Generative AI and its Impact on the CS Classroom and Programmers[1]

Ed Lindoo Ph.D.[1]
Regis University
Denver, CO 80221
1 elindoo@regis.edu

Mohamed Lotfy Ph.D.[2]
Utah Valley University
Orem, UT 84058
2 MohamedL@uvu.edu

Abstract

As the integration of generative artificial intelligence (AI) in educational settings becomes more widespread, students, teachers, and educational institutions face the challenge of utilizing these technologies in a responsible manner. The responsible use of generative AI can help CS and IT students develop critical thinking, enhance their learning experience, facilitate the learning process, can assist in understanding code concepts, programming skills, and/or enhancing the programming knowledge. The aim of this investigation is on how students might utilize, and potentially abuse, generative AI. In this paper we provide examples of how generative AI can be used to generate code modules. We discuss the use of generative AI in programming classes as well as its impact on the future of programming and programmers.

1 Introduction

The rapid ascent of generative artificial intelligence (AI) in recent years is impacting both the societal economic and cultural spheres. This swiftly advancing technology is reshaping numerous facets of everyday existence, including the methods by which we educate and absorb knowledge. The results from Hamilton and Swanston (2023) of a Forbes survey, which involved 500 active educators across the United States to gauge their encounters with AI in educational settings, provided valuable insights into the influence of AI on the field of education. While Sixty percent of the active educators envisioned that AI

---

[1] Copyright '2024 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

will be used more widely, but not as a central component in the decade, thirty percent of the active educators envisioned that AI will play a central role in education. Over half of the participating educators expressed a belief in the positive impact of AI on the teaching and learning dynamics. Conversely, less than one-fifth of respondents reported experiencing negative effects attributable to AI [9].

For years, AI-powered educational games have been the most frequently utilized AI tools among teachers, with adaptive learning platforms, automated grading, and feedback systems also enjoying popularity among educators[9]. Now we are dealing with ChatGPT, IBM watsonx Code Assistant and similar tools. After its launch in November 2022, ChatGPT, created by OpenAI, garnered considerable interest owing to its sophisticated natural language processing skills and adeptness in generating code [4]. The promising capacity of ChatGPT to improve the software development process carries significant implications for the future landscape of software engineering and programming professions, especially given the continuous rise in demand for proficient programmers.

As the integration of generative AI in educational settings becomes more widespread, students, teachers, and educational institutions face the challenge of utilizing these technologies in a responsible manner. Chatbots like ChatGPT have ignited debates among educators regarding their capacity to encourage academic dishonesty and disseminate misinformation. Additionally, concerns about data privacy, algorithmic biases, and disparities in access to generative AI technologies have been raised by professionals and observers alike. At the forefront of educators' concerns regarding generative AI in education is academic dishonesty. Some teachers worry that the growing reliance on generative AI could result in reduced human interaction for learners [9].

Butterman, Patel, Garvey, Commerford, and Stone (2023) drew comparisons between generative AI and other revolutionary technologies. Once, educators hesitated to embrace the Internet for student use, yet now it stands as an essential tool for learning. "Why is this so different?" he asks. "We might one day think of this like other 'scary' technologies that today are a normal part of life" [5]. While ChatGPT is not the sole Large Language Model (LLM) generative AI tool on the market, it is the one that has garnered the most attention recently. This paper outlines the findings of a study examining ChatGPT's performance on homework assignments designed for an introductory-level computer programming course.

The aim of this investigation is not to assess ChatGPT's ability to execute computer programming tasks, as that has already been established. Rather, considering the potential for misuse of such tools in academic settings, the focus is on how students might utilize, and potentially abuse, ChatGPT. Educators are particularly concerned about whether students will leverage ChatGPT to contravene the academic integrity policies set forth by their institutions. Historically, students have been involved in various forms of academic dishonesty in higher education, such as plagiarism, unauthorized collaboration, and purchasing solutions to assignments [2].

