



Kumon-Inspired Approach to Teaching Programming Fundamentals

Ivone Amorim  

PORTIC – Porto Research, Technology & Innovation Center
Polytechnic of Porto (IPP), Portugal

Pedro Baltazar Vasconcelos  

LIACC & Department of Computer Science
Faculty of Sciences, University of Porto, Portugal

João Pedro Pedroso  

CMUP & Department of Computer Science
Faculty of Sciences, University of Porto, Portugal

Abstract

Integration of introductory programming into higher education programs beyond computer science has led to an increase in the failure and drop out rates of programming courses. In this context, programming instructors have explored new methodologies by introducing dynamic elements in the teaching-learning process, such as automatic code evaluation systems and gamification. Even though these methods have shown to be successful in improving students' engagement, they do not address all the existing problems and new strategies should be explored. In this work, we propose a new approach that combines the strengths of the Kumon method for personalized learning and progressive skill acquisition with the ability of online judge systems to provide automated assessment and immediate feedback. This approach has been used in teaching *Programming I* to students in several bachelor degrees and led to a 10% increase in exam approval rates compared to the baseline editions in which our Kumon-inspired methodology was not implemented.

2012 ACM Subject Classification Social and professional topics → Computer science education; Applied computing → Interactive learning environments

Keywords and phrases Programming teaching, Programming education, Kumon method, Progressive learning, Online judge system

Digital Object Identifier 10.4230/OASICS.ICPEC.2024.5

Funding *Ivone Amorim*: Partially supported by CMUP, which is financed by national funds through FCT – Fundação para a Ciência e a Tecnologia, I.P., under the project with reference UIDB/00144/2020.

Pedro Baltazar Vasconcelos: Partially supported by: Base Funding – UIDB/00027/2020 of the Artificial Intelligence and Computer Science Laboratory – LIACC – funded by national funds through the FCT/MCTES (PIDDAC).

João Pedro Pedroso: Partially supported by CMUP, which is financed by national funds through FCT – Fundação para a Ciência e a Tecnologia, I.P., under the project with reference UIDB/00144/2020.

1 Introduction

Over the past decade, information technology has suffered a remarkable growth, with its evolution and application having significant impacts on our society, namely in education. Today, for graduates to easily integrate the labour market they need not only to acquire specific skills, but also to develop the agility to rapidly acquire new knowledge and address new challenges with creativity and critical thinking [12].

In the past, programming skills were associated mostly with computer science and engineering fields. However, today, Programming is seen as a fundamental area for the development of characteristics and skills such as creativity, problem-solving, persistence,



© Ivone Amorim, Pedro Baltazar Vasconcelos, and João Pedro Pedroso;
licensed under Creative Commons License CC-BY 4.0

5th International Computer Programming Education Conference (ICPEC 2024).

Editors: André L. Santos and Maria Pinto-Albuquerque; Article No. 5; pp. 5:1–5:13

OpenAccess Series in Informatics



OASICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

logical and critical thinking [17, 7]. Additionally, professionals from different areas can benefit from the ability to write codes for different applications. For instance, in biology or chemistry, analysing large datasets is often necessary, and this task can be facilitated by programming skills.

As a result, many diverse fields of knowledge now incorporate programming courses into their curricula [7]. Therefore, several students with different backgrounds and characteristics are involved in introductory programming [12]. For example, an instructor may have students in the same class who have no prior knowledge about programming, as well as students who have already learned different programming languages. This leads to difficulties in aligning classes with the students' interests and motivations. Other challenges include the lack of resources, such as computational labs, for practical classes. On the students' side, the already well-known difficulties they face are exacerbated by the different backgrounds students may have, which may lead them not to perceive programming as an important skill or competence. According to Gomes and Mendes [8], some of the main difficulties students may encounter while learning programming include inadequate teaching approaches, which sometimes prioritize theoretical concepts over improving students' problem-solving abilities; a lack of problem-solving skills necessary to understand the logic behind programming; the use of inappropriate self-learning methods to enhance academic programming success; and psychological factors.

Consequently, significant failure and dropout rates, as well as a lack of motivation, have been observed in programming courses [4, 16]. This has raised concerns among programming instructors, leading them to develop and utilize novel teaching and learning approaches to enhance students' learning experiences and ultimately increase success rates. The most used strategies include the introduction of dynamic elements in learning strategies such as automatic code evaluation systems [18, 11], namely online judge systems, gamification [15, 14], and systems that use visual representations of algorithms, such as Python Tutor¹. These strategies have shown to reduce some of the challenges instructors and students face when teaching and learning programming, respectively. However, they still do not offer a complete solution to address all the existing challenges.

