B2A, A Straight-Through Approach from Basics to Advanced Level: Case Study for Python Course*

Abbasi Shazmeen[1,*,†], Pavle Dakić[2,3,*,†] and Alwahab Dhulfiqar Zoltan[1,*,†]

[1]*Department of Programming Languages and Compilers, Faculty of Informatics, Eötvös Loránd University, 1/C Pázmány Péter st., Budapest, H-1117, Hungary*

[2]*Faculty of Informatics and Computing, Singidunum University, Danijelova 32, Belgrade, Serbia*

[3]*Institute of Informatics, Information Systems and Software Engineering, Faculty of Informatics and Information Technologies, Slovak University of Technology in Bratislava, Ilkovičova 2, 842 16 Bratislava, Slovakia*

Abstract

In this study, we present a straightforward teaching strategy for teaching the Python programming language. Our method emphasizes a clear progression from fundamental to sophisticated concepts. Beginning with a thorough explanation of core Python ideas, we show a beginner-level script that dissects each line of code and displays the results. One distinguishing feature of our system is that students use ChatGPT to optimize code, which encourages them to improve the core program with fewer lines. After getting the updated solutions, students carefully compare the original and optimized programs to discover improvements and potential flaws. This study underlines the effectiveness of the Straight-Through Approach, which focuses on improving Python understanding while also developing critical thinking and code optimization skills. Our findings show that this strategy is a simple and effective way for teachers to assist students in growing from basic to advanced Python abilities, resulting in a dynamic and successful learning environment. The proposed technique will teach students how to use AI-driven modules correctly, such as ChatGPT.

Keywords

Python Programming Language, Teaching Method, Knowledge management, Code Optimization, ChatGPT Integration, LLMs Testing System, Critical Thinking Skills

1. Introduction

In the ever-changing landscape of education, technology is crucial in transforming traditional methods and uncovering new opportunities for improving learning [1, 2]. Recent studies emphasize the transformative potential found within the intricate balance between teaching methods and tech advances.

During this transformative journey, the field of education faces a powerful force called artificial intelligence (AI). AI quietly infiltrates educational strategies beyond its traditional ties to complex algorithms and machine learning. Integrating it holds potential for customized learning, smart tutoring programs, and flexible evaluations [3, 4]. This article explores the diverse functions of artificial intelligence without explicitly mentioning it, seeking to uncover its hidden benefits in educational innovation. Furthermore, the intersection of teaching programming and AI creates a field ready for further investigation. One of the most significant parts is the knowledge management and processes, which requires an applied software approach based on real-world examples from practice [5].

Incorporating AI into coding practices offers educators a dynamic learning experience as they navigate evolving pedagogical demands. This method helps develop programming skills and promotes critical thinking and problem-solving. Recent research demonstrates how effective AI-driven methods are in helping with code optimization, debugging, and providing personalized feedback [6, 7].

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

*Corresponding author.

†These authors contributed equally.

a1jua9@inf.elte.hu (A. Shazmeen); pavle.dakic@stuba.sk (P. Dakić); dolfi@inf.elte.hu (A. D. Zoltan)

https://aalwahab.web.elte.hu/ (A. D. Zoltan)

0009-0009-4574-5819 (A. Shazmeen); 0000-0003-3538-6284 (P. Dakić); 0000-0002-7893-6250 (A. D. Zoltan)

The main challenge with this style of learning is that students must have a certain level of knowledge and experience with any programming language in order to ask specific questions to ChatGPT. Students had great experiences as a result of structured lessons, and they recognized the need of asking explicit questions that might potentially lead to improved results and code that answers the required problem. During our research, we used the Straight-Through Approach from Basics to Advanced (B2A) technique to assess students' knowledge and logical thinking.

This paper explores the intricate links between instructional methods, technological progress, and the various uses of AI in programming education, going beyond direct AI discussions. Our goal is to enhance the conversation on creative teaching methods by combining ideas from different academic sources, highlighting the importance of understanding the hidden benefits of AI in education and its impact on writing programs. Through this examination, we aim to offer a thorough grasp of the changing environment where education, programming, and technology come together. From another perspective, this technique will demonstrate to students the correct use of AI-driven modules. Consequently, students will also be able to independently learn other programming languages such as Java, C++, and more.

The work is organized as follows: material and methods, literature review, instructor-led teaching, results, discussion and conclusions.

2. Material and methods

We defined our investigation with a questionnaire that contained ten questions. The students responded, and we collected the results before analyzing the dataset using statistical methods, which led to the B2A method's conclusions. The dataset was developed based on the answers provided by the students, allowing us to better define the difficulty of the tasks that should be completed inside our research and classroom. We used a systematic technique approach dubbed A Straight-Through Approach from Basics to Advanced Level - B2A to maximize the educational benefits of ChatGPT for teaching Python programming. The method has five critical points:

1. Instructor-Led Teaching
2. Task to Code Optimization
3. Creating Student Reports
4. Examination of Differences
5. Sending and Assessment

All of the above key points are explained in more detail in the sections that follow.

