $p<0.01$). This supports the findings of Byrne and Lyons (2001); Konvalina, Stephens, and Wileman (1983); Owolabi et al. (2014b), that Mathematics is a good predictor of Programming.

In this current research, the researcher did not use high school Mathematics scores as a measure of mathematical ability. The current researcher used first year, first semester examination scores of two mathematics courses to assess students' mathematical ability. The researcher combined scores of all first semester courses (including non-Mathematics courses) to investigate whether they predict success in Programming.

In 2014, another research study was carried out in Nigeria by Owolabi et al. (2014b) to determine the predictive validity of Mathematics on success in Programming. To determine the mathematical ability of students, the researchers used a first semester Mathematics course called Algebra. The Basic Programming course completed in the second semester was used to determine programming ability. The programming language in which Basic Programming was taught was not mentioned. Researchers then used multiple regression Anova to analyse their data.

They found that mathematical ability contributed most and directly to the performance of students' Basic Programming marks ($\beta = 0.430$; $t = 5.973$; $p < 0.05$). They concluded that mathematical ability had a significant effect on the performance of students' Basic Programming marks. Their finding is in line with the findings of many researchers (Barlow-Jones et al., 2015; Gomes & Mendes, 2008; Van Der Westhuizen & Barlow-Jones, 2015b).

Algebra was not used to indicate mathematical ability in this research. The current researcher used two Mathematics first semester courses as a measure of mathematical ability. The researcher used the average scores of first semester examination marks to see if they can predict success in Programming.

At the University of Texas–Pan American (UTPA), Reilly and Tomai (2014) used a sample size of 558 students to conduct their research. They grouped students into two groups, namely, students who were ready to take Calculus and more advanced courses in the same semester as they first took CS1 (Programming course), and another group of students who were not ready to take Calculus. An Algebra course was a prerequisite for the Calculus course. However, there were students who were regarded as Calculus ready when entering the university without taking an Algebra course. Therefore, there were two ways for a student to be considered Calculus ready: (i) Straight from high school with high scores in Mathematics, and (ii) Students with low high school marks who must pass the university Algebra course.

The results showed that Calculus-ready students had a higher pass rate as compared to non-Calculus ready students. The majority of students taking CS1 were non-Calculus ready students. The number of students from the non-Calculus ready group who passed CS1 matched that of the Calculus ready students. Because many non-Calculus ready students passed, Reilly and Tomai (2014) concluded that the Algebra course or other Calculus prerequisites were not predictors of success in Programming.

In this research, Calculus was used as one of the predictors of success in Programming. In other words, the student has to take Calculus before taking Programming courses, unlike in Reilly and Tomai (2014)'s case where Calculus was done concurrently with Programming. In Reilly and Tomai (2014)'s study, Calculus was taken concurrently with Programming, which means that the prerequisite for Calculus was also the prerequisite for Programming. In the research conducted by Reilly and Tomai (2014) Algebra and high school Mathematics courses were used as prerequisites for Calculus. At the University of Zululand, Calculus is taken as a first year and first semester course and hence it does not have any prerequisite courses. In this research, Algebra and other courses considered by Reilly and Tomai (2014) were used as predictors of success in Programming. In Reilly and Tomai (2014) study, the reason for concluding that the Calculus prerequisite was not a predictor of success in Programming, was that the number of non-Calculus ready students who passed matched the number of Calculus ready students who passed.

The problem with this is that the majority of participants were non-Calculus ready students. Logically, if researchers have a large number of participants in one group (non-Calculus ready) than the other (Calculus ready), researchers can also expect many students (non-Calculus ready) to pass and match the number of the smaller group (Calculus ready). It is unfair to compare the number of students who passed if the two groups are not comprised of the same number of participants. In the current research, the researcher avoided this by using SPSS's Pearson product–moment correlation coefficient to analyse the data.

In another study carried out at Tshwane University of Technology (TUT) a group of 82 randomly selected students was used to determine whether their National Senior Certificate (NSC) Mathematics result was a predictor of success in Programming. To do that, Barlow-Jones et al. (2015) compared the students' Grade 12 Mathematics results with their performance in the two Programming courses. The Programming language that was used is VB.NET.

A Pearson product-moment correlation coefficient was computed to assess the relationship between the students' performance in Mathematics in Grade 12 and performance in Development Software 1 A. They found that there was a weak positive correlation between the two variables, $r = 0.345$, $n = 82$, $p = 0.02$. Therefore, Barlow-Jones et al. (2015) concluded that there is a weak correlation between a student's NSC result for Mathematics and the final mark obtained for Development Software 1 A. They also computed the Pearson product-moment correlation coefficient to assess the relationship between the students' performance in Mathematics in Grade 12 and performance in Development Software 1 B. They found that there was a weak positive correlation between the two variables, $r = 0.261$, $n = 82$, $p = 0.018$. They also concluded that there was a weak correlation between a student's NSC result for Mathematics and final mark obtained for Development Software 1 B. Their findings are consistent with the findings of Bergin and Reilly (2005); Byrne and Lyons (2001); Gomes and Mendes (2008); Shrock and Wilson (2001) but inconsistent with the findings of Reilly and Tomai (2014).

In the current research, the researcher used NSC Mathematics marks as mathematical ability. Mathematical ability was measured by students' Calculus and Discrete Mathematics scores which are university courses. In Barlow-Jones et al. (2015), they used two Programming courses which were taught using VB.NET. In this research only one Programming course taught in Java was used as the dependent variable. Non-Mathematics courses were also investigated to see if they could predict success in Programming.

In an eastern city in China, Qian and Lehman (2016) grouped a total of 69 7^{th} grade school students into two groups. Group A had 33 students, 17 boys and 16 girls. The average age of Group A was 12.3 years. Group B had 36 students, 23 boys and 13 girls with an average age of 13.2 years. Pascal Programming language was used and it was being taught in English. Stepwise regression analyses were conducted to investigate whether the various factors studied were able to predict students' Programming learning performance. Independent variables were gender, Mathematics score, Chinese score and English score. They found out that English and Mathematics ability has an influence on students' Programming performance. In the current research the participants were not grade 7 school students but university students. Mathematics and Science courses are going to be used to predict success in Programming, not English and gender.

2.8 ORDINAL STRENGTH OF PREDICTORS ON SUCCESS IN INTRODUCTION TO PROGRAMMING.

At Midwestern University 105 students voluntarily participated in a research study carried out by (Shrock & Wilson, 2001). C++ was the programming language used to assess Programming performance. The model included twelve possible predictive factors including mathematical background, attribution for success/failure (luck, effort, difficulty of task and ability), domain specific self-efficacy, encouragement, comfort level in the course, work style preference, previous programming experience, previous non-programming computer experience and gender.

They found that comfort level in the Computer Science class was the best predictor of success in the course. Mathematics background was second in importance in predicting success in this Computer Science class. Programming experience was found not to be a predictor of success in Programming. In this research, only four first year first semester course predictors were used. The scores of each of the modules were used to investigate their relationship with success in Programming. After that, each course was evaluated as a predictor of success in Programming and its ordinal strength was established.

In another study, Bergin and Reilly (2005) used questionnaires to collect data on the following from 80 students: (i) High School Leaving Certificate (LC) Mathematics grade, (ii) LC Physics grade, (iii) LC Biology grade, (iv) LC Chemistry grade, (v) highest LC Science grade, (vi) comfort level on the module, (vii) perceived understanding of the module material, (viii) prior Programming experience, (ix) prior non-Programming

Computer experience, (x) work-style preference (preference to work alone or as part of a group), (xi) encouragement from others to study Computer Science, (xii) number of hours per week working at a part-time job.

They used Pearson correlations to analyse the data. They found that high school Mathematics final examination results had a significant relationship with performance, $r = 0.46, p < 0.01$, followed by high school Physics with, $r = 0.59, p < 0.05$, and then high school Biology with, $r = 0.75, p < 0.05$. No relationship was found between high school Chemistry final examination and performance.

In the current study, four university courses were used. To assess mathematical ability, Discrete Mathematics and Calculus were used. A Physics course (Classic Mechanics) and a Chemistry course (General Chemistry) were used to assess their relationship with success in Introduction to Programming Systems. Anova was used to establish the ordinal strength of these courses.

An association rule mining technique was used by Anwar, Ahmed, and Khan (2012.) to expose the hidden knowledge on Programming, Mathematics and English using data that was available to them. They discovered that students who excelled in the Mathematics course also performed well in the Programming course and the reverse is true. They also found that students who achieved grade A in Mathematics and English also achieved grade A in Programming. The rules also showed that even though the students' English grades were very good, poor grades in Mathematics resulted in poor grades in the Programming course. They concluded that Mathematics was a better predictor of success in Programming than English. Association rule mining technique was not used in this research. Statistical methods were used to establish the ordinal strength of Physics, Chemistry and Mathematics courses as predictors of success in Programming in this study.

In another study, in Nigeria,Owolabi et al. (2014b), used a correlational design with achievement in Basic Programming as the dependent variable, while Mathematics ability, Mathematics anxiety, Computer anxiety, Programming anxiety, age and gender serve as the independent variables.

They found that Mathematics Ability (MA) contributed most and directly to the performance of students in Basic Programming ($\beta = 0.430$; $t = 5.973$; $p < 0.05$). The contribution of other predictors was: Computer anxiety ($\beta = 0.093$; $t = 1.247$; $p > 0.05$), Mathematics anxiety ($\beta = 0.031$; $t = 0.422$; $p > 0.05$), Age ($\beta = -0.058$; $t = 0.794$; $p > 0.05$), Gender ($\beta = 0.108$; $t = 1.484$; $p > 0.05$), and Computer Programming anxiety ($\beta = -0.113$; $t = -1.554$; $p > 0.05$). These results showed that Mathematics ability was the best predictor of success in Programming followed by gender, Computer anxiety, Mathematics anxiety, age and lastly Computer Programming anxiety.

In this research, only non-Programming performance in the form of first year first semester courses were used to predict performance in Programming. The ordinal strength of the courses was also established.

In South Africa, Van Der Westhuizen and Barlow-Jones (2015) used the data of 393 students who were enrolled for the two first-year courses between 2012 and 2015 at Tshwane University of Technology. They grouped their sample into two groups. Group 1 was a group of students who did Mathematics Core at high school and group 2 was a group of students who did Mathematical Literacy at high school. Students who were admitted based on their high school Mathematics marks (n=274) outnumbered students who were admitted based on their Mathematical Literacy marks (n=119). They compared high school Mathematics and Mathematical Literacy performance with performance in two Programming courses, namely, Software Development 1A and Software Development 1B.

