The Impact of Learning Styles on the Acquisition of Computer Programming Proficiency

Sanjay Ranjeeth

Abstract

Computer Programming forms the basis from which most students of information technology 'launch' themselves into further endeavours within the discipline. However, statistical analysis of students' performances in programming related assessment tasks reveals that the mastery of computer programming skills is not easily acquired. This assertion is supported by reports of high failure rates in computer programming courses at several academic institutes. This trend is also confirmed at the University of KwaZulu-Natal (UKZN) where programming related assessments have resulted in failure rates as high as 50%. In order to investigate this dilemma, a phenomenographic approach is used to discover how students experience the phenomenon of computer programming. The investigation is conducted with reference to the deep and surface learning styles framework. Student responses to interview questions on computer programming are classified according to this framework. It was found that at least 50% of the respondents adopted a surface approach towards the learning of computer programming. A point bi-serial correlation was drawn with the students' performance in a computer programming examination. There was a strong correlation between the learning styles adopted by the students and their performance in the computer programming examination.

Keywords: Computer programming, deep and surface learning, phenomenography, point bi-serial, qualitative, stratified random sampling, pedagogy of computer programming, post-modernism, interpretivism

Introduction

The learning of computer programming has been identified as problematic by the academic fraternity. Cognisance of this sentiment is attested to by a growing number of literary inquisitions that attempt to identify factors that may contribute towards obviating this stigmatic attachment. According to Efopoulos *et al*. (2005), there is a growing research impetus in the area of computer science education. This has resulted in the emergence of journals that have either exclusively focused on the teaching of computer programming or have a significant proportion of publications relating to the teaching/learning of computer programming (e.g. Computer Science Education, International Journal of Human-Computer Studies, Association for Computing Machinery (ACM), Journal Storage (JSTOR)). A possible explanation for this elevated interest in the teaching/learning of computer programming is that:

> software construction is a complex, socio-technical, cognitive process that requires a combination of technical, social, analytical and creative abilities (Rose, Heron & Sofat 2005).

Many studies on the learning of computer programming (e.g. Pea & Kurland 1984; Kaczmarczyk *et al*. 2010) allude to the deep misunderstanding of programming related concepts by adult novice programmers. Hence, attempts at resolving the impasse between novice and expert programming will not be an easy task. A consequence of this dilemma is a declining set of standards in elementary programming courses coupled with an increase in the failure rates (Warren 1991). This trend is also confirmed currently at the University of KwaZulu-Natal (UKZN) in the School of Information Systems and Technology (IS&T) where programming related assessments have resulted in failure rates as high as 50%. These sentiments tend to echo an unequivocal belief that the acquisition of competence in computer programming is no trivial achievement.

The Positivist Paradigm and Computer Programming

The enigmatic status regarding the pedagogy of computer programming seems be perpetuated by the failure of the positivist paradigm to embrace the complexities inherent in such studies. The use of empiricism without theoretical quality frameworks to underpin research efforts has resulted in a paradoxical situation where the combined effort is disparate in nature. According to Sheil (1981), the use of empirical methods to underpin research in the domain of computer programming has not been conclusive from the perspectives of reliability and generalisation. Hazzan *et al*. (2006) comment that:

> the ominous tendency of the Hawthorne effect to discredit experimental research has resulted in the spawning of post-modern methodologies that are empirically qualitative.

Traditionally, computer programming is considered to be a ‘scientific activity’. Hence, a natural consequence should be that research in the field would have strong inclinations towards positivism. However, Murnane (1993: 216) goes on to counter this argument by asserting that:

> no studies have shown that students who perform well in the traditional sciences have any particular advantages when it comes to programming …. [T]he development of a computer language may be a scientific process, but the authoring of a program written in that language is not.

Another significant contribution that can be added to the ‘mix’ is made by Strauss and Corbin (1990:17) who claim that:

> qualitative methods are used to better understand phenomenon about which little is already known or to gain in-depth information that may be difficult to convey quantitatively... embodying a research demeanour that is not fully dependent on statistical procedures and other means of quantification.

According to Sanders, Lewis and Thornhill (2003:83), the invocation of qualitative methods would classify a research project as interpretative. The interpretive approach embodies an:

> understanding of the social world from the 'inside', a world which considers the minds of people and their interactions with one another and their environment (Klopper 2008).