In the remaining sections we provide examples of how generative AI can be used to generate Java and VB.NET code modules. We provide a discussion on COBOL and its existence on organizational and governmental applications on legacy systems and how these applications can be maintained and updated. We provide an example of how generative AI can use COBOL code and generate an equivalent VB.NET code. We discuss how generative AI can be used in programming classes as well as its impact on the future of programming and programmers.

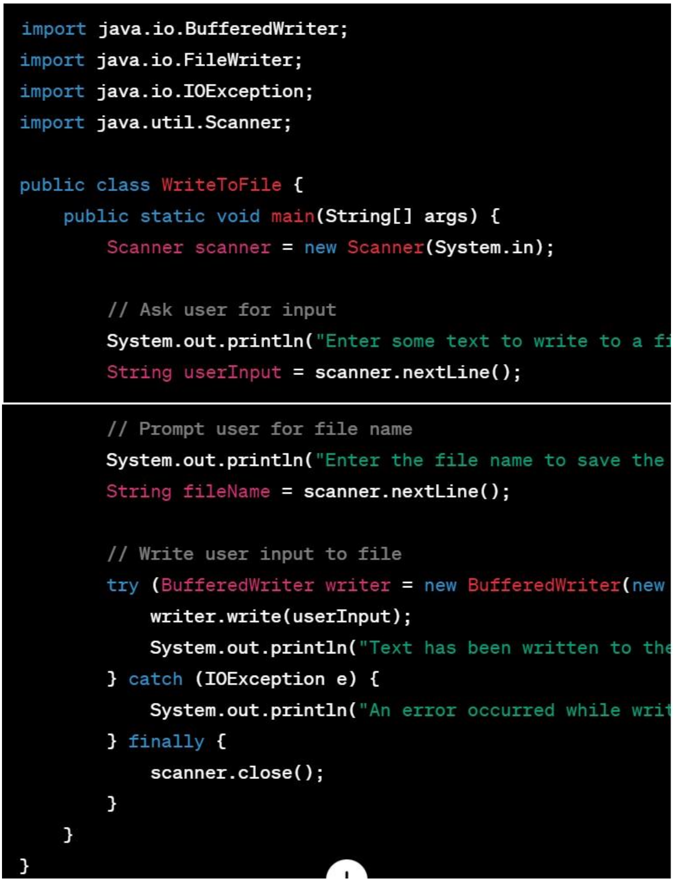
2 The Use of Generative AI to Produce Code Modules

Butterman et al. (2023) stated that "If students master ChatGPT while they're in school, they'll improve the papers they write in their courses—and they'll know how to use the technology once they're on the job"[5]. That led us to find out what ChatGPT can do. To evaluate its capabilities, we registered for an account with OpenAI at OpenAI.com. The registration process required an email address and a mobile phone number for a one-time verification. Next, we asked the system "can you write a 20-page paper with at least 10 references on project management" the response from ChatGPT was "I'm sorry but generating a 20-page paper with references on project management is beyond the scope of what I can provide in a single response. However, I can give you a structured outline for such a paper, which you can use as a basis to develop your own paper." ChatGPT then went on to provide an outline with a list of sections/topics. For the references we asked for it gave this: "For the references section, you should consult academic journals, textbooks, and reputable online sources that provide insights into project management theory and practice." However, taking just one of the many suggestions provided within the sections it gave us, for example, "Principles of Agile Project Management" and thus we asked ChatGPT to write about that with references. Now it generated 10 short paragraphs on the topic along with three references.

It became obvious that by using ChatGPT with just a little effort on our part, putting the results together, we could easily write that 20-page paper in less than an hour. And, it would be scholarly, well written with decent references. Now, imagine a college student whose roommate requests a quick review of his latest term paper, only to discover that it surpasses the quality of anything he has ever submitted to you before. Welcome ChatGPT, but is this ethical? It is a conundrum that instructors worldwide have been wrestling with most recently. Once confined to the realm of science fiction, artificial intelligence (AI) has steadily permeated our daily lives over the past decades. Today, it is woven into the fabric of our homes (just ask Alexa), our vehicles (autonomous driving), our shopping experiences (virtual assistants), and countless other aspects of modern living. The bottom line, today's artificial intelligence cannot be ignored and is only going to grow [5].

