HIGH SCHOOL STUDENTS` COMMON ERRORS IN PROGRAMMING

Davorka Radaković[1]*,
William Steingartner[2]

[1]University of Novi Sad,
Faculty of Sciences,
Novi Sad, Serbia

[2]Technical University of Košice,
Faculty of Electrical Engineering
and Informatics,
Košice, Slovakia

Abstract:

Identifying and classifying the commonness of errors made by novices learning to write computer programs has long been of interest to both: researchers and educators. Teachers understand the nature of these errors and how students act to correct them, hence more efficient teaching can be performed. Some errors are more frequent than others.

In this paper, we examine the most common errors in novice programming of first-year gifted mathematicians in Mathematical Grammar School. Regardless of extensive coverage of these types of errors during the lectures and in learning material, we have noticed that these still persevere when students write programs. Our findings imply that students who usually make all common mistakes have lower marks, but excellent students also make logical errors in conditions for loops. Therefore, we advise more practice in logical thinking with novice programmers and an introduction to formal semantics.

Keywords:

Computer Science Education, C# Language, Errors, Novice Programmers, Programming.

INTRODUCTION

In today's world, computer science education has an important part of the STEM curriculum. Students who are early exposed to STEM content commonly extend their interest in STEM subjects through elementary and high school up to the faculty level [1] [2]. Computer science education provides an abundance of new learning concepts and opportunities crosswise domains. Computational thinking in education prepares today`s school-aged students to live and work in an entirely digitized world [3].

Learning programming is sometimes very difficult for first-year high school students. Therefore, students produce significant errors in their code when they confront difficulties in learning programming [4] [5]. Analysis of frequent student errors is necessary for computer science professors to understand students' problems in learning programming. Accordingly, a computer science professor needs to obtain expertise in the subject matter and pedagogical knowledge of teaching the subject`s content.

Correspondence:
Davorka Radaković

e-mail:
davorkar@dmi.uns.ac.rs

DOI: 10.15308/Sinteza-2022-104-108

Due to the global COVID-19 pandemic, in the last two years, education has been implemented mainly in the home environment. Face-to-face teaching in schools was substituted with teaching online, using online platforms as new virtual classrooms. This pandemic brought remarkable disruption to education, as well as to High School Education, locally and all over the world [6] [7] [8] [9] [10] [11].

In this paper, the data collected during an introductory C# programming course in the first year was used to identify which errors are often made by Mathematical Grammar School scholars in two generations. One generation attended classes before the COVID-19 pandemic and the second generation attended classes during the COVID-19 pandemic. We have considered all different kinds of errors that were conducted to an incorrect solution. Further, we investigated connections between scholars' error-related behaviors and their achievements in C# introductory programming course.

Our work is innovative in several criteria. First, the population for the course is first-year Mathematical Grammar School gifted mathematicians scholars. Second, there was an inconsistency between the errors identified in papers by the researchers and those errors experienced by the high school scholars.

The paper is organized as follows: The next section gives an overview of related work. Common programming errors with methodology and results are shown in Section 3. The last section concludes the paper.

2. RELATED WORK

Ko and Myers in [12] gave the categorizations of programming errors linking the causes of errors. Identifying and helping to correct Java programming errors for Introductory Computer Science students using the education tool, Expresso, made for Java programming, is presented in [13]. This interactive tool generates error messages and additionally provides instructions on how to fix the code. The main purpose is to be used all along the beginning process of learning programming and for students to become more skilled with Java and gain a better comprehension of the essential programming concepts.

In 2005, [14] presents an integrated semantic and syntax error pre-processing system to benefit new programmers unravel the otherwise cryptic compiler error messages for them to concentrate more on design issues than implementation issues used in the United States Military Academy taken for an introductory programming course.

In [15], the authors investigated the types of errors most frequently accomplished by students practicing writing short fragments of Java code using the CodeWrite practice tool. Further, it was examined how long students spent resolving the most common syntax errors and discovered that certain types of errors are not solved any more quickly by the higher ability students. Furthermore, it was noticed that these errors waste a large amount of student time, advancing that targeted teaching interference may yield a significant outcome in terms of increasing student productivity.

