Problem Solving as a Predictor of Programming Performance

Glenda Barlow-Jones[1(✉)] and Duan van der Westhuizen[2]

[1] Department of Applied Information Systems, University of Johannesburg, Auckland Park, South Africa
glendab@uj.ac.za

[2] Department of Science and Technology Education, University of Johannesburg, Auckland Park, South Africa
duanvdw@uj.ac.za

Abstract. The purpose of this paper is to establish what correlation exists between students' problem solving ability and their academic performance in 1st-year programming courses. The students' achievement in the programming courses is specified as the dependent variable and four programming aptitude tests for logical reasoning, non-verbal reasoning, numerical reasoning and verbal logic are specified as the independent variables. The study group consists of 379 students. Our findings show a correlation between students' logical reasoning, numerical reasoning and verbal logic and performance in computer programming modules. The correlation between students' non-verbal reasoning and performance in computer programming modules was, however, not significant.

Keywords: Computer programming · Problem solving · Logical reasoning · Numerical reasoning · Verbal logic · Non-verbal reasoning

1 Theoretical Background

In this paper, we firstly explore some theoretical constructs around the notion of 'problem-solving'. We particularly invoke Bloom's revised taxonomy and literature around critical thinking. We then explore the difficulties that students typically experience in learning how to problem solve. Finally, we report on a project during which the correlation between the dimensions of problem-solving and student performance was calculated. It is difficult to determine what knowledge and skills first year programming students possess prior to their programming course. The main objective of computer programming is to implement programs that solve computational problems. In [3,10] we can find that problem solving ability is an indicator of programming performance. Critical thinking, also referred to as problem solving, reasoning or higher order thinking skills, can be defined as *"disciplined, self-directed thinking which exemplifies the perfections of thinking appropriate to a particular mode or domain of thought"* [18] and also as *"a process of gathering and evaluating data to make decisions and*

J. Liebenberg and S. Gruner (Eds.): SACLA 2017, CCIS 730, pp. 209–216, 2017.
https://doi.org/10.1007/978-3-319-69670-6_14

solve problems" [15]. Taxonomies of learning have been implemented worldwide to describe learning outcomes and assessment standards reflecting what learning stage a student is at. The original learning taxonomy developed by Bloom and several of his colleagues in 1956 identifies six levels of thought [2]:

- *Knowledge:* rote memorization, recognition, or recall of facts;
- *Comprehension:* understanding what the facts mean;
- *Application:* correct use of the facts, rules, or ideas;
- *Analysis:* breaking down information into component parts;
- *Synthesis:* combination of facts, ideas, or information to make a new whole;
- *Evaluation:* judging or forming an opinion about the information or situation.

These levels of thought start from the lowest order process to the highest order process with higher levels building on lower levels [2]. Once a student reaches the highest level they can be said to have grasped a subject matter. Bloom's Taxonomy was revised in 2011 to address the differences between comprehension and application and to better define the term evaluation. The changes made to the revised taxonomy are as follows [1]:

- *Remember* (previously 'knowledge');
- *Understand* (previously 'comprehension');
- *Apply* (previously 'application');
- *Analyse* (previously 'analysis');
- *Evaluate* (previously 'evaluation');
- *Create* (previously 'synthesis').

According to [9,12], students learn to write complete computer programs in their first year of a programming module which falls within the top two levels of Bloom's Revised Taxonomy of teaching and learning [1]. These two levels however, depend on the first four levels before a student is said to be able to grasp computer programming. For example, in computer programming, 'learning syntax' is the lowest order process [11] and efficiently utilising syntax in order to 'produce effective computer programs' is the highest order process [4]. Accordingly, lecturers expect students to be able to write programs within the first few weeks of their programming module [6]. These programs may be basic and get more difficult as the module progresses, however, many students may be left behind whilst still struggling to find solutions to basic problems. This means that the difficulty level at which the programming module starts, is already at too high a level for a novice programmer, which can lead to a lack of motivation and ultimately a student failing the module. Although novice students may have little experience programming, they do have experience solving problems in everyday life [19]. Problem solving is a mental process of analysing a given problem, developing a solution to the problem and presenting the solution [13]. When students solve problems either independently or in collaboration with other students they are learning by doing. While learning by doing is synonymous with problem solving [13] computer programming as a discipline is also synonymous with problem solving. However, according to [7], students struggle to solve problems for the following reasons:

- Students do not fully understand the problem either because they have not interpreted the problem statement correctly or they just want to start writing code.
- Students fail to transfer the knowledge that they have already acquired from past problems over to new problems.
- Students who take too long to find a solution just give up trying and wait for the solution to be given to them.
- Many students do not have enough mathematical and logical knowledge.
- Students lack specific programming expertise and struggle to detect simple syntactical and logical programming errors.

