A Look at First-Year Students' Pre-Knowledge on Computer Programming Over Time

Tomi Perša[1, *, †], Lili Nemec Zlatolas[1, †]

[1] University of Maribor, Faculty of Electrical Engineering and Computer Science, Koroska ulica 46, 2000 Maribor, Slovenia

Abstract

This study investigates the pre-existing knowledge of first-year students in Computer Programming courses, analysing the impact of secondary education backgrounds on their learning experiences. Over the course of five academic years, students at the University of Maribor were assessed through surveys and practical tests to determine their pre-knowledge of algorithmic thinking and problem-solving skills. The research highlights the influence of learning opportunities and the subsequent rise in students with prior programming experience. The findings present differences in self-reported programming knowledge across generations, emphasising the need for adaptive teaching strategies to bridge the knowledge gap. The results show the importance of early programming education and suggest a trend towards digital literacy in secondary education.

Keywords

Programming, teaching, JavaScript, learning, pre-knowledge

1. Introduction

An essential skill for an IT professional is learning the principles and logical thinking behind Computer Programming. Teaching it is a challenge for all kinds of students, as the algorithmic, literal, and logical way of thinking is not something we would use in everyday life. Conquering these hurdles is a challenge for novice and beginner students, as the explanations and live examples may not be enough for them to fully understand the concepts and procedures regarding Computer Programming.

University programs usually expect next to little knowledge in ICT-related subjects since these programs are meant to be for students from gymnasiums who have acquired learning potential but have not yet acquired their career path skills beforehand. Such courses aim to teach algorithmic and logical thinking while also providing declarative knowledge (basic understanding of programming constructs and language syntax) and procedural skills (appliance of declarative skills for problem-solving) [2]. Such skills are mostly achieved through practical examples and problem-solving, which in turn reinforces further knowledge, thus creating a cyclic process of learning. While introductory, such examples are aimed at learning the fundamentals, yet are sometimes circumvented by the use of generative AI or copying code without a proper understanding of its functionality [5]. These present a challenge, as such solutions show the ability of problem-solving but do not prove independence or proof of logical and algorithmic thinking ability. Much of the pre-knowledge and further programming learning is connected with a mathematical background, especially in the fields of logic and task-solving assignments [6]. Only through proper testing and grading can we see if the subject has acquired proper skills to work independently on their programming projects [5].

SQAMIA 2024: Workshop on Software Quality, Analysis, Monitoring, Improvement, and Applications, September 9--11, 2024, Novi Sad, Serbia

* Corresponding author.

† These authors contributed equally.

tomi.persa@um.si (T. Perša); lili.nemeczlatolas@um.si (L. Nemec Zlatolas)

0000-0002-2086-9825 (L. Nemec Zlatolas)

Students who have previous experience in programming are usually more self-assured in their knowledge, which often misleads their actual capabilities and thus sometimes perform worse than students who take their time to properly study the required constructs and concepts [7]. They have the capabilities required to conduct proper declarative knowledge and procedural skills but are more prone to mistakes. Subjects also don't approach better, potentially superior concepts, thus staying at the skill level which they are most comfortable with, often leading to false understandings of more advanced concepts and solutions [2]. There is also a matter of transitioning between different scripting languages, which results in basic syntax errors and incorrect answers due to the differences in syntax and structure between the languages they've learned previously and the ones they are currently using.

During the COVID pandemic, much of the educational process has moved online. Theoretical and practical classes in Computer Programming became more focused on providing solutions that would assist students in their learning process while also incorporating different online tools for providing feedback [8]. This meant an increased source of learning materials that would direct a beginner programmer to proper problem-solving and critical thinking [6]. Solution-seeking is time-consuming, but it can cater to subjects' learning capabilities and willingness to put time and effort into the acquirement of new skills [8]. Due to the abundance of accessible online materials, students who seek solutions for their assignments offline often find results that are out of their skill range, further distancing themselves from easier solutions and the course curriculum [6]. Certain students also prefer live or in-person examples of programming, giving them a more direct approach and quicker access to feedback should they hit an error during their assignments [9].