The diagnostic message frequencies in [5] show a relatively high occurrence of "cannot find symbol", "';' expected", "')' expected" and "illegal start of expression" diagnostics. The results are similar to the study [14].

One recent study [4] implemented a data-driven approach to identify Chinese high school students' common errors in a Java-based introductory programming course using the data that was collected in an automated assessment tool "Mulberry". Students' error related behaviours were further analyzed, and their relationships to success in introductory programming were explored. The study suggests that students' competence in improving code is important to their accomplishment in introductory programming.

Ettles, Luxton-Reilly and Denny analysed 15000 code fragments [16], created by novice programming students that contain logic errors. They classify the errors as algorithmic errors, misinterpretations of the problem, and fundamental misconceptions. Additionally, they identified that misunderstanding is the most frequent source of logic errors and leads to the most complicated errors for students to fix.

In [17] proposed, a method for the categorization of frequent errors in solution codes. The authors used paired source codes (incorrect - accepted) for these experiments. The longest common subsequence (LCS) algorithm is influenced to find the differences between wrong and accepted codes.

Due to the COVID-19 pandemic, many studies confirm the learning loss caused by schools closures and online learning [6] [7] [10] [18] [19] [20]. Besides this confirmation, studies also propose solutions to make up for the missed knowledge: consolidation of the curriculum, extending instructional time or improving the competencies of learning by supporting teachers to apply structured pedagogy and targeted instruction.

3. METHODOLOGY AND RESULTS

In this section, we describe the design of our study aimed to identify which errors are often made by first-year Mathematical Grammar School students compared with two generations, pre- and in- COVID-19 pandemic. At the end of this section, we focus on the obtained results and observations.

3.1. CONTEXT

We analyse the data that was collected during a first-year programming course for gifted mathematicians in Mathematical Grammar School "Jovan Jovanović Zmaj", Novi Sad, Serbia. The introductory C# programming classes were held five school hours per week. The major topics examined in this course are program structure, input/output, variables and operators, conditionals, and loops.

Two separate collections were performed. One started before the COVID-19 pandemic in the school year 2019/2020 and another entire during the pandemic in 2020/2021. The first group had 17 students, and the second had 19 students. Some students finished seventh and eighth grade in the Mathematical Grammar School. So, they had prior knowledge of programming.

Both sets of data collection are collected from students' written exams. Thus, students solved their tasks by writing on the paper without possibility to compile the program. Thus, students did not have the help of the compiler to warn them about mistakes, as is usually done in other studies [5] [12] [13] [15] [21]. Moreover, all solutions were inspected manually.

We analyzed the solutions obtained from the first three written exams.

3.2. CATEGORIES OF PROGRAMMING ERRORS

Our review of the literature demonstrated that researchers used a diversity of methods to identify common errors. Commonly accepted categorizations of errors are lexical, syntactic, semantic, and logic errors. Besides, these errors can be either static (compile-time) or dynamic (run-time) in nature. Nevertheless, the types of errors highly depend on the exercise [13].

Logical errors are the hardest of all error types to detect. Occasionally, referred to as semantic errors, there are situations where the programmer's code compiles successfully and executes but does not generate the proposed output for all possible inputs [16].

3.3. LIST OF COMMON HIGH STUDENT ERRORS

The following errors were identified in our research made by both groups of students:

- The comparison operator (==) vs the assignment operator (=);
- Unbalanced:
- parentheses ('(', ')');
- square brackets ('[', ']'),
- curly brackets ('{', '}'); and
- quotation marks;
- Inserting a semi-colon after the parentheses defining *if, for, foreach*, or *while* conditions;
- Separating the *for* loops with commas (',') instead of semi-colons (';');
- The equal sign in front of :
- the greater-than sign (or) ('=>') for a greater than or equal; and
- the (or less than or equal) ('=<'), instead of following them.
- Improper casting, i.e. missing of the cast; and
- Logical errors in loop conditions.

Also, we have noticed that students who write their code neat mostly do not have unbalanced errors. Another common error (up to 10%) was in conditions in the if-else statement are wrong due to the substitution of the comparison operator with the assignment operator. More, inserting a semi-colon before *if, for, foreach*, or *while* the block was common among 10% of students.