3. Literature Review

To gain a better understanding of the process of designing tools for learning Python with ChatGPT alongside related technologies, relevant works dealing with the challenges of learning and transferring knowledge to users were reviewed. By merging these authoritative sources, our study adds to the current body of knowledge in the field, offering a solid foundation and adding to the academic conversation on teaching programming languages.

The application of ChatGPT in various fields has garnered significant interest in recent years. Authors Zhang et al. [8] conducted a study to investigate the effectiveness of applying ChatGPT in dialogic teaching using electroencephalography. The study included undergraduate students who were placed into two groups: one interacting with ChatGPT and one that dealt with human teachers. While both groups performed similarly on retention tests, the group that engaged with ChatGPT performed worse on transfer tests. The study concluded that ChatGPT could assist students build a knowledge foundation and boost cognitive activity, but its ability to promote knowledge application and creativity was restricted. The findings show that combining ChatGPT along with conventional human teachers may be a more effective way to improve teaching quality.

This needs data integration across industries using CI/CD, dynamic microservices, and containers to facilitate knowledge transfer across a wide range of instruments. This includes data collection for kinetics experiments, parameter fitting and estimate, model construction, evaluation, and cluster validation [9]. Authors Badenhorst et. al. [10] present a workflow for completing this assignment that makes use of the Python computer language, notably the SciPy stack modules.

In contrast to other emerging studies that assess the accuracy of LLMs like ChatGPT on tasks from a selected area Górecki [11] investigate ways how to achieve a successful solution of a standard statistical task in a collaboration of a human-expert and artificial intelligence (AI). Through careful prompt engineering, Also, Górecki [11] covered how to separate successful solutions generated by ChatGPT from unsuccessful ones, resulting in a comprehensive list of related pros and cons.

Interest in natural language processing, specifically large language models, for clinical applications has exploded in a matter of several months since the introduction of ChatGPT. From the study of authors Gabriel et. al. [12] we can see that is important that understand the strengths and limitations of the rapidly evolving technology.

4. Instructor-Led Teaching

In circumstances where Instructor-Led Teaching is used, the Python programming language is taught using a basic teaching technique. As a result, the professor is able to establish and initiate communication with students. The teacher initiates the lesson by explaining a particular Python subject (like, list , tuple, set. class variable, instance variable, etc), clarifying ideas, and enhancing understanding of the core elements of the language.

After discussing, the teacher displays a Python program written in a typical, straightforward format, frequently split across multiple lines. This method indicates that a significant contribution to software engineering has been achieved by inventing a novel learning approach that uses previously unknown software capabilities. Specifically, by asking questions to the trained model contained in the current version of ChatGPT, the student can receive precise advice as well as the option to partially or completely complete the prescribed tasks. Through the aforementioned, everyone participates together in the creation and improvement of the teaching process.

The method consists of following key points:

1. Task to Code Optimization: students are assigned the job of utilizing ChatGPT to restructure the code that was initially shown. Their goal is to refine the code, making it more professional and concise by harnessing ChatGPT's features to improve the structure and decrease the line count.
2. Creating Student Reports: after finishing the re-factoring (optimization) job, students focus on understanding the optimized code version. Students must then create a comprehensive report explaining their comprehension of the optimized code. This report provides information on the modifications, the reasons for them, and the difficulties faced while restructuring.
3. Examination of Differences: students enhance their understanding by comparing the original code with the re-factored version using ChatGPT. This evaluation includes recognizing enhancements, streamlining, and possible compromises brought by ChatGPT, promoting a detailed comprehension of the optimization procedure.
4. Sending and Assessment: the last stage is when students hand in their reports, which represent their understanding of the optimized code, to the instructor for assessment. The teacher evaluates both the technical accuracy of the revised code and the level of comprehension shown in the student reports, offering helpful feedback to improve learning.

4.1. Teachers Can Discuss Student Submissions Aloud

The teacher evaluates both the technical accuracy of the optimized code and the level of comprehension shown in the student reports, giving helpful feedback to improve the learning experience. With this practical approach, our goal is to demonstrate the efficiency of the Straight-Through method, in which

ChatGPT acts as a useful instrument in fast-tracking students' advancement from beginner to expert stages in Python coding. Figure 1, displays the 6 steps that were mentioned.

