UNIWERSYTET HUMANISTYCZNO-PRZYRODNICZY IM. JANA DŁUGOSZA W CZĘSTOCHOWIE
Edukacyjna Analiza Transakcyjna 2022, nr 11

http://dx.doi.org/10.16926/eat.2022.11.11

Bożena WIECZOREK
https://orcid.org/0000-0002-4506-2841
Silesian University of Technology
e-mail: bozena.wieczorek@polsl.pl

Automated assessment system as a supportive tool for teaching programming in Academic Secondary School of SUT

How to cite [jak cytować]: Wieczorek, B. (2022). Automated assessment system as a supportive tool for teaching programming in Academic Secondary School of SUT. *Edukacyjna Analiza Transakcyjna*, *11*, 193–204.

Abstract

Automated assessment systems are used in education as tools to support teaching programming and algorithms as well as preparing students for challenging informatics competitions. They reduce the teacher's workload connected with manual verification of correctness and efficiency of computer programs. Furthermore, they enable learning of computer programming to be taken outside the traditional classroom as varied in difficulty tasks can be accessed and solved at any time and in any place. Academic Secondary School of Silesian University of Technology has deployed an automated assessment system in order to support computer programming education. The aim of this work is to find out how students' skills and interests in programming affect their feelings while using Szkopuł platform. For this purpose, a survey was conducted among students, and the results were analyzed in terms of the theory of transactional analysis.

Keywords: automated assessment systems, computer programming education.

Introduction

Academic Secondary School of the Silesian University of Technology in Gliwice is a newly-established secondary school in which the Silesian University of Technology provides substantive supervision and academic lecturers teach lead-

ing subjects such as mathematics, chemistry, biology, computer science and computer programming in C++ language. The education is based on proprietary syllabuses that broaden the core curriculum of science subjects. The Academic Secondary School's priority is to develop digital competences of each student by using a technology-rich learning environment and digital tools. Teaching methods enhanced by the use of technology are particularly useful during computer programming lessons as the students have unlimited access to computers. Therefore, programming education is supported by a number of educational platforms and portals like Khan Academy, Scholaris, Google Classroom, as well as the Szkopuł[1] automated assessment system (Wieczorek, 2020).

Transactional analysis (TA) in education focuses mainly on supporting the process of effective teaching or learning, self-development and increasing self-awareness of learners and educators (Pankowska, 2012, p. 15). According to Pankowska: "In the process of education, the aim is to develop the state of the Adult, because only in this state a person is able to become fully aware, to use his or her intellectual potential and to act responsibly" (2012, p. 24). Teachers should therefore encourage students to be independent and challenge them to support problem-solving skills. Teaching of programming seems to fit perfectly into the process of the Adult ego state development. Moreover, solving tasks on the automated assessment system appears to be a good approach to motivate students to develop programming skills and find solutions.

Some social research indicates that the language we use influences how we think, behave and perceive the world (Słaboń, 2001). This phenomenon is presented in the Sapir-Whorf hypothesis. Some of the assumptions of linguistic relativity have been confirmed by linguistic and non-linguistic tests conducted by researchers. Learning a programming language is similar in some ways to learning a foreign language. Novice programmers learn the syntax and semantics of a language to write simple instructions and then programs that solve more complex tasks. Students review code written by more experienced programmers, and try to understand how each line and the entire program works, just like learners who read texts in a foreign language trying to understand the meaning of individual sentences as well as the sense of the whole text. Finally, in learning a programming and a foreign language, practice is the most important thing. The inspiration for conducting the research was the theory of linguistic determinism, however, the topic proved to be too ambitious. Analyzing the answers to the survey questions, the author will try to find out whether how students participating in research react to automated grading systems depends on their experience in programming.

[1] https://szkopul.edu.pl

Issues and challenges in teaching computer programming

Teaching of computer programming creates significant challenges for educators (McAllister, Alexander, 2009). Difficulties in imparting knowledge in this field are related to the nature of the skills students need to develop. The core of programming consists in solving problems and creating programs as solutions. Students need to analyze the task, produce an algorithmic solution, and translate that algorithm into a program code. There is also another type of programming knowledge called program comprehension (Mannila, 2007). Learners are asked to demonstrate an understanding of how a specific program works. It is considered important to teach both skills: program generation and program comprehension.

