Analyzing Student Success in Programming and Other IT Courses

Dragana Knežević
Užice Department
Western Serbia Academy of Applied Studies
Užice, Serbia
dknezevic28@gmail.com

Ilja Stanišević
Valjevo Department
Western Serbia Academy of Applied Studies
Valjevo, Serbia
ilja.stanisevic@vipos.edu.rs

Abstract—**In the rapidly evolving world of higher education, particularly in the realm of IT, the value of programming competence in developing versatile and well-rounded professionals cannot be emphasized enough. With the growing integration of IT across various industries, students pursuing vocational studies are increasingly exposed to the programming curriculum. This study analyzes the academic progress of students over several years, not only on programming courses, but also in other IT, but non-programming courses. Focused on students from the Western Serbia Academy of Applied Studies, this study aims to illuminate the academic success of students in two groups of subjects: programming and non-programming professional courses. By comparing the achievements of students within these two groups within vocational studies, we seek to draw meaningful conclusions that can inform the future development of teaching activities.**

Keywords: applied studies, vocational studies, programming, professional courses, non-programming, undergraduate students.

I. INTRODUCTION

In the ever-evolving landscape of higher education, the integration of programming courses into the curricula of IT studies has become increasingly prominent, particularly within the realm of vocational studies. As technology continues to shape various industries, proficiency in programming is no longer a niche skill but a crucial asset for students pursuing careers in diverse fields. This paper will compare students' performance in courses that focus on different programming languages with courses that are highly specialized in information technology but do not include programming as a major component (i.e., non-programming professional courses).

The dynamic nature of programming languages and the rapid advancements in technology pose unique challenges and opportunities for educators. By analyzing the performance of students on both, programming and non-programming courses, we aim to discern patterns that could inform curriculum design, teaching methodologies, and ultimately enhance the overall educational experience for students in vocational programs.

Through comprehensive examination, we strive to provide valuable insights for educators, administrators, and policymakers involved in shaping the future of vocational education. As the demand for a workforce in IT sector is growing, understanding the nuances of programming education in vocational studies becomes essential for fostering a well-prepared and adaptable generation of professionals.

II. LITERATURE REVIEW

Many researchers specializing in pedagogical aspects of teaching in IT studies agree that instructing the principles of computer programming, commonly referred to as coding, is often one of the more intricate components of any computer science curriculum. Although students may not fully equip themselves to solve practical problems during IT studies, the studies provide an excellent foundation for further work and improvement.

Students should transform their problem-solving solutions into executable programs using a specified programming language. This step is fundamental for honing programming skills, as it provides students with the opportunity to put their theoretical understanding into practice when faced with real programming challenges [1].

Authors in [2] highlight that a few decades ago, the constructivist learning theory, emphasizing that students construct knowledge rather than passively receive it, was a rare or non-existent phenomenon in computer science, particularly in the context of learning programming. However, today, it seems that this is no longer the prevailing scenario, especially in recent years due to the increasingly common occurrence of online learning. This is particularly evident in the context of learning programming, as confirmed by authors in [3], who demonstrate that classroom instruction alone is insufficient to master all aspects of programming in any programming language.

Today, with the aim of enhancing interactivity and improving communication in remote learning, modern technologies and animated simulations can be employed. Synchronous communication technology can serve as a collaboration platform for instructors and students to share applications in real-time [4]. Additionally, authors in [5] emphasize the importance of using video presentations. The primary advantage for students is the ability to replay the same video lecture repeatedly, irrespective of location and at any time.

On the other hand, with the continuous growth of the internet, communities dedicated to specific issues, most commonly programming-related problems, are also expanding. This can have both positive and negative consequences. On one hand, addressing issues in this domain can be facilitated effortlessly, while on the other hand, it often translates into mechanical code copying without comprehension, which impacts both learning and the development of programming skills [6]. This directly impacts the achievement of one of the key objectives of courses focusing on programming languages: to teach students good programming practices, foster algorithmic thinking, and instruct them on applying logical concepts. Additionally, these courses aim to familiarize students with object-oriented principles such as encapsulation, inheritance, and polymorphism [7].