The evidence gathered from these literary sources suggests that research into the learning of computer programming should embrace post-modern philosophies such as interpretivism, as a viable option. This embodies a research ethos that accommodates the mixing of diverse ideas and methodologies that are qualitative in essence. This assertion is corroborated by the claim made by Berglund *et al*. (2006) that:

> the qualitative research paradigm enables the drawing of a more solid and significant conclusion about how students learn computing.

Research Question

The current study embraces the ideals of post-modernism and engages the phenomenographic research strategy to gain knowledge of the ways in which learners come to grips with the concepts and principles of computer programming. Phenomenography is defined by Eckerdal, Thune and Berglund (2005) as:

> an empirical, qualitative research approach where the object of interest is how a certain phenomenon is experienced by a certain group of people.

The deep and surface theoretical framework is employed to underpin this study. A precise research question reads as follows:

- What is the impact of learning styles (as embodied by the deep and surface framework for learning) on the acquisition of computer programming knowledge?

The sub-questions are as follows:

- What is the pre-dominant learning style employed by students learning computer programming?
- What is the correlation between learning styles and students' performance in computer programming assessment?

The Deep and Surface Framework

There have been many definitions of the concepts of deep and surface learning (e.g. Booth & Morton 1997: 34; Martin & Saljo 1976; Rhem 1995; Cope & Horan 1998; Hughes & Peiris 2006; Simon *et al*. 2006; Haripersad 2010). A common theme in these definitions is that the surface approach to learning entails memorisation, rote learning and consumption of knowledge from a quantity perspective for the purpose of reproduction at some assessment forum (such as an examination). The deep approach to learning entails intimate and quality driven understanding of content for the purpose of application and extension beyond the factual dimension. Lewandowski *et al*. (2005) cite various studies that are consistent with their assertion that: 'experts form abstractions based on deep (semantic) characteristics rather than on surface (syntactic) characteristics'. A listing of the characteristics of the deep and surface learning style framework gleaned from the sources mentioned above entail the following:

- Surface learning is related to passive processing that lacks reflection, uses low-level meta-cognitive skills and is extrinsically motivated.

- Deep learning is a product of active processing that is intrinsically motivated, reflective, and uses higher-level meta-cognitive strategies.

- Surface learning may result in good memory for facts and definitions, but has a limited ability to understand or use them.

- Deep learning, results in facility of thought derived from linking newly acquired facts and definitions into a conceptual framework of existing knowledge.

- Students who use surface learning may do well on tests that assess learning through knowledge of facts and definitions; they may not understand or be able to apply the memorised and superficially processed information.

- Students who use deep learning are able to understand, apply, and use information learned.

The impact made by this framework on the learning of computer programming cannot be ignored and is corroborated by a study undertaken by Booth (2001). Booth investigated the significance of the style of learning on the acquisition of competence in computer programming. She classified student programmers as either novices or experts. In order to make this classification, she conducted interview sessions with students to ascertain their level of understanding of computer programming. Individual responses were classified as a demonstration of either deep or surface knowledge of programming. The outcome of this exercise was an overall classification labelling a student as either an expert programmer if the majority of responses were classified as deep or a novice programmer if the majority of responses were classified as surface. The findings of Booth's phenomenographic study into the understanding of computer programming from a deep and surface perspective were that students who approached learning to program as learning to code in a programming language or as a pre-requisite for simply passing the course exhibited a surface approach to learning thereby rendering themselves as novice programmers. In contrast, students who focused on understanding the problem domain adequately in order to produce a product that could be used in a professional environment exhibited deeper learning traits thereby rendering themselves as experts. The current study undertook to replicate this study on students in the School of IS&T at UKZN.

Data Collection

It is reported in Hoepfl (1997) that interviews are the primary strategy for data collection when the phenomenographic approach to research is employed. The data collection strategy for the current study is commensurate with the assertion made by Hoepfl and aligned to a similar study conducted by Bruce *et al*. (2006).

The primary target population consisted of students who had completed the ISTN 212 course in computer programming in the School of IS&T. This course was offered at the Durban and Pietermaritzburg campuses of UKZN. The target population consisted of 261 students. Students were informed of the voluntary nature of their participation as well the non-obligatory demeanour of the interview questions. The interview sample consisted of 35 students from the target population who participated in a one hour semi-structured interview. In accordance with the sentiments expressed by Hoepfl (1997) the objective was to seek information-rich cases that can be studied in depth. This is an embodiment of an approach that prioritises the depth that can be ascertained from the data source at the expense of the volume of data.

