Kashihara, A. et al. (Eds.) (2024). Proceedings of the 32nd International Conference on Computers in Education. Asia-Pacific Society for Computers in Education

The Era of Learning Programming Through Program: Challenges and Potential of ChatGPT in Revolutionizing High School Programming Education

Tzu-Chi YANG
Institute of Education/Teacher Education Center,
National Yang Ming Chiao Tung University, Taiwan
tcyang_@nycu.edu.tw

Abstract: In the digital age, the importance of programming education has become increasingly prominent. Due to the open nature of programming languages, many tools and systems designed to support programming education in the past have been limited in their ability to effectively assist in the learning process. However, the emergence of generative AI, exemplified by ChatGPT, has demonstrated remarkable natural language processing capabilities and a vast repository of knowledge, overcome previous limitations and offering unprecedented potential for advancing programming education. ChatGPT has shown near-teacher-level proficiency in providing content, debugging, refining, evaluating, implementing, and even creating programs. This signifies that we are on the cusp of an era where programming can be learned through interaction with advanced AI. However, there is still a lack of substantial research on evaluating the benefits of integrating ChatGPT into programming education. Therefore, this paper aims to illustrate, through practical examples, how the introduction of ChatGPT can impact programming education, the potential challenges it may present, and how we can prepare to embrace this new era of learning programming through AI.

Keywords: Programming education, High school, AIED, Generative artificial intelligence, ChatGPT

1. Introduction

In recent years, as the world continues to digitalize, the use of digital technology has become a norm, and understanding how digital technology works has almost become a universal activity. Programming is the foundation of digitalization, and countries around the world recognize the importance of programming education in preparing students for the future, integrating it into K-12 curricula. Programming education is becoming increasingly significant in K-12 education (Hu, 2024). High school is considered a crucial stage for developing programming skills; however, there are still several issues in high school programming education that need to be addressed (Sun et al., 2024). More specifically, unlike other subjects, students can use various programming languages to accomplish the same tasks. Each programming language has its own syntax and environment, and accommodating these diverse requirements demands a significant amount of effort with only minimal benefits. Additionally, programming practice is open-ended, even when using a single programming language, different structures can be used to achieve the same task. Traditional learning systems and tools, which are pre-built with specific learning aids, are almost incapable of handling this issue. As a result, the current burden of programming education still falls on professional teachers, and the lack of human resources for programming teachers in most high schools limits the development of programming education (Sharma & Harkishan, 2022). Providing learning tools to assist high school students in learning programming has become an important issue.

In this regard, "ChatGPT" offers knowledge and interaction akin to that of human experts (OpenAI, 2024). It has demonstrated its capability to act as a teacher or tutor in fields such as medicine, biology, language, physics, computer science, and more, including understanding, verifying, evaluating, and explaining various programming languages, as well as offering suggestions for improving programming designs. This powerful AI technology has the potential to break through previous limitations and bring transformative changes to the field of programming education (Cooper, 2023). However, more practical examples and evidence are needed to illustrate ChatGPT's revolution in programming learning. Therefore, this paper explores the benefits, challenges, and strategies of using ChatGPT in programming education through the integration of this AI tool into programming learning.

