PREDICTING PROGRAMMER QUOTIENT

Meenu Raveendran[1], Dr. V. G. Renumol[2]

Division of IT, School of Engineering, CUSAT, Kochi, Kerala
[1]meenuraveendran@gmail.com
[2]renumolvg@gmail.com

Abstract— Computing education necessitates programming skill. But it varies in students. How can we quantify or measure the skill? We are yet to have a standardized measurement system for the programming ability. The concept of Programmer Quotient (PQ), which gives a measure of innate programming ability, attributes a value to one's ability to program, just like Intelligence Quotient (IQ). This would remain the same independent of the programming experience. In this paper, we consider few inherent skills such as Analytical ability, ability to synthesize etc. and try to correlate these skills to one's programming ability. A questionnaire was designed and used to measure the skill in these areas. Then a model was designed from the data collected. It can predict the programming skills of a student from his/her inherent skills, irrespective of the programming language.

Keywords—Programming ability, Programmer Quotient PQ, Programming skill, Analytical ability.

Meenu Raveendran[1]

Division of IT, School of Engineering, CUSAT, Kochi, Kerala
[1]meenuraveendran@gmail.com

1. INTRODUCTION

THE primary purpose of computing education is to impart to students a set of skills required to become a good programmer. But both teaching and learning of programming is considered to be really challenging by many [1].To learn programming, it requires correct understanding of the abstract concepts. Nature of the subject makes it difficult for many to learn it. For teaching, the size and heterogeneous nature of the student groups makes it difficult to design instruction, in such a way that it would be beneficial for everyone.[3]

Students find it hard to acquire the logic of computer programming. Many research works at the end of Introductory Programming Course (IPC) have shown that the rate of failure in these programming courses is relatively high [4] [5] [7].The high dropout and failure rates in IPCs are a universal problem that motivated many researchers to propose methodologies and tools to help students.

Another perception of the difficulty faced by students in programming is that not all students have the ability or the aptitude to program well [6]. This group argues that if we can measure this aptitude before admitting the students to a programming course, the failure rate and the dropout rates can be significantly reduced.Programming-related skills refer to the ability of a person in operating a computer to perform a task or multiple tasks. Handling a computer task with certain level of complexity often requires computer programming skills.[2]

This paper is based on the hypothesis that the programming ability is correlated with other measurable inherent skills or aptitudes of a person. But a test for measuring programming aptitude and identifying students who have a natural aptitude for programming has not yet been standardized.

Mcgettricket. al, introduced a concept of Programmer Quotient (PQ) to measure the programming ability [1]. The proposal was to develop the concept of PQ that would remain the same irrespective of the programming experience.

2. RELATED WORK

Research works in this field of computing education can be broadly classified into two. In the first type, the researchers have attempted to improve the methods of teaching programming. Several methods and tools have been

developed to assist the students to grab the concepts of programming. And the second category focuses on identifying what differentiates successful programmers from non-successful programmers. Some of the notable works are discussed below.

A.Improvement in Methodology

A great number of works deals with how the teaching methods or the delivery of the contents can be modified or improved so that the students are able to grasp the underlying concepts of programming. Simplified programming languages, environments and simulating software have been proposed to support teaching and self-learning different programming techniques.

A prominent work in early days suggests that visual way of learning programs can effectively improve the understanding.[7] In their paper, B. Kaučič et al.[16] has discussed a method to improve the teaching methodology for programming so that the success rate can be improved. They suggest that, a new generation of programming tools and activities can help to overcome the shortcomings of previous initiatives, and thus make computer programming more accessible to everyone and suitable for youngsters.

In their paper M.McCracken et al[5] conducted a multinational study of the assessment of programming skills in first year CS students. They found that the problems observed with programming skills seem to be independent of country and educational system and that the most difficult part for students seemed to be abstracting the problem to be solved from the exercise description

Janna Holovikiet. al found in their work that students should have certain basic abilities like mental schemas in regard to procedural thinking and action, functional understanding of processes, and a comfortable relationship to technological artifacts when they start learning programming[9].