The results revealed that having ‘school Mathematics mark’ slightly correlates with ‘Software Development 1A performance’ and is significant at the 0.05 level ($r = 0.063$, $p > 0.05$). A similarly weak correlation exists between having a ‘school Mathematics mark’ and ‘Software Development 1B performance’, which was significant ($r = 0.038$, $p > 0.05$). They then concluded that there was a weak relationship between having a ‘school Mathematics mark’ and performance in either of the two Programming language courses. They used point-biserial correlational analysis and it yielded a strong positive correlation between having Mathematical Literacy and performance in Development Software 1B which was significant at the 0.01 level ($r = 0.610$, $p < 0.05$), whereas a slight negative correlation, yet significant at the 0.05 level was found to exist between having Mathematical Literacy and performance in Development Software 1B ($r = -0.130$, $p < 0.05$). The same thing was also found between Development Software 1A. They concluded that Mathematical Literacy was a better predictor of success in Programming than Mathematics. The conclusion that the researcher made was not supported by the results. In this research, the current researcher investigated the ordinal strength of university courses, not high school subjects and discussed the results as they stand.

An unexpected discovery was discovered by Qian and Lehman (2016) in China. They found that English was a better predictor of success than Mathematics. They discovered this when they were using grade 7 school children. They used Pascal as the Programming language. This language was being taught in English. They computed English and Mathematics scores for Chinese students with their Pascal scores. In this study English was not used as a predictor of success in Programming. Data was collected from university students, not grade 7 students.

2.9 CAN UNIVERSITIES RELY ON SELECTED PREDICTORS AS PREDICTORS OF SUCCESS IN INTRODUCTION TO PROGRAMMING?

Table 2.1 presents a number of studies on predictors of success in Programming. These studies were carried out in different contexts and environments. These differences make it difficult to generalise findings from one context to the other. While local predictors of success have been identified, there is no evidence that these predictors generally hold. In South Africa, Barlow-Jones et al. (2015) found that there was a weak positive correlation between Mathematics and success in Programming. Their findings were in line with many researchers (Bergin & Reilly, 2005; Leeper & Silver, 1982; Owolabi et al., 2014b). In another research study, Reilly and Tomai (2014) found that Algebra, which is a Mathematics course, was not a predictor of success in Programming. As a result of these different findings, it is difficult to make generalisations.

In their findings, Qian and Lehman (2016), found that English could be used as a predictor of success in Programming. These findings are contrary to the generally accepted view that Mathematics ability is the best predictor of success in Programming. All these differences make it difficult to generalise these results hence the need for local investigations.

2.10 SUMMARY

In this chapter, world-wide Introduction to Programming Systems failure rate was discussed in detail. The average failure rate in the United States of America is around 33 % and in all other countries it is about 40%. Selection process and prediction background and history was also given. Most predictors of success in Introduction to Programming Systems were discussed, Good predictors include Mathematics and Science ability, Programming experience and Programming anxiety. Poor / inadequate predictors include number of hours playing video games. A table with 26 predictors of success in Introduction to Programming Systems was also included in this chapter. Mathematics and Science ability were discussed in detail since they are widely accepted as good predictors and also since they are the predictors that this study is focussing on. For all literature reviewed, the researcher highlighted how the current research differed. The chapter ended by reviewing the ordinal strength of predictors.

CHAPTER THREE

RESEARCH METHODOLOGY AND DESIGN

3.1 INTRODUCTION

A research design is the conceptual structure within which research is conducted. It constitutes the blueprint for the collection, measurement and analysis of data (Kothari, 2004). According to Kumar (2011) a research design is a procedural plan or blueprint for how a research study is to be completed, operationalising variables so they can be measured, selecting a sample of interest to study, collecting data to be used as a basis for testing hypotheses and analysing the results. Methodology provides a working plan and describes the activities necessary for the completion of the study (Kothari, 2004). In this chapter, the focus is on how the research was conducted. The chapter looks at the research paradigm, research design, research method, research instrument, sampling design and ethical considerations.

3.2 RESEARCH PARADIGM

The current study is embedded in the transfer of learning paradigm. Transfer of learning is the ability to take in information learned in one situation and apply that to another different situation. (Thorndike & Woodworth, 1901) introduced this theory. Far transfer and near transfer of knowledge are the two types of transfer of learning. Near transfer happens when a new situation resembles the old system (Kaiser, Kaminski, & Foley, 2013). Near transfer of learning is usually repetitive, such as tasks that reproduce a process or a procedure (Kaiser et al., 2013). Far transfer of learning happens when the new situation is dissimilar to the learning situation (Kaiser et al., 2013). Far transfer of learning involves applying principles, implementing strategies and exercising judgment. The current study is anchored by far transfer of learning. In this research, the current researcher explored how Mathematics and Science modules completed prior to Introduction to Programming affected the module (Introduction to Programming). The researcher aimed to establish if students can transfer old knowledge and skills (Mathematics and Science) to solve a new problem (Programming).

3.3 RESEARCH DESIGN

A research design is a plan, structure and strategy of investigation so conceived as to obtain answers to research questions (Creswell, 2012). It is a procedural plan that is adopted by the researcher to answer questions validly, objectively, accurately and economically (Kumar, 2011). The research used two research designs which are case study and correlational research designs. A research design has two core functions, firstly it identifies and expands upon the procedures and logical arrangement required to embark on a study, and secondly it stresses the significance of quality in these procedures to ensure their validity, objectivity and accuracy (Kothari, 2004).

A case study is a dominantly qualitative study design, but also prevalent in quantitative research (Kumar, 2011). The case study research design was used in this research, in a quantitative manner where the findings of the study can be replicated, retested and generalised. Case study research design according to (Polit & Beck, 2010) is an in-depth study of one entity or institution. This research used case study, in the sense that data from one institution was used. Data was gathered from all

Computer Sciences students from 2011 to 2016 at University of Zululand who were assessed in the five selected courses. Many researchers also used case study research design (Barlow-Jones et al., 2015; Golding & McNamarah, 2005; Van Der Westhuizen & Barlow-Jones, 2015b).

Correlational research design is also known as prospective research design. Relational research and prediction are two types of correlational designs. Relational research design is used to describe the association between or among variables, providing empirical proof proposing two or more variables are or are not related (Creswell, 2012). In this study, to assess the relationship between the predictors and scores of Introduction to Programming, an average score of predictors was calculated and used as the independent variable and the students' scores for Introduction to Programming was used as the dependent variable.

Prediction design is used to identify variables that can effectively predict some outcome or criterion (Creswell, 2012). In this design predictive relationship can be estimated with a statistical procedure called multiple regression. Multiple regression was used to estimate the collective as well as the individual contribution of all predictor variables. Some researchers who were studying predictors of success in Introduction to Programming also used correlational design in their studies (Barlow-Jones et al., 2015; Byrne & Lyons, 2001; Chowdhury et al., 1987; Golding & McNamarah, 2005).

3.4 POPULATION AND SAMPLING

Population refers to the entire collection of cases in which a researcher is interested (Polit & Beck, 2010). In this research, the researcher used the methods that were used by many researchers who investigated the predictor and its validity (Ayaya, 1996; Owolabi et al., 2014b; Petersen & Howe, 1979; Qian & Lehman, 2016). The sample that most studies agree upon is the students in their first year of study (Ayaya, 1996; Owolabi et al., 2014b; Petersen & Howe, 1979). In this study, the population comprised all Computer Sciences students who took Introduction to Programming Systems and predictive modules in the worldwide from 2011 to 2016.

Sampling is the method of picking a subset of population components to represent the entire population (Polit & Beck, 2010). From the population the researcher used a non-probability sampling technique used by other researchers like Van der Westhuizen and Barlow-Jones (2015a) and Owolabi, Olanipekun, and Iwerima (2014a) called convenient sampling. By convenient sampling, the sample is picked from a location convenient to the researcher (Kumar, 2011). The researcher is studying at University of Zululand and lives near the university. Hence the sample that is convenient to the researcher was data of all Computer Science students who took all the five modules at University of Zululand from 2011 to 2016.

The predictive modules were taken by students in their first year, first semester and Introduction to Programming was taken in their first year, second semester. In the current research the sample size was 399. The examination marks of these students for the selected courses were obtained from University of Zululand academic data base.

3.5 ETHICAL CONSIDERATIONS

The researcher conducted this research in a manner consistent with international and national acceptable standards governing research. Anonymity was very important since the research is dealing with students' marks. The researcher erased any detail that could be used to identify the identities of students to maintain anonymity of students. Student numbers were replaced by numbers. Only marks and numbers were analysed and stored as data. The data was kept secure and access to it was limited to lawful users for confidentiality purposes. The names and identifications of people who provided this research with data remained anonymous.

Consent from the head of Computer Science department (HOD) was solicited by making an appointment and meeting. During the meeting, the researcher presented the HOD with an ethical clearance, data extraction form and access letter requesting permission to collect and use marks of students. The HOD expressed willingness and informed consent to give out data after the researcher explained adequately the type of data needed and its purpose, and how the research will be of use to the department. After getting consent from the HOD the researcher went to the Assistant Registrar and repeated the same process to get informed consent. The Assistant Registrar wrote an email to the examination center to instruct them to provide the researcher with the requested data. The examination center personnel extracted the necessary data from the university database and sent it to the researcher.

Students' marks are considered data in public realm and as a result the researcher did not ask for consent from students.

The researcher made sure that the results were discussed without giving false information. Bias was avoided in this research as it is unethical. The researcher avoided bias by discussing the findings as they were without either hiding the results, or highlighting results disproportionately to their true existence. The researcher provided an annual report and another report at the end of the project in respect of ethical compliance to University of Zululand Research Ethics Committee (UZREC).