Becoming well versed in programming is time-intensive and requires a lot of practical exercises. Students cannot be passive recipients of knowledge conveyed by the teacher, because during each class they have to put what they learned into practice. Moreover, it is crucial for the teacher to find propriate methods to support students in developing computational thinking skills[2] essential for implementing computer programs. Encouraging students to gain more practical experience during after school activities is also important.

Another teaching difficulty is that learners present varying levels of abilities and interest in programming. It was observed[3] that passionate learners need challenging problems to solve in order to feel motivated. They predominantly enjoy participating in programming competitions, so they spend much time working on difficult individual projects. Students with a slower pace of learning or less able ones favor to study with a progressive worksheet to build up their self-confidence.

An additional issue for teachers is analysis and evaluation of source codes developed by students. As learners are expected to write a large number of programs, it is difficult to assess and comment on all of them. Solutions of simple tasks are usually short and therefore easy to test. However, if the evaluation of more complicated programs is done manually, it is extremely time-consuming, tedious and error-prone.

Automated assessment platform

Szkopuł is the e-learning platform for carrying out algorithmic contests with automatic assessment of user-submitted programs. It gives access to a vast number of problems to solve and an archive of tasks from various programming

[2] Problem solving and algorithmic thinking skills (decomposition, pattern recognition, pattern abstraction) are crucial competences related to computational thinking.

[3] The teacher observation during four years of teaching in the school.

competitions, including: Olympiad in Informatics, Junior Olympiad in Informatics and several international Olympiads (Central European OI, Baltic OI and International OI). Submitted programs are compiled and run on host machine against a set of tests prepared by the authors of the task. Each test consists of an input file, an output file, a timeout, and a memory limit. The content of the input file is passed to the standard input of the program. If the program ends correctly, execution does not timeout and memory limit is satisfied, its output is compared to the contents of the test output file and if they are the same, points are scored. Partial scoring, depending on the number of test cases passed, is supported and user's solution is graded on a scale from 0 to 100 points. Each contest has its rank list showing the number of points scored by competitors. It is planned to fortified Szkopuł platform with programming and algorithms courses to enable users to improve their skills in these areas[4].

Academic Secondary School started using Szkopuł platform after joining the Algorithmics and Programming Championship project (MAP – Mistrzostwa w Algorytmice i Programowaniu). The project is run by the Foundation for Information Technology Development in cooperation with two Polish universities: University of Warsaw and University of Wrocław. The aim of the project is to support the development of secondary school learners who are passionate about programming and algorithms in order to make them capable of competing in programming contests on national and international level. Teachers participating in the project run extra-curricular classes to activate particularly talented students. They can be assisted by mentors[5] or take part in algorithmic workshops to increase competences in this field. Original curricula as well as methodical support are proposed to teachers together with Szkopuł platform where a large collection of algorithmic problems is shared.

During classes, all the students learn instructions and data types in C++ programming language and develop their knowledge with presented algorithms or design paradigms. Knowing the theory, they start writing source codes to solve selected problems on Szkopuł platform (usually after one year of study). The only feedback each student receives after submitting the solution is the information about the execution time, correct or wrong answer on each test, timeout or memory limit overflow and the number of points the program scored (Figure 1). In case there was a wrong answer on one of the inputs, the correct answer is given.

Students of Academic Secondary School, who are particularly interested in algorithmics and programming, attend the MAP circle as an after school activity. A special contest is created on Szkopuł platform where the tasks require students to apply specific, optimal algorithms.

[4] The courses from main2.edu.pl service are planned to be transferred to Szkopuł platform.

[5] Computer science students who were finalists of the Olympiad in Informatics.

The survey and its results

In order to identify factors that learners like and dislike about Szkopuł and discuss the strengths and weaknesses of the platform, a survey was conducted among 60 students of Academic Secondary School who have completed at least their first year of programming. Among the participants, 14 attended MAP circle classes. For the purpose of further discussion, the group of students attending MAP circle will be referred to as the MAP group, and the other students as the School group. The survey included questions about programming skills and experience (questions 1 and 2), students' feelings while working with Szkopuł (questions 3, 4 and 5), and suggestions for improving the platform to suit students' needs (question 6).