According to [3], more attention should be paid to novice programming students' problem-solving abilities by encouraging them to practice problem solving, as learning to solve problems algorithmically contributes to learning to program. Students need to think about the processes they go through in solving everyday life problems and look at how to use the same processes to develop algorithms, for example: *"they need to identify things that are familiar to them, divide the problem into smaller problems and use existing solutions"* [5] — the very same things that [7] identifies as what students struggle with. The purpose of this paper is to establish what correlational relationship exists between students' problem solving ability and their academic performance in first-year level programming courses.

2 Method

The participants of the study were a group of 186 first year students enrolled for the National Diploma in Business Information Technology (NDBIT) at the University of Johannesburg (UJ), and 193 first year students enrolled for the National Diploma in Information Technology (NDIT) at the Tshwane University of Technology (TUT).[1] The research process was preceded by a thorough literature review.

2.1 Instrumentation

Four programming aptitude tests (which measured a student's ability to problem solve) were used with permission from the University of Kent Careers and Employability Service Department. All tests were completed in a test-like setting with hired venues and appointed invigilators.[2] Ten items from each test were used to reduce the load on students and due to time constraints.

[1] Explanation for readers from outside South Africa: the 'National Diploma' in South Africa consists of a vocational 2-year curriculum below the level of a Bachelor of Science degree.

[2] Students were required to complete consent forms stipulating what was expected from them during the research process. Permission to conduct the research was sought from the Ethics Committee at the Faculty of Education at the UJ and from the Research Ethics Committee at the TUT. In all cases, data were collected responsibly and recorded as accurately as possible.

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Question:
What is the missing letter in this series: c c d **?** e f g g h

Answer: e

Fig. 1. Logical reasoning test questions

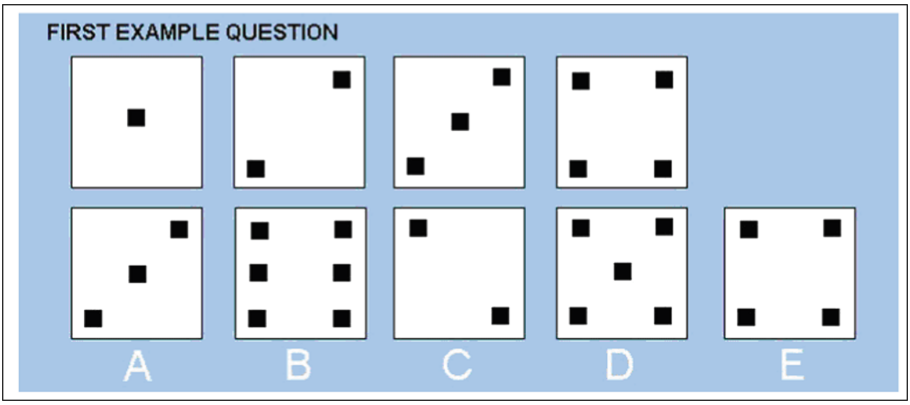

Fig. 2. Non-verbal reasoning test questions: In the first example question the top row of four boxes make up a series from left to right. You have to decide which of the 5 boxes underneath, marked A to E, will be the next in the sequence. For example in the first example, the top four boxes have 1, 2, 3, and 4 dots respectively. Obviously, the next box in the sequence will have 5 dots, which is box D. Answer 1 = D

Logical Reasoning Test. The first programming aptitude test was the logical reasoning test which involved letter sequences and tested the students' ability to think logically and analytically. The test involved looking at a specific sequence of letters and working out the next letter of the sequence: see the example in Fig. 1. The average score for the logical reasoning test was 6.7/10.

Non-verbal Reasoning Test. The second programming aptitude test was the non-verbal reasoning test which determined a student's ability to understand and analyse visual information and solve problems using visual reasoning — for example: identifying relationships, similarities and differences between shapes and patterns, recognizing visual sequences and relationships between objects, and remembering these. The non-verbal reasoning test enabled students to analyse and solve complex problems without relying upon or being limited by language skills. The test involved looking at a specific sequence and working out the next member of the sequence from the pictures given: see the example in Fig. 2. The average score for the non-verbal reasoning test was 4.65/10.