According to the literature several variables could qualify as potential detractors to any focused study on the learning of computer programming. The significant factors identified were the influence of culture (Bishop-Clark 1995; Rose, Heron & Safat 2005), previous programming experience (Byrne & Lyons 2001; Hagan & Markham 2002; Wilson & Shrock 2002; Allert 2004; Bergin & Reilly 2005; Govender 2009), gender (Bennedsen & Caspersen 2005; Lau & Yuen 2009) and mathematics and problem solving ability (Pillay & Jugoo 2005; Pioro 2006; Bennedssen & Caspersen 2008). In order to accommodate and possibly minimise the influence of these factors, a stratified random sampling strategy was employed. Interviewees were selected in order to obtain adequate representation from the spectrum of variables identified. Twenty students from each campus (Durban and Pietermaritzburg) of UKZN were invited to participate in the interview sessions. There was one student from the Pietermaritzburg campus and four students from the Durban campus who were absent from the interview sessions. However, this did not have a significant impact on the data as there was a liberal presence of respondents from the different variable classifications.

Data Presentation

The interview consisted of 12 questions. The main strategy involved classification of interviewee responses to each interview question as either deep or surface. This judgement was made on the basis of the quality and depth of understanding inherent in the responses to each question. In accordance with the dictates of qualitative research, the researcher's subjective judgement, underpinned by knowledge in the subject area was used to make the distinction between deep and surface responses. However, there were instances where an absolute classification could not be made. In these situations, a value of neutral was assigned to these responses.

In order to explain the logic used in making the classification according to the deep and surface framework, a few examples of verbatim responses extracted from the interview transcripts are presented:
Here is a student's response to the question: How would you inter-change the value of two variables named x and y?

> Create another constant variable X equalling ten, just as a value, to switch it and, then switch A from ten to thirty, then you just add two X, or if you want, well A could be X, so you could say A plus two A, gives you the new A, and B minus A equals the new B, and C minus A equals the new C.

An analysis of this response reveals that the student was able to accomplish the required task for the specific problem presented. However, the lack of generalisation curtailed any prospect of applying the same solution strategy to another set of input values for the same task. The respondent is thus classified as a surface learner in this instance. This classification is in direct contrast to the classification made for the following response for the same task by another student:

> I initialise a variable temporary, then I take the value of C, and I assign the value of B to C. Oh, before that, I will assign C to a temporary variable, and then I will assign B to C, and A to B and I will assign the temporary, that temporary to A.

In the instance above, this response is indicative of expert understanding of the problem domain where the solution offered is applicable in a generalised context. Hence, the student is classified as a deep learner in this instance. Another demonstration of the deep and surface framework is presented in relation to the applicability of an unconditional or 'for … do' loop. The student is given a scenario that required the computation of the average of a finite set of numbers. The interaction between student and interviewer is as follows:

> *Student*: I would use a Do loop (Student).
> *Interviewer*: Why can't a For loop be used?
> *Student*: Because For loops are generally associated with arrays.
> *Interviewer*: Why wouldn't you use a While loop?
> *Student*: You can use the While loop but I'm more comfortable with a For loop.

The student's response is correct in this instance. In a written examination that required implementation of the looping construct, the student would probably score maximum marks. However, the explanation for the choice of looping structure is indicative of a superficial or a surface learning demeanour. The student was not able to offer an adequate generalisation for the implementation of the 'For' looping construct. However, the student has been able to construct a personal cognitive model for looping that has been applied successfully to specific problem situations. This conclusion is established from the comment that '… "For" loops are generally associated with arrays'. The problem here is that this cognitive model is not completely reliable and there are exceptions where it will not work. Hence the response provided by the student in this instance is classified as surface learning.