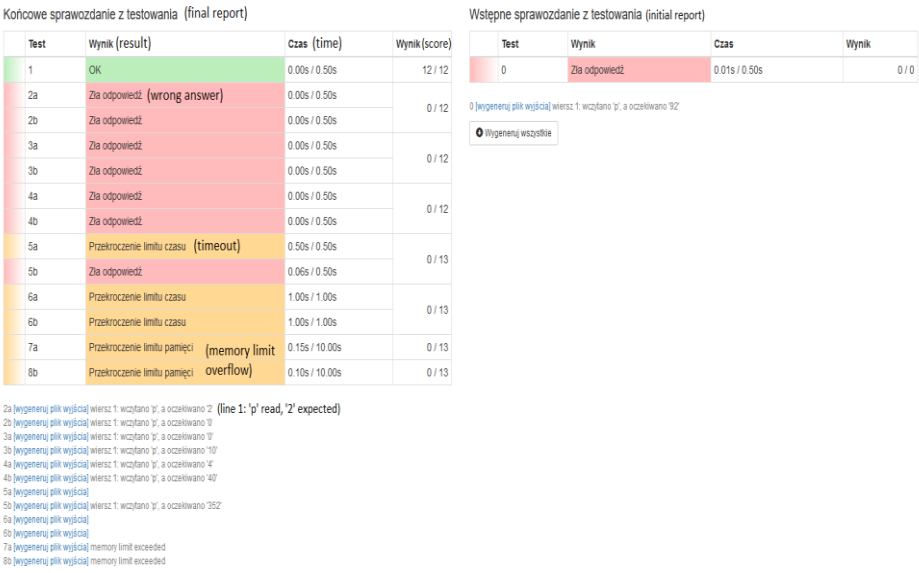

Figure 1
Example feedback of the assessed solution

Source: report of the author's submission.

Question 1 and 2: How long have you been programming? How do you rate your programming skills?

In the School group, 26 students out of 46 consider themselves novice programmers. As many as 35% of them have been learning programming for three or four years, which suggests that this period of time was not enough for them to acquire the appropriate skills. Among students who declared themselves to be intermediate programmers (17 out of 46), 88% needed three years or less to

feel comfortable while working on programming tasks. Only 3 students consider themselves experts.

The ratio between expert and intermediate programmers in the MAP group is completely different. Nearly 86% of the students code at the most advanced level, and it took them an average of 3 years to achieve that level.

Question 3: Your feelings while working on the platform.

Answering this question, students had to choose one from six options:

1. I don't like working on the platform. The tasks are too difficult and I can't solve them.
2. I'm not fond of working on the platform. The tasks are difficult and I have to spend a lot of time solving them. Sometimes the results are not satisfactory.
3. Solving tasks on the platform is a good idea, but sometimes the tasks are too difficult.
4. I like to work with Szkopuł platform – I can immediately check if my solution is good enough.
5. I like working on Szkopuł platform – I can check the accuracy of my solution and compare the results I achieved with my classmates.
6. Working on the platform is a great idea. It motivates me to search for optimal solutions.

The answers for the School and the MAP group are shown in Table 1 and Table 2, respectively.

Table 1
Responses of the School group to question number 3

	Option 1	**Option 2**	**Option 3**	**Option 4**	**Option 5**	**Option 6**
Novice (26 answers)	2	6	11	4	2	1
Intermediate (17 answers)	0	5	3	4	2	3
Expert (3 answers)	0	1	0	2	0	0

Source: own research.

Table 2
Responses of the MAP group to question number 3

	Option 1	Option 2	Option 3	Option 4	Option 5	Option 6
Intermediate (2 answers)	0	0	0	2	0	0
Expert (12 answers)	0	0	2	3	2	5

Source: own research.

As we can see, for most novice programmers in the School group the tasks on the platform are too difficult (options 1, 2 and 3). Only 7 out of 26 students enjoy solving tasks (options 3, 4 and 5) mainly because they can quickly check

their solutions. Among experienced and intermediate participants, about 50% think that working on the platform is a good idea. However, only 3 out of 20 are motivated to search for better algorithms (option 6).

In the MAP group, nearly 86% of participants like working on the Szkopuł platform, and 5 out of 14 feel motivated to perform well and spend more and more time seeking optimal solutions.

Question 4: What are you dissatisfied with while working on the Szkopuł platform?