Question:
A taxi driver works 46 weeks of the year and gets an average of 70 customers per week averaging 4 kilometers each at 90 cents per kilometer. His expenditure is as follows:

Car service/repair/insurance:	R1,250,00 per year
Petrol costs:	R0.06 per kilometer
Mortgage costs:	R250,00 per month
Other expenditure — food, electricity, etc.:	R125 per week

What is the total income in Rands of the taxi driver for the whole year?

Answer:
Average fare = 4 × 90c = R3.60
Income per week = 70 fares at R3.60 each = 70 × 3.60 = R252
Income for 46 weeks work = R252 × 46 = R11 592

Fig. 3. Numerical reasoning test questions

Question:
Simon, Cheryl and Dannii are going by train to Pretoria to watch a singing competition.
Cheryl gets the 2.15 pm train.
Simon's train journey takes 50% longer than Dannii's.
Simon catches the 3.00 train.
Dannii leaves 20 minutes after Cheryl and arrives at 3.25 pm.

When will Simon arrive?

Answer:
Dannii leaves at 2.35, arrives 3.25, therefore 50m journey.
Simon's journey takes 75m, therefore arrives at 4.15

Fig. 4. Verbal logic test questions

Numerical Reasoning Test. The third programming aptitude test was the numerical reasoning test which included mathematical questions: see the example in Fig. 3. The average score for the numerical reasoning test was 3.24/10.

Verbal Logic Test. The fourth programming aptitude test was the verbal logic test which included verbal logic puzzles, some of which had a numerical element. This test, tested the students' ability to think logically, analytically and numerically, and also to extract meaning from complex information: see the example in Fig. 4. The average score for the verbal logic test was 2.79/10.

Programming Examination Results. Data were also collected from the examination results of the students programming module, Development Software 1: UJ — Development Software 1A (DSW01A1) and Development Software 1B (DSW01B1) and; TUT — Development Software 1A (DS0171AT) and Development Software 1B (DS0171BT). Student numbers were used as the key field to link the data sets. The Development Software 1 (DS1) results were used as the dependent variable throughout the study.

3 Data Analysis

The four programming aptitude tests for logical reasoning, non-verbal reasoning, numerical reasoning and verbal logic were correlated with the DS1 final mark of the students. The results of the computation are presented in Table 1.

Table 1. Correlation of programming aptitude tests and DS1 mark

Test type	Correlations	DS1 mark
Logical reasoning test mark	Pearson correlation	.199
	Sig. (2-tailed)	.000
	N	341
Non-verbal reasoning test mark	Pearson correlation	.095
	Sig. (2-tailed)	.078
	N	347
Numerical reasoning test mark	Pearson correlation	.257
	Sig. (2-tailed)	.000
	N	348
Verbal logic test mark	Pearson correlation	.143
	Sig. (2-tailed)	.008
	N	341

3.1 Logical Reasoning

A Pearson product-moment correlation coefficient was computed to assess the relationship between the logical reasoning test mark variable and the students' performance in DS1 variable. Logical reasoning refers to a student's ability to think logically and analytically. There was a small, positive correlation between the two variables, $r = .199$, $n = 341$, $p = .000$. Overall, there was a small, positive correlation between the non-verbal reasoning test mark and performance in DS1.

3.2 Non-verbal Reasoning Test Mark

A Pearson product-moment correlation coefficient was computed to assess the relationship between the non-verbal reasoning test mark variable and the students' performance in DS1 variable. Non-verbal reasoning refers to a student's ability to understand and analyse visual information and solve problems using visual reasoning. There was no correlation between the two variables, $r = .095$, $n = 347$, $p = .078$. The results for this group show an insignificant correlation between students' non-verbal reasoning ability and performance in DS1.

3.3 Numerical Reasoning Test Mark

A Pearson product-moment correlation coefficient was computed to assess the relationship between the numerical reasoning test mark variable and the students' performance in DS1 variable. The numerical reasoning test included mathematical questions. There was a small, positive correlation between the two variables, $r = .257$, $n = 348$, $p = .000$. Overall, there was a small, positive correlation between the non-verbal reasoning test mark and performance in DS1.

3.4 Verbal Logic Test Mark

A Pearson product-moment correlation coefficient was computed to assess the relationship between the verbal logic test mark variable and the students' performance in DS1 variable. The verbal logic test included logical, analytical and numerical questions. There was a small, positive correlation between the two variables, $r = .143$, $n = 341$, $p = .008$. Overall, there was a small, positive correlation between the verbal logic test mark and performance in DS1.