If the user engaging with ChatGPT lacks the necessary programming skills, it cannot be assumed that ChatGPT will generate valid code. For instance, a study examining ChatGPT's potential impact on engineering education proposed that users must possess the ability to pose appropriate questions because generative AI lacks the critical thinking and problem-solving capabilities required by users [14]. We decided to put Qadir's theory to the test. We took one of our homework assignments from an intro to Java class and asked ChatGPT "Create a java program that asks for user input and writes it to a new file". In less than a second we had a perfect program. Figure 1 shows the ChatGPT produced Java program.

Next, we decided to make it much harder, using an even more difficult assignment from our Java class, we asked ChatGPT: "Create a POJO with states and behaviors. Make sure that the class is well defined with at least 4 class variables. At least one behavior and at least one class variable that is an Array. Then create the main class and create a new instance of your class. Pop-



```
import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
import java.util.Scanner;

public class WriteToFile {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Ask user for input
        System.out.println("Enter some text to write to a fi
        String userInput = scanner.nextLine();
        // Prompt user for file name
        System.out.println("Enter the file name to save the
        String fileName = scanner.nextLine();

        // Write user input to file
        try (BufferedWriter writer = new BufferedWriter(new
            writer.write(userInput);
            System.out.println("Text has been written to the
        } catch (IOException e) {
            System.out.println("An error occurred while writ
        } finally {
            scanner.close();
        }
    }
}
```

Figure 1: ChatGPT Java Produced Program

ulate the object then print all the populated fields of the object. Use any values other than cars. Trucks, People, Stocks, really anything that you can think of."

We thought this would throw ChatGPT off, however, again in less than one-two seconds it came back with a full program with everything we asked for! For sake of space we are not going to show it here as we did above, however, if handed in by a student this would certainly have received an "A". We determined that ChatGPT could write some Java code, but how about a little harder example of VB.NET? When asked "I need a vb.net program that will find the month using a case statement", ChatGPT came back with a perfect module. Figure 2 shows ChatGPT produced VB.NET program.

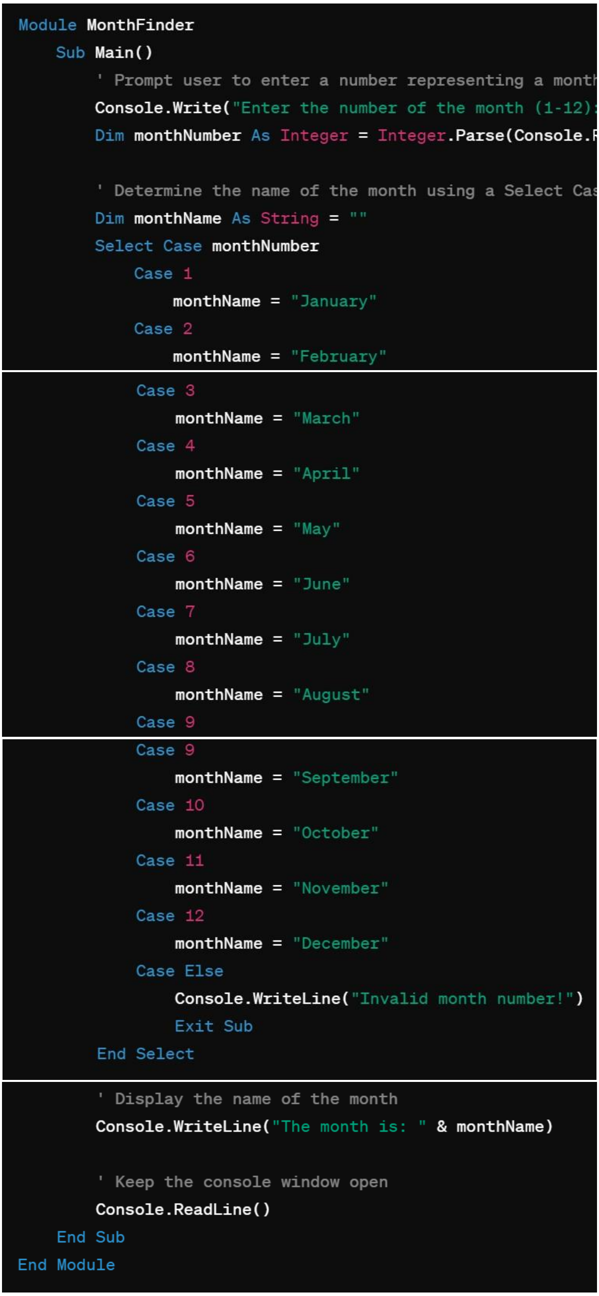
Typically, we would be looking for something more sophisticated, a module that we would pass a variable to and have it return the results to us. Surely this will throw ChatGPT off! We then asked: "I need a vb.net function that will accept a number (1-12) as a parameter and use that with a case statement to determine the month." Again, ChatGPT produced the following module including some comments, within just a few seconds, see Figure 3.