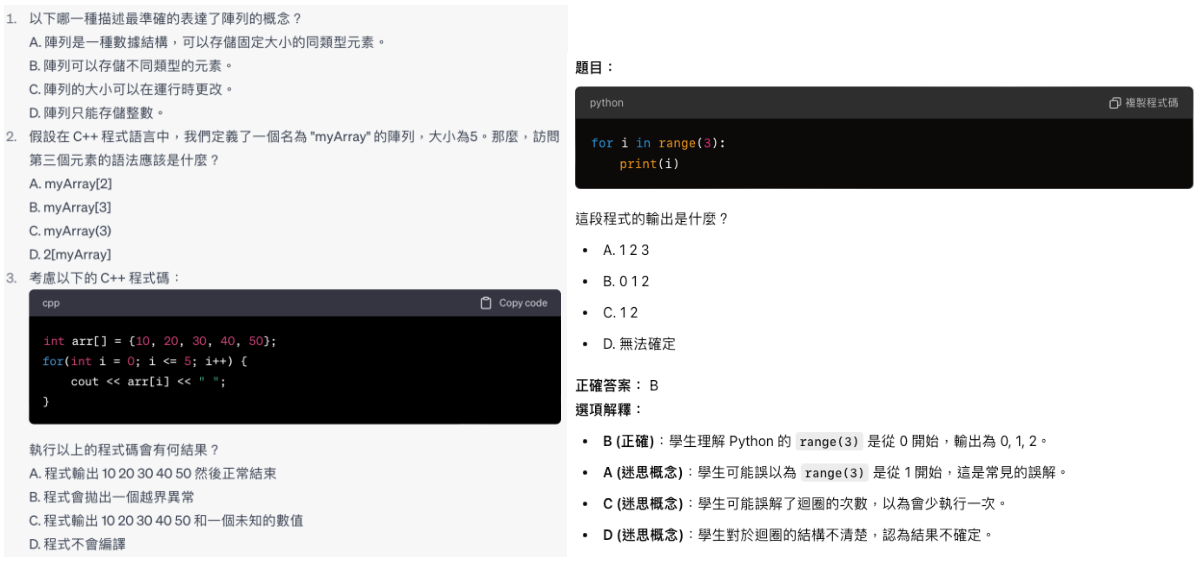
2. Related Works

Programming is a discipline that requires both skill and experience, requiring that students explore various problem-solving methods and apply their knowledge to develop applications that meet specific goals. Students must not only become familiar with programming syntax and organize this syntax to code, but they must also rely on experience to troubleshoot issues within their programs. Consequently, students often encounter difficulties and require multiple forms of support during the learning process (Hu, 2024). Without the help of a teacher, students may struggle to identify the true cause of errors, leading them to become stuck in a cycle where they are unable to progress. Previous research suggests that learners should receive continuous guidance throughout the programming process to avoid these pitfalls (McDonald, 2018). Real-time error checking and feedback during coding can significantly enhance learning outcomes. Conversely, if students do not receive timely guidance to resolve issues, they are likely to face increasing challenges in their learning (Sun & Hsu, 2019). While online assessment and feedback systems can examine the correctness of code and provide instant feedback, they may only partially mitigate this problem. Such systems can lead students to rely on trial-and-error methods, focusing on brute-force coding to solve tasks rather than understanding and applying more effective syntax structures, which ultimately limits the learning of novices (Daradoumis et al., 2019). As a result, teacher-led instruction remains the primary approach in programming education.

However, beyond the shortage of programming teachers, these educators often face the challenge of providing guidance to dozens of students within a limited amount of time, presenting significant obstacles in programming education (Woo & Falloon, 2023). ChatGPT, based on the Generative Pre-trained Transformer (GPT) model and Large Language Models (LLMs), offers a groundbreaking solution to the limitations of programming learning support tools. Unlike human languages, programming languages possess clear and finite structures and semantics, which ChatGPT can nearly fully comprehend, accurately generate, correct, execute, and evaluate without additional configuration, offering comprehensive support for learning programming languages (OpenAI, 2024). In this vein, ChatGPT, with its rich content knowledge and versatile methods, has demonstrated unprecedented potential in helping high school students learn programming knowledge and enhance their programming skills (Yilmaz & Yilmaz, 2023). Therefore, this study aims to design and implement the integration of ChatGPT into programming classrooms to explore the benefits and challenges of using ChatGPT in programming education.