4 Conclusion

The University of Kent's Careers and Employability Service Department assesses student's computer programming aptitude with tests measuring competencies such as numerical reasoning, logical reasoning, verbal reasoning and non-verbal reasoning which are required in computer programming jobs. These tests were adapted for this study. The findings show that there is a correlation between a student's logical reasoning ($r = .199$, $p = .000$), numerical reasoning ($r = .257$, $p = .000$) and verbal logic ($r = .143$, $p = .008$) and performance in computer programming modules. This supports findings of earlier studies [8,14,16] telling us that problem solving ability is a major predictor of performance in programming courses. The correlation between students' non-verbal reasoning and performance in computer programming modules was, however, not significant. This could be because the ability to use pictures in thinking is to a large degree a matter of practice, not aptitude [17]. A similar finding was reported in [14]; in this study Computer Science 1 (CS1) students were identified as having experienced difficulties with: decomposing problems, developing sufficient solutions, and re-using previously seen solutions (even for elementary problems). To this end they introduced the course Development of Algorithmic Problem-Solving Skills (DAPSS) to be taken in parallel to studying CS1. The main focus of DAPSS was to set aside the details of the programming language and concentrate on reflective processes, awareness to problem-solving behaviour and development of cognitive skills. Results showed that the DAPSS course had a positive effect on students' problem-solving skills which in turn improved their programming skills [14]. It is thus recommended that the teaching of problem-solving skills at the University of Johannesburg (UJ) and the Tshwane University of Technology (TUT) be introduced as part of the programming module, to provide opportunities to enhance students' programming performance and thinking processes.

References

1. Anderson, L.W., Krathwohl, D.R., Bloom, B.S.: A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives. Allyn & Bacon, Boston (2001)
2. Bloom, B.S., Engelhart, M.D., Furst, E.J., Hill, W.H., Krathwohl, D.R.: Taxonomy of Educational Objectives, Handbook. 1: Cognitive Domain. Longman, London (1956)
3. Chao, P.: Exploring students' computational practice, design and performance of problem-solving through a visual programming environment. Comput. Educ. **95**, 202–215 (2016)
4. Cooper, S., Dann, W., Pausch, R.: Alice: a 3-D tool for introductory programming concepts. Comput. Sci. Coll. **15**(5), 107–116 (2000)
5. Dale, N., McMillan, M., Weems, C., Headington, M.: Programming and Problem Solving with Visual Basic.NET. Jones & Bartlett Learning, Burlington (2003)
6. Gomes, A., Mendes, A.J.: Bloom's taxonomy based approach to learn basic programming. In: World Conference on Educational Multimedia, Hypermedia and Telecommunications, Association for the Advancement of Computing in Education (2009)
7. Gomes, A., Mendes, A.J.: Learning to program – difficulties and solutions. In: Proceedings International Conference on Engineering Education (2007)
8. Kimmel, S.J., Kimmel, H.S., Deek, F.P.: The common skills of problem solving: from program development to engineering design. Int. J. Eng. Ed. **19**(6), 810–817 (2003)
9. Lister, R.: On blooming first year programming, and its blooming assessment. In: Proceedings Australasian Conference on Computing Education (2000)
10. Liu, C., Cheng, Y., Huang, C.: The effect of simulation games on the learning of computational problem solving. Comput. Educ. **57**(3), 1907–1918 (2011)
11. Mayer, R.E.: Teaching and Learning Computer Programming: Multiple Research Perspectives. Routledge, Abingdon (2013)
12. Meerbaum-Salant, O., Armoni, M., Ben-Ari, M.: Learning computer science concepts with scratch. Comput. Sci. Educ. **23**(3), 239–264 (2013)
13. Muller, O.: Pattern oriented instruction and the enhancement of analogical reasoning. In: 1st Proceedings of the International Computing Education Research Workshop, pp. 57–67 (2005)
14. Muller, O., Haberman, B.: A Course Dedicated to Developing Algorithmic Problem-Solving Skills – Design and Experiment. PPIG, Limerick (2009)
15. Paul, R.W.: Critical thinking: what every person needs to survive in a rapidly changing world. Technical report. Center for Critical Thinking and Moral Critique, Sonama State University (1990)
16. Reed, D., Miller, C., Braught, G.: Empirical investigation throughout the CS curriculum. In: Proceedings 31st SIGCSE Technical Symposium on Computer Science Education, pp. 202–216 (2000)
17. Reynolds, C.: Intelligence Testing (2009)
18. Smith, C.J.: Processing thoughts: critical thinking. In: Ethical Behaviour in the E-Classroom: What the Online Student needs to know, pp. 31–43. Elsevier (2012)
19. University of Kent: Programming Aptitude Tests (2013)