

Programming aptitude testing as a prediction of learning to program

Markku Tukiainen and Eero Mönkkönen
*Department of Computer Science,
University of Joensuu, Finland
markku.tukiainen@joensuu.fi
emonkkon@cs.joensuu.fi*

Keywords: POP-I.B. preprogramming knowledge, POP-II.A. individual differences

Abstract

Identifying variables that predict programming aptitude may help educators and employers select potential students and employees. The local polytechnic school wanted to have a test that would predict success in learning programming concepts. In this paper we present an empirical study that attempts to evaluate how predictive the developed testing of programming aptitude was.

Introduction

Programming aptitude testing has coexisted with the programming profession and education for decades. Probably the best known early test used to predict programming aptitude was the IBM Programmer Aptitude Test (PAT). The test consisted of three parts. The first part required the participant to figure out the next number in series, the second part to figure out analogies represented in figures, and the third part to solve arithmetic problems. The test was widely used in industry (in 1966, 349 American and Canadian companies used IBM-PAT to screen job candidates), but the relationship between the test and the actual job performance of programmers were often quite low (Mayer & Stalnaker, 1968).

Wolfe (1971) developed a series of tests to overcome some of the limitations of earlier tests. His tests required participants to manipulate numbers according to a complicated set of procedures. These tests primarily assessed the participant's ability to follow a detailed procedural instructions. The progress of programming aptitude testing continued to add new features to test battery, the most recent tests include at least a basic set of tasks that measure, for example logical ability, interpretation of specifications, attention to detail, accuracy and reasoning with symbols (see e.g. PAAT or I-PAT, Winrow, 1999).

The admission to educational institutes requires scholastic aptitude testing in some countries. In the U.S. the SAT tests are used on a normal basis for college admissions, although at the present time some schools, for example University of California, are contemplating the possibility of scrapping the SAT requirements altogether (Organ, 2001). Like all tests, the SAT tests are imperfect. It is argued that they account for no more than about 20% of the variance in first-year college grade point averages (Ransdell, 2001). It is suggested that some of the tests used for academic assessment purposes, e.g. English for Specific Academic Purposes, are suggested to be abandoned altogether because of the difficulties inherent in testing (Clapham, 2000). The basic conclusion of these student performance studies is that aptitude, which is the potential to learn a task, is not necessarily a good indicator of performance on the task per se.

This paper reports a study, which was conducted to find out whether the programming aptitude of Finnish polytechnic students can be predicted using the results of scholastic aptitude tests that measure one's ability to make logical conclusions, learning ability and verbal ability.

Method

We invited 33 volunteers to participate in the experiment. They were administered the scholastic aptitude test before entering the study program, a pre-test of programming aptitude before the first programming course, and a post test as the final exam of the course.

Subjects

The subjects in the experiment were students of the Business Information Degree programme at North Karelia Polytechnic. All of them were first-year students and all volunteered for the experiment. They received no pay for their participation nor was any other incentive used. Their background education consisted of a normal Finnish elementary education with 12 years of public schooling, ending with a matriculation examination.

Materials

The pre-test and the post-test materials are presented in appendices A and B respectively at the end of this paper.

Data collected

Students' mathematics and Finnish high school grades (MATH = mathematics grade, FINN = Finnish language grade, scale 1-10), results of scholastic aptitude tests (LOG = logical aptitude, VERB = verbal aptitude, LEARN = learning aptitude, scale 1-5) and results of the programming aptitude test (scale 1-5) and results of the final exam (scale 1-5) of students' first programming course at the polytechnic were used in this study.

Design

All of the subjects participated at each of the tests. There were no differences made in the treatment of the participants. Of the original 33 subjects, 7 were removed from the later analysis because they did not for various reasons attend the final exam.