3. ChatGPT-Enhanced Programming Learning Activities

Integrating ChatGPT into programming course offers significant enhancements. One of the key tasks for educators is to create assessments that accurately gauge student understanding. Given the open-ended nature of programming, students may sometimes arrive at the correct solution while harboring misconceptions or alternative understandings of concepts. This can lead to situations where students know the correct answer but don’t fully grasp why it is correct, allowing misconceptions to persist. In programming courses, it's also crucial to regularly assess whether students have these misconceptions. ChatGPT can efficiently support this process by helping to design and refine assessments. For instance, instead of traditional test questions, we can use ChatGPT to create more sophisticated distractors—incorrect options in multiple-choice questions—that are tailored to expose specific misconceptions students might have (Figure 1). This allows for a more precise identification of areas where students need further instruction, ultimately leading to more effective learning outcomes. Additionally, ChatGPT can generate personalized feedback for students based on their specific mistakes (Figure 2), guiding them through the reasoning process to correct their misunderstandings. This individualized support can be particularly beneficial in large classes where direct teacher feedback for every student may not be feasible. By leveraging ChatGPT, educators can create a more interactive and responsive learning environment that adapts to the needs of each student (Figure 3).



1. 以下哪一種描述最準確的表達了陣列的概念？
   A. 陣列是一種數據結構，可以存儲固定大小的同類型元素。
   B. 陣列可以存儲不同類型的元素。
   C. 陣列的大小可以在運行時更改。
   D. 陣列只能存儲整數。
2. 假設在 C++ 程式語言中，我們定義了一個名為 "myArray" 的陣列，大小為5。那麼，訪問第三個元素的語法應該是什麼？
   A. myArray[2]
   B. myArray[3]
   C. myArray(3)
   D. 2[myArray]
3. 考慮以下的 C++ 程式碼：

```cpp
int arr[] = {10, 20, 30, 40, 50};
for(int i = 0; i <= 5; i++) {
    cout << arr[i] << " ";
}
```

   執行以上的程式碼會有何結果？
   A. 程式輸出 10 20 30 40 50 然後正常結束
   B. 程式會拋出一個越界異常
   C. 程式輸出 10 20 30 40 50 和一個未知的數值
   D. 程式不會編譯