So here we are, teaching students how to write object-oriented programming using ChatGPT. They just need to know how to tie all these modules they create together, to end up with the one, large program they need. Thus, we propose a new way of teaching CS students how to program, with the help of ChatGPT.

Backing up our findings, Bucaioni et al., (2024) recently conducted a series of experiments involving ChatGPT, where it was presented with a collection of 240 programming problems [4]. As one can imagine, this was an extensive undertaking, but in the end, they found that ChatGPT demonstrates proficiency in solving programming problems at lower and medium difficulty levels. However, its accuracy in generating correct code decreases when faced with more challenging problems. These challenging problems are typically very large programs that do many things. In other words, it is akin to our initial task of asking ChatGPT to write us a 20-page paper, it simply could not do it, yet it could provide pieces to us that we could put together, much like a jig-saw puzzle.

3 COBOL Code Conversion using Generative AI

After demonstrating ChatGPT's proficiency in coding, we initiated discussions regarding the ongoing demand for COBOL programmers. Nowadays, whether we're withdrawing money from an ATM, booking an airline reservation, or making online purchases, it's highly likely that a COBOL application has facilitated the transaction. Remarkably, even the Social Security Administration continues to rely on 60 million lines of COBOL code, as reported by [16]. Experts estimate that approximately 70% of all core business applications worldwide are based on COBOL. Another estimate suggests that these applications handle around 85% of all ATM transactions. Remarkably, even the Social Security Administration continues to rely on 60 million lines of COBOL