To attain the best outcomes, it's most effective to blend various learning and teaching methodologies. This is corroborated by authors in [8], who indicate that students exhibit significantly improved performance through active class participation and partial self-preparation for exams.

Many authors confirm that teaching in other areas of computer science poses challenges for both instructors and students. An author in [9] provides a guide for planning the principles of teaching exclusively for IT courses, applicable to all levels of education and any teaching context.

Some authors go a step further and include machine learning as a tool in addressing this issue, aiming to detect and predict the success of undergraduate students for the purpose of adapting programming courses to different levels of knowledge and prior experience among students [10].

Over the past few years, there has been a notable and increasing shift towards both necessary and voluntary online education. Online classes, in particular, have emerged as a prominent factor contributing to students' challenges in exam performance in recent times. This paper aims to explore the issues arising from online education, including the trend of reduced regular class attendance compared to a few years ago. The focus is specifically on monitoring the academic success of students in the field of IT, seeking insights into the multifaceted challenges associated with online instruction and the changing patterns of student engagement in academic activities. The goal is to investigate and identify trends in student success by examining passing rates and average grades over a period of five academic years.

III. SUCCESS OF STUDENTS AT WSAAS IN PROGRAMMING AND NON-PROGRAMMING COURSES

The Western Serbia Academy of Applied Studies (WSAAS) comprises two departments, Užice and Valjevo, catering to students enrolled in the Information Technology and Systems study program. This program involves learning multiple programming languages and encompasses dozens of professional courses throughout the three-year study. These courses cover programming languages like C, C++, C#, Java, JavaScript, PHP, and SQL. It's worth noting that all the mentioned programming languages rank among the top ten according to the TIOBE Programming Community Index [11] for the previous year.

In addition to students at WSAAS exploring multiple programming languages during their three-year studies, the majority of the remaining courses consist of specialized applied subjects in the IT field, although not specifically focused on programming. In these courses, students delve into operating systems, various application software, graphic and web design, computer networks, multimedia, etc.

During the semester, it is very difficult to measure how successful students will be in taking exams in a specific course. This becomes possible after the exam period, but it is best to take into account several exam periods, or even better, an entire school year. The results presented in this paper stem from the amalgamation of statistical data collected during exam periods in both departments over a five-year academic period.

This paper primarily concentrates on the two previously mentioned groups of courses: programming and non-programming-oriented courses but in the field of computer science (six in each group).

This section will show a comparative overview of student performance in those two groups of courses through several diagrams with graphical representations of data collected for both the Užice and Valjevo departments. All diagrams on the x-axis are labeled with the academic year or the course name, while the y-axis represents the passing rate and average grade.

Fig. 1 illustrates the average grade and the percentage of students who passed the exam for non-programming courses over the observed five-year period. Similarly, Fig. 2 displays the same values for programming courses. Both figures indicate a significant dropout in terms of student success in both groups of courses for the 2020/21 academic year. This is likely a result of the teaching and examination methods during that academic year, mainly conducted online due to the ongoing pandemic at that time in our country. It is noteworthy that this trend did not persist, and as the diagram shows, students were much more successful in their exams in the following academic year. This positive trend continued into the last academic year (2022/23) for non-programming courses, but unfortunately not for programming-oriented courses.

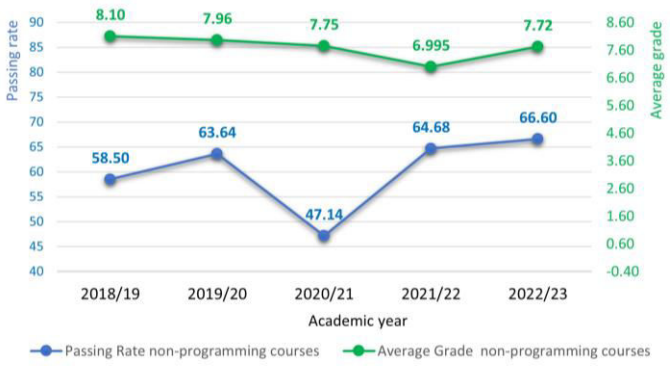

Figure 1. Average grade and % of students who passed the exam for non-programming courses, by year