題目：

```python
for i in range(3):
    print(i)
```

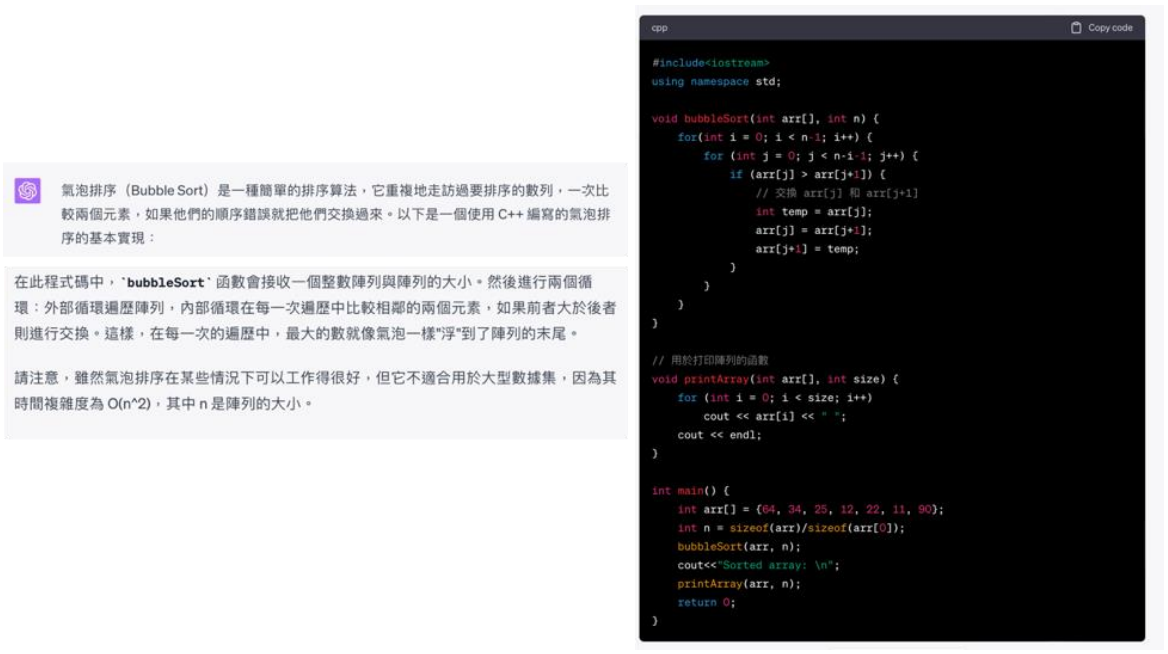
這段程式的輸出是什麼？

- A. 1 2 3
- B. 0 1 2
- C. 1 2
- D. 無法確定

正確答案： B
選項解釋：

- **B (正確)**：學生理解 Python 的 `range(3)` 是從 0 開始，輸出為 0, 1, 2。
- **A (迷思概念)**：學生可能誤以為 `range(3)` 是從 1 開始，這是常見的誤解。
- **C (迷思概念)**：學生可能誤解了迴圈的次數，以為會少執行一次。
- **D (迷思概念)**：學生對於迴圈的結構不清楚，認為結果不確定。

Figure 1. Generating formative assessments that can identify misconceptions



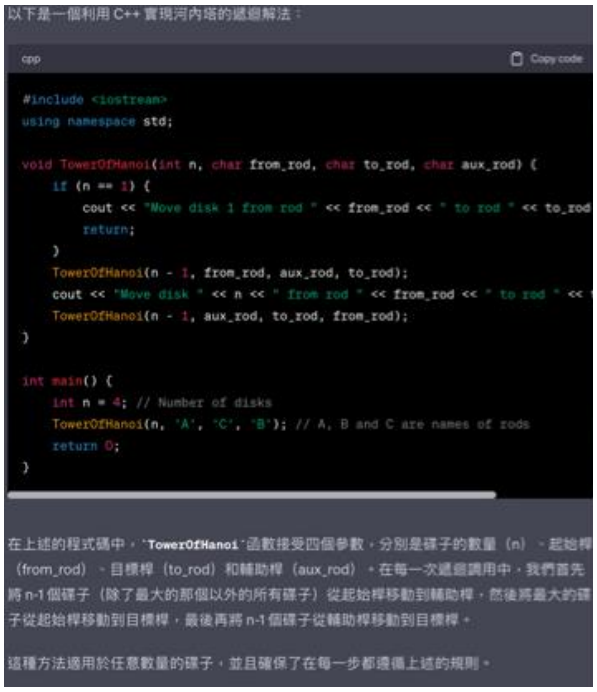
氣泡排序（Bubble Sort）是一種簡單的排序算法，它重複地走訪過要排序的數列，一次比較兩個元素，如果他們的順序錯誤就把他們交換過來。以下是一個使用 C++ 編寫的氣泡排序的基本實現：

```cpp
#include<iostream>
using namespace std;

void bubbleSort(int arr[], int n) {
    for(int i = 0; i < n-1; i++) {
        for (int j = 0; j < n-i-1; j++) {
            if (arr[j] > arr[j+1]) {
                // 交換 arr[j] 和 arr[j+1]
                int temp = arr[j];
                arr[j] = arr[j+1];
                arr[j+1] = temp;
            }
        }
    }
}

// 用於打印陣列的函數
void printArray(int arr[], int size) {
    for (int i = 0; i < size; i++)
        cout << arr[i] << " ";
    cout << endl;
}

int main() {
    int arr[] = {64, 34, 25, 12, 22, 11, 90};
    int n = sizeof(arr)/sizeof(arr[0]);
    bubbleSort(arr, n);
    cout<<"Sorted array: \n";
    printArray(arr, n);
    return 0;
}
```

在此程式碼中，`bubbleSort` 函數會接收一個整數陣列與陣列的大小。然後進行兩個循環：外部循環遍歷陣列，內部循環在每一次遍歷中比較相鄰的兩個元素，如果前者大於後者則進行交換。這樣，在每一次的遍歷中，最大的數就像氣泡一樣"浮"到了陣列的末尾。