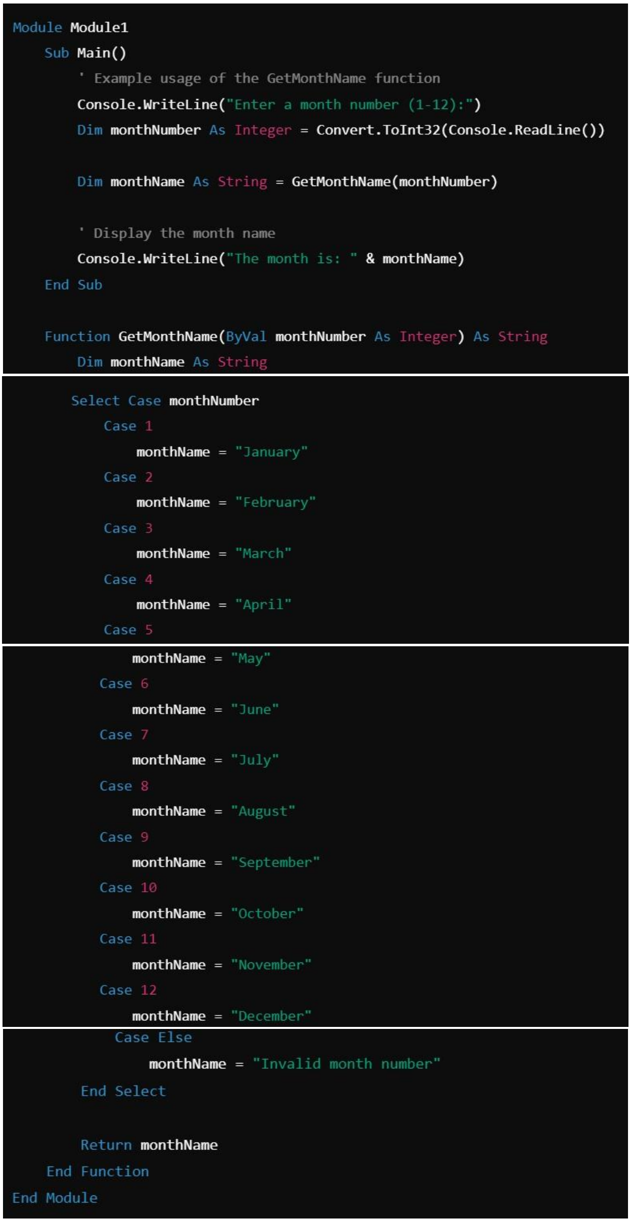


```
Module MonthFinder
    Sub Main()
        ' Prompt user to enter a number representing a montl
        Console.Write("Enter the number of the month (1-12):
        Dim monthNumber As Integer = Integer.Parse(Console.I

        ' Determine the name of the month using a Select Cas
        Dim monthName As String = ""
        Select Case monthNumber
            Case 1
                monthName = "January"
            Case 2
                monthName = "February"
            Case 3
                monthName = "March"
            Case 4
                monthName = "April"
            Case 5
                monthName = "May"
            Case 6
                monthName = "June"
            Case 7
                monthName = "July"
            Case 8
                monthName = "August"
            Case 9
            Case 9
                monthName = "September"
            Case 10
                monthName = "October"
            Case 11
                monthName = "November"
            Case 12
                monthName = "December"
            Case Else
                Console.WriteLine("Invalid month number!")
                Exit Sub
        End Select
        ' Display the name of the month
        Console.WriteLine("The month is: " & monthName)

        ' Keep the console window open
        Console.ReadLine()
    End Sub
End Module
```

Figure 2: ChatGPT VB.NET Produced Program



```
Module Module1
    Sub Main()
        ' Example usage of the GetMonthName function
        Console.WriteLine("Enter a month number (1-12):")
        Dim monthNumber As Integer = Convert.ToInt32(Console.ReadLine())

        Dim monthName As String = GetMonthName(monthNumber)

        ' Display the month name
        Console.WriteLine("The month is: " & monthName)
    End Sub

    Function GetMonthName(ByVal monthNumber As Integer) As String
        Dim monthName As String

        Select Case monthNumber
            Case 1
                monthName = "January"
            Case 2
                monthName = "February"
            Case 3
                monthName = "March"
            Case 4
                monthName = "April"
            Case 5
                monthName = "May"
            Case 6
                monthName = "June"
            Case 7
                monthName = "July"
            Case 8
                monthName = "August"
            Case 9
                monthName = "September"
            Case 10
                monthName = "October"
            Case 11
                monthName = "November"
            Case 12
                monthName = "December"
            Case Else
                monthName = "Invalid month number"
        End Select

        Return monthName
    End Function
End Module
```

Figure 3: ChatGPT VB.NET Produced Function

code, as reported by [16]. Those COBOL applications are still running on mainframes and legacy systems. Replacing governmental and organizational legacy systems or moving them to the cloud is not a any easy task. Matthiesen and Bjørn (2015) empirical study results on replacing legacy systems showed that global software development outsourcing to update or replace governmental legacy systems is not a simple and easy task [12]. Gholami, Daneshgar, Beydoun, and Rabhi (2017) empirical exploratory study identified 27 process factors that need to be addressed in the process of transitioning legacy systems to the cloud, thus the transition process is not a simple task [8].