Procedure

The high school marks were collected from the students' applications to the Polytechnic. The scholastic aptitude tests were carried out by a professional company. In these tests the applicants' verbal ability (VERB) was tested by having them follow a given set of instructions. Their logical ability (LOG) was tested through tasks in which they had to handle relations between different words. Their learning ability (LEARN) was tested through tasks where the applicants had to convert words into numbers according to a given set of instructions as fast as possible. The scholastic aptitude tests were carried out during the entrance exams to the polytechnic. The programming aptitude test was carried out after the students had been admitted to the school and before the start of their first programming course. The pre-test (a programming aptitude test) (Huoman, 1989) was one designed by M.Sc. J. Huoman, a graduate from the Department of Computer Science, University of Joensuu. The test consists of a collection of programming-like tasks that do not require previous programming experience. The post test was the final exam of the students' first programming course, which lasted one semester with 6 hours a week lectures and exercises. The tool used during the course was Ms Visual Basic.

Results

We equated the students' understanding of programming concepts in the final exam with the mastery of their first course materials. The questions in the final exam concerned the mastery of Ms Visual Basic concepts, control flow constructs, understanding of the data and control flow and two simple programming tasks.

| <i>Number of students</i> | ABILITY | | | Total | |
|---------------------------|---------|---|---|-------|---|
| | 2 | 3 | 4 | | |
| Final exam grade | 1 | 2 | 3 | 4 | 7 |
| | 2 | 1 | 3 | 1 | 5 |
| | 3 | 2 | 4 | 1 | 7 |
| | 4 | 4 | 1 | 1 | 6 |

| | | | | | |
|-------|---|----|---|--|----|
| | 5 | | 1 | | 1 |
| Total | 9 | 12 | 5 | | 26 |

Figure 1: Cross-tabulation of scholastic aptitude and the final exam grades

Figure 1 shows the distribution of the final exam grades and the grades from the scholastic aptitude tests. The attributes in the scholastic aptitude test (LOG, VERB, LEARN) have been unified in a single variable entitled ABILITY (the average of the original attributes). The justification for the unification was tested using Cronbach's test ($\alpha = 0.8413$).

There were no significant correlations between the grades from scholastic aptitude and the grades from the final exams. Neither were there any significant correlations between the programming aptitude test and the mathematics grade, the Finnish language grade and the attributes in the scholastic aptitude test (See Table 1).

If we take a closer look at grades from the programming aptitude test and the grades from the final test, we realize that there seems to be no interaction between them (See Figure 2).

| | | MATH | FINN | LOG | VERB | LEARN | HUOM |
|-------|----|--------|--------|--------|-------|-------|--------|
| FINN | P | ,912** | | | | | |
| | Sp | ,892** | | | | | |
| LOG | P | ,364* | ,320 | | | | |
| | Sp | ,321 | ,317 | | | | |
| VERB | P | ,370* | ,377* | ,485** | | | |
| | Sp | ,351* | ,377* | ,428* | | | |
| LEARN | P | ,378* | ,412* | ,362* | ,384* | | |
| | Sp | ,365* | ,411* | ,378* | ,398* | | |
| HUOM | P | -,209 | -,177 | ,201 | ,081 | ,023 | |
| | Sp | -,203 | -,107 | ,237 | ,095 | ,061 | |
| EXAM | P | -,280 | -,362* | ,241 | -,186 | -,220 | ,513** |
| | Sp | -,275 | -,304 | ,255 | -,167 | -,247 | ,518** |

Table 1 – . Cross-tabulation of variables and Pearson and Spearman correlations. * Significant correlation ($p < 0.05$). ** Very significant correlation ($p < 0.01$).

| Number of students | | Huoman test grade | | | | Total |
|--------------------|---|-------------------|---|---|---|-------|
| | | 2 | 3 | 4 | 5 | |
| Final exam grade | 1 | 5 | 1 | 1 | | 7 |
| | 2 | 4 | 1 | | | 5 |
| | 3 | 5 | 2 | | | 7 |
| | 4 | 1 | 4 | 1 | | 6 |
| | 5 | | | | 1 | 1 |
| Total | | 15 | 8 | 2 | 1 | 26 |

Figure 2: Cross-tabulation of Huoman aptitude test and the final exam grades

Table 1 shows that there is either a significant or very significant correlation between the attributes of scholastic aptitude tests. This seems quite natural. When the final exam grades are compared to the attributes in the scholastic aptitude test, the other attributes, with the exception of the logical attribute, have a negative correlation to the grades. So it seems that the final exam could emphasize the logic ability of the students.