In this work, we present a study where we analysed answers from students with different secondary education level backgrounds through the years on their different levels of pre-knowledge of Computer Programming, algorithmic thinking, problem-solving, and their self-perceived knowledge of the subject. The analysis was performed with active students at the start of the course and after they had attended the course's theoretical lessons and solved practical laboratory assignments. The experiment was conducted with students who took the course Fundamentals of Web Programming, which was later renamed to Programming for Media.

The structure of the paper is as follows. The description of the research methods is provided in Section 2 and the main contribution of the paper is in Section 3, where the results are presented. Finally, the discussion is in Section 4, and the conclusion of the paper is in Section 5.

2. Research methods

The study evaluated the pre-knowledge of the students in the Fundamentals of Web Programming course using a test. Further details are presented in the following sub-sections.

2.1. Data collection and participants

At the start of the semester, the students attending the course Fundamentals of Web Programming were allowed to fill out a survey that would assess their skills and knowledge of Computer Programming in any programming language. This study was done in 5 consecutive years (from 2019/2020 to 2023/2024), and we will compare the results of generations. The study was done among students of Media Communications (MC) at the Faculty of Electrical Engineering and Computer Science, University of Maribor. In the first three years of research, the course was also an elective course in the study program Information and Communication Technologies (IaCT). As presented in Table 1, there were a total of 425 participants over five consecutive years.

2.2. Measures

Measurement items were tested with a 7-point Likert scale and some yes/no questions. The measurement items and criteria are presented in Table 2. The survey consists of 25 questions, where

13 of which are used to acquire demographic information, information on previous education, and self-evaluation of the programming knowledge, which were only asked during the pre-test of the survey. The following 12 questions consist of theoretical and practical examples that test the pre-knowledge of programming in JavaScript.

Table 1
Number of participants

Study year	MC students	IaCT students	Total
2019/2020	57	25	82
2020/2021	63	45	108
2021/2022	70	48	118
2022/2023	60	0	60
2023/2024	57	0	57
Total	307	118	425

3. Data Analysis and Results

We have conducted a Data Analysis with SPSS 29 and MS Excel.

In Figure 1, the 100% loaded bar chart presents what students have reported on the topic if they have ever attended any programming course before entering the University by year of enrollment. As can be seen in the figure, the COVID-19 generation had a bit of a setback, whereas the number of students who received some prior programming knowledge before entering University has been rising again for the last two years. As programming is an important skill in today's digital world, we can presume that this number will be even higher in the next years.

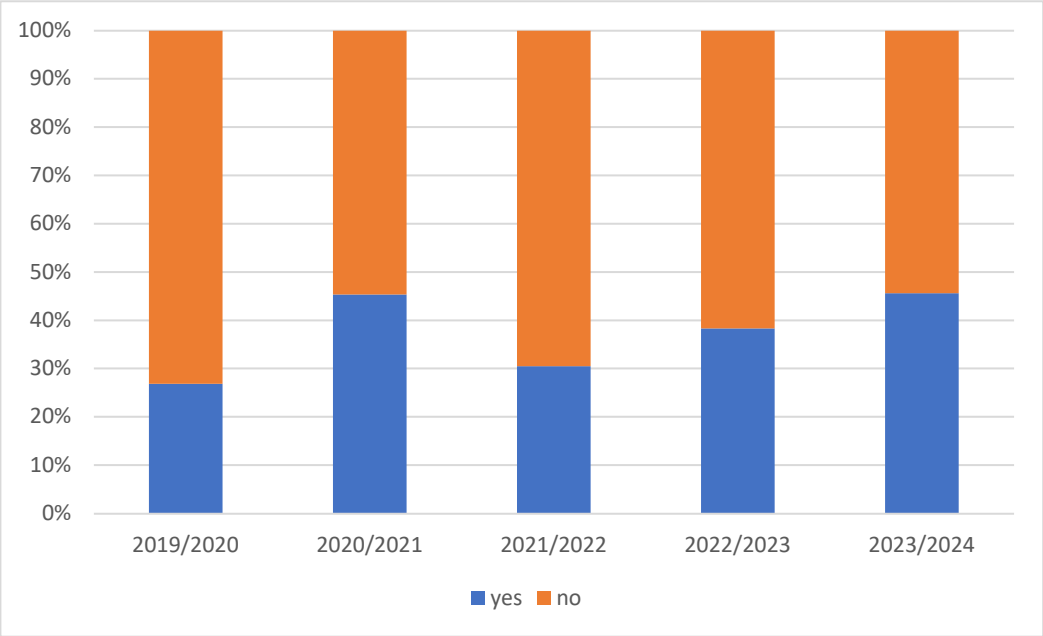

Figure 1: Students who attended the programming course before entering the University by year of enrollment.