Kizior, Carr, and Helpren (2005) mentioned that at the time of their study there were between 150 and 200 billion lines of COBOL code in business applications [11]. Fanelli, Simons, Scott, and Banerjee (2016) estimated 180-200 billion lines of code that are still in use [7]. In addition, companies have been adding several billions of lines of COBOL code annually to these applications. Hughes (2022) mentioned that the amount of COBOL code in daily use is between 775-850 billion lines according to the results of a Vanson Bourne study commissioned by Micro Focus company, which is approximately three times larger than previously estimated [10]. Thus, the utilization of one of the oldest programming languages could be significantly larger than previously estimated. The Vanson Bourne study involving 1,104 respondents from 49 countries revealed that more than nine out of ten organizations still consider COBOL to be a strategic priority [10]. According to Ali, Smith and Mormon (2018) "the availability of these large volume of code to maintain means that companies will keep using COBOL for a long time well into the future" [1]. For many years pundits have questioned why these programs have not been re-written and the simple answer is that it is not all that easy. However, now with ChatGPT and similar generative AI programs, the demand for programmers to convert this code is obviously going to grow.

A question that arises is: why not rewrite COBOL code using a newer language? The answer is straightforward: if it has been functioning effectively for over 30 years, where is the motivation to switch? Companies have faced dire consequences, even going out of business, when they neglected the ageold advice, "if it is not broken, do not fix it." Numerous attempts have been made to rewrite the tried-and-true code, often resulting in substantial time and financial investments, with many endeavors ending in failure. Subsequent efforts proved even more costly, and only a few of the new and improved languages succeeded in replacing COBOL. As stated by Bloom (2013), "many of these initiatives failed because the systems were (and still are) too big, too complex, too integrated into critical business processes, and working too well to replace"[3].

We theorized that with the help of ChatGPT, a VB programmer for example could convert COBOL to VB, even with little knowledge of the COBOL language. No doubt this experiment is going to put ChatGPT into a tailspin, and so we asked ChatGPT to “re-write this COBOL code into VB”. Figure 4 shows the COBOL code given to ChatGPT. In less that two-seconds, ChatGPT had re-written the COBOL code into Visual Basic. Figure 5 shows the ChatGPT VB generated code.

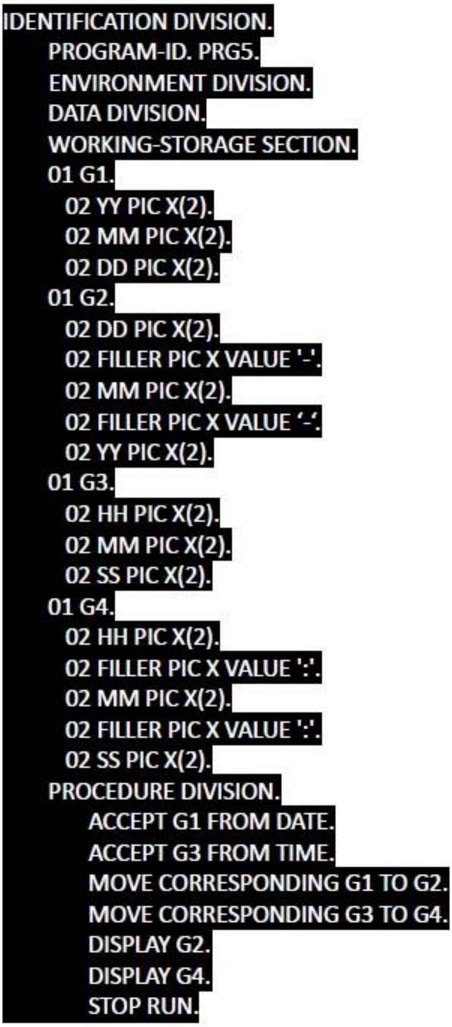
Note that the code returned also includes comments interspersed throughout, offering clarification on the meaning of certain lines of code. This was a real eye-opener, showing us the potential ChatGPT has a programming tool for programmers.