There is a very significant correlation between the Huoman programming aptitude test and the final exam. So it seems that the Huoman test and the exam measure the same abilities of the students. Because no other dependencies to the measured background variables were found, this made us wonder if there are any other background variables that could better explain the interaction. We decided to ask prior programming experience from the subjects. They all answered the email questionnaire concerning their experience.

Thus the group was divided into two study groups. The first group ($n = 11$) had no background knowledge of programming, whereas the second group ($n = 15$) had some experience of programming (on an average they had experience on one programming language with 1 to 5 programs with a length of 1-100 lines). Their background was controlled by their final mathematics and Finnish language marks on their high school diploma and the scholastic aptitude test. There were no significant differences between the groups.

When we take into consideration the background knowledge of the subjects, we see a difference in the distribution of Huoman aptitude test results and the grades from the final exam (See Figures 3 and 4).

In Figure 3 (the group with no prior programming knowledge), it seems that the Huoman programming aptitude test gives no evidence of success in the programming exam. The only noticeable trend could be that if a student is very successful in the Huoman test (grades 4 to 5), then she will be successful in the programming test as well.

In Figure 4 (the group with prior programming knowledge), there seems to be a more systematic arrangement. The Huoman aptitude test seems to provide a floor limit to the success in the programming test. This is not a very clear effect, but seems to be coherent.

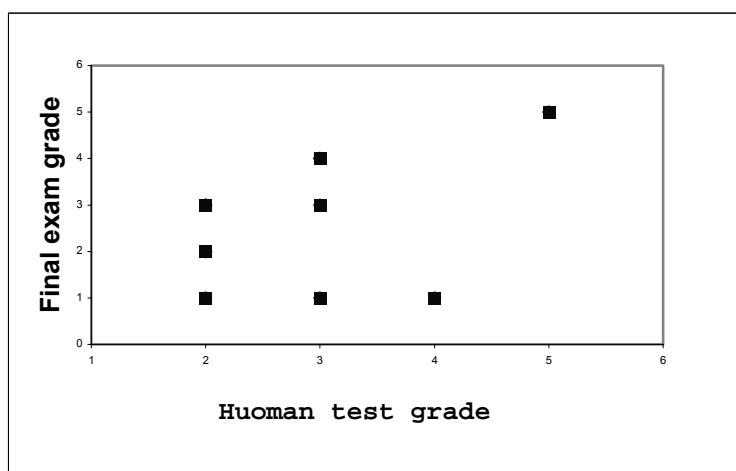


Figure 3: Group with no prior programming knowledge

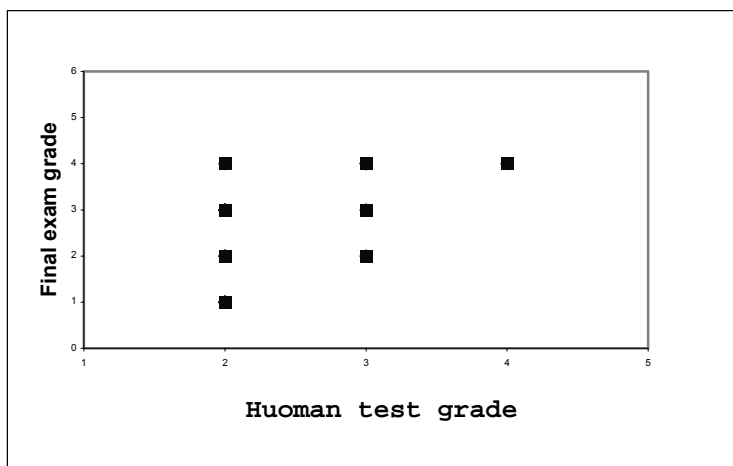


Figure 4: Group with prior programming knowledge

Discussion

The immediate question that raises from the study is then what to measure as a prediction for programming aptitude. Some research (see e.g. Whitley 1996) has suggested, for example the importance of cognitive style or psychological type to predict one's aptitude. Using the Myers-Briggs type indicator it has been determined that in the dimension SN (sensing-intuiting), intuitive persons are more prone to succeed in programming than sensitive persons. This seems to be in contrast with the stereotypic ISTJ-personality (introvert, sensing, thinking, and judging) associated with programmers (Lyons, 1985).