In this work, we explore a new approach that combines the strengths of the Kumon method [21] for personalized learning and progressive skill acquisition with the ability of online judge systems to provide automated assessment and immediate feedback.

The rest of this paper is organized as follows. Section 2 provides an explanation on some key concepts regarding online judge systems and the Kumon method, and discusses some related work. Section 3 presents the course to which our methodology was designed, details the online judge system used and explains our Kumon-inspired approach. Section 4 presents the case study methodology employed to assess the effectiveness of our approach. Section 5 presents and discusses the results obtained. Finally, Section 6 presents the conclusions of our work and proposes future directions to further explore and validate this approach.

2 Background and Related Work

2.1 Online judge systems

The assessment of students by instructors can be done for several reasons: to provide feedback to students on their learning path, to assess previous knowledge in a specific subject, to evaluate teaching methodologies, to identify at-risk students, among other reasons [5].

¹ <https://pythontutor.com/>

However, due to the increasing number of students with diverse backgrounds that instructors have to assess and support in introductory programming courses, a wide range of assessments is required. This demand may lead to instructors being overloaded with work and reduces the time they have for other tasks. Online judges are automated assessment tools [2] designed for the reliable evaluation of algorithm source code submitted by users. These tools may be a crucial help for instructors in these assessment processes. Online judges are now popular in various applications, including in programming contests and education. The use of online judges in the classical educational system has advantages for both instructors and students. For instructors, they allow the assessment of students' assignments automatically, increasing assessment accuracy and significantly reducing the time needed for evaluation. Consequently, many more exercises can be prepared and assigned to students [24]. On the other hand, students receive almost instant feedback on their answers, which motivates them to perform more exercises, promotes active learning by helping them to easily understand the main difficulties they should address, and fosters independence.

There are several factors that may limit the accessibility of existing online judges for use in introductory programming courses. According to Asuncion et al. [2], one of the problems is that most existing online judges were not designed for introductory programming classes. Therefore, the exercises they provide may not align with the curricular units of these courses. For example, the well-known online judge Codeforces² was primarily designed for programming contests. Consequently, some of their problems considered “easy” may not be suitable for introductory programming courses, as they may require concepts typically covered in a data structures and algorithms course rather than in introductory programming classes. Additionally, the same authors claim that creating and uploading new exercises for many online judges is not easy, if possible at all.

Fortunately, several reviews of existing online judge systems and their application in education have been conducted over the years, helping instructors in finding the right tool for their purpose [1, 9]. Recently, Wasik et al. [24] provided a comprehensive review of the state of the art for these systems. They classified them according to their principal objectives into categories which include “Development platforms.” This refers to systems that are often provided as open source projects or binary archives that can be downloaded and installed locally, providing full administrative privileges to the user. They can be used to host a programming competition or a course using the user's own infrastructure. Moreover, most can be adapted to user needs and integrated with external services. Well known platforms to prepare and perform programming contests are DOMjudge³, Mooshak⁴, and SIO2⁵. There are other development platforms which are dedicated to support the educational process, such as CloudCoder [19], BOSS [10], and Web-CAT [6]. Unfortunately, some of these latter systems are not open-source, and others have been updated for the last time more than 15 years ago. In this work, we use an online judge system named Codex that falls into the category of “Development platform”. This system was developed at the Faculty of Sciences of the University of Porto, and its source-code is publicly available through the GitHub platform⁶. Further details on this system are presented in Section 3.2.

² <https://codeforces.com/>

³ <https://www.domjudge.org/>

⁴ <https://mooshak.dcc.fc.up.pt/>

⁵ <https://github.com/sio2project>

⁶ <https://github.com/pbv/codex>

2.2 Kumon-inspired methods and potential benefits

The Kumon Method is a pedagogical method that emerged in 1954 in Osaka, Japan, by the hand of Toru Kumon, a Mathematics teacher of a high school, who developed this to help his son’s mathematical education. This method has been extended to other subjects, such as “Reading” and “English as a foreign language”. Initially popularized in Japan and subsequently in the United States of America [21], the method has gained global recognition and is now implemented in learning centres all around the world, mainly for teaching Mathematics and English⁷.

The main objective of the Kumon Method is to maximize the learning potential of each individual. To achieve this, a series of habits and abilities have to be acquired, namely:

- Self-learning; the students learn how to learn by themselves, without depending on another person.
- Study habits; students are encouraged to divide the study effort into smaller, more regular steps.
- To foster concentration: if students are not able to focus on a specific task, it will be difficult for them to learn effectively.
- Self-confidence, that allows students to face any educational challenge.
- Motivation to learn, to perceive learning as something enjoyable that will help them to grow as persons.

