Environment. Technology. Resources, Rezekne, Latvia
Proceedings of the 10th International Scientific and Practical Conference. Volume III, 29-35

Computer Programming Aptitude Test as a Tool for Reducing Student Attrition

Juris Borzovs, Laila Niedrite, Darja Solodovnikova
University of Latvia, Faculty of Computing.
Address: Raina blvd 19, Riga, LV 1586, Latvia

Abstract. **The stable trend to lose from one-third to half of students in the first study year of computing studies motivated us to explore, which methods are used to determine in advance such applicants, who have no change to overcome the first study year. Initially, a research about the factors influencing the attrition in Faculty of Computing at the University of Latvia was conducted. The research revealed that the trend of non-beginning studies might indicate the wrong choice of the study field and possible lack of understanding of what is programming by enrolled students (applicants as well as pupils).**

The study provides the review of the situation with programming aptitude tests in the world, which could serve as one of the solutions to the dropout reduction. An action plan is proposed, which is based on the exploration of students and evaluation of activities already conducted at the Faculty of Computing of the University of Latvia to reduce dropout (School of Young Programmers, Compensative Course in High School Mathematics, Mentoring programs). Moreover, the supplementation of these activities by one of the existing programming attitude tests (or a combination of several tests) or a necessity to develop a new similar test is considered.

Keywords: **Aptitude test, attrition rate, computer science education, data analysis.**

I INTRODUCTION

In recent years the observed practically stable trend to lose from one-third to half of students in the first year of computing studies motivated us to explore, whether the world has not found indeed a method, how to determine in advance such applicants that have no chance to overcome the first study year.

On the one hand, the dropout students and the teaching staff have wasted their resources. On the other hand the Ministry of Science and Education and experts that are evaluating the study programs frequently associate the high dropout rate with a low effectiveness of the implementation of the study program (another matter – whether it is reasonable) and ask what is done to reduce the attrition.

To reduce the attrition, it was decided initially to perform a research about the factors influencing the attrition in Faculty of Computing at the University of Latvia [1]. Our study investigated students enrolled into the computer science bachelor and programmer professional programs in one year (2013). It were originally assumed, that following factors could have a potential impact on attrition:

1. High school grades (admission score)
2. Compensative course in high school mathematics
3. Intermediate grades for core courses
4. Prior knowledge in programming.

The results of the study in more detail can be found in [1]. Further, a short summary of the main conclusions is given to justify the choice of appropriate solutions to reduce the attrition.

There exists a large group of students, who in fact do not begin studies and who are later expelled. The majority of expelled students drop out in the 1st semester of the 1st year. These students include both students with weaker and with quite good high school grades. Since the 1st year dropout consists mostly of students, who have not really begun studies, so the high school grades do not have significant effect on dropout, but they may have an impact on the further study process.

The hypothesis that a programming background is an important factor influencing dropout was rejected, since we found out that the ratio between students with and without prior knowledge remained the same at the start and end of the first study year. This was concluded based on the data from self-assessment questionnaires. Also there is still a large number of students, who do not really begin studies, which could be explained by the wrong choice of a study program and insufficient insight into "what is programming", what the program developers are doing at work, and during the self-assessment test the knowledge in an informatics at school was possibly incorrectly treated as knowledge in programming.

ISSN 1691-5402

DOI: http://dx.doi.org/10.17770/etr2015vol3.175

It is not necessary to prove, that every man has a different capability, however, in the society an opinion dominates, that every „normal" student is capable to acquire the basics of the most important sciences, e.g. language, mathematics etc. Therefore, they are included into the school curriculum. There is still an open question, whether in this set of skills and knowledge, also the programming should be included.

Even in most of high schools the programming is not on the list of mandatory subjects, and so it is in the whole world, despite the recent actions (e.g. [2]).

It is not proved, however, there is a widespread opinion that the programming is not among skills that should be taught mandatory, because a significant part of "normal" people can not learn them.

In the 1st study year it is meant usually by „programming" just writing a computer program in some programming language. Certainly, such limited understanding is quite far from all methods of developing computer programs [3], and includes not more than 20% of tasks in professional software development. However, it can be hardly imagined that software developer can be without these program coding skills.

According to the research results [4], universities should prepare an action plan to reduce the attrition, which, at the same time, does not lower the quality of studies [5] or admission requirements [6].