On the other hand, if we consider the skills portfolio needed in the IT industry, there is a demand for very different sets of talent other than just pure programming or coding (Nakayama & Sutcliffe, 2001). What the industry needs more is not the aptitude tests for programming but the establishment of credentials that accurately represent the ability of software engineering (Shaw, 2000). In programming we will probably continue to see a ratio of 20:1 differences in performance as has been customary. The aptitude just presents the potential to learn to program not the performance in programming. If we think about the factors affecting the performance in programming one cannot deny the role of native talent. Yet one's psychological style is important. Great programmers are also great tinkerers (Curtis, 1991), they have the intrinsic motivation to see how a program or a system works.

Conclusion

The ability to predict an individual's potential to learn programming concepts is important for many reasons (Weinberg, 1998). Our immediate concern in this study was to find tests other than the scholastic aptitude tests to help in the selection of the students to the polytechnic programming degree program. Although the Huoman programming aptitude test predicted no more than 25 percent of the final exam results (which is the common finding of almost any of the measured predictors, see e.g. Evans & Simkin, 1989), it was considered to predict the floor limit of performance in the first programming course for those who had prior knowledge of programming concepts.

Acknowledgements

We would like to thank Professor Jorma Sajaniemi for encouraging and stimulating the writing of this paper. We wish also to acknowledge Professor Sajaniemi as the supervisor of the empirical study presented here.

References

Clapham, C. (2000) Assessment for academic purposes: where next? *System*, **28**: 511-521.

- Curtis, B. (1991) Techies as non-technological factors in software engineering? In the proceedings of the *International Conference on Software Engineering (Proceedings of the 13th international conference on Software engineering 1991)*, Austin, Texas, United States, 147-148.
- Evans, G.E. & Simkin, M.G. (1989) What best predicts computer proficiency? *Communications of the ACM*, **32** (11): 1322-1327.
- Huoman, J. (1986) *Predicting programming aptitude*. Master's Thesis, Department of Computer Science, University of Joensuu: Joensuu (in Finnish).
- Mayer, D. B. & Stalnaker, A. W. (1968) Selection and evaluation of computer personnel – the research history of SIG/CPR. In the proceedings of the *1968 ACM National Conference (23rd ACM National Conference)*, 657-670.
- Nakayama, M. & Sutcliffe, N.G. (2001) IT skills portfolio research in SIGCPR proceedings: analysis, synthesis and proposals. In the proceedings of the *Special Interest Group on Computer Personnel Research Annual Conference (Proceedings of the 2001 ACM SIGCPR conference on Computer personnel research 2001)*, San Diego, California, United States, 100– 113.
- Organ, D. W. (2001) Predicting success in school and at work. *Business Horizons*, May-June 2001: 1-2.
- Ransdell, S. (2001) Predicting college success: the importance of ability and non-cognitive variables. *International Journal of Educational Research*, **35**: 357-364.
- Shaw, M. (2000) Software engineering education: a roadmap. In the proceedings of the *International Conference on Software Engineering (Proceedings of the conference on The future of Software engineering 2000)*, Limerick, Ireland, 371–380.
- Weinberg, G. M. (1998) *The psychology of computer programming (silver anniversary edition)*. New York: Dorset House Publishing.
- Whiley Jr, B.E. (1996) The relationship of psychological type to computer aptitude, attitudes, and behavior. *Computers in Human Behavior*, **12** (3): 389-406.
- Winrow, B. (1999) The Walden programmer analyst aptitude test. *Dr. Dobb's Journal*, Fall 1999, Software Careers. (<http://www.ddj.com/documents/s=894/ddj9914b/9914b.htm>).
- Wolfe, J. M. (1971) Perspectives on testing for programming aptitude. In the proceedings of the *1971 Annual Conference of the ACM*, 268-277.
- Wolfe, J. M. (1971) Wolfe programming aptitude test (school edition). In the proceedings of the *Ninth annual SIGCPR conference*, Illinois, United States, 180 - 185.