Answering this multiple choice question, students could choose from one to seven options:

1. There is no such thing.
2. Programming task that is difficult to understand.
3. Input/output that must have a specific format.
4. Lack of comprehensive information about the error in my code.
5. No information on the algorithm that should be used to solve the task at 100 points.
6. Lack of hints to help improve my solution.
7. Lack of access to the test cases which assess my solution.

The answers for the School and the MAP group are shown in Figures 2 and 3, respectively.

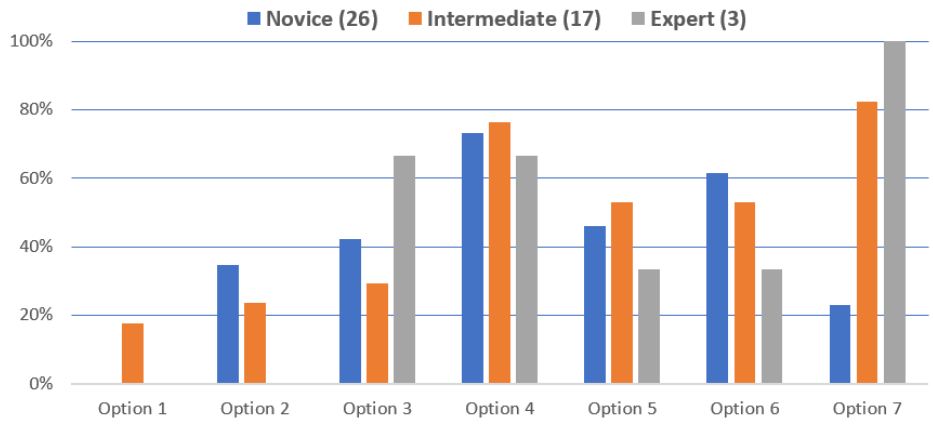

Figure 2
Responses of the School group to question number 4

Source: own research.

As shown on graphs, the School group students marked significantly more options than programmers from the MAP group. In the first group, each participant selected an average of 3 options, compared to 1.6 in the second group.

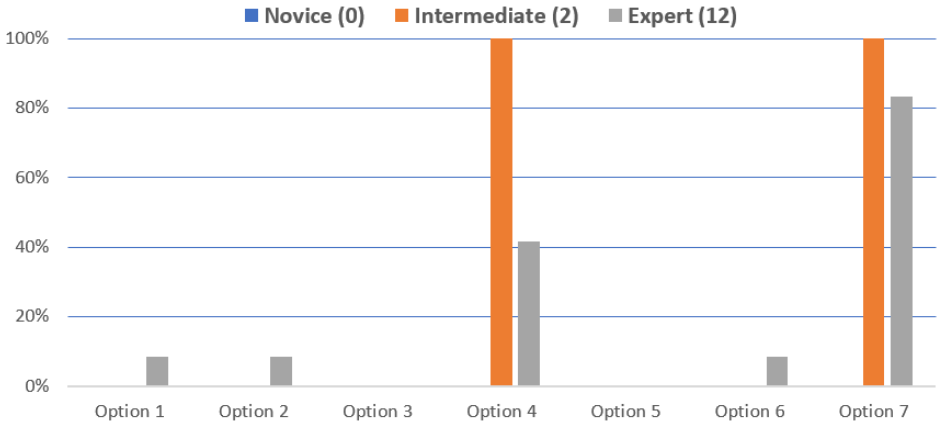

Figure 3
Responses of the MAP group to question number 4

Source: own research.

Most novice programmers consider the lack of error information in their codes and the lack of hints to improve the solutions to be a major drawbacks of Szkopuł platform. It seems that these students need positive support from the teacher. They need strokes[6] to direct them towards good solutions, or to confirm that they are on the way to finding them. Dorota Pankowska noted that: "The strokes properly given to students […] not only strengthen their self-esteem, but also influence the process and effects of education" (Pankowska, 2010, p. 178). For novice programmers, such positive encouragement seems essential to build their confidence in learning programming.

Among the more experienced programmers, the majority dislike the lack of access to test cases. This option was chosen by more than 80% of intermediate participants and 100% of experts in the School group and 100% of intermediate participants and more than 80% of experts in the MAP group. These participants in most cases do not need additional teacher support. They try to find optimal solutions on their own, and access to test cases would help them correct errors in their codes.