3.6 INSTRUMENTS

The research was a desktop study which used secondary data that was collected from the examination centre. The researcher did not interfere with the examination. The researcher only collected marks from student academic records from the University of Zululand database. Data from the university database, was in the form of lists. Each year had five lists, one for each module. A data extraction sheet was designed and used as the instrument to collect. This sheet was designed as a table with six columns and many rows. Each student's performance was in its row. The researcher carefully extracted students' performances from the lists and put them in the data extraction sheet. All student numbers were replaced by numbers. An example of the data extraction sheet is attached as **Appendix 2.**

3.7 STUDY OF VARIABLES.

3.7.1 Predictor variables

The main independent variables that were utilised in this study were Mathematics courses (Discrete Mathematics and Calculus), a Physics course (Classic Mechanics) and a Chemistry course (General Chemistry). Mathematics courses were used to assess mathematical ability of students. Many researchers like Bergin and Reilly (2005), Leeper and Silver (1982) and Nowaczyk (1983) discovered that a good mathematical background correlates positively with success in Programming. This is because Mathematics and Science develops problem-solving skills which are very vital in Programming. Physics and chemistry courses were used to assess science ability of students. All these courses were done in the first semester of each year.

3.7.2 Criterion variable

Introduction to Programming Systems is the criterion variable in this study. This course is done in the second semester of every year after students took predictor variables in the first semester. This module is taught in Java.

3.8 METHODS OF DATA ANALYSIS

This refers to the methods that were used to analyse data in order to answer the research questions. Researchers (Tabachnick & Fiddell, 2013) came up with the formula: $N>50 + 8m$ (where m is the number of independent variables) to calculate the minimum sample size. Since, the current research had four independent variables the minimum sample size is 82. The study's sample size was 399 which is bigger than the one that is recommended by (Tabachnick & Fiddell, 2013). The students' data that was collected from the university includes student numbers and performance in the five modules. Data mining and extrapolation was employed. Outliers (very high or very low scores) were removed. This was done because data was analysed using SPSS's multiple regression, which is very sensitive to outliers.

The first research question was: Do selected predictors have an effect on performance in Introduction to Programming Systems? This question was answered by using simple correlation. A simple correlation is a mathematical measure of a relationship between two variables (Pallant, 2013). SPSS's Pearson product–moment correlation coefficient was used to investigate the relationship between the predictors and performance in Introduction to Programming Systems. The average score of the four predictor modules was calculated and used as the independent variable while student's Programming score was the dependent variable. A correlation may be expressed on a continuum ranging from +1.00 to -1.00. A coefficient of +1.00 indicates a perfect positive relationship. This means that as the average score of predictor increases so does the Introduction to Programming Systems score. A coefficient of -1.00 indicates a negative relationship. This means that as the average score of predictor increases, the score for Introduction to Programming Systems decreases. On the other hand, a coefficient of 0 indicates no relationship between the average score of predictors and the score for Introduction to Programming Systems. The size of the absolute value, not taking into consideration the sign gives an indication of the strength of the relationship. The bigger the absolute value, the stronger the relationship between the predictor and criterion. To interpret the values between 0 and 1, the researcher used guidelines according to (Cohen, 1988).

small	r = 0.10 to 0.29
medium	r = 0.30 to 0.49
large	r = 0.50 to 1.00

SPSS's standard or simultaneous multiple regression analysis was used to answer the second research question: Which of the selected predictors has high predictive validity? The scores of all independent (or predictor) variables were entered into the equation simultaneously. To come up with the ordinal strength of the predictors, each predictor was assessed in terms of its predictive power. Prior to conducting standard multiple linear regression analyses, an inter-correlation matrix using SPSS's Pearson product-moment correlations was studied to determine the extent to which each of the predictors was related to the criterion variable. The fundamental assumptions of regression were also examined to evaluate the appropriateness of the regression models. The Normal Probability Plot (P-P) was used to check for normality.

To answer the third research question, the findings of research question one and two were analysed.

3.9 SUMMARY

This chapter provided an overview of the research design and methodology that was employed for this study. Correlational and case study research designs were used in this study and were discussed in detail providing other researchers who also used the same research designs and reasons why these research designs were appropriate for the study. The chapter also explained and gave reasons why the convenient sampling method was used. Since the research adopted case study as one of its research designs, convenient sampling was the most appropriate sampling method.

A detailed step by step explanation on how data was collected was also discussed. This chapter also highlighted how the research was careful in following all the ethical requirements in order to obtain credible data. After collecting data, the researcher was very careful to match every student's performance within all five modules so as to achieve reliable results.

The chapter ended by looking at how data was analysed. Pearson's correlation coefficient was used to analyse data and the results were used to answer the first research question. SPSS's multiple regression analysis was used to analyse data and the results were used to answer the second research question. The results of these two tests are discussed in the next chapter.

CHAPTER FOUR

PRESENTATION AND INTERPRETATION OF RESULTS

4.1 INTRODUCTION

This chapter focuses on the presentation and analysis of results. The presentation of results is divided into three sections. The first section gives the Normal P-P of regression tests that was carried out on the model before using it. The second section presents descriptive statistics, regression and correlation between the predictors and the criterion on yearly bases for 2011 and regression summary for 2012 to 2016. The third section, discusses how data from all these years were combined and treated as one in answer to the research questions. The section also describes descriptive statistics, regression and correlation between the variable Introduction to Programming Systems and selected predictor modules.

4.2 DESCRIPTION OF DATA

The data that was used in this research were final examination scores of students in five modules. Students' scores are considered scale data. This data was collected from the university database. A data extraction sheet (see **Appendix 2**) was designed and used to extract data from the source documents. The data extraction sheet was used to collect data from the different years. Similar results were obtained from each year which showed that the instrument was reliable. The research measured the performance of students in five modules. These scores went through moderation before being published by the university as official performance of students. This renders the test scores unquestionable and hence valid for the purpose of this research.

4.3 MODEL TESTS

The model used in this research consists of four predictor variables and one criterion variable. Data from these variables must be normal in order for the model to be valid. The Normal P-P test was carried out to test for data normality as shown in **figure 4.1**.

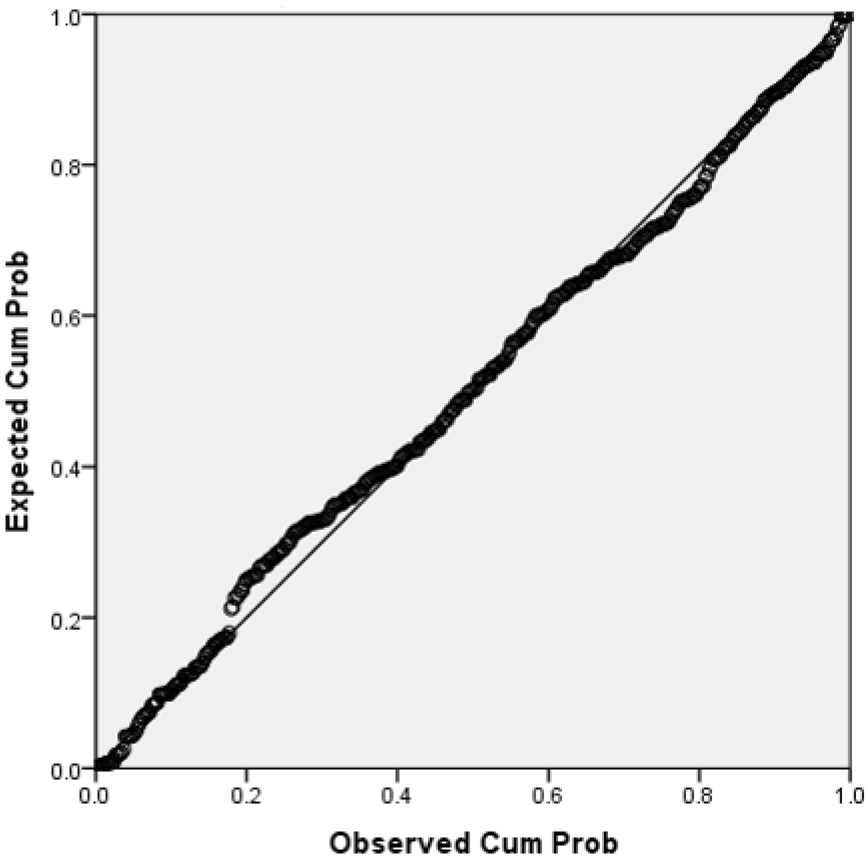

Figure 4.1 Normal P-P Plot

The continuous line in **figure 4.1** shows the normal distribution of the sample. The data points closely follow the line which means the sample is normally distributed and can therefore be used for analysis.

4.4 PRESENTATION AND DISCUSSION OF 2011 RESULTS

4.4.1 Descriptive statistics

Descriptive statistics (mean and standard deviation) were computed for year 2011 and sample size (n) was 77. The means of the modules are given in **table 4.1**.

Table 4.1 Descriptive statistics of the variables in the study.

Descriptive Statistics (n =77)			
	Mean	Std. Deviation	n
Introduction to Programming Systems	50.13	13.830	77
Calculus	41.94	17.061	77
General Chemistry	43.03	13.300	77
Classic Mechanics	46.09	14.719	77
Discrete Mathematics	46.97	14.164	77

The mean of Introduction to Programming Systems ($\overline{x}$ = 50.13, SD =1 3.830) was higher than that of all predictor modules. Among the predictors, Discrete Mathematics had the highest mean ($\overline{x}$ = 46.06, SD = 14.164) followed by Classic Mechanics with ($\overline{x}$ = 46.09, SD =14.719) then General Chemistry with ($\overline{x}$ = 43.03, SD = 13.300), and lastly Calculus with ($\overline{x}$ = 41.94, SD = 17.061).

The results show that the criterion had the highest mean compared with all predictor module means. The predictor modules are completed in first year first semester by students and the criterion is completed in the second semester. The means of predictor modules are lower than those of the second semester.

4.4.2 Correlation

Correlation measures the strength or degree of linear association between predictors and Introduction to Programming Systems (criterion). The correlation coefficients of the predictor modules are given in **table 4.2.**

Table 4.2: Correlation matrix of criterion and predictors

Correlations

	Introduction to Programming Systems	Calculus	General Chemistry	Classic Mechanics	Discrete Mathematics
Introduction to Programming Systems	1.000	0.699	0.399	0.628	0.620
Calculus	0.699	1.000	0.423	0.622	0.669
General Chemistry	0.399	0.423	1.000	0.362	0.458
Classic Mechanics	0.628	0.622	0.362	1.000	0.721
Discrete Mathematics	0.620	0.669	0.458	0.721	1.000

Significant at 0.05.

The results show that there is a positive correlation between the predictors and Introduction to Programming Systems. The following modules: Calculus ($r = 0.699$), Classic Mechanics ($r = 0.628$) and Discrete Mathematics ($r = 0.620$) showed a strong positive correlation with Introduction to Programming Systems since their Pearson's r values were bigger than 0.500. General Chemistry ($r = 0.399$) showed a weak correlation with the criterion.