B. Factors Affecting Programming Ability

In Predictors of Success in a First Programming Course, [11] Simon et al, studied the relationship of programming ability with that of one's spatial visualization, behavioral aspects and Attitude. It has been found that, the behavioral aspects like searching a phone book has clear association with one's ability to program well. The learning approach also has a positive correlation on the effectiveness.

In two related papers Dehnadi, Richard et al. reported that the students can be classified based on the mental models they used to solve the questions: i) Consistent and ii) Inconsistent [6] [10]. Consistent students use the same model to solve all questions while the inconsistent group used different models for different questions. They found that consistent students tend to perform slightly better than the inconsistent students and they suggested that students can be divided into programming sheep and non-programming goats [6]. But in paper[10] they confirmed that this test doesn't work if the intake is already experienced.The paper [6] claimed that with the test used one can divide the students into 2 categories: the programming-sheep and the non-programming goats.

But recently, the above work was retracted by the co-author Richard Bornet [12]. While he accepts it to be true that novices who answer 'consistently' in the test are more likely to pass a programming course, he retract his claims that Dehnadi discovered a correct way to measure the programming skills and students can be divided into programming sheep and non-programming goats.

Kurland et al. [7] in a pioneer work on the development of programming skills in high school students and found that Math and reasoning abilities have direct correlation with the programming skills.Simon et al,[11] studied the relationship of programming ability with that of one's spatial visualization, behavioral aspects and attitude. They found that there are several factors positively correlate with one's programming ability. And they are logical thinking, problem solving attention to detail, consideration of alternatives, mathematics, knowledge of programming, ability to learn, knowledge of computers, modularizing, and planning.

Jens Bennedsen et.al[13] claimsthat general abstraction ability has a positive impact on performance in computing science, but they did not find a correlation between abstraction ability and overall performance in learning computing science. In their work [14], Renumol et.al, has identified a set of 8 cognitive processes through which a student may go through while programming. In a similar work,[15] Ambrosio, et.al, has pointed that abstraction and command sequencing are the 2 cognitive abilities which are identified as key to explain the difficulties the students encounter when learning to program. Underlying the abstraction difficulties are the student's problem solving abilities. Problem solving is composed of Analysis and Synthesis.

Some of the existing papers on methods to test the programming skills are described by David Clark [19] and SaeedDehnadi [17]. They have tried the effectiveness of testing the programming skills using MCQs from the programming knowledge. David Clark in his work[19], assessed students' programming skills with MCQs from 4 levels of cognitive skills namely, Knowledge, Comprehension, Application, and Problem-solving. It was found that comprehension, application and analysis questions were good discriminators for the students' skill, but Knowledge questions were not significant discriminators. In [17], Dehnadi tried to give an objective interpretation of the results from previous experiments using mental models for predicting success in programming course.

In their paper, 'Programming aptitude testing as a prediction of learning to program' [18], MarkkuTukiainen et.al.devised a Programming Aptitude Test (PAT) with three parts: Number series, figure analogies, and arithmetic problems. Their test was considered to predict the floor limit of performance in the first programming course for those who had prior knowledge of programming concepts. In his thesis work, Otto Seppala researched about the Advances in assessment of programming skills[20], how to improve existing assessment techniques and tools so that we can more accurately assess students' programming skills and give them

better feedback. He used visual feedback, reading and tracing of the code to improve the programming skills of students.

As these works suggested there may be few factors which distinguish the successful programmers from others. The paper is a preliminary effort to identify some of these factors which contribute to PQ. The objective of this work is to create a model to predict the programming ability of a student by assessing his/her cognitive skills. The resultant model is expected to predict the programming ability with a certain level of accuracy

3. OVERVIEW OF THE SYSTEM

The programming ability of a person might depend on numerous factors ranging from one's gender or the type of education one has had. In this project we concentrate on various cognitive and aptitude skills of the students and their effect in their PQ such as analytical ability, ability to synthesize, inductive and deductive reasoning, translation, decision making and problem solving skills. The effects of gender, family background, emotional condition or any other similar aspects on one's PQ are not considered.