請注意，雖然氣泡排序在某些情況下可以工作得很好，但它不適合用於大型數據集，因為其時間複雜度為 O(n^2)，其中 n 是陣列的大小。

Figure 2. Obtaining example code through ChatGPT



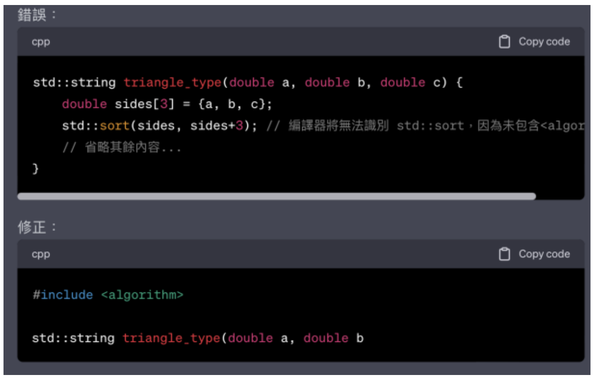
以下是一個利用 C++ 實現河內塔的遞迴解法：

```cpp
#include <iostream>
using namespace std;

void TowerOfHanoi(int n, char from_rod, char to_rod, char aux_rod) {
    if (n == 1) {
        cout << "Move disk 1 from rod " << from_rod << " to rod " << to_rod
        return;
    }
    TowerOfHanoi(n - 1, from_rod, aux_rod, to_rod);
    cout << "Move disk " << n << " from rod " << from_rod << " to rod " << t
    TowerOfHanoi(n - 1, aux_rod, to_rod, from_rod);
}

int main() {
    int n = 4; // Number of disks
    TowerOfHanoi(n, 'A', 'C', 'B'); // A, B and C are names of rods
    return 0;
}
```

在上述的程式碼中，`TowerOfHanoi` 函數接受四個參數，分別是碟子的數量（n）、起始桿（from_rod）、目標桿（to_rod）和輔助桿（aux_rod）。在每一次遞迴調用中，我們首先將 n-1 個碟子（除了最大的那個以外的所有碟子）從起始桿移動到輔助桿，然後將最大的碟子從起始桿移動到目標桿，最後再將 n-1 個碟子從輔助桿移動到目標桿。

這種方法適用於任意數量的碟子，並且確保了在每一步都遵循上述的規則。

Figure 3. Assisting in learning the Tower of Hanoi with ChatGPT.



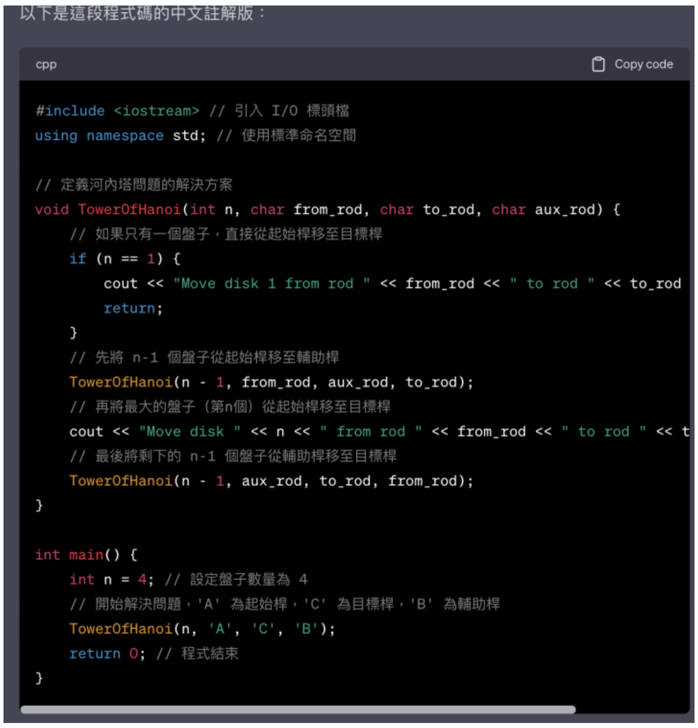
錯誤：

```cpp
std::string triangle_type(double a, double b, double c) {
    double sides[3] = {a, b, c};
    std::sort(sides, sides+3); // 編譯器將無法識別 std::sort，因為未包含<algor
    // 省略其餘內容...
}
```

修正：

```cpp
#include <algorithm>

std::string triangle_type(double a, double b
```