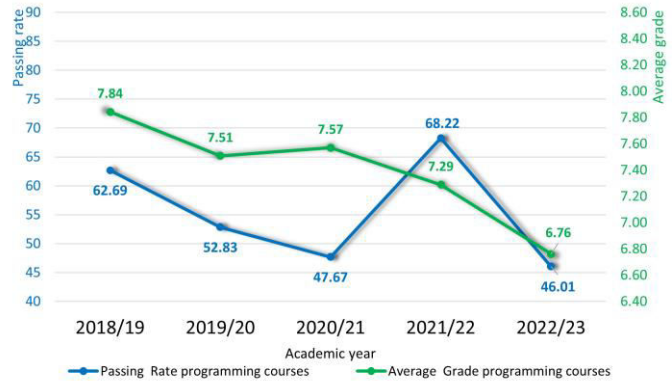

Figure 2. Average grade and % of students who passed the exam, for programming-based courses, by year

To gain insight into the success of students in each individual course, the next two figures illustrate the average grades and the passing rate for all students who passed the exam in non-programming (Fig. 3) and programming courses (Fig. 4), considering the observed five-year period.

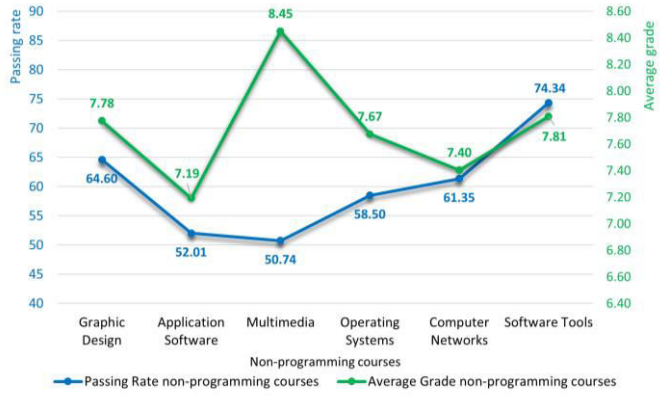

Figure 3. Average grade and passing rate, during five-years academic period for non-programming courses

The diagrams clearly show that the students' success in exams is not in direct correlation with the average grade. For non-programming courses, the highest passing rate is recorded for the course covering software tools (almost 75%), while the average grade for this course is 7.81. On the other hand, the multimedia-related course, despite having the highest average grade (8.45), exhibits the lowest passing rate among all the observed courses (50.74).

Observing programming-oriented courses, the programming course with the highest passing rate (PHP) is not simultaneously the course with the highest average grade. The lowest pass rate is in courses based on the C# programming language, where students achieve a higher average grade than in PHP course. The lowest average grade is attained by students in the programming course based on the C/C++ programming language, while the highest grades are achieved in courses based on the JAVA programming language.

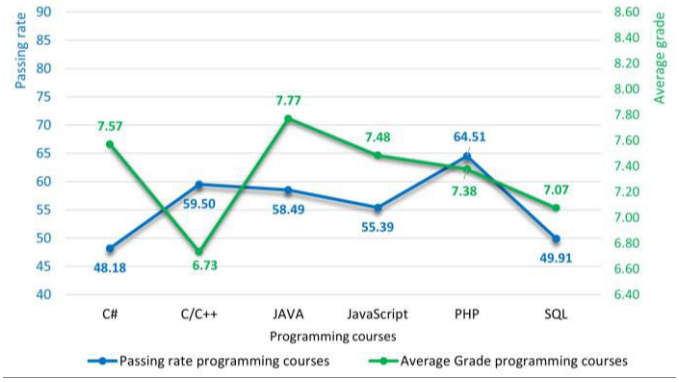

Figure 4. Average grade and passing rate, during five-years academic period for programming-based courses

Nevertheless, to improve clarity regarding the academic performance of students in the two observed course groups, the next two figures present a comparative analysis of passing rates (Fig. 5) and average grades (Fig. 6) over the five-year academic period.