In the following part of the paper, a review about the programming aptitude tests is given, that can be used to reduce attrition. We will propose an action plan that is based on investigation of students and on analysis of already undertaken actions to lower the attrition rate ("The School of Young Programmers" [7], Compensative course in high school mathematics, Mentoring program). We will also outline considerations about supplementation of these actions with existing programming aptitude tests or combination of them, also the necessity to develop new targeted tests will be discussed.

II MATERIALS AND METHODS

Attempts to pre-select the most appropriate students for programming are at least 50-60 years old [8]-[10]. Searching *'computer programming aptitude test'* in *Google Scholar*, at least 60 publications for the period from 1960 until 2014 are obtained, but in neither of them, including the youngest publications, „silver bullet" unfortunately was not found.

All existing approaches to computer programming aptitude determination are conditionally divisible into two parts: the ones based on the psychological tests (for instance, [11]-[18]) and the ones based on solving specifically designed non-programming tasks (for instance, [19]-[21]).

In practice, hybrid tests [22] are often used, which contain elements to check the „logical reasoning, numerical problem solving, pattern recognition, ability to follow complex procedures and attention to detail". So, in addition to problem solving, such tests include also evaluation of various specific personality traits.

There also exist tests, which are used to self-asses the existing programming knowledge. Programming simulation includes „pseudocode, control structures (e.g. loops), look-up tables, sets, arrays, boolean true/false, looping and other programming structures".

If the time comes, when an adequate idea of programming and one's skills in this area will be obtained before applying to the university computer science study program, everything written above will become obsolete. While this is not the case, the most promising psychological [16], [18] and problem solving, for example, [22] self-test summary should be offered to the prospective students.

Psychological Tests

In one of the studies [16], Myers-Briggs personality types detection test is used to determine, which types of college students are doing better in the programming introductory course.

Myers-Briggs test is available, for example, at this website [23]. The test determines a total of 16 different personality types, 4 aspects are evaluated:
1) general attitude: Extraverted vs. Introverted;
2) perception function: Sensing vs. Intuition;
3) judging function: Thinking vs. Feeling;
4) perception – judging domination: Judging vs. Perceiving.

The study [16] concludes that „sensing students performed better on programming assignments than intuitive students, and that judging students achieved higher programming averages than perceptive students". Commenting on the findings of these types, it can be noted that *Sensing* means that an individual relies on a specific, topical information, but *Intuition* means that a person relies on his or her vision of the world. On the other hand, people with judging dominance perceive the world as an ordered structure that follows the set of laws, in contrast to the perceiving people, who perceive the world as a structure that can take different forms and results.

Another study [18] describes how results of various tests performed by students, correlate with their programming ability. It was determined that 2 tests SQ [24] and EQ [25] used together, showed the best correlation results. A test „Cambridge Personality Questionnaire" (SQ - Systemizing Quotient - test) was used, which determines how easily an individual understands object systems. The second test used in the study, was „*The Cambridge Behaviour Scale*" (*Empathy Quotient test - EQ*), which describes how easily an individual understands human emotions. A difference between the two test results: SQ - EQ. ($r = .67$) revealed the correlation with programming results. Separately, each of these tests showed significantly poorer correlation.

The authors of the study explained the obtained results with the fact that people with SQ significantly higher than EQ prefer dealing with ordered systems in everyday life, instead of dealing with people, who they do not understand. The authors also suggest that a large SQ – EQ value indicates the aptitude for studies that require great effort and practice that definitely corresponds to the programming studies.

Problem Solving Tests

One of the further examined examples [22] describes a test that already exists at the university, while the other shows the possibility to realize a test in a modern environment, attractive and habitual for pupils [26].

The test of the University of Kent includes questions about 1) „Logical thinking and problem solving", which essentially evaluates the ability to prevent problems arising during the development of IS, 2) „Pattern and syntax recognition", which essentially tests the ability to discern differences and pay attention to details, to verify the information, 3) „Ability to follow complex procedures", which assesses the ability to organize, process events, objects in a logical sequence, according to some regulations, assess the impact of an action on the future.

The test includes 26 questions. Answers are assigned points and groups are defined by the number of points, but the selection of specific questions and division into groups is unclear, since the test is based on the practical experience of its usage. This gives an insight into the test, which has already been implemented and used in practice at the university, however, the usage of such test in our case should be further explored and evaluated.

The second example, „Aptitude and Logical Reasoning" [26] test, which is implemented as a mobile application and is available in Google Play, is not primarily intended to measure the programming aptitude, but helps to improve problem solving skills and prepare for various exams and similar tests. There are many such mobile applications, which suggests that both the test and the its implementation environment are topical and demanded among users.