There were cases where it was not possible to make distinct classifications. This could be attributed to situations that required excessive intervention by the interviewer or cases where the responses given were somewhat confusing. These responses were interspersed with elements of deep understanding, but the impact of these responses was diluted by comments that may have been indicative of superficial understanding. In the following excerpt, the student was required to comment on the applicability of the 'While' loop in preference to a 'Repeat' loop. The following response was given:

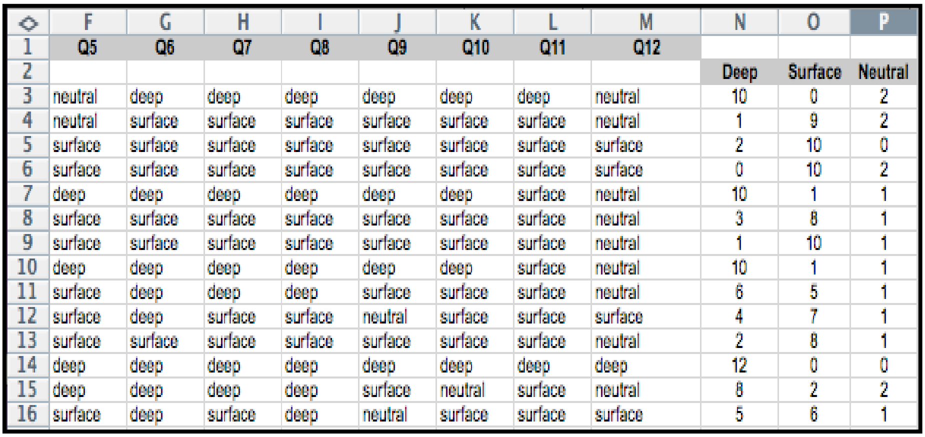
> When they ask the question 'do you want more items?' and then do

everything it is the While Loop; If they ask the question after doing everything it is a Repeat Loop.

In this case the answer is correct and also indicative of meaningful understanding of the difference between both loops. However, the student did not allude to the significance of the fact that the ‘Repeat’ loop was always executed at least once. The inability to make this generalisation begins to impart a measure of doubt to the inclination of classifying this response as indicative of deep learning. In cases such as this, there was no classification made, reflective of the researcher’s neutral stance towards the response.

Data Analysis

After classifying the individual responses as either deep/surface/neutral, there was the additional dilemma of making an overall classification from the tally of individual deep and surface classifications. However, an analysis of the tally of deep and surface scores for each student revealed a discernable difference between the scores for each classification. This observation simplified the task of making an overall classification. A ‘snapshot’ of the deep and surface tallies is presented in Figure 1:



◇	F	G	H	I	J	K	L	M	N	O	P
1	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12			
2									Deep	Surface	Neutral
3	neutral	deep	deep	deep	deep	deep	deep	neutral	10	0	2
4	neutral	surface	surface	surface	surface	surface	surface	neutral	1	9	2
5	surface	surface	surface	surface	surface	surface	surface	surface	2	10	0
6	surface	surface	surface	surface	surface	surface	surface	surface	0	10	2
7	deep	deep	deep	deep	deep	deep	surface	neutral	10	1	1
8	surface	surface	surface	surface	surface	surface	surface	neutral	3	8	1
9	surface	surface	surface	surface	surface	surface	surface	neutral	1	10	1
10	deep	deep	deep	deep	deep	deep	surface	neutral	10	1	1
11	surface	deep	deep	deep	surface	surface	surface	neutral	6	5	1
12	surface	deep	surface	surface	neutral	surface	surface	surface	4	7	1
13	surface	surface	surface	surface	surface	surface	surface	neutral	2	8	1
14	deep	deep	deep	deep	deep	deep	deep	deep	12	0	0
15	deep	deep	deep	deep	surface	neutral	surface	neutral	8	2	2
16	surface	deep	surface	deep	neutral	surface	surface	surface	5	6	1

Figure 1: ‘Snapshot’ of the Deep and Surface Data Tallies

In the instances where this difference was not that discernable or there were a significant number of neutral responses to individual questions, an overall classification of neutral was made. A graphical summation of the classifications is shown in Figure 2.