Figure 5. Passing rate for both, non-programming and programming courses, by year

Analyzing the passing rates of students throughout the observed period, 2020/21 stands out with the lowest passing rate for both groups of courses (approximately 47%). Despite the encouraging improvement in both course groups in the following academic year (2021/22), unfortunately, this trend

did not persist. In the last observed academic year (2022/23), while students in WSAAS non-programming courses achieved the highest success rates (almost 67%), the opposite held true for programming courses. Throughout the observed period in this academic year, programming courses recorded the lowest passing rate (46%).

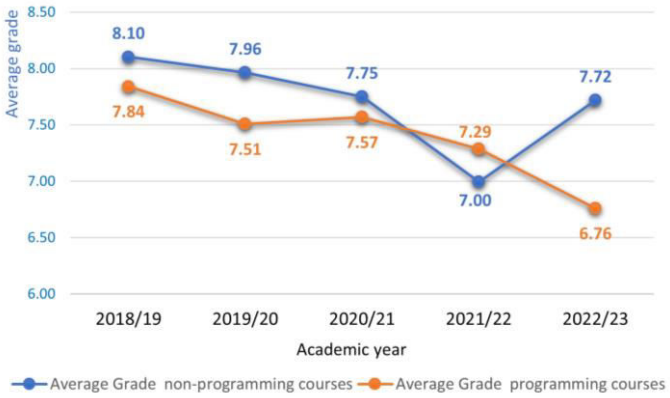

Figure 6. Average grade for both, non-programming and programming courses, by year

Observing the average grades over the five-year period, a slight downward trend is noticeable in both groups of courses. While 2020/21 stood out earlier with the lowest passing rates for both groups, this is not the case when it comes to the average grade levels. In contrast to the previous diagram (Fig. 5), Fig. 6 shows that the most significant drop in average grades occurred in the 2021/22 academic year. Interestingly, the same year stands out with the highest passing rate. However, in the last year, in correlation with the increase in passing rates for non-programming courses, there is also a rise in the average grades for these courses. Similar correlation exists between the average grades of students and their success in passing programming courses, but in a negative sense.

In general, throughout the observed five-year period, non-programming courses have higher average grades in each academic year (except for 2021/22). However, no clear conclusion can be drawn regarding the passing rates.

For a more in-depth examination, one can delve into more specific diagrams that offer insights into the average grades of students in each of the six non-programming courses across the observed five academic years (Fig. 7). A similar approach is taken for programming courses (Fig. 8).

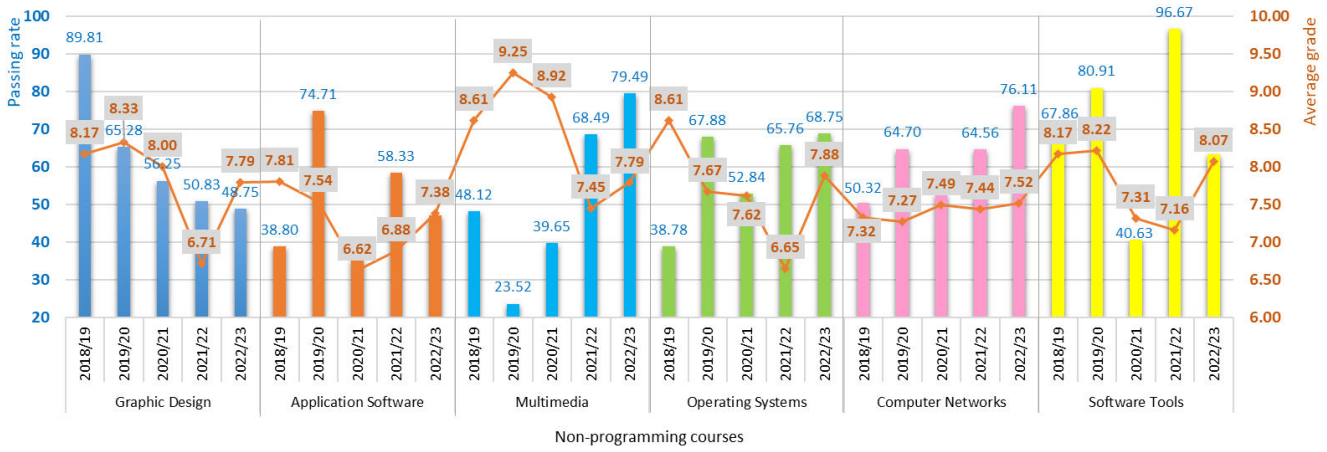

Figure 7. Average grade and passing rate for non-programming courses, by course and year