Question 5: What do you like about working on the Szkopuł platform?
Participants could choose from one to seven options:

1. There is no such thing.
2. Interesting programming tasks.
3. Difficulty of tasks – working on the solution is time-consuming, but it develops my skills in programming and algorithmics.
4. Specific format of input data and results.

[6] Strokes as an element of interpersonal communication are in the field of TA interest.

5. Rapid assessment of my solution.
6. The ranking, which allows me to see how I'm doing compared to other students.
7. Tasks – solving them gives me satisfaction.

The graphs in Figure 4 and Figure 5 show the responses of the School group and the MAP group, respectively.

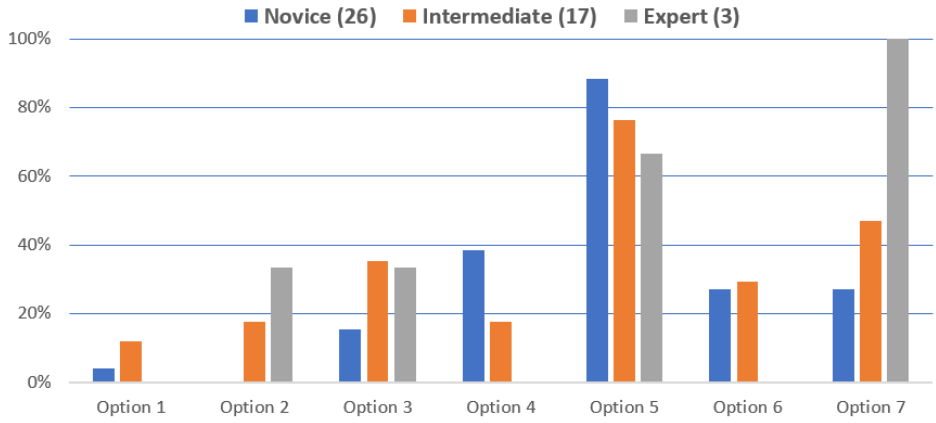

Figure 4
Responses of the School group to question number 5

Source: own research.

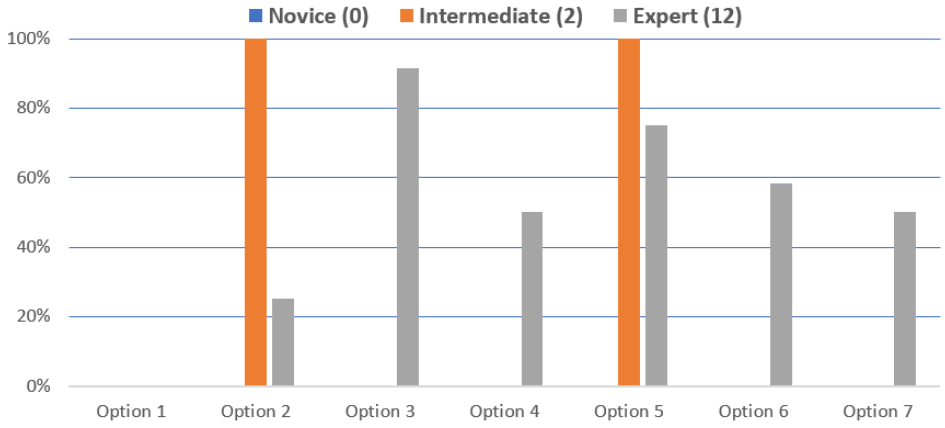

Figure 5
Responses of the MAP group to question number 5

Source: own research.

In response to this question, the Map group students have selected more options on average than participants from the School group (3.3 and 2.2 options, respectively). 38 out of 46 students in the School group enjoy the speed of automated assessment on the platform. For more experienced programmers, the satisfaction from solving tasks on Szkopuł is important (all experts and nearly half of the intermediate participants).

In the Map group, the rapid evaluation of the solution was also a frequently chosen option (nearly 79% of students). Almost 92% of the experts in this group, however, believe that the difficulty of the tasks and the experience they gain while solving them is a great advantage of the platform. Most of them like the ranking (58%), the specific format of I/O data (50%) and they feel satisfaction from solving tasks (50%).

Question 6: How do you think the Szkopuł platform could be improved?