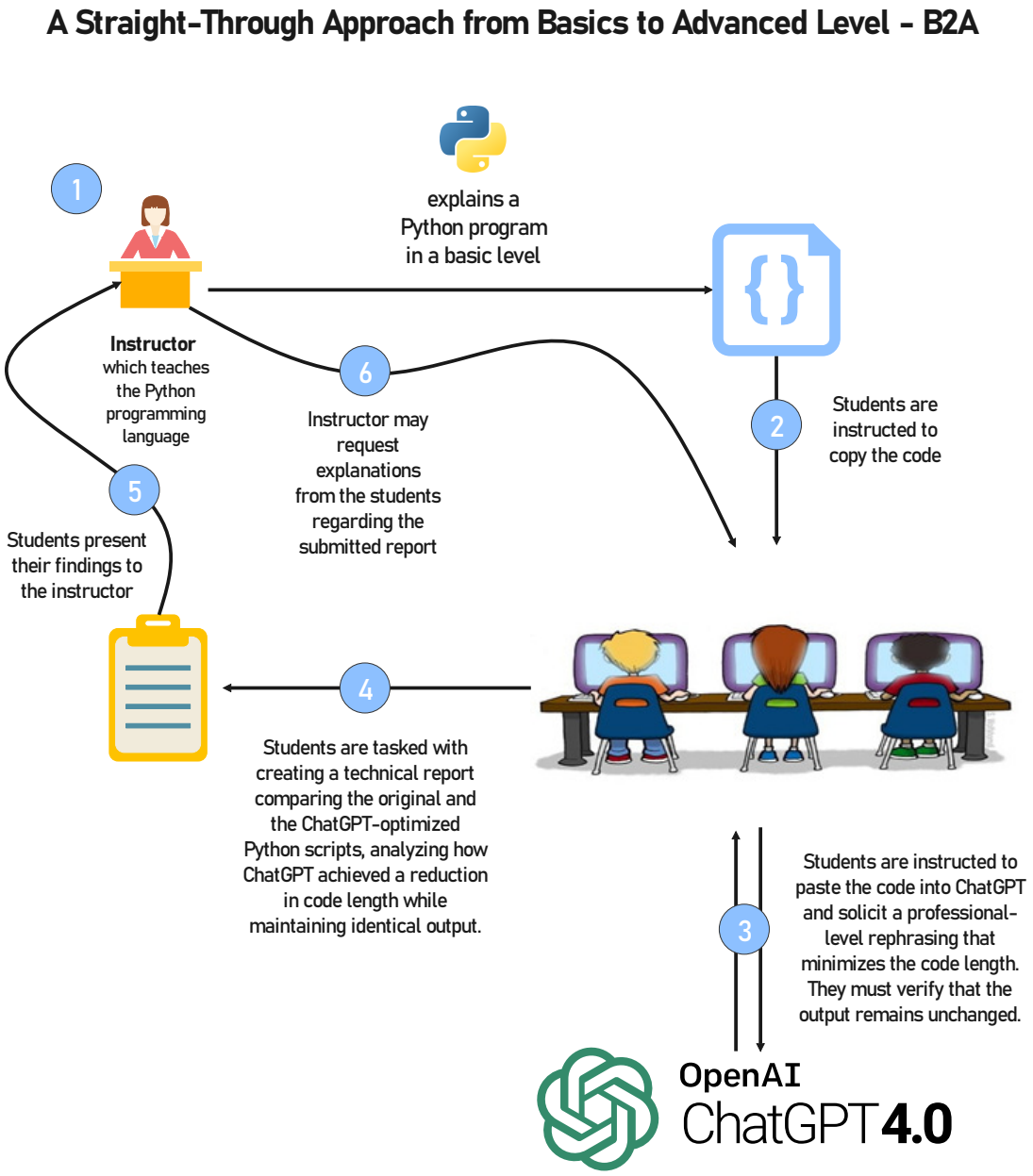

Figure 1: The 6 step of B2A; A Straight-Through Approach from Basics to Advanced Level. Source: author's contribution.

The entire process can be explained in more detail in the following 6 steps:

1. The instructor explains a Python program written in a basic level, and shows the output to them
2. Students are instructed to paste the code into ChatGPT and solicit a professional-level rephrasing that minimizes the code length. They must verify that the output remains unchanged.
3. Using Python script which is on advanced level.
4. Students are tasked with creating a technical report comparing the original and the ChatGPT-optimized Python scripts, analyzing how ChatGPT achieved a reduction in code length while maintaining identical output.
5. Students present their findings to the instructor, elucidating the conclusions drawn from their analysis of the Python scripts before and after ChatGPT's optimization.
6. Instructor may request explanations from the students regarding the submitted report.

All these steps are linked together, demonstrating the progression from educational material in its original form to finalized multiple-choice questions, with the utilization of sophisticated NLP methods and tools in a Python programming environment.