4.4.3 Multiple regression

Multiple regression is used to check if predictors have significant relationships with the criterion. Regression identifies the strength of the effect that predictors have on the dependant variable.

Table 4.3: Multiple regression of the predictor variables on the performance in introduction to programming systems.

Model Summary				
Model	R	R Square	Adjusted R Square	Std. Error of the Estimate
	0.749	0.561	0.537	9.412

The multiple regression correlation coefficient (R) showing the linear relationship between the four predictors (Calculus, Discrete Mathematics, General Chemistry and Classic Mechanics) and Introduction to Programming Systems is ($R = 0.749$). This means that there was a strong linear relationship between the predictors and Introduction to Programming Systems. The R square value is 0.561. This means that predictors to Introduction to Programming Systems account for 56,1% of the variation in the performance in Introduction to Programming Systems.

The analysis of variance (ANOVA) of this regression model was used to determine whether there are any statistically significant differences between the means of the variables.

Table 4.4: Multiple regression ANOVA

ANOVA					
Model	Sum of Squares	df	Mean Square	F	Sig.
Regression	8158.871	4	2039.718	23.027	.000
Residual	6377.831	72	88.581		
Total	14536.701	76			

Table 4.4 shows the Multiple Regression ANOVA of the predictors of performance in Introduction to Programming Systems. **Table 4.3** shows that there is a strong linear relationship between the predictors and the criterion. That linear relationship was verified using multiple regression ANOVA. About 56,1% (8158.871) of the variation in regression is explained by the model while 43.9% (6377.831) is residual. This means that the model variable (predictors) explain more than half of the model variability. The results show that $p<0.01$, which is less than 0.05, meaning that there was a significant linear relationship between predictors and performance in Introduction to Programming Systems.

The coefficients that indicate the relative effects of each of the four predictors on the performance in Introduction to Programming Systems are shown in **table 4.5**.

Table 4.5: Coefficients

Model	Unstandardised Coefficients B	Unstandardised Coefficients Std. Error	Standardised Coefficients Beta	t	Sig.	Remark
(Constant)	16.037	4.417		3.631	.001	
Calculus	.354	.090	.437	3.949	.000	Significant
General Chemistry	.074	.093	.071	.799	.427	Not Significant
Classic Mechanics	.230	.110	.245	2.092	.040	Significant
Discrete Mathematics	.116	.123	.119	.939	.351	Not Significant

The results show that Calculus ($\beta = 0.354$; $t = 3.631$; $p < 0.05$) and Classic Mechanics ($\beta = 0.230$; $t = 2.092$; $p < 0.05$) were the only modules with $p<0.05$ which means that these were the only modules that had a significant effect on Introduction to Programming Systems. General Chemistry ($\beta = 0.074$; $t = 3.949$; $p < 0.43$) and Discrete Mathematics ($\beta = 0.116$ $t = 0.939$; $p < 0.35$) resulted in $p>0.05$ which means that they had an insignificant effect on Introduction to Programming Systems.

4.5 SUMMARY OF RESULTS FOR ALL THE YEARS

An analysis of the raw data for the years 2012 to 2016 was done in a similar way to the one in **Section 4.4** and the results are summarised here in **table 4.6.**

Table 4.6 Summary of results for all the years

Model strength, Predictor coefficients, probabilities and significance status									
	2011 (R Square = .561)			**2012 (R Square = .263)**			**2013 (R Square = .265)**		
Variables	B	p	Status	B	p	Status	B	p	Status
Calc	.354	.000	Sig.	.207	.051	Not Sig.	-.121	.395	Not Sig.
Chem	.074	.427	Not Sig.	.040	.723	Not Sig.	-.165	.437	Not Sig.
Mech	.230	.040	Sig.	.291	.016	Sig.	.060	.691	Not Sig.
Dis. Math	.116	.351	Not Sig.	-.007	.944	Not Sig.	.485	.003	Sig.

Model strength, predictor coefficients, probabilities and significance status									
	2014 (R Square = .387)			**2015 (R Square = .432)**			**2016 (R Square = .313)**		
Variables	B	p	Status	B	p	Status	B	p	Status
Calc	.141	.108	Not Sig.	.176	.096	Not Sig.	.101	.088	Not Sig.
Chem	.045	.739	Not Sig.	.088	.373	Not Sig.	.119	.022	Sig.
Mech	.216	.040	Sig.	.098	.377	Not Sig.	.120	.022	Sig.
Dis. Math	.475	.000	Sig.	.303	.004	Sig.	.111	.800	Not Sig.

Key: Calc - Calculus; Chem – General Chemistry; Mech – Classic Mechanics; Dis, Math – Discrete Mathematics.

The results in **table 4.6** show that, in 2011, explained variation for the model is 56.1%. The remainder of the variation is explained by other factors outside of the given four predictors. This means that 56.1% of the performance in Introduction to Programming Systems is due to the four predictors (Calculus, Discrete Mathematics, General Chemistry and Classic Mechanics). Subsequently the explained variation is 26.3%, 26.5%, 38.7%, 43.2%, and 31.3% for the years 2012 to 2016 respectively.

Results in **table 4.6** also show that Calculus was not a significant predictor across all the years, except in 2011. Chemistry was a significant predictor only in 2016. Mechanics was a significant predictor in 2011, 2012, 2014 and 2016 suggesting a strong presence in the model. Finally, Discrete Mathematics was a significant predictor in 2013, 2014 and 2015, making it the next best predictor.

The weak explanatory power of the model coupled with erratic significance levels of predictors in the individual years seem to suggest that time (years) is not a factor in the model's predictive power. To test if time is a factor, a dummy variable test was conducted where time (years) was treated as a dummy variable with 2011 as the base year. The results are shown in **table 4.7**.

Table 4.7 Time dummies for the different years

	(1)	(2)	(3)	(4)
Dependent: Introduction to Programming Systems	ordinary least squares	ordinary least squares	ordinary least squares	ordinary least squares
Calculus	0.372***	0.343***	0.241***	0.189***
	(0.028)	(0.031)	(0.041)	(0.043)
Chemistry		0.158***	0.121***	0.089**
		(0.045)	(0.045)	(0.044)
Physics			0.251***	0.178***
			(0.052)	(0.057)
Discrete Mathematics				0.207***
				(0.047)
2012.year				0.356
				(1.831)
2013.year				2.399
				(2.107)
2014.year				-1.630
				(1.553)
2015.year				-1.634
				(1.870)
2016.year				1.928
				(1.622)
Constant	34.058***	28.357***	23.346***	20.435***
	(1.436)	(2.114)	(2.381)	(2.461)
Observations	399	399	399	399
R-squared	0.236	0.255	0.310	0.357
Year FE	NO	NO	NO	YES
F	175.8	90.29	71.87	33.14

***Significant at $p<0.05$

The time variable coefficients were found to be 1.831 (for 2012), 2.107 (for 2013), 1.553 (for 2014), 1.870 (for 2015), and 1.928 (for 2016). These coefficients were all not significant at 5% level of significance, meaning time is not an important factor. After discovering that time was not a factor, the researcher then decided to use the overall model to answer the research questions.

4.6 OVERALL MODEL

Table 7 shows that the time variable coefficients were not significant at 5% level of significance. The meaning of this is that, the conditions under which the modules were studied were not a factor. This is as good as saying the students were exposed to similar conditions as though they were in the same class, and as a result their data can be treated as one. Data from six data extraction sheets (each year had its own data extraction sheet) was used to compile a single data extraction sheet (overall model data extraction sheet). This data from the overall model data extraction sheet was used to form the overall model and was analysed as one entity to answer the research questions.

4.6.1 Descriptive statistics

The mean and standard deviation of all the years was computed and the sample size of the study was 399. The means of the modules are given in **table 4.8.**

Table 4.8. Descriptive statistics of the variables in the study.

Descriptive Statistics			
	Mean	Std. Deviation	n
Introduction to Programming Systems	50.76	12.766	399
Calculus	44.90	16.676	399
General Chemistry	44.40	11.457	399
Classic Mechanics	44.67	14.027	399
Discrete Mathematics	47.94	14.331	399

Table 7 shows the means and standard deviations of the modules. Here are the means of the modules in descending order: Introduction to Programming Systems ($\bar{x}$ = 50.76, SD = 12.766), Discrete Mathematics ($\bar{x}$ = 47.94, SD = 14.331), Calculus ($\bar{x}$ 44.90, SD = 16.676), Classic Mechanics ($\bar{x}$ = 44.67, SD = 14.027) and General Chemistry with ($\bar{x}$ = 44.40, SD = 11.457).

The results show that the mean of Introduction to Programming Systems was higher than the means of predictors. The mean for Introduction to Programming Systems is followed by means of Mathematics modules (Discrete Mathematics and Calculus). The means of Mathematics modules are followed by mean of the Physics module (Classic Mechanics) and then General Chemistry.

4.6.2 Correlation

Research Question 1: Do selected predictors have an effect on performance in Introduction to Programming Systems?

To answer this question a correlation matrix (**Table 4.9**) showing the correlation coefficient between Introduction to Programming Systems and the selected predictors was used.

Table 4.9: Correlation Matrix Showing the Relationship Between Introduction to Programming Systems and Selected Predictor Variables.

Correlations

	Introduction to Programming Systems	Calculus	General Chemistry	Classic Mechanics	Discrete Mathematics
Introduction to Programming Systems	1.000	.486	.263	.465	.480
Calculus	.486	1.000	.270	.514	.524
General Chemistry	.263	.270	1.000	.250	.300
Classic Mechanics	.465	.514	.250	1.000	.492
Discrete Mathematics	.480	.524	.300	.492	1.000

Significance at $p<0.05$

The results show that there is a positive correlation between the predictors and Introduction to Programming Systems. From the results, Mathematics module, Calculus had the highest correlation coefficient of 0.486 followed by another Mathematics module, Discrete Mathematics with 0.480. Physics module, Classic Mechanics with 0.465 followed the Mathematics modules and lastly General

Chemistry with 0.236. All these coefficients were positive but weak since they are all less than 0.500.

4.6.3 Regression

The correlation coefficient does not imply causality. It may show that there is a strong correlation between Introduction to Programming Systems and one of the predictors, however it does not mean that the predictor is responsible for the performance in Introduction to Programming Systems. That information can be obtained from regression analysis. Multiple Regression was used to answer the second research question which is: Which of the selected predictors has high predictive validity?