Appendix A. Human Programming aptitude test.**Task 1**

The company has information of their employees in three different lists. You have acquired all three lists, which all have little bit different information depending on the purpose of the list.

The lists have following information of the employees:

List 1: the number, the name, the occupation and the department

List 1 is ordered by the number of employee to ascending order

List 2: the name, the number, the address, the phone number and the social security number

List 2 is organized to alphabetic order by the name

List 3: the number, the social security number, salary and some secret information

List 3 is ordered by the number to ascending order

Your job is to make a report of those employees whose salary is greater than 2000 €. The report has to display the name, the address, the department and the salary of the employees.

Describe how you would solve the problem.

Task 2

Let i and j be integers.

List all values of i and j , that make the expression to be always true

a) $(i \geq 1)$ and $(i \leq 5)$

The expression is true when i has values:

b) $(j \geq 3)$ or $(j \leq 7)$

The expression is true when j has values:

Task 3

Write a letter sequence, that is next on the series. Try to determine the general form of the series:

a) bce , bbcde , bbbcdde , _____

The general form:

b) bcace , bcacacace , bcacacacacace , _____

The general form:

c) _____ , abccdd , abbccdd , abbbccdd , ...

The general form:

Task 4

Your job is to sum up 50 numbers and at the end report the sum and the count of numbers that were positive numbers (>0).

Describe how you would solve the problem or write in a pseudo-code.

Task 5

Your job is to deduce which of the given numbers is the smallest. How many comparisons you have to make if you have:

a) 3 numbers _____

b) 6 numbers _____

c) n numbers _____ ($n \geq 1$)

Appendix B: The final exam questions.**Task 1.**

The following statements/reserved words are from Ms Visual Basic. Describe shortly what they are:

- a) Option Base 1
- b) Cint
- c) Boolean
- d) String
- e) InputBox

Task 2.

- a) Write the following loop using For-statement

```
k=1
```

```
Do until k > 50
```

```
    Number_table (k) = 0
```

```
    print k
```

```
    k = k + 1
```

```
Loop
```

- b) Write the following Case - statement using conditional statements (If – Then – Else):

```
Select Case intNumber
```

```
Case 1
```

```
    Print "Number is 1"
```

```
Case 6
```

```
    Print " Number is 6"
```

```
Case Else
```

```
    Print "All the other numbers"
```

```
End Select
```

Task 3.

What does the following program print?

```
Dim a,b,c,d,e As Integer
```

```
a= 2
```

```
if a = 1 then
```

```
    a = a+1
```

```
end if
```

```
if a = 2 then
```

```
    a = a+2
```

```
end if
```



```
if a = 3 then
    a = a+3
end if
if a = 4 then
    a = a+4
end if
Print "a="+a
b = 2
Select Case b
case 1
    b = b + 1
case 2
    b = b + 2
case 3
    b = b + 3
case 4
    b = b + 4
Print "b="+b
c = 11 \ Int(8/3);
Print "c="+c
d = 20
for e = 1 to 4
    d = d - e
Next e
Print "d="+d
a = 1
e = 1
while a < 8
    a = a + 1
    if a < 4 then
        e = e + a
    else
        e = e + 2
    end if
wend
Print "e="+e
```

Task 4.

Write a program that first sorts into ascending order and then prints out the numbers in the Credits –table. The Credits –table has been declared:

Option Base 1

Dim Credits (15) As Integer

Task 5.

Write a program that reads numbers from the user, until the number 0 is encountered or 20 numbers have been inputted. The numbers must be saved into the Information –table. After this the program should go through Information –table, and figure out:

- a) Is the number 50 in the table?
- b) How many numbers are < 50 ?
- c) What is the average of all the numbers?