In this project we have identified few cognitive and aptitude abilities for our experiment. Some of these identified areas are part of Bloom's Taxonomy of learning objectives. Some are on the basis of our experience, assumptions and also some exploratory options. We need to verify if a correlation exists between ability in these areas and programming ability

A. Phase I: Designing the Questionnaire

For the modeling of the predictor, we have identified the following set of factors.

1. Analytical ability – Ability to examine and break information into parts by identifying motives or causes; making inferences and finding evidence to support generalizations.[21]
2. Synthesis ability – Ability to combine facts, ideas, or information to make a new whole[21]
3. Induction – Ability to move from specific observations to broader generalizations "(bottom up" approach)[22]
4. Deduction – Ability to move from the more general to the more specific ("top-down" approach.)[22]
5. Translation – Ability to understand, express and re-express the 'same' or 'equivalent' meaning in another form[23]
6. Decision making – Ability to select a logical choice from the available options[24]
7. Problem solving – Ability to work through details of a problem to reach a solution [25]

Once the independent factors were identified for the model, the questions for assessing each of these abilities were identified. Datasets of each of these areas consisting of 50 questions were created. The questions were identified from text books written for the purpose of measuring these skills and also from the web.

With enough number of questions in place, the required questionnaire was designed. The questionnaires consists of five questions each from Analytical, Synthesis, Induction, Deduction, Translation, Decision Making and one question from Problem Solving. This is because; the question in the problem solving section is descriptive type while others are multiple choice questions.Problem solving questions involves giving a situation to a student and asking him/her to write down the steps that would be taken to tackle the situation effectively. The answer given by a student is evaluated using the guidelines from the method shown byNational Center for Research on Evaluation, Standards, and Student Testing (CRESST)[26]

The questionnaire is then distributed among different sets of students – the known sample. These groups of students formed the training groups, about whose programming ability is already known.Thus their programming ability can be quantified by a value, which we call the programmer quotient. The questionnaire was distributed among several training groups belonging to undergraduate engineering programs in Information Technology and Electronics & Communication Departments.

B. Designing theModel

The questionnaires answered by the students with different PQs are evaluated and the results are analyzed. Then we tried to correlate or map the aptitude abilities to the programming ability. Analysis of the questionnaire evaluation gives us a general trend of the factors that might contribute to the programming ability of students.

By analyzing the data from the questionnaires, a model for Programmer quotientis built using statistical data analysis tools. We assume PQ to be a variable dependent on several independent variables. Thus the subjective assessment is translated to an objective assessment, which gives a quantitative value for the programmer quotient. The model thus predicts the PQ of a student, based on the assessment of the questionnaire answers.

This model is then tested on other student groups. The test groupswere given the questionnaire and their answers were evaluated. The data is then put to the model and the model predicts their PQ.

4. RESULTS

Different sets of questionnaires were created and distributed among known sample of students and the answers were analyzed. Multiple regression analysis was done on the data collected to develop the model.

A. Method

Data was collected from a known sample of nearly 100 students. The students were from undergraduate programmes of Information Technology and Electronics & Communication in School of Engineering, Cochin University of Science & Technology, India.

Questionnaires were distributed among the students and theirprogramming ability was known from their practical work, viva voce and usage of algorithms/logics in solving

various problems assigned in their practical sessions. Inputs regarding their programming ability were also taken from the teaching and lab faculty, to arrive at the final Programmer Quotient assigned for the students.

The data was then stored in MS Excel. Then using the multiple regression analysis technique, a model was generated. For each sample of students, the model was generated and also, the model was generated for each of the variable.

The finally generated model of the form $Y = b_0 + b_1 X_1 + b_2 X_2 + \ldots,\ldots\ldots + b_k X_k$ is given by:

$$\mathbf{PQ = 1.08 + 0.64 * X_1 - 0.42 * X_2 + 0.02 * X_3 + 0.76 * X_4 + 0.04 * X_5 + 0.29 * X_6 + 0.17 * X^7}$$

Where X1 – Analytical Ability
X2 – Synthesis Ability
X3 – Inductive Ability
X4 – Deductive Ability
X5 – Translation Ability
X6 – Decision Making
X7 –Problem Solving

From the model generated, it is clear that Analysis ability and Deduction ability may be the crucial factors in determining the Programmer Quotient and also, Inductive ability and Translation ability may not be a good differentiator.