Several research studies have assessed the effectiveness of this method, primarily focusing on improving mathematical skills [13, 22]. However, there is no known study on the application of this method in teaching programming. Although some works propose similar strategies, such as intensive training [3], none have directly applied the principles of the Kumon method to foster the development of the aforementioned habits and abilities in programming students.

3 Progressive Method for Teaching Programming

In this section, we begin by introducing the course for which our Kumon-inspired approach was designed. Following that, we provide an overview of Codex, the online judge system utilized in our methodology. Lastly, we present our novel progressive method for teaching programming fundamentals.

3.1 The Course – Programming I

Programming I is a first year course for several bachelor degrees at the Faculty of Sciences from the University of Porto. It is mandatory for the bachelor degrees of Agricultural Engineering, Engineering Physics, Geospatial Engineering, Bioinformatics, and Physics. Besides, it is an optional course for the bachelor degrees of Biology, Geology, and Chemistry.

The main objectives of this course are: Get acquainted with personal computers in the GNU/Linux operating system and their usage; Learn how to write computer programs using Python and execute them in a terminal; Acquire competence in the implementation of simple algorithms; Acquire good code structuring and programming style; Learn some basic data structures and algorithms; Get acquainted with program debugging and testing.

The main learning outcomes and competences expected from students who successfully perform this course are: understanding the role of programming for solving problems in their degree, acquaintance with the basic components of a recent programming language, ability

⁷ <https://kao.kumonglobal.com/our-global-network/>

to write programs that allow accomplishing useful goals, and confidence in the usage of the Python language and its standard library. The students have 2 hours per week of face-to-face theoretical classes and 2 hours per week of laboratory classes, for 14 weeks, resulting in a total of 56 contact hours.

3.2 The online judge system: Codex

Codex⁸ is a web system for setting up programming exercises with automatic assessment, which is intended for learning environments, similar to a judge system. It was developed and is currently being used at the Faculty of Sciences of the University of Porto for introductory courses on Python, Haskell and C programming [23]. Its main features are:

Simple exercise authoring. Exercise descriptions are written in a human-readable Markdown format that can easily be copied, mailed, kept in a version repository, compared for changes, etc.

Allows assessing program units. Exercises can assess single functions, classes or methods as well as complete programs.

Provides automatic feedback. Rather than report just an accept/reject result, Codex can report the failed examples to students.

Multiple types of exercises. Besides code testing, Codex also allows multiple-choice and fill-in questionnaires.

In Codex, student's submissions are classified as follows:

- *CompileError*: rejected attempt due to a compile-time error or warning; for an interpreted language such as Python, this is typically a syntax error;
- *RuntimeError*, *TimeLimitExceeded* or *MemoryLimitExceeded*: the execution was aborted due to a runtime error or resource exhaustion;
- *WrongAnswer*: testing failed in at least one case;
- *Accepted*: all tests passed.

For the *Programming I* course we used input-output and unit testing with the *Doctest* Python library [20]. The *Doctest* files were generated semi-automatically and comprise a large number of test cases (typically, about one thousand); this not only allows testing the student's attempts more thoroughly, but also prevents over-fitting solutions to a small number test cases.

For beginner exercises that are typically not computationally intensive, the turn-around for student feedback is quick (typically 2-4 seconds).

Another important advantage of Codex is that it allows instructors to get real-time insights regarding the performance of students in laboratory assignments. Therefore, it helps to identify students who are struggling early on, allowing for timely intervention and support.

3.3 Our Kumon-based approach

Our approach for teaching and assessing programming fundamentals in Python is inspired in the Kumon method that emphasizes self-learning through repetitive practice of progressively challenging exercises. Instructors have developed three different types of assignments that allow students to progressively develop their Python skills and which are fully aligned with the course syllabus. The way these exercises are provided to students and the methodology

⁸ <https://github.com/pbv/codex>

5:6 Kumon-Inspired Approach to Teaching Programming Fundamentals

employed for course evaluation were designed to encourage self-learning, build the habit of studying, increase self-confidence, and ultimately motivate students to learn. These are all key objectives of the Kumon method. Additionally, our approach allows students to have some control over their own progress through individualized learning, and immediate feedback.