Surprisingly, the most common mistakes were logical errors in loop conditions with 30% of students` exams in first evaluations, notwithstanding the pervasive coverage of these types of errors during the lectures.

No significant difference was noticed between students' error making and obtained marks between face-to-face lecturers and online, unlike the reports given in [18] [19] which observed that in core subjects, like math and reading, there are alarming signs that in some grades students might be falling even further behind pre-pandemic expectations. Also, the average learning loss graded by the length of the school closure during the pandemic was presented in [20]. We could consider the main reasons for having the same students` accomplishments before and during the pandemic. Teaching was conducted without interruption online as face-to-face, with the same hours; and the groups are rather small, up to 19 students.

3.4. STUDENT ERRORS AND MARKS IN THE COURSE

Introductory programming is usually difficult for high school students. However, our students are gifted and have academic performance in math and science as one of the most important predictors of their achievement in introductory programming courses [4]. Although the observed groups of students have passed an additional exam for entering the course for gifted mathematicians, some students in high school discontinued their excellent mastering of subjects. Consequently, students with many errors in their exams had pure marks.

On the other hand, it is noticed that up to 20% of students with very good and excellent marks make logical errors in loop conditions. Hence, logical thinking should be more practiced during the lessons.

4. CONCLUSION AND FUTURE WORK

This paper has analysed high school students' solutions to programming exercises in an introductory C# course. We have identified the most common errors high school students made and given their categorization.

We can agree with [22], that the deficiencies in students' strategic knowledge are one of the main reasons programming is a challenge for students.

Students in the second group were followed in their second year, and only students with pure marks still made logical errors in loop conditions. Given the current state of affairs, our recommendation for expanding programming tuition would be to introduce a theoretical introduction to formal semantics. Students would thus become acquainted with the theoretical features of languages, which would help them realize the important features at the level of syntax and its connection to the resulting semantics and thus avoid several mistakes in writing programs [23] [24].

For future work, we could additionally consider code and "algorithmic" smells to instruct students not only to write correct but also adequate and proper programs. The result of our research is significant for teachers and researchers wishing to advance programming pedagogy.

5. ACKNOWLEDGEMENTS

This work was supported by project KEGA 011TUKE-4/2020: "A development of the new semantic technologies in educating of young IT experts", granted by the Cultural and Education Grant Agency of the Slovak Ministry of Education. The international cooperation is grounded and supported by CEEPUS network CIII-CZ-1503-02-2122 - Development of Computational Thinking.

6. REFERENCES

[1] H. B. Gonzalez and J. J. Kuenzi, "Science, Technology, Engineering, and Mathematics (STEM) Education: A Primer," 1 August 2012. [Online]. Available: https://fas.org/sgp/crs/misc/R42642.pdf. [Accessed 1 April 2022].

[2] N. DeJarnette, "America's children: providing early exposure to stem (science, technology, engineering and math) Initiatives," *Education*, vol. 133, no. 1, p. 77–84, 2012.

[3] "TeachEngineering STEM Curriculum for K-12," [Online]. Available: https://www.teachengineering.org/curriculum/browse. [Accessed 1 April 2022].

[4] Y. Qian and J. Lehman, "An Investigation of High School Students' Errors in Introductory Programming: A Data-Driven Approach," *Journal of Educational Computing Research*, vol. 58, no. 5, pp. 919-945, 2020.

[5] D. McCall and M. Kölling, "A new look at novice programmer errors," *ACM Transactions on Computational Education*, vol. 19, no. 4, p. 1–38, 2019.

[6] M. Bakator and D. Radosav, "Managing Education in the COVID-19 era," in *International Conference on Information Technology and Development of Education – ITRO 2020*, 2020.

[7] S. Pokhrel and R. Chhetri, "A Literature Review on Impact of COVID-19 Pandemic on Teaching and Learning," *Higher Education for the Future*, pp. 133-141, 2021.

[8] V. Vilić, "Cyber Security and Privacy Protection During Coronavirus Pandemic" in *Sinteza2021: International scientific conference on information technology and data related research*, Belgrade, Serbia, 2021.