The answers to this open-text question were significantly different in both groups. In the School group, 37 out of 46 participants suggested improvements to the platform. 73% of them indicated that the platform should give students some guidance how to correct errors and some hints that would guide users towards better solutions. According to some participants, programming tasks should be more easily understood and the test cases should be accessible. There were also three answers in which students identified a need for access to reference solutions.

In the Map group, 72% of students suggested improvements to Szkopuł. The most common proposal was to make test cases available on the platform, which would help correct errors in the program or improve the solution. Two participants suggested that the programming tasks should include more examples of input data and the results that should be returned. The need to know the reference solution was brought to the attention of one individual.

Conclusion

Students who attend the MAP circle classes have more coding practice, as they write computer programs during programming lessons, during circle activities and at home. They evaluate their skills highly considering themselves as experts, though some have been programming in C++ for a year or two (3 students). In the School group, students needed four or more years of programming to identify themselves as advanced programmers.

Similar differences can be observed in the students' feelings while working on Szkopuł. The majority of the School group participants do not like to work on the platform because they consider the tasks too difficult. They need some guidance how to correct implemented algorithms and. In the MAP group, nearly 86%

of participants enjoy working on the Szkopuł platform, and 5 out of 14 feel motivated to perform well and spend more and more time seeking optimal solutions. Moreover, almost 92% of the experts in this group, believe that the difficulty of the tasks and the experience they gain while solving them is a great advantage of the platform.

We can say that Adult ego state of programmers in the MAP group is better developed. These students perceive the tasks on the platform as the challenges they want to face. Students with less practice in programming and algorithmics need teacher guidance and positive strokes. It can be concluded that learners at this stage of programming skills should work on Szkopuł platform under the supervision of an educator who will support them. Analysis of student responses shows that they need feedback and comments from teachers. Transactional analysis theory emphasizes the need for recognition signs. Despite its usefulness, the automated assessment system at this stage of development cannot yet replace the teacher-student relationship.

References

Mannila, L. (2007). Novices' progress in introductory programming courses. *Informatics in Education*, *6*, 139–152.

McAllister, G., Alexander, S. (2009). Key aspects of teaching and learning in computer science. In: H. Fry, S. Ketteridge, S. Marshall (ed.), *A handbook for teaching and learning in higher education* (pp. 282–299). New York: Routledge.

Pankowska, D. (2010). *Nauczyciel w perspektywie analizy transakcyjnej*. Lublin: Wydawnictwo Uniwersytetu Marii Curie-Skłodowskiej.

Pankowska, D. (2012). Analiza transakcyjna w edukacji czy edukacyjna analiza transakcyjna? – próba porządkowania znaczeń*. Edukacyjna Analiza Transakcyjna, 1*, 13–30.

Wieczorek, B. (2020). IT teaching in Academic Secondary School of the Silesian University of Technology. In: B. Pituła. A. Waligóra (ed.), *Modern aspects of teaching at technical university* (p. 11–26). Toruń: Wydawnictwo Adam Marszałek.

Słaboń A. (2001). Hipoteza Sapira-Whorfa. In: M. Pacholski, A. Słaboń, *Słownik pojęć socjologicznych* (p. 58). Kraków.

System automatycznej oceny jako narzędzie wspierające nauczanie programowania w Akademickim Liceum Ogólnokształcącym SUT

Streszczenie

Systemy oceny automatycznej są używane w edukacji jako narzędzia wspierające nauczanie programowania i algorytmów oraz przygotowanie uczniów do trudnych konkursów informatycznych. Redukują one obciążenie nauczyciela związane z ręczną weryfikacją poprawności i wydajności programów komputerowych. Ponadto umożliwiają naukę programowania poza tradycyjną klasą, ponieważ zróżnicowane zadania o różnym stopniu trudności mogą być rozwiązane w dowolnym czasie i miejscu. Akademickie Liceum Politechniki Śląskiej wdrożyło system oceny automatycznej w celu wsparcia edukacji programowania. Celem tej pracy jest ustalenie, jak umiejętności i zainteresowania uczniów programowaniem wpływają na ich odczucia podczas korzystania z platformy Szkopuł. W tym celu przeprowadzono ankietę wśród uczniów, a wyniki zostały przeanalizowane pod kątem teorii analizy transakcyjnej.

Słowa kluczowe: systemy oceny automatycznej, edukacja programowania, analiza transakcyjna.