The three types of assignments developed are the following:

- **Theoretical assignments:** Following every theoretical class, students were given theoretical assignments in the form of quizzes to encourage them to actively participate in those classes and solidify their understanding of the presented concepts. These assignments were intended to be completed outside the classroom within a limited time frame. Students were allowed to consult any resources to help them in providing correct answers.
- **Worksheet assignment:** These are weekly worksheet assignments with problems provided to students approximately a week before the corresponding laboratory class. Students are encouraged to attempt to solve these problems outside the classroom, and any doubts can be clarified with instructors during laboratory sessions. Its main goal is to encourage frequent study and self-learning among students.
- **Laboratory assignments:** These assignments are sets of coding problems made available to students through the Codex system and are fundamental elements of our approach. A total of 10 assignments, each corresponding to a different level of difficulty, are provided over a 14-week period. Students can only access these assignments in the classroom and must successfully complete the current level to unlock the next one. The success in one level means that the student was able to submit a correct answer to all questions in one of the three possible attempts. If a student is not able to submit a correct answer in any of the three attempts, he or she has to attempt another problem set of the same level in the following label class. Hence, in a given lab class, different students may be attempting problems at distinct levels according to their own progress. This strategy was designed to ensure that students only proceed a new level when the knowledge from the previous one has been assimilated. Additionally, students must complete at least the first five levels to qualify for the final exam. The number of questions in these weekly assignments varies between 6 and 8 depending on the topics assessed.

The assessment strategy for determining students' final grades was also designed to align with the principles of Kumon's method. This involved considering grades obtained from both the Theoretical and Laboratory assignments as key components contributing to the final grade. Additionally, a final exam covering all the course topics is conducted in a codex-based environment under the same conditions of laboratory assignments: maximum of three attempts per exercise and no access to external resources.

The final grade is determined using the following formula:

$$\text{Final Grade} = 0.2t + 0.2l + 0.6e, \quad (1)$$

where t represents the grade obtained from theoretical assignments, l denotes the grade from laboratory assignments, and e denotes the grade from the final exam.

4 Case Study Methodology

To evaluate the effectiveness of our novel methodology, we compared the learning results of students enrolled in the *Programming I* course, as outlined in Section 3.1, over a period of four school years. Our Kumon-inspired approach was implemented in two of these years, while in the other two years, we did not utilize this method. Below, we outline the teaching methodology employed in each school year considered in our study.

- **2019:** In this school year, the students's had three different types of assignments: weakly laboratory assignments (different from the ones described in Section 3.3), intermediate test and final exam. The final grade was computed using

$$\text{Final Grade} = 0.2l + 0.2i + 0.6e,$$

where l , i , e denote the grade from laboratory, intermediate test, and final exam assignments, respectively. Students were considered eligible to take the final exam only if at least one of the following conditions was met: having a non-null grade in the intermediate test or correctly answering at least 50% of the laboratory assignments. For a student to be approved, they needed to achieve a grade of at least 40% on the final exam. The laboratory assignments were provided through Codex, with each assignment consisting of only one question and without including levels and progressive learning. The intermediate test and the final exam were also conducted through Codex.

- **2020:** This was the first school year in which our Kumon-inspired approach was implemented, and the final grade was determined using Equation (1). The students were eligible to go to the final exam only if they had answered correctly to at least 50% of the laboratory assignments. For a student to be approved, it had to have a grade on the final exam not less than 45%. It is important to note that in this year our Kumon-inspired approach was still being adjusted.
- **2021:** In this year, our Kumon-inspired approach was also utilized, albeit with more stringent conditions for a student to qualify for the final exam. More specifically, a student was eligible for the final examination only if they had successfully completed at least 50% of both theoretical and laboratory assignments. To be approved, a student needed to achieve a grade of at least 50% on the final exam.
- **2023:** In the school year 2023, technical issues resulting from renovation works in the Computer Science building caused difficulties with the network infrastructure. Despite being provided with worksheets and laboratory assignments through Codex, as described in Section 3.3, the students did not have the appropriate conditions to ensure fair assessment process using our Kumon-inspired approach. As a result, we decided to assess students only through a final examination, which was conducted in Codex. For a student to be approved, they had to have an exam grade of at least 50%.

The school years 2019 and 2023 serve as the baseline against which we compare the effectiveness of our Kumon-inspired approach in the other years (2020 and 2021). In the school year 2022, the team of professors in charge of teaching *Programming I* was different, as well as the methodological approach applied. Therefore, this year was not considered in our study.

It is important to add that in the 2021 and 2023 school years, students were allowed to seek help from their more advanced peers in laboratory classes. However, students assisting their colleagues were prohibited from writing code for them. In 2019 and 2020, students were not incentivized to seek help from other students.

A total of 937 students from different bachelor's degrees were part of our study, with most of them coming from Physics, Engineering Physics, and Geospatial Engineering.