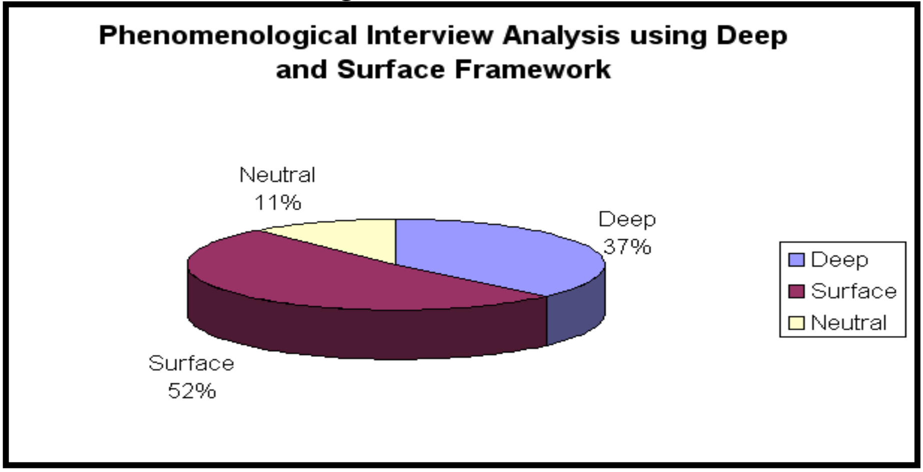


Figure 2: Results of the Interview Analysis using the Deep and Surface Protocol

Responses from 4 of the interviewees were classified as neutral. Hence, this leaves a complement sample of 31 as the focus of some descriptive statistical analysis. Looking at the graphical summary in Figure 2, it is evident that the majority of the students from the sample (52%) exhibited a surface approach to the learning of computer programming. Adopting a comparative stance, it is interesting to note that the proportion of deep learners turned out to be 42% and the proportion of surface learners was 58%. A by-product of this analysis is an inquisition into the feasibility of conducting an extrapolation from the sample onto the population.

An Inferential Dimension

The qualitative nature of this study precludes any adherence to the norms of quantitative methodology. However, in order to expand the potential of this study, an incursion into inferential statistics is undertaken. It should be noted that this may justifiably be viewed with suspicion since it is not the intention of the study to make inferences about the population on the basis of the sample used. In the case in discussion, the sample size is 35. Lind *et al*. (2005: 273) remark that:

> statistical theory has shown that samples of at least 30 are sufficiently large to allow us to assume that the sampling distribution follows the normal distribution.

The underlying qualitative ethos of the data gathering activities meant that with a sample size of approximately 12%, a confidence interval could be constructed for the population proportion. Applying the formula for a population proportion, $p \pm \sqrt{\left(\frac{p(1-p)}{n}\right)} = 0.58 \pm 1.96\sqrt{\frac{0.58(1-0.58)}{261}}$ and using a 95% confidence interval, we obtain the interval range to be between 50% and 62%. This inference is that at least 50% (with a 62% worse-case scenario) of the population will display a surface learning demeanour towards computer programming.

A Correlative Analysis with Formal Assessment

A correlation between the students' learning styles (deep or surface) was drawn with their performance in a computer programming examination. This correlation entailed a comparison between a dichotomous variable (learning style) and a continuous variable (examination mark) thereby necessitating the implementation of point bi-serial analysis. It is suggested in Glass and Hopkins (1996) that the point bi-serial correlation is mathematically equivalent to the Pearson product moment correlation. The result of this correlation is shown in Table 1.

		Exam-Mark	Deep-Surface
Exam-Mark	Pearson Correlation	1	.896 (**)
	Sig. (2-tailed)		.000
	N	34	34
Deep-Surface	Pearson Correlation	.896 (**)	1

	Sig. (2-tailed)	.000	
	N	34	34

** Correlation is significant at the 0.01 level (2-tailed).

Table 1: Point Bi-serial Correlation of Examination Mark and Learning Style

Answers to the Research Questions

The first sub-question of the current study entailed an inquiry to ascertain the pre-dominant learning style adopted by students in their learning of computer programming. The summary of results obtained (shown in Figure 2) indicate that at least 50% of the students adopt a surface style of learning in their efforts to acquire computer programming knowledge.

The second sub-question entailed an inquiry to ascertain a possible correlation between the learning style adopted and students' performance in computer programming assessment. The results of the correlation analysis (shown in Table 1) depict a strong relationship (0.896) between student performance in computer programming assessment and the learning style adopted. Students who adopt a deep approach to the learning of computer programming score significantly higher marks in computer related assessment than students who adopt a surface approach to the learning of computer programming.

The main research question entailed an inquiry to establish the impact of learning styles on the acquisition of computer programming knowledge. The evidence provided by the answers to the 2 sub-questions indicates that students who adopt a surface strategy towards the learning of computer programming will not acquire mastery of computer programming skills. This may be reflected by a poor performance in computer programming assessment. The strong correlation between learning styles and performance in computer programming assessment indicate that the reverse is also true i.e. students who adopt a deep strategy towards the learning of computer programming will acquire mastery in computer programming. This may be reflected by a very good performance in computer programming assessment.