Table 2
Questions used for the study.

QUESTIONS	POSSIBLE ANSWERS
1 Did you already attend a programming subject (before enrolling on the faculty)?	Y / N
2 Which programming language did you use? Answer only if you ticked the answer in the previous question "YES". (Multiple choice)	• C • C++ • C# • Java • JavaScript • Pascal • Python • PHP • Other
3 How long have you been learning programming? Answer only if you answered "YES" to question 1	• six months or less • 6-12 months • 1-2 years • 2-3 years • 3-4 years • More than four years
4 Where did you learn programming? Answer only if you answered "YES" to question 1 (multiple choice)	• Independently • In a course in primary school • In a course in secondary school • At extracurricular activities in school • Online
5 I have a lot of knowledge in programming. 6 I know how to use the programming language JavaScript. 7 I know how to use one of the available programming languages. 8 I know how to use variables. 9 I know how to use arrays. 10 I know how to use conditional statements. 11 I know how to use loops. 12 I know how to use functions. 12 I know how to use objects. 13 I know how to use DOM.	• 1 – strongly disagree • 2 – disagree • 3 – more or less disagree • 4 – undecided • 5- more or less agree • 6 – agree • 7 – strongly agree

In Table 3, the amount of time of prior programming learning is presented by enrollment year per student. This table is presented for the Media Communications study program to avoid the influence of the other study program on the statistics.

Table 3
The number of participants of the Media Communications study program who had none or some knowledge of programming before entering University.

Study year	Did not learn before	Six months or less	6-12 months	1-2 years	2-3 years	3-4 years	more than four years
2019/2020	40	3	1	7	3	3	0
2020/2021	42	5	4	2	4	5	1
2021/2022	50	3	3	6	5	3	0
2022/2023	37	5	8	5	2	3	0
2023/2024	25	12	6	6	2	5	1
Total	**194**	**38**	**33**	**40**	**22**	**26**	**3**

In Figure 2, the programming languages that the students used when taking programming courses are presented. As can be seen from the figure, the most commonly learned programming languages are C++ and JavaScript. However, we can also notice increasing interest in C# whereas the interest in C++ and Java interests are decreasing in the last years.

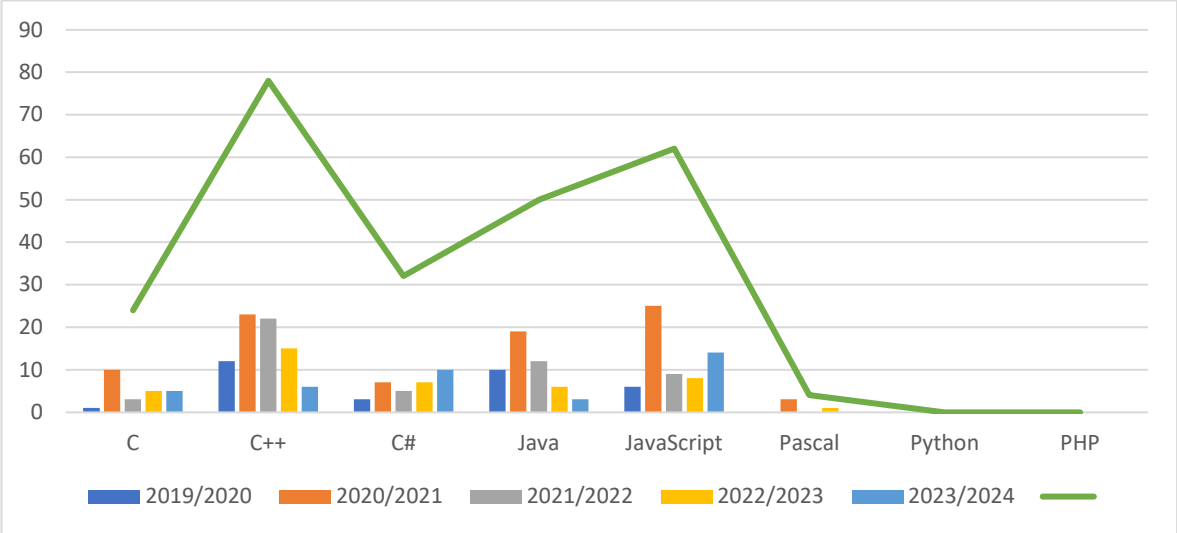

Figure 2: The programming language that the students used before previous programming courses.