III RESULTS AND DISCUSSION

According to the conclusions of the primary study of the distribution of students [1], a majority of students really do not begin studies, therefore, activities targeting both „early" dropped out students and applicants are considered. Several ideas are planned to be studied further, for example, the possibility for students to assess themselves their suitability for a chosen field of study, by offering a variety of self-assessment tests - personality tests, logic tests, mathematics tests. For example, the contents of the latter test could be based on the compensative course in high school mathematics, which is already conducted at the Faculty of Computer Science. It could be implemented as a computerized test for self-assessment that candidate students could take before they choose the study program.

For example, the contents of the latter test could be based on the compensative course in high school mathematics, which is already conducted at the Faculty of Computer Science. It could be implemented as a computerized test for self-assessment that candidate students could take before they choose the study program. The groundwork in this direction has been started in the project "School of Young Programmers" of the Faculty of Computer Science at the University of Latvia [7]. The purpose of this project is to promote the comprehension about programming in high schools.

A detailed insight into both the study about students, the analysis of existing activities aimed at reducing the dropout and the new action plan follows.

Results of the Initial Survey - Reasons for the Choice of the Faculty of Computing

The reasons for the students' choice of the Faculty of Computing of the University of Latvia could be determined from the questionnaire completed by prospective students beginning their studies. In 2013 235 questionnaires were processed. Questionnaires are anonymous, summarized results are available. It is allowed to mark several reasons for the choice, and the most marked reasons are shown in the Table1.

I like everything related to computers	127	54,0%
Potentially high salary in the future	108	46,0%
I think that there is a high quality of study	98	41,7%
I have been programming and I like it	97	41,3%
I hope to learn a profession where I could easily get a job	89	37,9%
I am interested in mathematics	91	38,7%

TABLE 1.

REASONS FOR THE CHOICE OF THE FACULTY OF COMPUTING

The reasons with a positive impact on the further study process were identified as „I have been programming before and I like it" and „I am interested in mathematics". Many students also marked the reasons that can indicate misunderstanding of the profession, such as „I like everything related to computers", i.e. underestimation of the true nature of the program is possible, because the idea of computing may be influenced by the use of computers according to the contents of a high school informatics course. Another large part of students choose the reasons that could indicate even a lack of interest for the study content („potentially high salary in the future" and „I hope to learn a profession where I could easily get a job").

Since there is no information about the respondents of the questionnaire, the precise analysis of the connection between reasons and dropout is not possible, but conclusions can be drawn that before enrolment, students underestimate that the curriculum includes mathematical courses (less than a half of students are interested in mathematics), and that they will have to program. 41.3% of new students selected the answer „I have been programming and I like it", but for other students, it is unknown, what the outcome of their familiarization with programming will be, and it would be better to offer the opportunity to get an idea of programming earlier - before enrolment into the study program. While the programming subject is not mandatory in the high school curriculum for everyone, it would be good to create such opportunity.

The table does not include the reasons marked by less than 20% of respondents, including the influence of parents and friends, the opportunity to easily get into the group financed by the state, which are also risky reasons with high potential of dropouts, because these answers do not indicate students' interest and idea of the profession.

School of Young Programmers

The School of Young Programmers (SYP) [7] was created to promote the comprehension about computing [Fig.1]. The lectures available in the online environment allow to understand what a programming is. In 2014 at the lectures of the School of Young Programmers, a modern, intuitive programming language Scratch designed especially for schools, was studied. Scratch is a project of the lifelong learning group "Lifelong Kindergarten" of the Massachusetts Institute of Technology (USA). Although this language is easy to learn, it is possible to create quite complex programs in Scratch 2.0. "The School of Young Programmers" is offered by the Faculty of Computing of the University of Latvia with the support of the Fund of the University of Latvia and JSC "4finance".

Fig.1. E-environment of the School of Young Programmers

SYP offers tutorials of the programming language Scratch, program examples, video materials and environment [27], which allows pupils, who have not studied programming at high school, to learn programming by themselves in a short time. *Scratch* can be used at different levels of difficulty for pupils of different ages, therefore, it can also be used to evaluate, whether a pupil is interested in and is good at programming, before making a decision about studying at the Faculty of Computing or elsewhere.