5. Results

Based on the previously reviewed literature review and starting points, within this section we present the achieved results. Using a case study, we attempt to highlight the potential of direct AI application for the learning needs of Python programming language.

5.1. Case Study: Improving Code Optimization in Python Instruction with the Help of ChatGPT

In this part, we offer a detailed case study demonstrating the use of our teaching method, B2A emphasizing the integration of ChatGPT for enhancing (optimizing) Python code in teaching. The case study progresses using a methodical approach, showing how it improves the student learning experience. This case study done on different groups with different languages (English, Hungarian, Slovenian, and Slovakian)

For applied Methodology we have used the six steps of the B2A method were applied to students in the Python course[1], using the course as a pilot for the method. The method was delivered according to the following six steps.

5.1.1. Instructor-Led Teaching

The instructor starts by teaching certain Python subjects (In this context we use the List subject), offering clear explanations of basic principles. Students are introduced to a standard Python program, which serves as a fundamental code for optimization. Students are taught a fundamental List program to identify the biggest number out of three inputs. The code provided in figure Listing 1 acts as the initial step for future optimization projects.

5.1.2. ChatGPT Optimizing Task

Students utilize ChatGPT to restructure the given code, striving for a more polished and optimized version. The updated code, produced by ChatGPT. Code listing 2 shows the optimized version of the list code showed in code listing 1.

Listing 1: Python program shows the basic use of list. Source: author's contribution.

```
"" # Input three numbers from the user
  number1 = int(input("Enter the first number: "))
  number2 = int(input("Enter the second number: "))
  number3 = int(input("Enter the third number: "))

  # Assume the first number is the largest initially
  largest_number = number1

  # Compare with the second number
  if number2 > largest_number:
  largest_number = number2

  # Compare with the third number
  if number3 > largest_number:
  largest_number = number3

  # Print the largest number
  print("The largest number is:", largest_number)
```

[1] https://python.inf.elte.hu/index.html

Within the displayed code in listing 1, students can try to use the if function through a short example. The given code example analyzes the entered values and shows which is the largest number.

Listing 2: Optimized code for the program shown in the Listing 1. Source: author's contribution.

```
"" # Input three numbers from the user and store them in a list
  numbers = [int(input(f"Enter number {i+1}: ")) for i in range(3)]

  # Print the largest number from the list using the max() function
  print("The largest number is:", max(numbers))
```

5.1.3. Creating student reports

Students are tasked with exploring optimized code and producing a comprehensive report. This report should detail the modifications made, the rationale behind them, and the challenges encountered during refactoring. In this exercise, students will observe that ChatGPT employs list comprehension—an advanced Python technique—to optimize the code. Consequently, students should learn about list comprehension, gather examples, and prepare questions for the next class. This preparation will enable the instructor to address their questions and discuss list comprehension more effectively during the lessons.

5.1.4. Comparison Analysis

Students compare the original code with the ChatGPT-optimized code, identifying enhancements, optimizations, and potential trade-offs. This exercise fosters a deeper understanding of the optimization process. Students should consolidate their findings, ideas, and questions into a report for evaluation by their teachers.

5.1.5. Sending and Assessing

Students submit their reports to the teacher for assessment. The evaluation includes checking the accuracy of the revised code and the depth of understanding demonstrated in the students' evaluations. Constructive feedback is provided to enhance the learning experience. Additionally, teachers can orally question students about list comprehension to gauge their understanding of the topic.

5.1.6. Assisting in providing verbal feedback on student submissions

In this case, the teacher can assess the report that was turned in and interact with the students. ChatGPT used list comprehension to shorten the list. Students have not previously been taught the advanced Python technique of list comprehension in basic code. Through this method, positive students' and teacher experiences in the learning process are obtained with the possibility of continuous improvement of the learning process. In this situation, ChatGPT used list comprehension to make the management of lists more efficient. List comprehension is a sophisticated functionality in Python that enables the succinct and effective generation, modification, and traversal of lists. ChatGPT efficiently decreased the complexity of list operations by utilizing list comprehension, showcasing advanced skills in Python programming. At this level, teachers can say during the lesson: "Let's explore further the idea of list comprehension and how it can be used in Python programming".

The results of the case study show that the B2A approach is effective in using ChatGPT to accelerate students' transition from basic to advanced Python programming. By continuously improving code, students develop coding skills and understand the importance of efficiency and professionalism. Using ChatGPT in programming education greatly supports the course's pedagogical objectives. In this case study, we highlight the educational advantages of incorporating ChatGPT into Python teaching, especially when focusing on improving code efficiency. The method enables students to smoothly transition and achieve a comprehensive and enduring level of expertise in Python programming. The

research adds to the discussion on creative teaching methods, highlighting the hidden benefits of AI-powered tools in the realm of programming learning.