5 Results and Discussion

To analyse the potential impact of our Kumon-inspired approach on teaching and learning Programming fundamentals, we began by calculating statistics on the number of approved students. More specifically, we computed the percentage of students who obtained approval

5:8 Kumon-Inspired Approach to Teaching Programming Fundamentals

among both those enrolled and those effectively assessed. Table 1 provides an overview of the number of students enrolled in the course, the number of students assessed, and the grade statistics by school year.

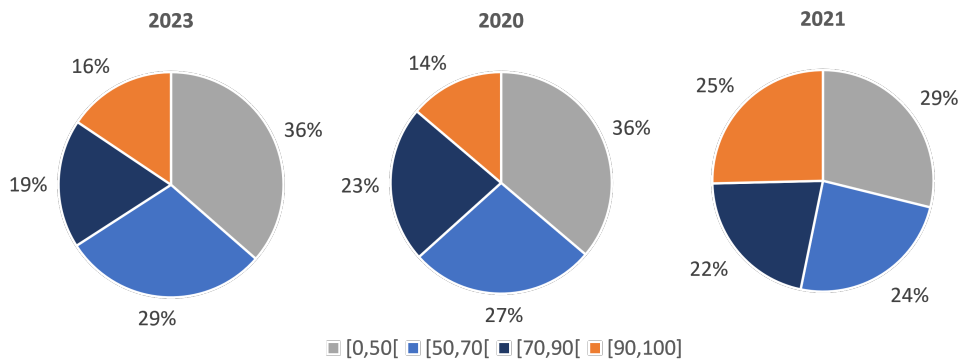
■ **Table 1** Overview of student enrolment, assessment, and grades statistics by school year.

	Baseline		Kumon approach	
	2019	2023	2020	2021
N° of enrolled students	223	207	259	248
N° of students assessed	159	173	210	201
N° of approved students	92	110	134	143
Grades mean	62.52	51.45	52.95	59.95
% approved/enrolled	41.26	53.14	51.74	57.67
% approved/assessed	57.86	63.58	63.81	71.14

In Table 1, we can observe that 2021 was the school year with the highest percentage of approved students among those assessed, with an approval rate of around 71%. Moreover, this year was the one with the highest percentage of approved students among the enrolled ones, reaching approximately 58%. When comparing the results of the school year 2021 with the baseline years, it becomes clear that the percentage of approved students is significantly higher when the Kumon-inspired approach is applied. Furthermore, when comparing this year with 2020, a much higher approval rate is observed. These results suggest a positive correlation between the grades obtained and the level of engagement required from students. For instance, in 2021, students had to complete at least half of both their theoretical and laboratory assignments, which led to better final grades compared to 2020, where they only needed to complete half of their laboratory assignments. It is important noting that these results may also have been affected by the fact that in 2021, struggling students were allowed to get assistance from more advanced peers, as detailed in Section 4. Interestingly, comparing the results between 2020 and 2023, we notice a slight overall improvement in the latter year. This suggests that allowing students to request help from their peers has a positive impact on their learning achievements. Another interesting point to observe in Table 1 is that years in which the Kumon-inspired approach was (even partially) applied had a higher rate of assessed students among those enrolled.

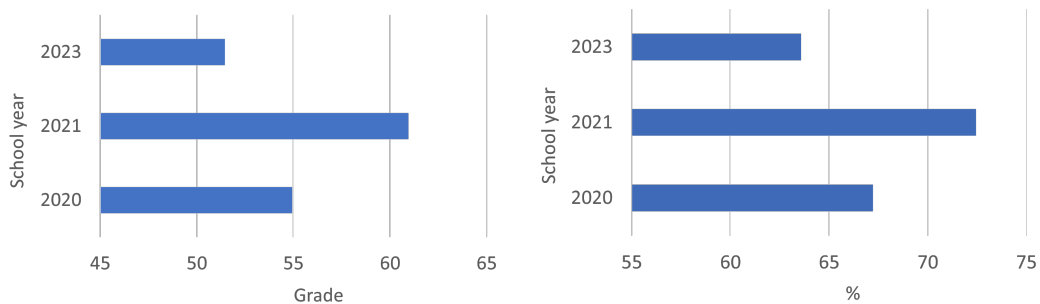
To evaluate the potential impact of our Kumon-inspired methodology on students' performance, we conducted an additional analysis. We divided the grades into four intervals: $[0, 50[$ (indicating insufficient performance), $[50, 70[$ (indicating satisfactory performance), $[70, 90[$ (indicating good performance), and $[90, 100]$ (indicating excellent performance), and calculated the percentage of students falling within each interval. Figure 1 displays the results obtained by school year. The grading system in 2019 was different from the rest of the years, so we could not include it in this analysis.