Table 4.10: Multiple regression of the predictor variables on the performance in introduction to programming systems (n=399).

Model Summary

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate
	.586	.343	.337	10.397

Table 4.10 presents the multiple regression results of the predictors on the performance of Introduction to Programming Systems. The multiple regression correlation coefficient (R) showing the linear relationship between the four predictors (Calculus, Discrete Mathematics, General Chemistry and Classic Mechanics) and Introduction to Programming Systems is 0.586. This means that there was a strong linear relationship between the predictors and Introduction to Programming Systems. The R square value is 0.343. This means that predictors of Introduction to

Programming Systems account for 34.3% of the variation in the performance in Introduction to Programming Systems.

Table 4.11: Multiple regression ANOVA.

ANOVA					
Model	Sum of Squares	df	Mean Square	F	Sig.
Regression	22275.151	4	5568.788	51.521	.000
Residual	42586.703	394	108.088		
Total	64861.855	398			

Table 4.11 shows the Multiple Regression ANOVA of the predictors on performance in Introduction to Programming Systems. The F-test showed that, F-ratio = 51.521, $p<0.05$. This indicated that jointly (the predictors put together), the entire model has a significant effect on the performance of students in Introduction to Programming System.

Research Question 2: Which of the selected predictors has high predictive validity?

To answer the second research question, **Table 4.12** of coefficients was used. Standardised coefficients are used to determine which one of the independent variables is more important. These coefficients are used to determine the ordinal strength of the predictors. Unstandardised coefficients are used to determine what effect 1 unit change in an independent variable will have on the dependent variable. Unstandardised coefficients are used to formulate the estimated model.

Table 4.12: Coefficients

Model	Unstandardised Coefficients		Standardised Coefficients	t	Sig.	Remark
	B	Std. Error	Beta			
(Constant)	20.445	2.514		8.131	.000	
Calculus	.181	.039	.237	4.604	.000	Significant
General Chemistry	.086	.048	.077	1.782	.075	Not Significant
Classic Mechanics	.193	.046	.212	4.226	.000	Significant
Discrete Mathematics	.203	.045	.228	4.478	.000	Significant

The results show that Calculus (β= 0.237; t = 4.604; p < 0.05), Discrete Mathematics (β = 0. 228; t = 4.478; p < 0.05) and Classic Mechanics (β = 0. 212; t = 4.226; p < 0.05) were the modules with p<0.05 which means that these were the only modules that had a significant effect on Introduction to Programming Systems. General Chemistry (β = 0. 077; t = 3.949; p < 0.43) showed p>0.05 which means it had an insignificant effect on Introduction to Programming Systems.

Standardised coefficients showed that Calculus had the highest relevant effect on Introduction to Programming Systems with (β = 0.237; t = 4.604; p<0.01). The interpretation of the results is that if the overall mark obtained in Calculus increased by one mark then the mark in Introduction to Programming Systems will increase by 0.237 ceteris paribus. Discrete Mathematics followed Calculus with (β = 0.228; t =4.478; p<0.01). This means if marks of all other modules were kept constant, the mark in Introduction to Programming Systems would increase by 0.228 when the mark in Discrete Mathematics increased by 1. Classic Mechanics has (β = 0.212; t = 4.226; $p<0.01$). An improvement in the Classic Mechanics mark would raise the student's mark in Programming by 0.212 holding constant all other modules. General Chemistry (β = 0.077; t = 1.782; $p<0.1$) is only significant at the 10 percent level which is bigger than 5 percent and hence it has an insignificant effect on Introduction to Programming Systems.

Using the coefficients of the predictors to determine the ordinal strength of the predictors, the ordinal strength is as follows in order of importance: Calculus, Discrete Mathematics, Classic Mechanics and General Chemistry.

4.6.4 Estimated model

The equation of the estimated model shows the regression coefficients that represent each predictor's contribution to the prediction of Introduction to Programming Systems. The constant of the equation (20.445) means, if all the regression coefficients of predictors are zero, then the mark in Introduction to Programming Systems will be 20,445. The coefficients indicated that if all other predictors are kept constant the mark in Introduction to Programming Systems would increase by the magnitude of the coefficient. The regression equation is as follows:

$Y = 0.181X_1 + 0.086X_2 + 0.203X_3 + 0.192X_4 + 20.445$, where

Y is Introduction to Programming Systems

X_1 is Calculus

X_2 is General Chemistry

X_3 is Discrete Mathematics

X_4 is Classic Mechanics

Research Question 3: How can universities use these predictors as criteria for success in Introduction to Programming Systems?

To answer this research question, the results of the first two research questions are used. In the first research question, the results showed that there was a correlation between the selected predictors and Introduction to Programming Systems. In the second research question, the results showed that General Chemistry had a weakly significant effect on Introduction to Programming Systems, while Calculus, Discrete Mathematics, Classic Mechanics had the most relevant effect on Introduction to Programming Systems. These results can be used by universities to offer students Calculus, Discrete Mathematics and Classic Mechanics as predictors of performance in Introduction to Programming Systems and find another module to replace General Chemistry as a predictor.

4.7 SUMMARY

This chapter focused on the statistical tests that were carried out to make sure that the model produced reliable results. Normal P-P Plot Test was used and it showed that the data was normally distributed. The chapter also discussed the analyses that were carried out to answer the research questions. The first research question was answered by calculating a Pearson correlation coefficient. The results showed that all the four predictors had a positive correlation with performance in Introduction to Programming Systems. Though all the predictors showed a positive correlation with Programming, the Pearson coefficient was less than 0.5 which indicated it is a weak positive correlation. The second research question was answered by multiple regression analysis. All predictor variables had a significant effect on Introduction to Programming Systems. However, it turns out that Calculus, Discrete Mathematics and Classic Mechanics are the modules that had the most relevant effect on Introduction to Programming Systems. General Chemistry had a weak correlation but insignificant effect on Introduction to Programming Systems given that it was only significant at 10 percent level compared to Calculus, Discrete Mathematics and Classic Mechanics which are significant at 1 percent level. The third research question was answered by analysing the results of the first two research questions. The results of the first two showed that Calculus, Discrete Mathematics and Classic Mechanics can be used by universities as predictors of Introduction to Programming Systems while General Chemistry is not a good predictor.

CHAPTER 5

DISCUSSION, CONCLUSION AND RECOMMENDATIONS

5.1 INTRODUCTION

This chapter provides a discussion of the findings, implications derived from the findings and conclusion. The limitations of the study are presented, and recommendations for further study are also suggested.

5.2 FINDINGS

The three research objectives were as follows:

5.2.1 To establish the predictive validity of selected predictors on performance in Introduction to Programming Systems.

5.2.2 To establish an ordinal strength of selected predictors on success in Introduction to Programming Systems.

5.2.3 To find out if universities can rely on selected predictors as predictors of success in Introduction to Programming Systems.

The findings will be discussed as follows; findings of descriptive statistics and then the findings of all the research objectives.

5.2.1 Findings of descriptive statistics

The mean of Introduction to Programming Systems ($\bar{x}$ = 50.76) was higher than the means of Discrete Mathematics ($\bar{x}$ = 47.94), Calculus ($\bar{x}$ = 44.90), Classic Mechanics ($\bar{x}$ = 44.67) and General Chemistry with ($\bar{x}$ = 44.40).

These results showed that the criterion module (Introduction to Programming Systems), completed in the second semester, had the highest mean compared with all the predictor modules (Discrete Mathematics, Calculus, Classic Mechanics and General Chemistry) done in first year, first semester. The difference in the means of first and second semester modules might be a result of students trying to adapt to university life during the first semester (Riehl, 1994). These findings can be used to inform the Department of Computer Sciences to present Programming modules to students in the second semester rather than the first semester. However, these findings can also mean that Programming modules are performed better than Mathematics, Physics and Chemistry modules. To really understand the implication of these findings more research needs to be carried out.

5.2.2 The predictive validity of selected predictors on performance in introduction to Programming Systems

The results from chapter 4, **table 4.9** showed that there was a weak positive correlation between the Introduction to Programming Systems module (criterion) and all the selected predictors, namely, Mathematics modules (Calculus and Discrete Mathematics), Classic Mechanics and General Chemistry. However, the strength of the correlation varied. Mathematics modules (Calculus and Discrete Mathematics) had the highest correlation with Introduction to Programming Systems ($r = 0.486$ and 0.480, $p<0.01$) respectively. These findings are consistent with the literature (Bergin & Reilly, 2005; Bohlmann & Pretorius, 2008; Byrne & Lyons, 2001; Chowdhury et al., 1987; Gomes & Mendes, 2008; Leeper & Silver, 1982; Owolabi et al., 2014a). Nevertheless, these findings are not consistent with the findings of Van der

Westhuizen and Barlow-Jones (2015a), who found that Mathematics is not a predictor of success in Introduction to Programming Systems.

A Physics module, Classic Mechanics followed the two Mathematics modules ($r = 0.465$, $p < 0.01$). This was expected because of the relationship between Mathematics and Physics. Physics is more of an application of Mathematics since the two subjects are highly integrated. In some cases, Mechanics is considered to be a section of Mathematics. This finding is in line with that of Bergin and Reilly (2005), who found that there was a weak positive correlation between Introduction to Programming Systems and Science subjects. A Chemistry module, General Chemistry showed the lowest positive correlation with Introduction to Programming Systems ($r = 0.236$). This finding is also consistent with the findings of (Bergin & Reilly, 2005) who found that there was no correlation between Chemistry and Introduction to Programming Systems.