5.2. Planning and Design of Slides for Python Course

When developing the Python course content, we focused on designing instructional slides to enhance student learning and engagement. Figure 2 provides an example of this design approach. Each slide in the course is systematically structured to improve comprehension and practical application of Python programming concepts. As illustrated in Figure 2, a sample slide is divided into four clear sections.

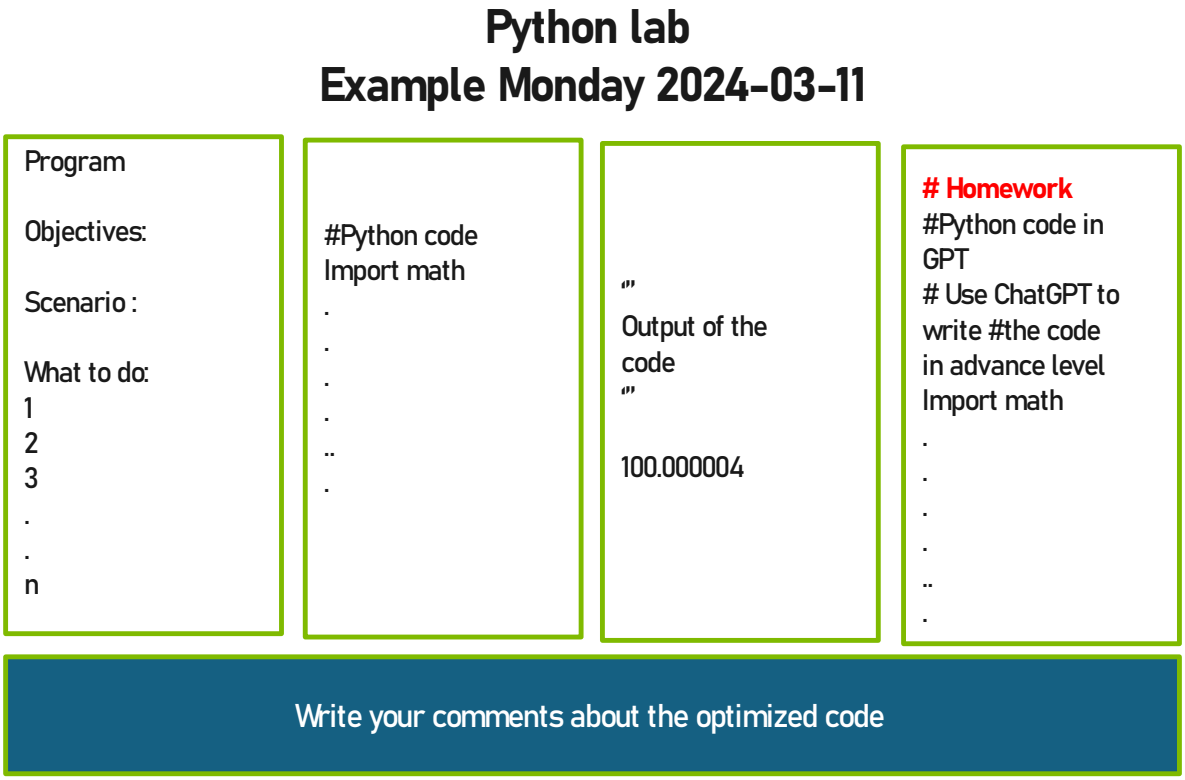

Figure 2: Prototype of the lesson slide. Source: author's contribution.

5.2.1. Goals and Scenarios

The initial section, placed in the far left block, describes the goals for the Python task to be completed. Underneath the goals, a scenario gives context to the programming assignment, giving students a practical way to grasp its relevance in the real world. Right below the given scenario, the task that needs to be completed is clearly outlined. This brief explanation of the assignment helps students understand the main goal and get ready for the coding task.

5.3. Coding Implementation

The main focus of the slide is showcasing the code related to the assignment. The code is deliberately written in a simple layout to make it easier for instructors to provide detailed explanations during the course. This method promotes engagement and understanding among students by paying attention to the instructor's explanation of the code logic.

5.4. Output of the simple code

The third section of the slide is dedicated to displaying the output of simple code. This allows students to compare this output with that of the optimized code, ensuring both outputs are precisely the same. This comparison is crucial for students to understand the effectiveness of the optimizations made.

5.5. Enhanced Code and Student Feedback

Found in the top right and bottom right sections, these parts offer chances for students to get involved and think critically. Students have been assigned the responsibility of enhancing the given code by using the features of ChatGPT, a cutting-edge language model. The enhanced code is then positioned in the top right section, enabling students to analyse their answers against the original implementation. Afterwards, students are encouraged to share their thoughts and insights on the improved code in the bottom right section, promoting a greater grasp of coding standards and algorithmic effectiveness.