As can be observed from Figure 1, the 2021 school year was the one that has the highest percentage of students with an excellent performance. More specifically, 25% of students scored within the interval $[90, 100]$, compared to 14% in 2020 and 16% in 2023. In terms of students with insufficient performance, 2021 recorded the lowest percentage (29%), further suggesting a significant improvement in overall student performance due to our approach. When comparing 2020 and 2023, both school years showed the same percentage of students with insufficient performance. However, in 2023, a higher percentage of students demonstrated performance above the level of sufficiency. This strengthens the hypothesis that the peer assistance provided during lab classes positively impacts students' performance.



■ **Figure 1** Percentage of students with final grades within each grade interval and grouped by school year.

Finally, to assess the impact of our progressive learning methodology on the knowledge acquired by students during the course, we analysed their exam grades. Since the exam covers all taught subjects and is conducted at the end of the course, the grades obtained provide a good indicator of the knowledge acquired and skills developed. As such, we computed the average exam grades among students with a lab level of 5 or higher during the 2020 and 2021 school years. We also calculated the percentage of those students that scored 50% or higher on their exams. The results obtained are shown in Figure 2. For the school year 2023, the exam grade corresponds to the course grade, since no theoretical or laboratory assignments were requested.



(a) Exam grade average.

(b) Percentage of students with an exam grade ≥ 50 .

■ **Figure 2** Overview of exam grade statistics for students which achieved level of 5 or higher in the 2020 and 2021 school years, and for all assessed students in 2023.

The statistics presented in Figure 2 demonstrate the impact of implementing the Kumon-inspired progressive learning approach on exam grades, particularly among students with a lab level of 5 or higher during the 2020 and 2021 school years. Those who had access to this approach showed notably higher exam grades compared to those who did not. Moreover, among these students, those who were requested to complete at least 50% of the theoretical assignments saw even more significant improvements, as evidenced by the data from the 2021 school year. More specifically, in 2021, students with a lab level of 5 or higher achieved an average exam grade of 61. Approximately 75% of these students attained an exam grade of 50% or higher, indicating a high success rate for those who followed this approach. In

contrast, when the progressive learning approach was not applied, students achieved an average exam grade of 51%, with 64% of them reaching an exam grade of 50% or higher. This translates to an increase of more than 10% in exam approval rates between the school year when the Kumon-inspired methodology was employed and the theoretical assignments were required, and the year when students did not have access to the progressive learning method. These findings strongly suggest that our new methodology enhances the teaching-learning process and improves students performance in final examinations. When comparing the exam grades between the 2020 and 2023 school years, it becomes evident how the implementation of progressive learning positively influenced student performance in final examinations. Notably, in 2020, students were required to complete the first five laboratory assignments successfully, whereas no such prerequisite was imposed in 2023. This observation substantiates the positive impact of the progressive learning approach on students' achievement.

5.1 Threats to validity

Testing and validating new teaching methodologies in a school environment has its own challenges, due mainly to ethical considerations, since all students should be given the same opportunities and conditions to learn but also because of all external factors which can not be controlled and may impact the students' success (e.g. socio-economic context).

Considering this, we have identified the following main factors that may have had an impact in the results obtained:

- **COVID-19 pandemic:** During the school years in which the Kumon-inspired approach was applied (2020 and 2021), some lockdowns were imposed in Portugal. However, Programming I is a course taught in the first semester, which was not directly affected by those lockdowns. Moreover, in those schools years, the theoretical classes of Programming I were delivered online synchronously, and the laboratory classes were conducted face-to-face. This allowed to fully implement our Kumon-based approach despite the imposed restrictions. Therefore, the results obtained are considered robust, and we believe they were not significantly affected by external pandemic-related factors.
- **Socio-economic context:** It is well known that the socio-economic background can significantly impact students' success. In the specific case of our Kumon-inspired approach, students are required to actively participate in their learning process by practising laboratory assignments outside of classrooms and completing theoretical assignments outside of class as well. Students from different socio-economic backgrounds may have varying levels of access to resources such as computers and internet connectivity, which can negatively impact their learning outcomes. However, at our faculty, students have 24-hour access to computer laboratories and can also work anywhere within the faculty premises using their personal computers with free internet connection. Although this does not guarantee equal conditions for all students, it does help to minimize disparities. Additionally, our study benefited from the participation of a total of 937 students, which enhances its validity and reliability.
- **Variability in exams difficulty levels:** The difference in the levels of difficulty among exams across different school years may have influenced the outcomes observed in our study. Specifically, we acknowledge that exams administered after 2020 were generally more challenging compared to those in 2019. However, this reinforces the advantages of our Kumon-inspired approach, as students were able to achieve better overall results despite having more difficult exams.