5.2.3 An ordinal strength of selected predictors of success in introduction to Programming Systems.

The table of coefficients **(Table 4.12)** showed standardised coefficients which were used to determine which one of the independent variables were significant. It was established that the coefficients of Calculus, Discrete Mechanics and Classic Mechanics were significant and only General Chemistry is insignificant. The results established that the Mathematics module, Calculus ($\beta = 0.237$; $t = 4.604$; $p<0.01$) had the highest relevant effect on Introduction to Programming Systems. This result indicated that, if the mark obtained in Calculus increased by one mark then the mark in Introduction to Programming Systems would increase by 0.237 ceteris paribus. This implies that Calculus is a good predictor of Introduction to Programming Systems. The second important Mathematics module in this model was Discrete Mathematics (β = 0.228; t =4.478; p<0.01). If marks in all other predictors were kept constant, the mark in Introduction to Programming Systems would increase by 0.228 when the mark of Discrete Mathematics is increased by one. This implies that Discrete Mathematics is a good predictor of Introduction to Programming Systems. Both Mathematics modules were significant at 1% level. This finding shows that Mathematics ability is a good predictor of success in Introduction to Programming Systems, which is consistent with the findings of (Owolabi et al., 2014a). Other things being constant, scores in Introduction to Programming Systems will increase by 0.212 when the mark in Classic Mechanics is increased by one. The results of Classic Mechanics were significant at 1% level. This implies that Classic Mechanics is a good predictor of Introduction to Programming Systems. Lastly, if the marks of Discrete Mathematics, Calculus and Classic Mechanics were kept constant, the mark in Introduction to Programming System would increase by 0.077 when the mark in General Chemistry (β = 0.077; t = 1.782; p <0.1) is increased by one. This result was significant at 10% level. This shows that there is a positive correlation between the criterion and the predictor but it is not statistically significant. In other words, General Chemistry is not a good predictor of success in Introduction to Programming Systems. These results revealed that the ordinal strength of the predictors is as follows: Calculus, Discrete Mathematics, Classic Mechanics and lastly General Chemistry which is not a good predictor.

5.2.4 Can universities rely on selected predictors as predictors of success in Introduction to Programming Systems?

Since time dummies reflected the absence of year specific effects, this means that, the year and the conditions under which these modules were studied for six years did not affect the result. These results show a trend for the past six years. It can be concluded that universities can rely on these results and that Calculus, Discrete Mathematics and Classic Mechanics are good predictors of success in Introduction to Programming Systems while General Chemistry is not.

5.3 IMPLICATIONS OF THE RESULTS

The findings of this study indicated that there is a positive significant correlation between Mathematics modules (Calculus and Discrete Mathematics) and Introduction to Programming Systems. Mathematics is one of the subjects used by University of Zululand to select students for Computer Science programmes. Due to Mathematics' power to predict students' future achievement in Computer Programming, the selection criterion for entering University of Zululand's Computer Science programmes is considered to be appropriate. On the other hand, students who fail these modules in their first semester are at risk of failing Introduction to Programming Systems and hence should be advised to choose other modules or to retake these modules before taking Introduction to Programming Systems.

High school Physical Sciences is another subject considered by University of Zululand to select students for Computer Science programmes (Zululand, 2017). This subject combines two topics, Physics and Chemistry. These two topics are given equal weighting at grade 12 level. High school Physics covers three main concepts, Waves and Light, Mechanics, and Electricity and Magnetism. The Mechanics section accounts for less than 20% of the final mark (Education, 2011). General Chemistry accounts for 50% of the final mark. This means that over 80% of the final mark is not Mechanics. This augurs correctly with the current study. The findings of this research showed that the Physics module Classic Mechanics is a good predictor, whilst General Chemistry is not a good predictor of success in Introduction to Programming Systems. Due to the lack of predicting power of General Chemistry, which accounts for 50% of high school Physical Sciences final mark, incorporating Physical Sciences as a significant predictor for studying Introduction to Programming Systems is inappropriate.

5.4 RECOMMENDATIONS

The results imply that more Mathematics and Physics modules can be used to prepare students for Introduction to Programming Systems. This also suggests that instead of having a Chemistry module in the first semester, it should rather be swapped with Mathematics or Physics modules which are currently in semester two. The results of semester one modules, which will then include predicting modules could then be used to advise students who are at risk of failing, not to take Introduction to Programming Systems or to retake failed modules before taking it. The same results can also be used to encourage capable students to take Introduction to Programming Systems who may not have originally intended to take it. All this will help to increase the pass rate in Introduction to Programming Systems and help mend the image already tarnished by high failure rates.

5.5 LIMITATIONS OF THIS STUDY

The researcher only used data from one university out of twenty-three universities in South Africa. Therefore, generalisation of the findings from this study to the entire population of Computer Science programmes must be approached with caution.

5.6 AVENUES OF FURTHER RESEARCH

The results showed that the model (of four predictors) explained about 34% of the success in Introduction to Programming Systems, which leaves about 66% explained by other factors. Therefore, future research could be done to identify other predictors and factors that could explain the unexplained variation.

The reviewed literature and the results showed that there is a weak positive correlation between ability in Mathematics and performance in Introduction to Programming Systems. Future studies could focus on finding out why there is always weak correlation, or to identify other modules which may be better predictors of performance in Introduction to Programming Systems.

Reviewed literature has also shown that there is high failure rate in Introduction to Programming Systems. The descriptive results showed that the mean of Introduction to Programming Systems is higher than the means of all predictor modules. This may be because the pass rate of predictor modules is lower than that of Introduction to Programming. Future studies should focus on comparing students' failure rates in other modules to see if Introduction to Programming Systems has high failure rates or if those high failure rates exist across other modules as well.

5.7 CONCLUSION

Introduction to Programming Systems is considered to be a difficult module with high failure rate (Bennedsen & Caspersen, 2007). Predictors of success in Introduction to Programming Systems can be used to predict which students are likely to fail, and encourage them not to take this course. These predictors can also be used to advise capable students to take Introduction to Programming Systems. This might reduce the failure rates in Introduction to Programming Systems since more capable students will be taking this course. Mathematics (Calculus and Discrete Mathematics) and Physics (Classic Mechanics) modules are good predictors of Introduction to Programming Systems, while General Chemistry is not a good predictor of success in Introduction to Programming Systems. Universities can give students predictor modules before admitting them to the Introduction to Programming Systems course, as this will help students to prepare for it and also assist universities to advise students accordingly.

REFERENCES

Anastasi, A. (1976). *Psychological Testing* (4th ed.). New York: Macmillan.

Anastasi, A., & Urbana, S. (1997). *Psychological Testing (7th edition)* (7th ed.). Upper Saddle River, NJ: Prentice – Hall.: Pearson.

Anderson, L. W., Krathwohl, D. R., Airasian, P., Cruikshank, K., Mayer, R., Pintrich, P., . . . Wittrock, M. (2001). A taxonomy for learning, teaching and assessing: A revision of Bloom's taxonomy. *New York. Longman Publishing. Artz, AF, & Armour-Thomas, E.(1992). Development of a cognitive-metacognitive framework for protocol analysis of mathematical problem solving in small groups. Cognition and Instruction, 9*(2), 137-175.

Anwar, M. A., Ahmed, N., & Khan, W. (2012.). Analysis of Students' Grades in Mathematics, English, and Programming Courses: A KDD Approach. *International Journal of Future Computer and Communication., Vol. 1.*(No. 2.).

Ayaya, O. A. (1996). The prediction of academic performance in the first-year: a case study at the National University of Lesotho. *South African Journal of Higher Education, 10*(2), 101 – 113.

Barlow-Jones, G., Van der Westhuizen, D., & Coetzee, C. (2015). *Is Mathematics Still Relevant As An Admission Criterion For Entry Into An Information And Communication Technology Course At A South African University.* Paper presented at the ICICTE 2015 Proceedings

Bennedsen, J., & Caspersen, M. E. (2005). *An investigation of potential success factors for an introductory model-driven programming course.* Paper presented at the In Proceedings of the first international workshop on computing education research New York.

Bennedsen, J., & Caspersen, M. E. (2007). Failure rates in introductory programming. *SIGCSE Bulletin (Association for Computing Machinery, Special Interest Group on Computer Science Education),, 39*(2), pp.32-36.

Bergin, S., & Reilly, R. (2005). *Programming: factors that influence success. .* Paper presented at the Proceedings of the 36th SIGCSE technical symposium on Computer science education, ,St. Louis, Missouri, USA. .

Bohlmann, C., & Pretorius, E. (2008). Relationships between mathematics and literacy: Exploring some underlying factors. Pythagoras,. *South African Journal of Hgher Education*, 42-55.

Bosquea, L. D., Martinez, R., & Torres, J. L. (2015). *Decreasing Failure in Programming Subject with Augmented Reality Tool.* Paper presented at the 2015 International Conference on Virtual and Augmented Reality in Education.

Byrne, P., & Lyons, G. (2001). *The Effect of Student Attributes on Success in Programming.* Paper presented at the Proceedings of the 6th Annual Conference on Innovation and Technology in Computer Science Education., ACM New York, NY, USA.

Cafolla, R. (1988). Piagetian Formal Operations and Other Cognitive Correlates of Achievement in Computer Programming. *J. Educ. Technol. Syst., 16*(1).

Caspersen, M. E. (2007). *Educating Novices in The Skills of Programming* (PhD Dissertation), University of Aarhus, Denmark

Cavanagh, S. (2003). Many State Tests Said to Be Poor Predictors of College Readiness. *EducationWeek, 23*(9), 13.

Chowdhury, A., A., Nelson, C., V., Fuelling, C. P., & McCormick, R. L. (1987). *Predicting Success of a Beginning Computer Course Using Logistic Regression.* Paper presented at the Proceedings of the 15th annual conference on Computer Science.

Cohen, L., & Manion, L. (1980). *Research Methods in Education*. London: Groom Helm Ltd.

Creswell, J., W. . (2012). *Educational Research. Planning, Conducting, and Evaluating Quantitative and Qualitative Research.* (M. Karen Ed. 4th ed.). Boston, USA: Pearson Education.

Cronbach, L. J. (1971). *Test validation*. Washington, D. C: American: Council on Education.

Dawes, P., Yeld, N., & Smith, M. J. (1999). Access, selection and admission to higher education: maximizing the use of the school-leaving examination. *South African Journal of Higher Education, 13*(3), 97 – 104.

Education, D. o. B. (2011). *Curriculum and Assessment Policy Statement Grade 10-12. Physical Sciences*.

Erdogan, Y., Aydin, E., & Kabaca, T. (2006). Exploring the Psychological Predictors of Programming Achievement. *Journal of Instructional Psychology, Vol 35*(No.3), 264-269.

Evans, G. E., & Simkin, M. G. (1989). What best predicts computer proficiency? *Communication of the ACM, 32*(11), 1322-1327.

Faulkner, L. R. (2002, April 05, 11.). Class rank predicts success. *USA Today*.

Foxcroft, C., & Roodt, G. (2001). *An Introduction to Psychological Assessment*. Oxford: Oxford University Press.