The further details of the regression analysis are given below in the table. The value of R^2 determines the predictive ability of the model. It tells us how much of the variance of "Y" we have explained in the regression. The closer R^2 is to 1, the better the model and the prediction are. Adjusted R2 is the percentage measurement of goodness of fit. The higher this value, the more accurate the equation is.

Regression Statistics	
Multiple R	0.865233
R Square	0.748628
Adjusted R Square	0.730673
Standard Error	1.015735
Observations	106

Here, the value of R^2 is 0.74, which indicates that the model has a reasonably good predictive capability.The standard error is an estimate of the standard deviation of the coefficient, the amount it varies across cases. It can be thought of as a measure of the precision with which the regression coefficient is measured. Here the SE of 1 makes the equation acceptable.

Another value is the P-value, which is the significance factor. These values provide the likelihood that they are real results and did not occur by chance. The lower the P-Value, the higher the likelihood that the coefficient is valid. For example, a P-Value of 0.016 for a regression coefficient indicates that there is only a 1.6% chance that the result occurred only as a result of chance.

Parameter	*P-value*
Analysis	1.60066E-08
Synthesis	0.002234168
Induction	0.809137687
Deduction	7.70271E-12
Translation	0.645938007
Decision Making	0.001225338
Problem Solving	0.066975398

Here, from the p-values for various factors, we can assume that the significance of Induction and Translation is very less in predicting the PQ.

With satisfying values of R^2 and p, the model thus was tested among a sample of 10 students. The PQ predicted by the model was found to be a good prediction of their programming ability.

5. CONCLUSION AND FUTURE WORK

The experiments conducted with a sample of about 100 students over 7 factors showed promising results. Analytical ability and deduction ability could very well be concluded as two main factors which may contribute to one's programming ability. And we could also assume that some of these factors like Induction and Translation may not be clear distinguishing factors affecting the programming ability of the student.

The model has many real-world applications. The model can be used at the admission test level to screen the students admitted to the CS & IT courses. This will ensure that only students with the programming capability join the course and thus will reduce the failure rate. It will also help in improving the morale of the students. At industry level, the model can be used to identify employees with great programming skills. They can thus be allotted to projects requiring good deal of programming skill and vice versa.

The evaluation and analysis of the method developed suggest interesting directions for future work. Our model met almost all the objectives that we aimed, to a particular extend. Next step is to concentrate on how to improve the accuracy as a future study.The work could be enhanced by considering more factors, which are considered to be affecting the programming ability. This could include various other factors like Abstraction ability, Critical thinking etc. Including some other factors like gender could also provide an interesting insight to the future work.

ACKNOWLEDGMENT

I would like to express my sincere gratitude to my guide for giving me valuable ideas and guidance.I also thank the various teaching and laboratory faculty in SOE, CUSATfor their help inconducting the experiments.

REFERENCES

[1] Andrew Mcgettrick, Roger Boyle, Roland Ibbet, John Lloyd, Gillian Lovegrove, And Keith Mander;Grand challenges in computing: Education – A summary. Computer Journal Vol. 48 No.1, 2005, 42–48.

[2] Sheeson E. Chang; Computer anxiety and perception of task complexity in learning programming-related skills; Computers in Human Behavior; Volume 21, Issue 5 September 2005, Pages 713–728, Elsevier

[3] EssiLahtinen, KirstiAla-Mutka, Hannu-MattiJärvinen; A Study of the Difficulties of Novice Programmers; ITiCSE '05 Proceedings of the 10th annual SIGCSE conference on Innovation and technology in computer science education, Pages 14-18; ACM New York, NY, USA ©2005;

[4] Raymond Lister, Elizabeth S. Adams, Sue Fitzgerald, William Fone, John Hamer, Morten Lindholm, Robert McCartney, Jan Erik Mostr, Kate Sanders, Otto Sepp¨al¨a, Beth Simon, and Lynda Thomas.; A multi-national study of reading and tracing skills in novice programmers, 2004

[5] Michael McCracken, Vicki Almstrum, Danny Diaz, Mark Guzdial, Dianne Hagan, Yifat Ben- David Kolikant, Cary Laxer, Lynda Thomas, Ian Utting, and TadeuszWilusz; A multinational, multi-institutional study of assessment of programming skills of first-year CS students. In workinggroup reports from ITiCSE on Innovation and technology in computer science education, Canterbury, UK, 2001. ACM SIGCSE, 2001 - dl.acm.org.