Mentoring Program

The mentor program is designed at the University of Latvia to support a freshmen at the beginning of their study, because they have to contend with a large amount of new information and settle into an unfamiliar environment. The program works at almost all faculties, including the Faculty of Computing [28]. At the Faculty of Computing, the program is implemented by the Student Authority, in collaboration with the Alumni of the Faculty of Computing, the Student Council of the University of Latvia, Accenture Latvia and FranklinCovey. The participants of the program - mentors, who are senior students, have the opportunity to participate in free training to get new experience useful for their career. Freshmen, in turn, have the opportunity to get an advisor, who can provide information, advice and support. The mentoring program at the Faculty of Computing is designed to help students to earn a diploma, providing support directly in the first year when the dropout is the highest.

The activity of the mentoring program is described by the following statistics: 42 mentors and 66 freshmen applied for the program. Among freshmen, 80.3% of those who took part in the mentoring program, registered for the second semester, but only 63.21% of students who did not have mentor, registered for the second semester. So, among mentored freshmen, there are better results, however, different conclusions could be drawn: either the program really helped first-year students, or more active, interested students, who would cope with the studies either way, registered for the program.

Current Situation with the Compensative Course

The Compensative Course in High School Mathematics was introduced in 2009 with the purpose to expand knowledge of students of the Faculty of Computing, to improve the situation with the study process both for teachers and for students. Students are able to get knowledge on topics, which were not promptly acquired in high school, and the absence of this knowledge complicates the understanding of the university course content. The course also allows teachers to work with the audience with more homogeneous knowledge.

The course content includes topics that were identified during interviews with teachers and taking into account teachers' proposals about the essential topics of their courses, which should be known already from high school.

„Compensative Course in High School Mathematics" is mandatory for the students of the Faculty of Computing with the admission examination score less than 700 (1000 is the maximum possible). The course is taught by means of lectures on the the planned course topics. The course is conducted in the 1st and 2nd semesters of the first study year and the course topics are divided into the groups: Algebra and Functions (in the fall semester), as well as Trigonometry and Geometry (in the spring semester). Students have to take a test on each of these groups of topics and when they get an assessment „passed" for each of the two groups in the corresponding semester, the final test for the semester is obtained. Students are allowed to take each of these tests repeatedly several times. The e-learning environment Moodle is used to inform students about their results. However, the tests themselves have not been implemented in the e-learning environment. If a student passes a test faster than at the end of the semester (if a student is sure that he or she has acquired unclear issues, then he or she can take tests more quickly), then a student is allowed not to attend this course anymore. This course was offered to 122 students out of 254 students enrolled in 2013.

The Compensative Course in High School Mathematics directly represents both the necessary training in mathematics and students' personal qualities, such as motivation, commitment and perseverance, because a student does not have to acquire a new difficult content, but has to learn deeper topics, which have already been studied at high school, to avoid potential problems in further studies.

It should be recalled that this course targets those 122 students with admission score of less than 700, which includes grades for high school mathematics. Therefore, analysing students by the mathematics knowledge aptitude indicator, it is more important to analyse further exactly the weakest, less suitable, group of students, to plan further support activities. However, as a study of dropout showed [1], there is also a high dropout rate among students with more than 700 points, which is why the pre-enrolment activities aimed at prevention of dropout due to other reasons, including the wrong choice, are recommended to be conducted among this group of students as well.

Therefore, some of the results are described at a glance, which could be used to distinguish separate student subgroups that could benefit from different supportive activities to reduce dropout. The full results of the dropout study can be found at the paper [1].

Results of a study on the outcomes of the Compensative Course in High School Mathematics allowed to distinguish 3 groups of students: „do-nothing students" - 15 students (100% of them dropped out), which did not attempt to pass any of the tests, the 2nd group consisting of 74 students who sooner or later passed the tests (30% of them dropped out) and „failers" - 33 students who failed to pass the tests (88% of them dropped out). Conclusions about students were made based on the available information about the history of passing the course. Among „test failers", there are mostly the students who have quickly given up (attempted to pass the tests only once), but more than a half of those, who passed the tests, are the students, who attempted to pass the tests several times. These data indicate that the result depends not so much on the level of high school knowledge, as on other reasons – interest or motivation (or lack of them) to work.