6 Conclusion

This study demonstrates the potential of combining the strengths of the Kumon method with the automatic assessment abilities of online judge systems. Our approach encompasses three different types of assignments aligned with the principles of Kumon. Theoretical assignments encourage active participation in theoretical classes and solidify newly acquired knowledge. Worksheet assignments promote frequent study and foster self-learning skills. Finally, laboratory assignments provide progressive learning opportunities to increase self-confidence. An online judge system facilitates the implementation of our strategy, providing automated assessment and immediate feedback. Together, these strategies enable the achievement of most of the objectives of the Kumon method, fostering a successful teaching-learning process.

In addition to outlining the methodology, results from school years with our approach implemented showed significant improvements in student performance and engagement. More specifically, the 2021 school year stands out as the most successful, with the highest percentage of approved students among the ones assessed and enrolled. This success can be attributed to our progressive approach and the requirements set forth by our methodology, including the completion of theoretical and laboratory assignments, which encouraged active participation and self-learning. Furthermore, analysis of students' performances revealed a notable increase in the percentage of students achieving excellent grades in the 2021 school year compared to other years. Conversely, the percentage of students with insufficient performance decreased significantly, indicating an overall improvement in student outcomes. Moreover, our analysis of exam grades suggests a significant impact of our progressive learning methodology on students' performance. Students who participated in the Kumon-inspired approach exhibited higher exam grades.

In the future, it would be interesting to investigate students' perceptions of this new methodology, for example, through interviews. Students who have experienced more traditional programming approaches would provide valuable insights in this context. Their feedback may help to further understand the effectiveness of our methodology and guide the design of new approaches. It would also be important to assess the developed skills, habits, and abilities, particularly in the areas of self-learning, self-regulation, study habits, self-confidence, and motivation levels. Additionally, studying the impact of peer-assisted learning could help to understand collaborative learning dynamics and their influence on academic achievement. Conducting such research could help to refine educational practices and promote effective learning environments for programming fundamentals courses.

References

- 1 Kirsti M Ala-Mutka. A survey of automated assessment approaches for programming assignments. *Computer Science Education*, 15(2):83–102, 2005. doi:10.1080/08993400500150747.
- 2 Aldrich Ellis Asuncion, Brian Christopher Guadalupe, and Gerard Francis Ortega. The abc workbook: Adapting online judge systems for introductory programming classes. In *Proceedings of the 30th International Conference on Computers in Education*, volume 2, pages 395–400. IEEE, 2022. URL: https://icce2022.apsce.net/uploads/P2_W05_052.pdf.
- 3 Yoram Bosse, David Redmiles, and Marco A. Gerosa. Pedagogical content for professors of introductory programming courses. In *Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education*, ITiCSE '19, pages 429–435, New York, NY, USA, 2019. Association for Computing Machinery. doi:10.1145/3304221.3319776.
- 4 Chin-Soon Cheah. Factors contributing to the difficulties in teaching and learning of computer programming: A literature review. *Contemporary Educational Technology*, 2020.