[6] SaeedDehnadi and Richard Bornat; The camel has two humps'; Feb-2006; www.cs.mdx.ac.uk/research/PhDArea/saeed/paper1.pdf

[7] D. Mldlan Kurland; Roy D. Pea; Catherine Clement; Ronald Mawby; A Study Of The Development Of Programming Ability And Thinking Skills In High School Students; J. Educational Computing Research, Vol. 2(4), 1986

[8] Jeffrey Carver, Lorin Hochstein, Jason Oslin; Programming Ability: Do we know it when we see it? An Empirical Study of Peer Evaluation;

[9] JaanaHolvikivi; Conditions for Successful Learning of Programming Skills; N. Reynolds and M. Turcsányi-Szabó (Eds.): KCKS 2010, IFIP AICT 324, pp. 155–164, 2010. © IFIP International Federation for Information Processing 2010

[10] Richard Bornat, SaeedDehnadi, Simon; Mental models, Consistency and Programming Aptitude; 2008, Australian Computer Society, Inc. Tenth Australasian Computing Education Conference (ACE2008), Wollongong, Australia, January 2008.

[11] Simon, Sally Fincher, Anthony Robins; Predictors of Success in a First Programming Course; 2006, Australian Computer Society, Inc. Eighth Australasian Computing Education Conference (ACE2006), Hobart, Tasmania, Australia, January 2006

[12] Richard Bornat; Camels and humps: a retraction*; July 24, 2014

[13] Jens Bennedsen, Michael E. Caspersen; Abstraction Ability as an Indicator of Success for Learning Computing Science? ICER'08, September 6–7, 2008, Sydney, Australia, ACM

[14] Renumol. V. G.,DharanipragadaJanakiram, and Jayaprakash. S. 2010. Identification of cognitive processes of effective and ineffective students during computer programming. ACM Trans. Comput. Educ. 10, 3, Article 10 (August 2010);

[15] Ambrosio, A.P, Fábio Moreira Costa, Leandro Almeida, Amanda Franco, and JoaquimMacedo; Identifying cognitive abilities to improve CS1 outcome; 41st ASEE/IEEE Frontiers in Education Conference; October 12 – 15, 2011

[16] B. Kaučič*, T. Asič; Improving Introductory Programming with Scratch; MIPRO 2011, May 23-27, 2011, Opatija, Croatia; IEEE

[17] SaeedDehnadi; Testing Programming Aptitude; Workshop of the Psychology of Programming Interest Group, 2006 - hssc.sla.mdx.ac.uk

[18] MarkkuTukiainen and EeroMönkkönen; Programming aptitude testing as a prediction of learning to program; 14th Workshop of the Psychology of Programming Interest Group, Brunel University, June 2002

[19] David CLARK, Testing Programming Skills with Multiple Choice Questions; Informatics in Education, 2004, Vol. 3, No. 2, 161–178

[20] Otto Seppala; Advances in assessment of programming skills; Aalto University publication series DOCTORAL DISSERTATIONS, 98/2012

[21] Allen and Noel, 2002; Types and levels of educational obje;ctives; 2005 University of Central Florida UCF Academic Program Assessment Handbook; http://oeas.ucf.edu/doc/Bloom_Taxonomy.pdf ;

[22] http://www.socialresearchmethods.net/kb/dedind.php;

[23] http://www.academia.edu/1268673/Translation_Ability_and_Translatorial_Competence_Expert_and_Novice_Use_of_Dictionaries;

[24] http://www.businessdictionary.com/definition/decision-making.html;

[25] http://www.businessdictionary.com/definition/problem-solving.html;

[26] National Center for Research on Evaluation, Standards, and Student Testing (CRESST)