In Figure 3, the information on where the students learned computer programming before coming to the University is presented. As presented on the graph, most commonly students learned programming in a course in secondary school. Not many students learned programming in primary school or at extracurricular activities in schools.

Finally, we conducted an ANOVA test to compare five different generations in their self-reported knowledge of programming. We have calculated a mean for each student for questions 5-13 (7-point Likert scale). In Table 4, we have presented the mean and standard deviation by study year for these questions together. The Levene statistics significance is greater than .05 (.065), therefore we can assume equal variances across all groups. The ANOVA test showed that the knowledge of programming is significantly different among different generations (p-value <.001, F-score 4.723, df = (4, 418)).

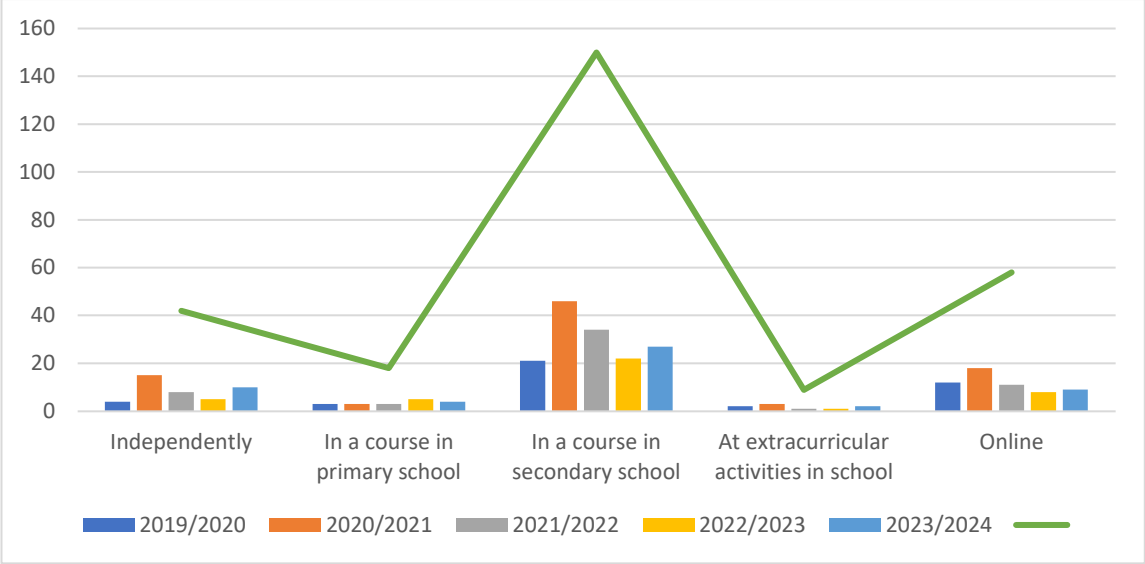

Figure 3: Where did the students learn computer programming before enrolling in University?

Table 4
Mean and standard deviation of self-reported knowledge on programming.

Study year	N	Mean	Std. Dev.
2019/2020	81	3,20	1,73
2020/2021	108	3,59	1,98
2021/2022	118	2,97	1,67
2022/2023	60	2,60	1,71
2023/2024	56	2,52	1,76
Total	**423**	**3,07**	**1,81**

We also conducted a Bonferroni posthoc test where we saw significant differences between the 2020/2021 and 2022/2023 groups (mean .95, p-value .01) as well as 2020/2021 and 2023/2024 groups (mean 1.07, p-value 0.00).