The current study indicates that the learning style adopted by students does have an impact on computer programming performance. However, it should be noted that this assertion does not necessarily constitute a causal relationship. The answers to the research questions were obtained via abductive reasoning, defined by Klopper (2008) as an approach where:

> the researcher accounts for her/his own inter-subjective influence on subjects' responses and how s/he interprets the results.

Conclusion

This study was conducted in the context of poor student performance in computer programming. This study has established that a possible cause for poor performance is a lack of deep understanding of fundamental concepts that underpin the discipline of computer programming. The statistical inference from this study reveals that at least 50% of the population of students who will be enrolled in computer programming courses will adopt a surface approach towards the learning of computer programming. The implication is that these students will adopt an approach that entails the learning of programming code from a syntactic and semantic perspective in order to pass the course. This implication needs to be noted from a pedagogical perspective, necessitating the use of teaching and assessment strategies that encourage the inculcation of deep learning traits thereby minimising the prospect of superficial learning with the explicit purpose of passing a course.

Previous research in the pedagogy of computer programming has focused on the effect of quantitative variables such as gender, mathematical ability, previous programming experience and culture as possible predictors of programming performance. However, none of these studies have been conclusive in providing a definitive explanation for poor performance in computer programming related assessment. A significant, peripheral observation from the current study is that the teaching and learning of computer programming is a social phenomenon and the qualitative approach should be viewed as a viable option for future research efforts in this area. The deep and surface framework used in this study proved to be a reliable predictor of computer programming performance.

Recommendations

An obvious recommendation from this study is that computer programming instruction has to be delivered in an environment that facilitates the adoption of a deep learning approach. However, the presentation of computer programming instruction at many institutes, including IS&T at UKZN is done on a platform that consists largely of complex visual development environments. This seems to be contrary to the requirements of a framework that promotes deep learning of computer programming. The complexity of the environment adds to the cognitive burden of learning the fundamentals of computer programming. In order to obviate this unwarranted complexity, a development environment consisting of an editor and a compiler will be sufficient to re-direct student focus onto obtaining a deep understanding of the principles that underpin computer programming.

References

Allert, J 2004. Learning Style and Factors Contributing to Success in an Introductory Computer Science Course. *Proceedings of IEEE International Conference on Advanced Learning Technologies*, IEEE Computer Society.

Bennedsen, J & M Caspersen 2005. Revealing the Programming Process. *Proceedings of the 36th SIGCSE Technical Symposium on Computer Science Education* 37: 186-190.

Bennedsen, J & M Caspersen 2008. Abstraction Ability as an Indicator of Success for Learning Computing Science. *Proceedings of the Fourth international Workshop on Computing Education Research.*

Bergin, S & R Reilly 2005. Programming: Factors that Influence Success. *Proceedings of the 36th SIGCSE Technical Symposium on Computer Science Education*.

Berglund, A, M Daniels & A Pears 2006. Qualitative Research Projects in Computing Education Research: An Overview. *ACM International Conference Proceeding Series* 165: 25-33.

Bishop-Clarke, C 1995. Cognitive Style, Personality and Computer Programming. *Computers in Human Behaviour* 11,2: 241-260.

Booth, S 2001. Learning to Program as entering the Datalogical Culture: A

Phenomenographic Exploration. *9th European Conference for Research on Learning and Instruction*, Fribourg, Switzerland.

Booth, S & F Morton 1997. *Learning and Awareness.* Lawrence Erlbaum Associates Publishers.

Bruce, S, G Mohay, G Smith, I Stoodly & R Tweedale 2006. *Transforming IT Education: Promoting a Culture of Excellence.* Informing Science Press California.

Byrne, P & G Lyons 2001. The Effect of Student Attributes on Success in Programming. *Sixth Annual Conference on Innovation and Technology in Computer Science Education.*

Cope, C & P Horan 1998. Toward an Understanding of Teaching and Learning about Information Systems. *ACM International Conference Proceeding Series* 3:188 - 197.

Eckerdal, A, M Thune & A Berglund 2005. What Does it Take to Learn Programming Thinking? *Proceedings of the 2005 International Workshop on Computing Education Research*.