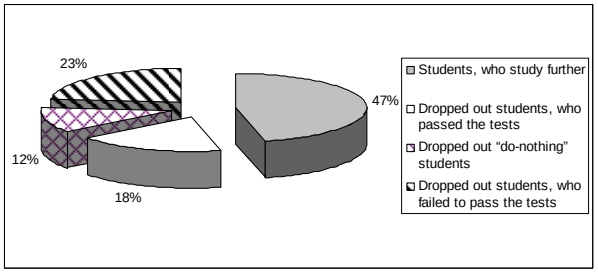

Fig.2. Groups of test takers

So the following groups of students can be distinguished (see Fig.2):

1) Students, who study further – high school knowledge is acceptable after taking the course
2) Dropped out students, who passed the tests – high school knowledge is acceptable after taking the course, the motivation has been sufficient to pass the course, however, students do not continue studies
3) Dropped out „do-nothing" students – wrong choice of the program and lack of motivation, no data on the knowledge in this course
4) Dropped out students, who failed to pass the tests – lack of motivation, high school knowledge is not acceptable as the course was not passed.

Proposed Plan for Dropout Reduction

The Table 2 shows the evaluation of the above-mentioned groups, which activities and when should be planned.

According to the characteristics of each group, the time when activities should be conducted, as well as the type of the support activities, is identified. The existing compensative course in high school mathematics should be retained at the beginning of studies for students with admission scores less than 700 points. Since the previously given distribution into groups is possible only after the fact of dropping out, then, as a preventive measure to reduce dropout caused by other reasons, such as difficulties in settling

in the university environment, difficulties in orientation in the study process, a mentoring program should be developed and popularized at the beginning of studies.

TABLE 2.
ACTIVITIES FOR DROPOUT REDUCTION

Groups	When the activity should be offered?	Activity
Students, who study further	At the beginning of studies	Existing compensative course in mathematics
Dropped out students, who passed the tests	At the beginning of studies	Mentoring program, existing compensative course in mathematics
Dropped out "do-nothing" students	Before enrolment	Aptitude tests
Dropped out students, who failed to pass the tests	Before enrolment	Aptitude tests

On the other hand, to prevent the enrolment of students inappropriate for the programmer profession, which correspond to the latter two groups, profession aptitude tests should be offered, as well as educational and promotional activities for pupils, explaining what is programming and which knowledge is required for a programmer, where one of the tools would be the promotion of the School of Young Programmers.

Our proposals in the area of the professional aptitude test are the following:

1) To implement the assessment test of the compensative course in mathematics as an online test to evaluate the knowledge in mathematics,
2) For the evaluation of programming skills, if perspective students have such skills, the supplementation of materials of the School of Young Programmers with a set of exercises organized in the form of a test, where the execution of these exercises would not exceed a certain time limit. The test would serve as an initial insight into the aptitude measurement and attraction of interest, and the rest of the materials available in the e-environment could be then used for the in-depth understanding.
3) Psychological tests [16], [18], the description of which has already been given above,
4) Problem solving tests [22], [26] - a combination of the best examples.

The proposed package should be promoted before the enrolment.

IV CONCLUSIONS

There are various tests for programming aptitude measurement, at the same time, the specificity of the national education system and the university should be taken into account to evaluate the plan for the most appropriate activities to reduce dropout.

In this paper, the action plan, which, in addition to the programming aptitude tests implemented in a self-assessment form, also includes other activities that should be used both before and after the enrolment and require the involvement of the faculty staff, was proposed.

One of the most important further tasks is the combination of the best examples of the problem solving tests – including both the content and the form – by offering content in the Latvian language as well as conducting the content selection or novelty. The further tasks are related to this test and approbation of the whole action plan and determination of aptitude criteria.

V REFERENCES

[1] J. Borzovs, L. Niedrīte, and D. Solodovņikova, "Factors affecting attrition among first year computer science students: the case of University of Latvia", 2015, submitted for publication.

[2] Computing at School, [Online]. Available: http://www.computingatschool.org.uk/ [Accessed: Mar.17., 2015]

[3] D.Bricklin, "Why Johnny can't program", August 2002, [Online], Available: http://www.bricklin.com/ wontprogram.htm [Accessed: Mar.17., 2015]

[4] F. Araque, C. Roldán, and A. Salguero, "Factors influencing university drop out rates", Computers and Education, vol. 53(3), pp. 563-574, 2009.

[5] L. Paura and I. Arhipova, "Cause Analysis of Students' Dropout Rate in Higher Education Study Program", Procedia-Social and Behavioral Sciences, vol. 109, pp. 1282-1286, 2014.