Gibbs, D. C. (2000). The effect of a constructivist learning environment for fielddependent/ independent students on achievement in introductory computer programming. *SIGCSE Bulletin (Association for Computing Machinery, Special Interest Group on Computer Science Education),, 32*(1), 207-211.

Golding, P., & McNamarah, S. (2005). *Predicting Academic Performance in the School of Computing & Information Technology.* Paper presented at the 35th ASEE/IEEE Frontiers in Education Conference, Indianapolis.

Gomes, A., & Mendes, A. (2008). *A study on students' characteristics and programming learning.* Paper presented at the Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications 2008, Chesapeake.

Gregory, R. J. (1996). *Psychological Testing* (2nd ed.). Boston: Allyn & Bacon.

Greyling, J. H. (2000). *The compilation and validation of a computerised selection battery for Computer Science and Information systems students.* (Doctoral thesis), University of Port Elizabeth.

Hagan, D., & Markham, S. (2000). *Does it help to have some programming experience before beginning a computing degree program?* Paper presented at the Proceedings of the 5th annual SIGCSE/SIGCUE ITiCSEconference on Innovation and technology in computer science education, , Helsinki, Finland.

Huysamen, G. K., & Roozendaal, L. A. (1999.). Curricular choice and the differential prediction of the tertiary-academic performance of men and women. *South African Journal of Psychology., 29*(2), 87-94.

Jegede, P. O. (2009). Predictors of java programming self efficacy among engineering students in a Nigerian University. *arXiv preprint arXiv:0909.0074*.

Jenkins, T. (2002). On the difficulty of learning to program. *Proceedings of the 3rd Annual Conference of the LTSN Centre for Information and Computer Science, August 27-29, Loughborough, UK. Re- trieved April 1, 2009, 3*.

Kaiser, L., Kaminski, K., & Foley, J. (2013). Learning Transfer in Adult Education: New Directions for Adult and Continuing Education. *Wiley*, Pg 104.

Konvalina, J., Stephens, L. J., & Wileman, S. A. (1983). Identifying factors inuencing computer science aptitude and achievement. *AEDS Journal, 16*(2).

Kothari, C., R. (2004). *Research Methodology* (Second Revised ed.): New Age International Publishers.

Kumar, R. (2011). *Research Methodology a step-by-step guide for beginners* (3rd ed.). London: SAGE Publications.

Leeper, R. R., & Silver, J. L. (1982). Predicting success in a first programming course. . *Technical Symposium on Computer Science Education , Proceedings of the thirteenth SIGCSE technical symposium on Computer science education, Indianapolis, Indiana, United States. , 13*, 147-150.

Lindblom-Ylanne, S., Lonka, K., & Leskinen, E. (1999). On the predictive value of entry-level skills for successful studying in medical school. *Higher Education, 37*, 239 – 258.

Louw, A. D., Meyer, J. C., & van Schalkwyk, D. J. (1998). Die keuring van studente vir die Nasionale Diploma in Omgewingsgesondheid aan die Kaapse Technikon. *South African Journal of Hgher Education, 12*(2), 149-166.

Miller, R., & Bradbury, J. (1999). Academic performance of English first and second language students : Selection criteria. *South African Journal of Science, 95*(1), 30-34.

Norwawi, N. M., Hibadullah, C. F., & Osman, J. (2005). *Factors Affecting Performance in Introductory programming.* Paper presented at the In Proceedings of the International Conference on Qualitative Sciences and Its Applications

Nowaczyk, R. H. (1983). *Cognitive skills needed in computer programming.* Paper presented at the Annual Meeting of the Southeastern Psychological Association, Atlanta, Georgia.

Owolabi, J., Olanipekun, P., & Iwerima, J. (2014a). Mathematics Ability and Anxiety, Computer and Programming Anxieties, Age and Gender as Determinants of Achievement in Basic Programming. *GSTF Journal on Computing (JoC), 3*(4), 109.

Owolabi, J., Olanipekun, P., & Iwerima, J. (2014b). Mathematics Ability and Anxiety, Computer and Programming Anxieties, Age and Gender as Determinants of Achievement in Basic Programming. *GSTF Journal on Computing (JoC), 3*(4).

Pallant, J. (2013). *SPSS Survival Manual. Step By Step Guide To Data Analysis Using IBM SPSS* (5th ed.). New York: McGraw Hill.

Parliament. (2014). *Umalusi on standardising matric results*. Parliament .gov.za Retrieved from http://www.parliament.gov.za/live/.

Petersen, C., & Howe, T. (1979). Predicting academic success in introduction to computers. *AEDS Journal, 12*(4).

Polit, D. F., & Beck, T. C. (2010). *Essentials of nursing research : appraising evidence for nursing practice* (Seventh Edition ed.). China: Wolters Kluwer Health | Lippincott Williams & Wilkin.

Qian, Y., & Lehman, J. D. (2016). Correlates of Success in Introductory Programming: A Study with Middle School Students. *Journal of Education and Learning, Vol. 5*(No. 2).

Ramalingam, V., LaBelle, D., & Wiedenbeck, S. (2004.). *Self-efficacy and mental models in learning to program.* Paper presented at the Proceedings of the 9th annual SIGCSE conference on Innovation and technology in computer science education,, Leeds, United Kingdom.

Reilly, C. F., & Tomai, E. (2014). *An Examination of Mathematics Preparation For and Progress Through Three Introductory Computer Science Courses*. Paper presented at the Frontiers in Education Conference, Texas, USA

Riehl, R. J. (1994). The Academic Preparation, Aspirations, and First-Year Performance of First-Generation Students. *College and University, 70*(1), 14-19.

SAnews. (2015, 25 June 2015). Probe into matric group copying continues. *SAnews.* Retrieved from http://www.sanews.gov.za/south-africa/

Shrock, S., & Wilson, B. C. (2001). *Contributing to success in an introductory computer science course: A study of twelve factors.* Paper presented at the SIGCSE Bulletin ACM New York, NY, USA.

Shukur, Z., Alias, M., Hanawi, S. A., & Arshad, A. (2003). *Faktor-faktor Kegagalan: Pandangan pelajar Yang mengulang Kursus Pengaturcaraan C.* Paper presented at the Paper presented in Bengkel Sains Pengaturcaran, Kuala Lumpur.

Shultz, K. S., & Whitney, D. J. (2005). *Measurement Theory In Action*. California: Sage Publications, Inc.

Smith, S. (2015). Matric cheats uncovered in seven out of nine provinces. *Times Live,* p. 1.

Tabachnick, B. G., & Fiddell, L. S. (2013). *Using multivariate statistics* (6th edn ed.). Boston: Pearson Education.

Tai-sheng, F., & Yi-ching, L. (2002). Is Math Ability Beneficial to Performance in College Computer Science Programs? . *Journal of National Taipei Teachers College,, Vo1.* (no. XV), Pg 69-98.

Thorndike, E. L., & Woodworth, R. S. (1901). The influence of improvement in one mental function upon the efficiency of other functions. *Psychological Review, 8*, 247–261.

Torrey, L., & Shavlik, J. (2010). *Handbook of Research on Machine Learning Applications and Trends : Algorithms, Methods, and Techniques.* (Vol. 1): IGI Global.

University of Zululand. (2014). *University of Zululand Annual Report*. Retrieved from

Van der Westhuizen, D., & Barlow-Jones, G. (2015a). High school mathematics marks as an admission criterion for entry into programming courses at a South African university. *The Independent Journal of Teaching and Learning, 10*(1), 37-50.

Van Der Westhuizen, D., & Barlow-Jones, G. (2015b). High school mathematics marks as an admission criterion for entry into programming courses at a South African university. *The Independent Journal of Teaching and Learning (The IIE).*

Ventura, P. R. (2003). *On the Origins of Programmers: Identifying Predictors of Success for an Objects First CS1.* (PhD. dissertation), The State University of New York at Buffalo.

Vihavainen, A., Kurhila, J., Paksula, M., & Luukkainen, M. (2011). *Extreme Apprenticeship Method in Teaching Programming for Beginners.* Paper presented at the Proceedings of the 16th annual joint conference on Innovation and technology in computer science education.

Watson, C., & Li, F. W. B. (2014). Failure rates in introductory programming revisited. *Annual Joint Conference Integrating Technology into Computer Science Education*, 39.

White, G., & Sivitanides, M. (2005). An Empirical Investigation of the Relationship Between Success in Mathematics and Visual Programming Courses. *Journal of Information Systems Education, Vol 14*(No 4), pg 09-415.

Wiedenbeck, S. (2005). *Factors affecting the success of non-majors in learning to program.* Paper presented at the Proceedings of the 2005 international workshop on Computing education research, Seattle, WA, USA.

Wilson, B. C. (2002). A study of factors promoting success in computer science including gender differences. *Computer Science Education, 12*(1), 144-164.

Wirth, N. (2002). *Computing science education: the road not taken.* Paper presented at the Proceedings of the 7th annual conference on Innovation and technology in computer science education, Aarhus, Denmark.

Zaaiman, H., Van der Flier, H., & Thijs, G. D. (1998). Selecting South African higher education students.: critical issues and proposed solutions. *South African Journal of Higher Education, 12*(3), 96-101.

Zululand, U. o. (2017). *Faculty of science and agriculture undergraduate prospectus*. Retrieved from

APPENDICES

APPENDIX 1

Table A1: Overview of research in programming predictors

Researcher(s) and year	N	Language	Significant predictors
Newsted 1975	131	FORTRAN	Perceived ability, college GPA10, and student effort accounted for 41% of the variance.
Kurtz, 1980	23	FORTRAN	Level of formal reasoning accounted for 63% of the variance.
Leeper et al., 1982	92	Not specified	Number of high school English, Mathematics, Science, and Foreign Language units, SAT12 verbal score, SAT Mathematics score, and high school rank accounted for 26% of the variance.
Barker et al., 1983	353	Two languages	Piagetian intellectual development accounted for 12% of the variance.
Konvalina et al., 1983	382	BASIC	Students who completed the course (228) had significantly more Mathematics background than students who withdrew (154).
Hostetler, 1983	79	FORTRAN	GPA, diagramming and reasoning score, Mathematics background, and personality accounted for 77% of the variance.
Nowaczyk, 1983	286	COBOL	Performance in prior Mathematics and English courses accounted for a statistically significant amount of variation (but does not say how much).