Figure 8. Average grade and passing rate for programming courses, by course and year

Examining each of the non-programming courses individually, the diagram in Fig. 7 reveals unusual fluctuations both in terms of passing rates and average grades. The course in graphic design consistently experiences a decline in passing rates each year, while other courses either show an increase or irregular oscillations. On the other hand, programming courses (Fig. 8) exhibit an unbalanced and unstable trend, with the most significant drop occurring in the last academic year for almost all courses except the one covering the PHP programming language.

IV. CONCLUSION

Despite programming being considered one of the most challenging areas in the IT field, the presented results do not provide clear evidence to support this theory. Over the years, students' success fluctuates considerably, marked by frequent setbacks. This holds true for both groups of courses, although it is expected that students would more easily grasp knowledge in non-programming courses.

The passing rate, although averaging above 50% over the entire observed period, shows concerning declines in certain academic years and for specific courses. The consequences of online education caused by the pandemic in the past period undoubtedly have a significant impact on the results, but they cannot be solely and primarily blamed for the (un)success of students. Research should aim to shed light on the issues expressed here, characterized by a declining trend in both specific programming and non-programming courses.

REFERENCES

[1] M. N. Ismail, N. A. Ngah, and I. N. Umar, "Instructional strategy in the teaching of computer programming: a need assessment analyses," *The Turkish Online Journal of Educational Technology*, vol. 9, no. 2, pp. 125–131, 2010.

[2] M. Ben-Ari, "Constructivism in computer science education," *Journal of computers in Mathematics and Science Teaching*, vol. 20, no. 1, pp. 45–73, 2001.

[3] J. Goode, J. Margolis, and G. Chapman, "Curriculum is not enough: The educational theory and research foundation of the exploring computer science professional development model," in *Proceedings of the 45th ACM technical symposium on Computer science education*, 2014, pp. 493–498.

[4] X. Huan, R. Shehane, and A. Ali, "Teaching Computer Science Courses in Distance Learning.," *Journal of Instructional Pedagogies*, vol. 6, 2011.

[5] R. Shehane and S. Sherman, "Visual Teaching Model for Introducing Programming Languages.," *Journal of Instructional Pedagogies*, vol. 14, 2014.

[6] J. Pöial, "Challenges of Teaching Programming in StackOverflow Era," in *Educating Engineers for Future Industrial Revolutions: Proceedings of the 23rd International Conference on Interactive Collaborative Learning (ICL2020), Volume 1 23*, Springer, 2021, pp. 703–710.

[7] O. Hrnjaković, M. Tot, A. Kupusinac, and R. Doroslovački, "Različiti pristupi u metodici nastave osnova programiranja u srednjoj školi i na fakultetu," in *XXV Skup TRENDOVI RAZVOJA*, in T1.4-4. 2019, pp. 1–4.

[8] I. Stanišević, S. Obradović, and A. Tošić, "Aktivno učenje u nastavi informatičkih predmeta," presented at the 18. Telekomunikacioni forum TELFOR 2010, Beograd, 2010, pp. 1141–1144.

[9] O. Hazzan, T. Lapidot, and N. Ragonis, *Guide to teaching computer science*, 2nd ed. Springer London, 2014.

[10] N. Stanković, "Faktori učenja i predviđanje uspešnosti u programiranju primenom veštačkih neuronskih mreža," University of Kragujevac, Faculty of Technical Sciences Čačak, 2021.

[11] "TIOBE Index," TIOBE Programming Community Index Definition. [Online]. Available: https://www.tiobe.com/tiobe-index/programminglanguages_definition/