Figure 4. Debugging with the help of ChatGPT



以下是這段程式碼的中文註解版：

```cpp
#include <iostream> // 引入 I/O 標頭檔
using namespace std; // 使用標準命名空間

// 定義河內塔問題的解決方案
void TowerOfHanoi(int n, char from_rod, char to_rod, char aux_rod) {
    // 如果只有一個盤子，直接從起始桿移至目標桿
    if (n == 1) {
        cout << "Move disk 1 from rod " << from_rod << " to rod " << to_rod
        return;
    }
    // 先將 n-1 個盤子從起始桿移至輔助桿
    TowerOfHanoi(n - 1, from_rod, aux_rod, to_rod);
    // 再將最大的盤子（第n個）從起始桿移至目標桿
    cout << "Move disk " << n << " from rod " << from_rod << " to rod " << t
    // 最後將剩下的 n-1 個盤子從輔助桿移至目標桿
    TowerOfHanoi(n - 1, aux_rod, to_rod, from_rod);
}

int main() {
    int n = 4; // 設定盤子數量為 4
    // 開始解決問題，'A' 為起始桿，'C' 為目標桿，'B' 為輔助桿
    TowerOfHanoi(n, 'A', 'C', 'B');
    return 0; // 程式結束
}
```

Figure 5. Collaborating with ChatGPT to annotate code for better understanding.

ChatGPT simultaneously possesses outstanding evidence-based and innovative capabilities, providing guided discussions, external resources, and a variety of appropriate support, such as debugging (Figure 4) and paraphrasing explanations (Figure 5). Through this process, students can refine their answers or solutions by incorporating the improvement suggestions provided by ChatGPT.

4. Experiment Design and Data Collection

To address our research question, this study was conducted with 76 second-year students from three classes at a high school in northern Taiwan. Prior to participating in the experiment, all students had received instruction in basic programming concepts from the same teacher. At the end of the experiment, we invited the students to participate in interviews. The aim was to understand their perceptions of using ChatGPT in the process of learning programming languages and to gather insights into their specific experiences. The interview questions included, How do you feel about using ChatGPT in class compared to previous (or expected) experiences? What do you think are the advantages/disadvantages of learning with ChatGPT? Would you like to continue using ChatGPT to learn programming? Why or why not?.

5. Discussion

As the aforementioned, to gain a more comprehensive understanding of the impact of using ChatGPT in programming education, we integrated ChatGPT into a high school programming course and subsequently surveyed students about their experiences and perceptions of learning programming languages with ChatGPT. The findings revealed that directly implementing ChatGPT in an educational setting may still present some challenges. For instance, ChatGPT might diminish the learning experience of some students due to the following reasons:

Communication Barriers: When questions are vaguely phrased, ChatGPT may not fully address students' inquiries. This can lead students to spend considerable time explaining their questions rather than focusing on problem-solving. As a result, not only does it fail to assist them in resolving the issue at hand, but it also creates new challenges. For example, students reported that "It's not as simple as I thought; you need to be very precise with your questions to get the answers you want," "I found using ChatGPT in class less convenient than expected; I had to repeatedly clarify my requests to get a response," and "I feel that using ChatGPT didn't really help answer my questions."

Self-Efficacy and Attribution: Since ChatGPT often provides precise solutions and code, learners may start attributing their success to artificial intelligence, which could weaken their self-efficacy in terms of personal achievement (Gutwill, 2018). As some students mentioned, "Although it’s convenient to get direct answers, I miss out on the process of thinking and organizing my thoughts, so I don’t necessarily learn from the problems," and "I believe that effective learning in programming still requires personal understanding and hands-on coding."

Unbalancing Knowledge Interaction: While ChatGPT can offer rich and accurate knowledge and guidance, if it fails to appropriately gauge the learner's level and adjust its responses accordingly, it may unintentionally disrupt this balance. For instance, it may provide overly simple content when advanced challenges are sought or offer more complex explanations when a student is already feeling anxious. Students provided feedback such as "The answers to the same question are always the same," "It cannot respond in a simple and understandable way," and "Some of the knowledge in its answers is beyond what I’ve learned, so I don’t really understand it."