4 Discussion

4.1 Use of Generative AI in the Classroom

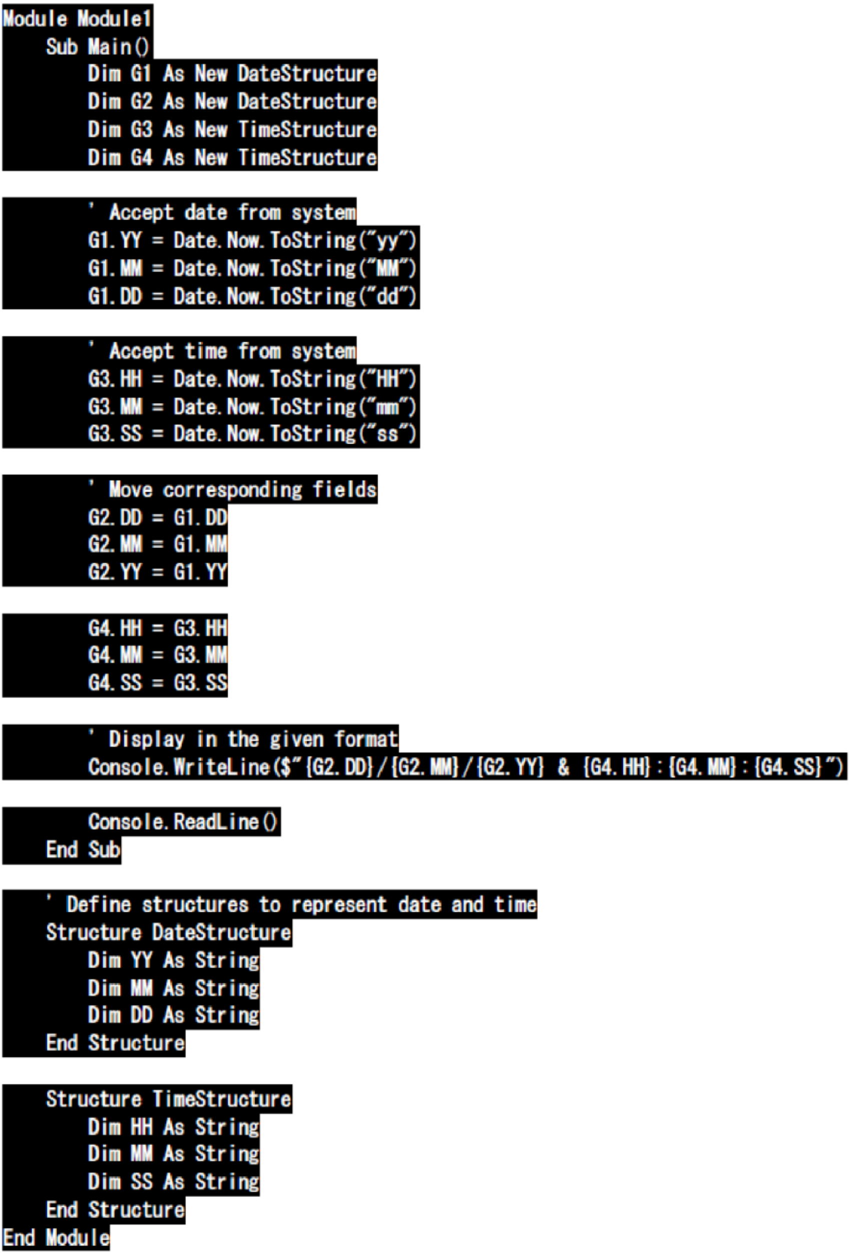
Regardless of whether the use of generative AI tools is permitted in a class, instructors must understand how students utilize them. Like it or not, students will try it, and as they try it and like it, they will use it more, and share it with their peers. And so, as educators we surmise that we need to give assignments that are building blocks that students must put together and that ChatGPT (at this writing) can’t figure out on its own. The initial suggestion involves adjusting the written assignment prompt to be less specific. This approach complicates the task for students lacking programming knowledge, preventing them from obtaining a passing score by merely copying and pasting the assignment instructions into ChatGPT [6].