We hope that organizing our slides in this way will help students fully understand Python programming concepts and improve their ability to write efficient and elegant code in an engaging and interactive learning environment. Additionally, this approach encourages interactivity by allowing students to write on their slides and add notes, which can aid in retention when they prepare for exams.

5.6. Application of Methodology on Hungarian, Slovakian, Slovenian and English Groups

We applied the approach to four different separate groups: The groups were asked the same questions in a survey conducted via Google Sheets. Table 1 summarizes the objective of each question. The survey questions aimed to assess the students' reactions to this teaching style. The overall results indicate that the B2A approach is well-received by the students and can be effectively applied across all groups as a ChatGPT-based teaching style for Python language and other programming languages.

Table 1
Survey Questions and Their Purpose

Q. Number	Question Summary	Purpose of Question
1	Overall Experience Rating	Gauge overall satisfaction and identify improvement areas
2	Contribution of ChatGPT	Assess the impact on understanding advanced Python concepts
3	Confidence in Code Writing	Evaluate the influence on participants' confidence levels
4	Enhancement of Critical Thinking	Identify influence on critical thinking and problem-solving
5	Effectiveness of Program and Tasks	Assess perceived effectiveness in bridging proficiency gap
6	Acceleration in Learning	Understand perceived impact on learning pace
7	Effectiveness of Comparative Analysis	Evaluate pedagogical value in reinforcing code optimization
8	Comprehensive Learning Experience	Measure overall efficacy of the instructional approach
9	Impact of Methodology on Communication	Assess methodology's effectiveness in enhancing communication

5.7. Including Examining system based on LLM

Large Language Models (LLMs) [13] are planned to be utilized to generate random questions from provided PDF lectures. This approach aims to automate the examination system, eliminating the need for

teachers to handle question generation and corrections. LLMs will randomly assign different multiple-choice questions (MCQs) to students. The testing system is based on the virtualization methods [14].

The implementation of this system is currently underway. Figure 3 illustrates the architecture of the system. Starting process is with creating multiple-choice questions through the implementation of text embeddings and tokens in Python environments is made possible with OpenAI technology. Important resources include Jupyter notebooks, Hugging Face libraries, and OpenAI's API. Typically, the procedure involves examining text data (like Python lectures or PDFs), creating embeddings to grasp the context, and creating questions from that understanding. To reduce the delay and enhance security, we plan to host the testing system using the concept of edge computing [15] [16]

As with any new technology there are certain upsides or risks to the validity and accuracy of the results obtained by ChatGPT. Which therefore requires a manual check by an expert in the field that one is trying to learn or apply in a practical sense. Because of this, the model within ChatGPT requires manual learning or supervised learning, which in a later connection can produce results with a higher percentage of accuracy.