Werth, 1986	58	Pascal	Significant correlation found for high school Mathematics, hours working at a part-time job, Piagetian intellectual development and cognitive style.
Mayer et al., 1986	57	BASIC	Word problem translation skills accounted for 50% of the variance.
Austin, 1987	76	Pascal	High school composite achievement, quantitative and algorithmic reasoning abilities, vocabulary and general information abilities, self-assessed Mathematics ability, and measures of an introverted/analytical style and extroverted level accounted for 64% of the variance of the variance.
Cafolla, 1988	23	BASIC	Cognitive development accounts for 34% of the variance.
Evans et al., 1989	117	BASIC	High school Mathematics courses, prior BASIC experience, hours playing video or computer games accounted for at most 23% of the variance of six different outcome variables (e.g. homework score, mid-term exam, and final exam).
Gibbs, 2000	50	BASIC	Within a constructivist learning environment, cognitive style was not found to influence Programming achievement.
Goold et al., 2000	39	C	Dislike for programming, gender, average on other modules, and raw secondary score (English plus other courses) accounted for 43% of the variance.

Hagan et al., 2000	97	JAVA	The more programming languages a student knew prior to taking the course, the higher the performance.
Byrne et al., 2001	110	BASIC	Mathematics (r = 0.353) and Science (r = 0.572) correlates with Programming performance.
Rountree et al., 2002	472	Java	The strongest single indicator of success was the grade the student expected to get on the course.
Stein, 2002	160	Java	Students who study Calculus do at least as well as students who study Discrete Mathematics.
Wilson, 2002	105	C++	Comfort level, Mathematics background, and attribution of success/failure to luck accounted for 44% of the variation. The number of hours playing computer games prior to the course had a negative effect while experience of a prior formal Programming class had a positive effect.
Holden et al., 2003	159	Java	Prior experience (independent of language) is an advantage in the first course in a programming sequence, but not in later courses.
Ramalingam et al., 2004	75	C++	Mental model, self-efficacy, and previous programming and computer experience accounted for 30% of the variance.
Ventura et al., 2004	499	Java	Prior programming experience was not a predictor of success for their objects-first CS1.

Bergin et al., 2005a	80	Java	Student's perception of his/her own understanding of the course, gender, Mathematics score, and comfort level accounted for 79% of the variance.
Ventura, 2005	499	Java	Percent lab usage, comfort level, and SAT Mathematics score accounted for 53% of the variance. Measures of effort are the primary predictors of success followed by comfort level, and then academic predictors (e.g. Mathematics) with marginal gains.
Wiedenbeck, 2005	120	C++	Prior computer and programming experience, self-efficacy, and knowledge organisation accounted for 30% of the variance.
Bergin et al., 2006	102	Java	Mathematics score, number of hours playing computer games, and programming self-esteem accounted for 80% of the variance. Mathematics score and programming self-esteem were found to have a positive relationship with performance while number of hours playing computer games was found to have a negative effect

APPENDIX 2

Table A2: Data extraction form

NUMBER	Introduction to Programming Systems	Classic Mechanics	Discrete Mathematics	Calculus	General Chemistry
1	51	50	48	50	45
2	50	13	31	59	32
3	50	28	16	40	26
4	31	30	51	32	44
5	51	58	77	64	46
6	50	57	59	48	44
7	68	50	42	57	40
8	50	56	25	42	55
9	50	58	50	58	44
10	54	50	37	73	32
11	61	50	53	56	40
12	55	51	67	56	44
13	54	45	37	43	42
14	68	66	56	73	46
15	67	50	55	60	50
16	58	57	62	69	75
17	60	55	28	50	45
18	48	45	39	76	26
19	58	50	50	50	46
20	58	60	50	50	46
21	54	40	41	36	58
22	48	58	50	51	45
23	60	42	40	53	41

ANNEXURE A: HIGHER DEGREE'S COMMITTEE REPORT.

TITLE: THE PREDICTIVE VALIDITY OF SCORES OBTAINED IN FIRST SEMESTER EXAMINATION ON PERFORMANCE IN INTRODUCTION TO PROGRAMMING.				
Name and Document	**Risk Profile**	**Decision**	**Committee's comments**	**Person/s responsible**
Mutambara D S1040-16	Low risk Reason Data collection from Desktop Special circumstances None.	Approved	The Committee: a) Approved the request for ethical clearance. • Breakdown of budget should be incorporated into the proposals in future.	Prof. D.R. Nzima

ANNEXURE B: ETHICAL CLEARANCE

UNIVERSITY OF ZULULAND
RESEARCH ETHICS COMMITTEE
(Reg No: UZREC 171110-030)

RESEARCH & INNOVATION
Website: http://www.unizulu.ac.za
Private Bag X1001
KwaDlangezwa 3886
Tel: 035 902 6887
Fax: 035 902 6222
Email: MangeleS@unizulu.ac.za

ETHICAL CLEARANCE CERTIFICATE

Certificate Number	UZREC 171110-030 PGM 2016/316						
Project Title	The predictive validity of scores obtained in first in first semester examination on performance in introduction to programming						
Principal Researcher/ Investigator	D Mutambara						
Supervisor and Co-supervisor	Prof PT Sibaya						
Department	Educational Psychology and Special Education						
Nature of Project	Honours/4th Year		Master's	x	Doctoral		Departmental

The University of Zululand's Research Ethics Committee (UZREC) hereby gives ethical approval in respect of the undertakings contained in the above-mentioned project proposal and the documents listed on page 2 of this Certificate.

Special conditions:
(1) This certificate is valid for 2 years from the date of issue.
(2) Principal researcher must provide an annual report to the UZREC in the prescribed format [due date-31 October 2017]
(3) Principal researcher must submit a report at the end of project in respect of ethical compliance.

The Researcher may therefore commence with the research as from the date of this Certificate, using the reference number indicated above, but may not conduct any data collection using research instruments that are yet to be approved.

Please note that the UZREC must be informed immediately of

- Any material change in the conditions or undertakings mentioned in the documents that were presented to the UZREC
- Any material breaches of ethical undertakings or events that impact upon the ethical conduct of the research

D Mutambara - PGM 2016/316 Page 1 of 2

Classification:

Data collection	Animals	Human Health	Children	Vulnerable pp.	Other
					X
Low Risk		Medium Risk		High Risk	
X					

The table below indicates which documents the UZREC considered in granting this Certificate and which documents, if any, still require ethical clearance. (Please note that this is not a closed list and should new instruments be developed, these would require approval.)

Documents	Considered	To be submitted	Not required
Faculty Research Ethics Committee recommendation	X		
Animal Research Ethics Committee recommendation			X
Health Research Ethics Committee recommendation			X
Ethical clearance application form	X		
Project registration proposal	X		
Informed consent from participants			X
Informed consent from parent/guardian			X
Permission for access to sites/information/participants			X
Permission to use documents/copyright clearance			X
Data collection/survey instrument/questionnaire			X
Data collection instrument in appropriate language		Only if necessary	
Other data collection instruments		Only if used	

The UZREC retains the right to

- Withdraw or amend this Certificate if
  - Any unethical principles or practices are revealed or suspected
  - Relevant information has been withheld or misrepresented
  - Regulatory changes of whatsoever nature so require
  - The conditions contained in this Certificate have not been adhered to

- Request access to any information or data at any time during the course or after completion of the project

The UZREC wishes the researcher well in conducting the research

Professor Gideon De Wet
Chairperson: University Research Ethics Committee
Deputy Vice-Chancellor: Research & Innovation
07 November 2016

CHAIRPERSON
UNIVERSITY OF ZULULAND RESEARCH ETHICS COMMITTEE (UZREC)
REG NO: UZREC 171110-30
07 -11- 2016
RESEARCH & INNOVATION OFFICE

D Mutambara - PGM 2016/316 Page 2 of 2

ANNEXURE G: ACCESS LETTER REQUESTING PERMISSION TO CONDUCT RESEARCH

University of Zululand
PO Box X1001

KwaDlngezwa

3886**.**

The Registrar.

University of Zululand

PO Box X1001

KwaDlngezwa

3886**.**

2 May 2017.

Dear Sir/Madam.

REQUEST FOR PERMISSION TO CONDUCT RESEARCH

I am a registered Master's student in the Department of Education at the University of Zululand. My supervisor is Prof D.Nzima.

The proposed topic of my research is: The Predictive Validity Of Scores Obtained In First Semester Examinations On Performance In Introduction To Programming. The objectives of the study are:

1. To establish the predictive validity of selected predictors on performance in Introduction to Programming.
2. To establish an ordinal strength of selected predictors on success in Introduction to Programming.
3. To find out if universities can rely on selected predictors as predictors of success in Introduction to Programming.

I am hereby seeking your consent to access computer science students' academic records from 2010 to 2016. To assist you in reaching a decision, I have attached to this letter:

(a) A copy of an ethical clearance certificate issued by the University

Should you require any further information, please do not hesitate to contact me or my supervisor. Our contact details are as follows:

David Mutambara: cell:0718744944 email: vadmutambara@gmail.com

Prof Nzima D.R. cell:083 6487129 email:NzimaD@unizulu.ac.za

Upon completion of the study, I undertake to provide you with a bound copy of the dissertation.

Your permission to conduct this study will be greatly appreciated.

Yours sincerely,

David Mutambara

ANNEXURE H: DECLARATION BY CANDIDATE

I acknowledge that I have read and understand the University's policies and rules applicable to postgraduate research, and I certify that I have, to the best of my knowledge and belief, complied with their requirements.

I declare that this proposal, save for the supervisory guidance received, is the product of my own work and effort. I have, to the best of my knowledge and belief, acknowledged all sources of information in line with normal academic conventions.

I further certify that the proposed research will be original, and that the material to be submitted for examination has not been submitted, either in whole or in part, for a degree at this or any other university.

I have subjected this document to the University's text – matching and/or similarity – checking procedures and I consider it to be free of any form of plagiarism.

Signature: ___________________________

Date: _______________________________

ANNEXURE H: DECLARATION BY SUPERVISOR(S)

I am satisfied that I have given the candidate the necessary supervision in respect of this proposal and that it meets the University's requirements in respect of postgraduate research proposals.

I have read and approved the final version of this proposal and it is submitted with my consent.

SUPERVISOR

Signature: ____

Print name: ______________________________________

Date: ____

CO–SUPERVISOR

Signature: ____

Print name: ______________________________________

Date: ____

CO–SUPERVISOR

Signature: ____

Print name: ______________________________________

Date: ____