[9] W. Steingartner, M. Jankura and D. Radaković, "Visualization of Formal Semantics - Possibilities of Attracting Formal Methods in Teaching" in *Sinteza2021: International scientific conference on information technology and data related research*, Belgrade, Serbia, 2021.

[10] S. Meinck, J. Fraillon and R. Strietholt, The impact of the COVID-19 pandemic on education: international evidence from the Responses to Educational Disruption Survey (REDS), Paris, France; Stichting IEA Secretariaat Nederland; Amsterdam Netherlands: UNESCO / International Association for the Evaluation of Educational Achievement (IEA) 2022, 2022.

[11] K. A. Godber and D. R. Atkins, "COVID-19 Impacts on Teaching and Learning: A Collaborative Autoethnography by Two Higher Education Lecturers," *Frontiers in Education*, vol. 6, 2021.

[12] A. J. Ko and B. A. Myers, "Development and Evaluation of a Model of Programming Errors," in *IEEE 2003 Symposia on Human Centric Computing Languages and Environments (HCC'03)*, Auckland, New Zealand, 2003.

[13] M. Hristova, A. Misra, M. Rutter and R. Mercuri, "Identifying and correcting Java programming errors for introductory computer science students," in *34th SIGCSE technical symposium on Computer science education (SIGCSE '03)*, New York, NY, USA, 2003.

[14] J. Jackson, M. Cobb and C. Carver, "Identifying Top Java Errors for Novice Programmers," in *Frontiers in Education 35th Annual Conference*, Indianapolis, USA, 2005.

[15] P. Denny, A. Luxton-Reill and E. Tempero, "All syntax errors are not equal," in *17th ACM annual conference on Innovation and technology in computer science education (ITiCSE '12)*, Haifa, Israel, 2012.

[16] A. Ettles, A. Luxton-Reilly and P. Denny, "Common logic errors made by novice programmers," in *20th Australasian Computing Education Conference (ACE '18)*, New York, NY, USA, 2018.

[17] M. M. Rahman, S. Kawabayashi and Y. Watanobe, "Categorization of Frequent Errors in Solution Codes Created by Novice Programmers," in *3rd ETLTC International Conference on Information and Communications Technology (ETLTC2021)*.

[18] S. G. Jaramillo, "COVID-19 Policy Documents Series - Proposals of solutions for the crisis "COVID-19 and primary and secondary education: the impact of the crisis and public policy implications for Latin America and the Caribbean","UNDP In Latin America and the Caribbean", 9 October 2020. [Online]. Available: https://www.latinamerica.undp.org/content/rblac/en/home/library/crisis_prevention_and_recovery/covid-19-y-educacion-primaria-y-secundaria--repercusiones-de-la-.html. [Accessed 1 April 2022].

[19] U.S. Department of Education, "Education in a Pandemic: The Disparate Impacts of COVID-19 on America's Students," 2021. [Online]. Available: https://www2.ed.gov/about/offices/list/ocr/docs/20210608-impacts-of-covid19.pdf. [Accessed 1 April 2022].

[20] "The global education crisis – even more severe than previously estimated," 4 January 2022. [Online]. Available: https://blogs.worldbank.org/education/global-education-crisis-even-more-severe-previously-estimated. [Accessed 1 April 2022].

[21] C. Lacave and A. I. Molina, "The Impact of COVID-19 in Collaborative Programming. Understanding the Needs of Undergraduate Computer Science Students," *Electronics*, vol. 10, no. 14, p. 1728, 2021.

[22] E. Albrecht and J. Grabowski, "Sometimes It's Just Sloppiness - Studying Students' Programming Errors and Misconceptions," in *51st ACM Technical Symposium on Computer Science Education*, New York, NY, USA, 2020.

[23] W. Steingartner, J. Perháč and A. Biliňski, "A Visualizing Tool for Graduate Course: Semantics of Programming Languages," *IPSI BgD Transactions on Internet Research*, vol. 15, no. 2, pp. 52-58, 2019.

[24] W. Schreiner, "Logic and Semantic Technologies for Computer Science Education," in *Informatics'2019, 2019 IEEE 15th International Scientific Conference on Informatics*, Poprad, Slovakia, 2019.