Efopoulos, V, V Dagdilelis, G Evangelidis & M Satratzemi 2005. WIPE: A Programming Environment for Novices. *Proceedings of the 10th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education.* New York: ACM Press.

Glass, V & KD Hopkins 1996. *Statistical Methods in Education and Psychology*. Boston: Allyn and Bacon/ Prentice Hall.

Govender, I 2009. The Learning Context: Influence on Learning to Program. *Computers & Education* 53,4.

Hazzan, O, Y Dubinsky, L Eidelman, V Sakhnini & M Teif 2006. Qualitative Research in Computer Science Education. *ACM SIGCSE Bulletin* 38:1.

Hoepfl, MC 1997. Choosing Qualitative Research: A Primer for Technology Education Researchers. *Journal of Information Technology* 9,1: 47-63.

Haripersad, R 2010. Deep and Surface Learning of Elementary Calculus Concepts in a Blended Learning Environment. *Proceedings of the 7th WSEAS International Conference on Engineering Education.*

Hughes, J & J Peiris 2006. Assisting CS1 Students to Learn: Learning Approaches and Object-Oriented Programming. *Proceedings of the 11th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education.*

Kaczmarczyk, L, E Petrick, J East, & G Herman 2010. Identifying Student Misconceptions of Programming. *Proceedings of the 41st ACM Technical Symposium on Computer Science Education.*

Klopper, R 2008. *Principles of Qualitative Research*. Lecture Given at the School of Information Systems & Technology, University of KwaZulu-Natal.

Lewandowski, G, A Gutschow, R McCartney, K Sanders, & D Shinners-Kennedy 2005. What Novice Programmers Don't Know. *Proceedings of the 2005 International Workshop on Computing Education*.

Lau, W & A Yuen 2009. Exploring the Effects of Gender and Learning Styles on Computer Programming Performance: Implications for Programming Pedagogy. *British Journal of Educational Technology* 40,4: 696 - 712.

Lind, D, G Marchal & S Wathan 2005. *Statistical Techniques in Business Economics*. McGraw-Hill, NY.

Marton, F & R Säljö 1976. On Qualitative Differences in Learning: Outcome and Process. *British Journal of Educational Psychology* 46: 4-11.

Murnane, JS 1993. The Psychology of Computer Languages for Introductory Programming Courses. *New Ideas in Psychology* 11,2: 213-228.

Pea, DR & M Kurland 1984. On the Cognitive Effects of Learning Computer Programming. *New Ideas in Psychology* 2,2: 137-168.

Pillay, N & VR Jugoo 2005. An Investigation into Student Characteristics Affecting Novice Programming Performance. *ACM SIGCSE Bulletin Archive* 37,4: 107-110.

Rhem, J 1995. *Deep/Surface Approaches to Learning*: *An Introduction*. The National Teaching and Learning Forum. Accessed on 2nd March 2009 http://www.ntlf.com/html/pi/9512/article1.htm

Rose, E, J le Heron, & I Sofat 2005. Student Understanding of Information Systems Design, Learning and Teaching: A Phenomenographic Approach. *Journal of Information Systems Education* 16,1: 183-195.

Sheil, BA 1981. The Psychological Study of Programming. *ACM Computing Surveys (CSUR) Archive* 13,1: 101 - 120.

Sanders, M, P Lewis & A Thornhill 2003. *Research Methods in Business*. Pearson Education Limited, New Jersey

Simon, S, S Fincher, A Robins, B Baker, I Box, Q Cutts, M de Raadt, P Haden, J Hamer, M Hamilton, R Lister, M Petre, K Sutton, D Tolhurst &

J Tutty 2006. Predictors of Success in a First Programming Course. *Proceedings of the 8th Australian Conference on Computing Education* 52: 189 - 196.

Strauss, A & J Corbin 1990. *Basics of Qualitative Research: Grounded Theory Procedures and Techniques*. Newbury Park, CA: Sage Publications Inc.

Warren, P 2001. Teaching Programming Languages Using Scripting Languages. *Journal of Computing Sciences in Colleges* 17,2: 205-216.

Wilson, BC & S Shrock 2002. Contributing to Success in an Introductory Computer Science Course: A Study of Twelve Factors. *Proceedings of the 32th SIGCSE Technical Symposium on Computer Science Education*. ACM Press, NY.

S Ranjeeth
School of Information Systems & Technology
University of KwaZulu-Natal
South Africa
ranjeeths@ukzn.ac.za