[6] L. Grebennikov and M. Shah, "Investigating attrition trends in order to improve student retention", Quality Assurance in Education, Vol. 20 (3), pp. 223 – 236, 2012.

[7] "School of Young Programmers" [Online]. Available: http://www.df.lu.lv/nacstudet/jauno-datoriku-skola/, [Accessed: Mar.17., 2015].

[8] T.C. Rowan, "Psychological tests and selection of computer programmers", Journal of the Association for Computing Machinery, Vol. 4, pp. 348-353, 1957.

[9] W.J. McNamara, J.L. Hughes, "Manual for the revised Programmer Aptitude Test", New York: international Business Machines Corporation, 1959.

[10] A.W. Stalnaker, "The Watson-Glaser Critical Thinking Appraisal as a predictor of programming performance", Proceedings of the Third Annual Computer Personnel Research Group, pp. 75-77, 1965.

[11] E. Spranger, " Types of Men: The Psychology and Ethics of Personality", Johnson Reprints, New York, 1966.

[12] C.K. Capstick, J.D. Gordon, and A. Salvadori, "Predicting performance by university students in introductory computing courses", ACM SIGCSE Bulletin Vol. 7(3), pp. 21-29, 1975.

[13] R.E. Mayer, J.L. Dyck, and W. Vilberg, "Learning to program and learning to think: What's the connection?" Commun. of ACM Vol.29 (7), pp. 605-610, 1986.

[14] D.A. Scanlan "The mental abilities associated with programming aptitude", CSC'88 Proceedings of the ACM sixteenth annual conference on computer science, p.737, 1988.

[15] G.E. Evans and M.G. Simkin, "What best predicts computer proficiency?" Commun. of ACM Vol.32(11), pp. 1322–1327, 1989.

[16] C. Bishop-Clark and D.D. Wheeler, "The Myers-Briggs personality type and its relationship to computer programming", Journal of Research on Computing in Education, Vol. 26(3), pp. 358-370, 1994.

[17] C.C. Cegielski, J. Dianne, and D.J. Hall, "What makes a good programmer?", Commun. of the ACM, Vol. 49(10), pp. 73-75, 2006.

[18] S. Wray, "SQ Minus EQ can Predict Programming Aptitude", Proceedings of the PPIG 19th Annual Workshop, pp. 243-254, 2007.
[19] M. Tukiainen and E. Mönkkönen, "Programming aptitude testing as a prediction of learning to program", In: J. Kuljis, L. Baldwin, and R. Scoble (Eds). Proc. PPIG 14, pp. 45-57, 2002
[20] S. Dehnadi, R. Bornat "The camel has two humps", 2006, [Online], Available: http://wiki.t-o-f.info/uploads/EDM4600/The%20camel%20has%20two%20humps.pdf [Accessed: Mar.17., 2015]
[21] T. Lorenzen, H.-L. Chang, "MasterMind©: a predictor of computer programming aptitude", ACM SIGCSE Bulletin, Vol. 38 (2), pp. 69-71, 2006.
[22] University of Kent, Computer Programming Aptitude Test [Online], Available: http://www.kent.ac.uk/careers/tests/computer-test.htm [Accessed: Mar.17., 2015]
[23] Carl Jung's and Isabel Briggs Myers' typology test, [Online], Available: http://www.humanmetrics.com/hr/you/personalitytype.aspx [Accessed: Mar.17., 2015]
[24] Autism Research Centre, Cambridge, "Cambridge Personality Questionnaire" (SQ test). Available: http://www.autismresearchcentre.com/arc_tests, [Accessed: Mar.17., 2015]
[25] Autism Research Centre, Cambridge, "The Cambridge Behaviour Scale" (EQ test), Available: http://www.autismresearchcentre.com/tests/eq_test.asp, [Accessed: Mar.17., 2015]
[26] Android Apps on Google Play: Aptitude and Logical Reasoning, [Online]. Available: https://play.google.com/store/apps/details?id=com.madguy.aptitude.lr [Accessed: Mar.17., 2015]
[27] I. Gorbāns, "Programming environment for every students – Scratch or basics of programming in few hours", E-book, 2014, [Online]. Available: http://skolas.lu.lv/mod/book/view.php?id=29857 , [Accessed: Mar.17., 2015]
[28] Mentoring program of Faculty of Computing at University of Latvia, [Online]. Available: http://www.df.lu.lv/zinas/t/28406/ [Accessed: Mar.17., 2015]