6. Conclusion

Based on our findings, it is crucial to understand that integrating ChatGPT into programming education is not simply a straightforward enhancement of the learning process. To effectively harness the potential of these tools, educators must carefully consider their possible implications, capitalizing on ChatGPT’s strengths while mitigating potential drawbacks. In light of this, we propose several recommendations for the future implementation of programming education: *Ensuring Teacher Competency in Integrating ChatGPT:* While teachers possess expertise in programming knowledge and pedagogy, integrating ChatGPT into programming education also requires them to guide students in effectively using the tool. Therefore, teachers should first assess their familiarity with ChatGPT to seamlessly incorporate it into their lessons. *Ensuring Students' Critical Thinking and Learning Progress:* Without confirming that students have sufficient knowledge and critical thinking skills, the benefits of using ChatGPT for learning programming cannot be fully guaranteed. When students use ChatGPT for programming, it is important to ensure that they comprehend the interactive modes, content, and potential unexpected outcomes associated with ChatGPT, and to prevent them from merely mimicking responses. Teachers should design reflective or critical activities to encourage students to evaluate and critique ChatGPT's responses, thereby fostering critical thinking and genuinely enhancing their knowledge acquisition.

Acknowledgements

This work was supported by the National Science and Technology Council in Taiwan under Grant 112-2628-H-A49-001-MY2.

References

Hu, L. (2024). Programming and 21st century skill development in K-12 schools: A multidimensional meta-analysis. *Journal of Computer Assisted Learning*, *40*(2), 610-636. https://doi.org/10.1111/jcal.12904

Sun, D., Zhu, C., Xu, F., Li, Y., Ouyang, F., & Cheng, M. (2024). Transitioning from introductory to professional programming in secondary education: Comparing learners' computational thinking skills, behaviors, and attitudes. *Journal of Educational Computing Research*, *62*(3), 647-674. https://doi.org/10.1177/07356331231204653

Sharma, P., & Harkishan, M. (2022). Designing an intelligent tutoring system for computer programing in the Pacific. *Education and Information Technologies*, *27*(5), 6197-6209. https://doi.org/10.1007/s10639-021-10882-9

Cooper, G. (2023). Examining Science Education in ChatGPT: An Exploratory Study of Generative Artificial Intelligence. *Journal of Science Education and Technology*, 1-9. https://doi.org/10.1007/s10956-023-10039-y

OpenAI. 2024. " Introducing ChatGPT" Accessed August 15, 2024]. https://openai.com/blog/chatgpt/.

Hu, L. (2024). Programming and 21st century skill development in K-12 schools: A multidimensional meta-analysis. *Journal of Computer Assisted Learning*, *40*(2), 610-636. https://doi.org/10.1111/jcal.12904

McDonald, C. (2018). Why is teaching programming difficult?. *Higher education computer science: A manual of practical approaches*, 75-93. https://doi.org/10.1007/978-3-319-98590-9_6

Yilmaz, R., and Yilmaz, F. G. K. 2023. "Augmented Intelligence in Programming Learning: Examining Student Views on the Use of ChatGPT for Programming Learning." *Computers in Human Behavior: Artificial Humans* 1 (2): 100005. https://doi.org/10.1016/j.chbah.2023.100005.

Woo, K., and Falloon, G. 2023. "Coding Across the Curriculum: Challenges for Non-Specialist Teachers." *In Teaching Coding in K-12 Schools: Research and Application*, 245–261. Cham: Springer International Publishing. https://doi.org/10.1007/978-3-031-21970-2_16.

Daradoumis, T., J. M. M. Puig, M. Arguedas, and L. C. Liñan. 2019. "Analyzing Students' Perceptions to Improve the Design of an Automated Assessment Tool in Online Distributed Programming." *Computers & Education* 128: 159–170. https://doi.org/10.1016/j.compedu.2018.09.021.

Gutwill, J. P. (2018). Science self-efficacy and lifelong learning: emerging adults in science museums. *Visitor Studies*, *21*(1), 31-56. https://psycnet.apa.org/doi/10.1080/10645578.2018.1503875