- 5 Rodrigo Duran, Jan-Mikael Rybicki, Juha Sorva, and Arto Hellas. Exploring the value of student self-evaluation in introductory programming. In *Proceedings of the 2019 ACM Conference on International Computing Education Research*, ICER '19, pages 121–130, New York, NY, USA, 2019. Association for Computing Machinery. doi:10.1145/3291279.3339407.
- 6 Stephen H. Edwards and Manuel A. Perez-Quinones. Web-cat: automatically grading programming assignments. In *Proceedings of the 13th Annual Conference on Innovation and Technology in Computer Science Education*, ITiCSE '08, page 328, New York, NY, USA, 2008. Association for Computing Machinery. doi:10.1145/1384271.1384371.
- 7 José Figueiredo and Francisco José García-Peñalvo. Strategies to increase success in learning programming. *2022 International Symposium on Computers in Education (SIIE)*, pages 1–6, 2022. URL: <https://api.semanticscholar.org/CorpusID:254911096>.
- 8 Anabela Gomes and Antonio Mendes. Learning to program - difficulties and solutions. In *Proceedings of the International Conference on Engineering Education – ICEE 2007*, pages 283–287, January 2007.
- 9 Petri Ihanntola, Tuukka Ahoniemi, Ville Karavirta, and Otto Seppälä. Review of recent systems for automatic assessment of programming assignments. In *Proceedings of the 10th Koli Calling International Conference on Computing Education Research*, Koli Calling '10, pages 86–93, New York, NY, USA, 2010. Association for Computing Machinery. doi:10.1145/1930464.1930480.
- 10 Mike Joy, Nathan Griffiths, and Russell Boyatt. The boss online submission and assessment system. *J. Educ. Resour. Comput.*, 5(3):2–es, September 2005. doi:10.1145/1163405.1163407.
- 11 Hieke Keuning, Johan Jeuring, and Bastiaan Heeren. Towards a systematic review of automated feedback generation for programming exercises. In *Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education*, ITiCSE '16, pages 41–46, New York, NY, USA, 2016. Association for Computing Machinery. doi:10.1145/2899415.2899422.
- 12 Alain Kabo Mbiada, Bassey Isong, Francis Lugayizi, and Adnan Abu-Mahfouz. Introductory computer programming teaching and learning approaches: Review. In *2022 International Conference on Electrical, Computer and Energy Technologies (ICECET)*, pages 1–8, 2022. doi:10.1109/ICECET55527.2022.9873427.
- 13 L. Orcos, R. M. Hernández-Carrera, M. J. Espigares, and Á. Alberto Magreñán. The kumon method: Its importance in the improvement on the teaching and learning of mathematics from the first levels of early childhood and primary education. *Mathematics*, 7(1), 2019. doi:10.3390/math7010109.
- 14 José Carlos Paiva, José Paulo Leal, and Ricardo Queirós. Authoring game-based programming challenges to improve students' motivation. In Michael E. Auer and Thrasylvoulos Tsiatsos, editors, *The Challenges of the Digital Transformation in Education*, pages 602–613, Cham, 2020. Springer International Publishing.
- 15 Mário Pinto and Teresa Terroso. Learning Computer Programming: A Gamified Approach. In Alberto Simões and João Carlos Silva, editors, *Third International Computer Programming Education Conference (ICPEC 2022)*, volume 102 of *Open Access Series in Informatics (OASIS)*, pages 11:1–11:8, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/OASIS.11.11.11.
- 16 Yizhou Qian and James Lehman. Students' misconceptions and other difficulties in introductory programming: A literature review. *ACM Trans. Comput. Educ.*, 18(1), October 2017. doi:10.1145/3077618.
- 17 Atajan Rovshenov and Firat Sarsar. Research trends in programming education: A systematic review of the articles published between 2012-2020. *Journal of Educational Technology and Online Learning*, 6(1):48–81, 2023. doi:10.31681/jeto1.1201010.
- 18 Ján Skalka, Martin Drlík, and Juraj Obonya. Automated assessment in learning and teaching programming languages using virtual learning environment. In *2019 IEEE Global Engineering Education Conference (EDUCON)*, pages 689–697, 2019. doi:10.1109/EDUCON.2019.8725127.

- 19 Jaime Spacco, Paul Denny, Brad Richards, David Babcock, David Hovemeyer, James Moscola, and Robert Duvall. Analyzing student work patterns using programming exercise data. In *Proceedings of the 46th ACM Technical Symposium on Computer Science Education, SIGCSE '15*, pages 18–23, New York, NY, USA, 2015. Association for Computing Machinery. doi:10.1145/2676723.2677297.
- 20 Python standard library. Doctest library documentation. URL: <https://docs.python.org/3/library/doctest.html>.
- 21 Nancy Ukai. The kumon approach to teaching and learning. *Journal of Japanese Studies*, 20:87, 1994. URL: <https://api.semanticscholar.org/CorpusID:150129343>.
- 22 Usmedi, Amelia Agita, and Ergusni. The effect of application kumon learning method in learning mathematics of ability troubleshooting mathematics of students. *Journal of Physics: Conference Series*, 1429(1):012005, 2020. doi:10.1088/1742-6596/1429/1/012005.
- 23 Pedro Vasconcelos and Rita P. Ribeiro. Using Property-Based Testing to Generate Feedback for C Programming Exercises. In Ricardo Queirós, Filipe Portela, Mário Pinto, and Alberto Simões, editors, *First International Computer Programming Education Conference (ICPEC 2020)*, volume 81 of *Open Access Series in Informatics (OASICs)*, pages 28:1–28:10, Dagstuhl, Germany, 2020. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/OASICs.ICPEC.2020.28.
- 24 Szymon Wasik, Maciej Antczak, Jan Badura, Artur Laskowski, and Tomasz Sternal. A survey on online judge systems and their applications. *ACM Comput. Surv.*, 51(1), January 2018. doi:10.1145/3143560.