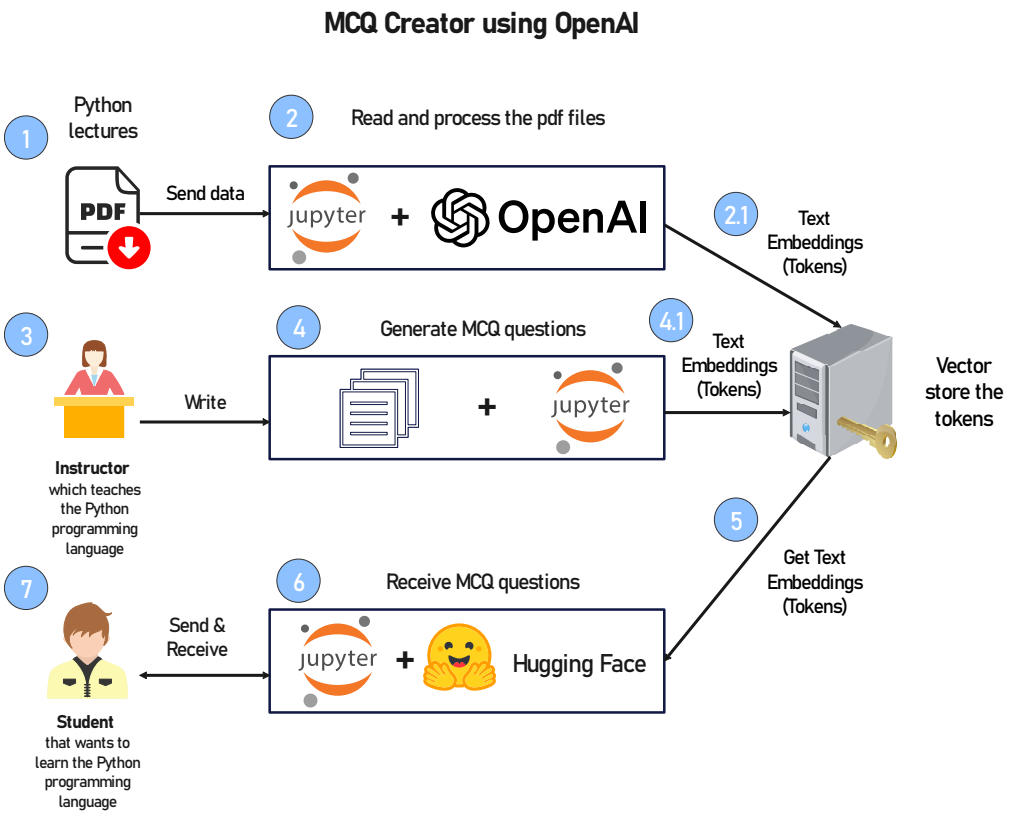

Figure 3: The testing system-based LLM. Source: author's contribution.

Shown and suggested use on Figure 3 can be explained in more detail with the following steps:

1. Input (PDF/Python Lectures):
2. Displaying the source materials being inputted into the system.
3. Processing (Text Extraction, Pre-processing):
4. The stages in which text is extracted and primed for analysis.
5. Embedding Generation (Open AI, Hugging Face):
6. Illustrating the conversion of text into embeddings.
7. Question Generation:
8. Focusing on creating multiple-choice questions using embeddings.
9. Output (Questions for Python Students):
10. Finalizing the set of questions for Python students.

All these steps are linked together, demonstrating the progression from educational material in its original form to finalized multiple-choice questions, with the utilization of sophisticated NLP methods and tools in a Python programming environment. Where students beginners can refine their search queries and become more adept at formulating the questions they ask ChatGPT with more trial and error.

6. Discussion

Engaging students more actively in the educational process is critical to improving their learning experience and academic success with programming languages. Incorporating interactive and dynamic teaching approaches can help create a more stimulating and participatory environment. One useful method is to prepare small activities and demonstrations of up to 30 minutes. This strategy has multiple benefits and can be used in various ways to increase student involvement. To expand on the early success of these tactics with discussion, it would be useful to investigate how peer teaching and collaborative projects may be scaled in bigger classroom settings. The dynamics of student interaction with the discussion aspect can considerably boost the final results in the development of soft skills in such situations, providing vital insights into learners' overall growth.

Shorter assignments assist students to keep their attention and avoid cognitive overload, resulting in improved focus and recall of material from videos or in class. Interactive demos inspire students to participate actively, which enhances their learning and fosters critical thinking skills. Short assignments also allow them to receive fast feedback from teachers, which helps students correct mistakes and reinforces learning in real time. The variety of these exercises accommodates different learning types, keeping the classroom lively and reducing boredom or lack of enthusiasm. We believe that shorter activities and interactive demonstrations of up to 30 minutes can help students engage in the educational process substantially more effectively.

Students can also explore optimized code and produce a comprehensive report detailing modifications, rationales, and challenges encountered. This task helps them observe advanced techniques like list comprehension in Python, enhancing their coding skills and analytical thinking. These approaches cater to various learning styles, making education more dynamic. Continuous assessment and feedback ensure these methods meet students' evolving needs, creating a more engaging and effective educational environment.

7. Conclusions

In conclusion, integrating ChatGPT into teaching via the B2A method simplifies the learning trajectory, eliminating the need for distinct basic, intermediate, and advanced levels. B2A encourages student engagement and self-learning while ensuring safe use of ChatGPT. Teachers find instruction more accessible with B2A, and can interact with students easily. LLMs facilitate testing by generating randomized questions.

B2A has garnered positive feedback from students across various international groups, as indicated by survey results. This approach, initially applied to a Python course, has broader applicability and can be extended to other programming languages such as Java, C++, etc., if lectures and labs are designed to suit the B2A approach.

To enhance the learning experience, students can be tasked with investigating optimized code and writing a full report outlining improvements, rationales, and issues encountered. This work allows pupils to observe sophisticated techniques such as list comprehension in Python, which improves their coding skills and analytical thinking. These approaches make learning more dynamic and accommodate a variety of learning styles. Continuous assessment and feedback guarantee that these strategies suit students' changing requirements, resulting in a more interesting and effective learning environment. The incorporation of shorter activities and interactive demonstrations into the educational process opens up various intriguing options for further investigation. Future research could look into the

long-term effects of these approaches on the retention of students and comprehension. Furthermore, studies might look into how different topics or fields benefit differently from these approaches.