4. Discussion

The findings indicate significant differences in the pre-knowledge of students enrolled in the Fundamentals of Web Programming course over five academic years. The data suggests that the COVID-19 pandemic may have impacted students' opportunities to gain programming knowledge before enrollment in the university, as can be seen in Figure 1, where between COVID-19 in 2021/2022, there was a big step down of students who ever attended programming courses before entering University. However, the rise in students with prior programming experience is encouraging and aligns with the increasing importance of digital literacy in the modern world.

The diversity in programming languages learned before university, with C++ and JavaScript being the most prevalent, reflects the broad spectrum of educational backgrounds among the students. The shift in interest from C++ to C# could be indicative of industry trends influencing educational choices. As also presented in Figure 2, we can observe a lower interest in C++ and Java technologies, whereas C, C#, and JavaScript technologies are rising among the interests of students. Moreover, the fact that most students learned programming in secondary school courses, as presented in Figure 3, suggests that early exposure to programming could be crucial in shaping future university curricula to better

prepare students for advanced courses. All results of the study were gained by students who self-reported their knowledge, so this might be a misleading aspect in terms of the validity of the study.

5. Conclusion

The study demonstrates that there is a significant difference in the self-reported programming knowledge among five consecutive generations of students. There could be a number of reasons for this; some factors could include changes in secondary education curricula, the impact of the COVID-19 pandemic, evolving industry demands and, thus, higher interest in programming among younger generations. The results underscore the need for universities to continually adapt their introductory programming courses to accommodate the varying levels of pre-knowledge students bring to the classroom. Future research could focus on the effectiveness of these adaptations and their impact on student's academic performance and career readiness in the field of computer science.

Acknowledgements

The authors acknowledge the financial support from the Slovenian Research and Innovation Agency (Research Core Funding No. P2-0057).

References

[1] S. Hopcan, E. Polat, and E. Albayrak, 'Whether to flip Extreme Apprenticeship: which is more effective in programming instruction?', *Educ. Inf. Technol.*, vol. 27, no. 8, pp. 10731–10756, Sep. 2022, doi: 10.1007/s10639-022-11055-y.

[2] Y. Zhang, L. Paquette, J. D. Pinto, and A. X. Fan, 'Utilizing programming traces to explore and model the dimensions of novices' code-writing skill', *Comput. Appl. Eng. Educ.*, vol. 31, no. 4, pp. 1041–1058, 2023, doi: 10.1002/cae.22622.

[3] X. Wang, Y. Wang, F. Yang, W. Le, and S. Wang, 'Measuring Programming Ability for Novice Programmers', *J. Internet Technol.*, vol. 23, no. 3, Art. no. 3, May 2022.

[4] M. Salinas, P. Leger, H. Fukuda, N. Cardozo, V. Duarte, and I. Figueroa, 'Evaluations of Integrated Programming Environment for First-Year Students in Computer Engineering', *JUCS - J. Univers. Comput. Sci.*, vol. 29, no. 1, Art. no. 1, Jan. 2023, doi: 10.3897/jucs.81329.

[5] R. Mason, Simon, B. A. Becker, T. Crick, and J. H. Davenport, 'A Global Survey of Introductory Programming Courses', in *Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1*, Portland OR USA: ACM, Mar. 2024, pp. 799–805. doi: 10.1145/3626252.3630761.

[6] K. Bubnó and V. L. Takács, 'Mathematical and computational awareness before and after the pandemic', *Front. Educ.*, vol. 7, Sep. 2022, doi: 10.3389/feduc.2022.933339.

[7] P. Denny, B. A. Becker, N. Bosch, J. Prather, B. Reeves, and J. Whalley, 'Novice Reflections During the Transition to a New Programming Language', in *Proceedings of the 53rd ACM Technical Symposium on Computer Science Education*, Providence RI USA: ACM, Feb. 2022, pp. 948–954. doi: 10.1145/3478431.3499314.

[8] N. Peimani and H. Kamalipour, 'Online Education in the Post COVID-19 Era: Students' Perception and Learning Experience', *Educ. Sci.*, vol. 11, no. 10, Art. no. 10, Oct. 2021, doi: 10.3390/educsci11100633.

[9] T. Srivatanakul, 'Emerging from the pandemic: instructor reflections and students' perceptions on an introductory programming course in blended learning', *Educ. Inf. Technol.*, vol. 28, no. 5, pp. 5673–5695, May 2023, doi: 10.1007/s10639-022-11328-6.