Another area of concern is how technology might facilitate these interactive ways. Research could look into the usefulness of different educational apps and simulations in improving student engagement and learning results. Furthermore, the effect of immediate feedback on student performance and motivation needs further examination. Future work will involve enhancing the testing system, integrating it with Canvas, refining LLMs for reliability, and comparing multiple LLM modules to optimize the testing process.

Acknowledgments

The work of Pavle Dakić was supported with access to Erasmus+ ICM 2023 No. 2023-1-SK01-KA171-HED-000148295, the Cultural and Educational Grant Agency of Slovak Republic (KEGA) Model-based explication support for personalized education (Podpora personalizovaného vzdelávania explikovaná modelom) - KEGA (014STU-4/2024), Scientific Grant Agency of Slovak Republic (VEGA) under grant No. VG 1/0675/24, National project "Increasing Slovakia's resilience to hybrid threats by strengthening public administration capacities", project code ITMS2014+:314011CDW7 and supported by the European Social Fund, finally co-funded by the European Regional Development Fund (ERDF) and the Operational Program Integrated Infrastructure for the project: National infrastructure for supporting technology transfer in Slovakia II – NITT SK II, co-financed by the European Regional Development Fund.

References

[1] P. Dakić, D. Lojaničić, H. Issa, M. Bogavac, Choosing, creating and developing managers, Oditor 7 (2021) 105–134. doi:10.5937/oditor2103105d.

[2] H. Pallathadka, L. K. Pallathadka, T. B. Devi, T. K. Devi, S. K. Singh, 21st century approach to technology driven education-an empirical study, Int. J. of Aquatic Science 12 (2021) 5471–5477.

[3] S. Yu, Y. Lu, An introduction to artificial intelligence in education, Springer, 2021.

[4] N. Hroncova, P. Dakić, Research study on the use of CI/CD among slovak students, in: 2022 12th International Conference on Advanced Computer Information Technologies (ACIT), IEEE, 2022. doi:10.1109/acit54803.2022.9913113.

[5] P. Dakić, Importance of knowledge management for CI/CD and Security in Autonomous Vehicles Systems, 2024. URL: https://jita-au.com/index.php/2024/06/13/importance-of-knowledge-management-for-ci-cd-and-security-in-autonomous-vehicles-systems/.

[6] T. Golis, P. Dakić, V. Vranić, Automatic deployment to kubernetes cluster by applying a new learning tool and learning processes, in: SQAMIA 2023 Software Quality Analysis, Monitoring, Improvement, and Applications, 2023. URL: https://ceur-ws.org/Vol-3588/p16.pdf.

[7] D. Cambaz, X. Zhang, Use of ai-driven code generation models in teaching and learning programming: a systematic literature review, in: Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1, 2024, pp. 172–178.

[8] J. Zhang, Y. Liu, W. Cai, L. Wu, Y. Peng, J. Yu, S. Qi, T. Long, B. Ge, Investigation of the effectiveness of applying chatgpt in dialogic teaching using electroencephalography, 2024. doi:10.48550/ARXIV.2403.16687.

[9] P. Dakić, Software compliance in various industries using ci/cd, dynamic microservices, and containers, Open Computer Science 14 (2024). doi:10.1515/comp-2024-0013.

[10] M. Badenhorst, C. J. Barry, C. J. Swanepoel, C. T. van Staden, J. Wissing, J. M. Rohwer, Workflow for data analysis in experimental and computational systems biology: Using python as 'glue', Processes 7 (2019) 460. doi:10.3390/pr7070460.

[11] J. Górecki, Pair programming with large language models for sampling and estimation of copulas, 2023. doi:10.48550/ARXIV.2303.18116.

[12] R. A. Gabriel, E. R. Mariano, J. McAuley, C. L. Wu, How large language models can augment perioperative medicine: a daring discourse, Regional Anesthesia & Pain Medicine 48 (2023) 575–577. doi:10.1136/rapm-2023-104637.
[13] S. Fakhoury, A. Naik, G. Sakkas, S. Chakraborty, S. K. Lahiri, Llm-based test-driven interactive code generation: User study and empirical evaluation, arXiv preprint arXiv:2404.10100 (2024).
[14] F. Eyvazov, T. E. Ali, F. I. Ali, A. D. Zoltan, Beyond containers: Orchestrating microservices with minikube, kubernetes, docker, and compose for seamless deployment and scalability, in: 2024 11th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions)(ICRITO), IEEE, 2024, pp. 1–6.
[15] A. Dhulfiqar, M. A. Abdala, N. Pataki, M. Tejfel, Deploying a web service application on the edgex open edge server: An evaluation of its viability for iot services, Procedia Computer Science 235 (2024) 852–862.
[16] A. Dhulfiqar, N. Pataki, M. Tejfel, Chatbot-based querying of iot devices in edgex, Proceedings http://ceur-ws. org ISSN 1613 (2023) 0073.