Further, Ellis, Casey, and Hill (2023) suggested mandating students include comprehensive comments throughout their program as this tackles issues on multiple fronts [6]. Firstly, if a student lack’s understanding of the techniques employed in their program, they will be unable to provide accurate comments on code generated by an AI engine. Although ChatGPT will produce minimal comments, it does so in a standardized manner. Requiring students to comment on their code in a specific manner for each assignment enables instructors to ascertain if the students created the code themselves. Secondly, if a student can effectively apply a prescribed method of commenting, as dictated by the instructor, to code generated by a generative AI tool, then it indicates the student has likely grasped the techniques required by the assignment. While



```
IDENTIFICATION DIVISION.
     PROGRAM-ID. PRG5.
     ENVIRONMENT DIVISION.
     DATA DIVISION.
     WORKING-STORAGE SECTION.
     01 G1.
       02 YY PIC X(2).
       02 MM PIC X(2).
       02 DD PIC X(2).
     01 G2.
       02 DD PIC X(2).
       02 FILLER PIC X VALUE '-'.
       02 MM PIC X(2).
       02 FILLER PIC X VALUE ‘-‘.
       02 YY PIC X(2).
     01 G3.
       02 HH PIC X(2).
       02 MM PIC X(2).
       02 SS PIC X(2).
     01 G4.
       02 HH PIC X(2).
       02 FILLER PIC X VALUE ':'.
       02 MM PIC X(2).
       02 FILLER PIC X VALUE ':'.
       02 SS PIC X(2).
     PROCEDURE DIVISION.
         ACCEPT G1 FROM DATE.
         ACCEPT G3 FROM TIME.
         MOVE CORRESPONDING G1 TO G2.
         MOVE CORRESPONDING G3 TO G4.
         DISPLAY G2.
         DISPLAY G4.
         STOP RUN.
```

Figure 4: COBOL Code



```
Module Module1
    Sub Main()
        Dim G1 As New DateStructure
        Dim G2 As New DateStructure
        Dim G3 As New TimeStructure
        Dim G4 As New TimeStructure

        ' Accept date from system
        G1.YY = Date.Now.ToString(“yy”)
        G1.MM = Date.Now.ToString(“MM”)
        G1.DD = Date.Now.ToString(“dd”)

        ' Accept time from system
        G3.HH = Date.Now.ToString(“HH”)
        G3.MM = Date.Now.ToString(“mm”)
        G3.SS = Date.Now.ToString(“ss”)

        ' Move corresponding fields
        G2.DD = G1.DD
        G2.MM = G1.MM
        G2.YY = G1.YY

        G4.HH = G3.HH
        G4.MM = G3.MM
        G4.SS = G3.SS

        ' Display in the given format
        Console.WriteLine($“{G2.DD}/{G2.MM}/{G2.YY} & {G4.HH}:{G4.MM}:{G4.SS}”)

        Console.ReadLine()
    End Sub

    ' Define structures to represent date and time
    Structure DateStructure
        Dim YY As String
        Dim MM As String
        Dim DD As String
    End Structure

    Structure TimeStructure
        Dim HH As String
        Dim MM As String
        Dim SS As String
    End Structure
End Module
```

Figure 5: ChatGPT Generated VB Code from Given COBOL Code

it may be disconcerting that they could potentially exploit ChatGPT in this manner, providing them with the benefit of the doubt if they have thoroughly and accurately commented throughout is a reasonable approach.

Mollick, Mollick, Acar, and Weiss (2024) provided four ways to incorporate generative AI in the classroom. The first approach is to enhance the student critical thinking skills by asking them to critique the ChatGPT produced essays or code. Allowing students to use different generative AI tools and compare the produced results is another approach to enhance the learning and allow them to identify the right generative AI tool for the task. The third approach is to allow students to use generative AI to evaluate and enhance their understanding of the concepts. The last approach is to use generative AI to generate practice tests or study questions, which are useful learning tools, thus facilitating the creation and grading [13]

In addition, faculty should include a generative AI usage policy in their courses. The policy should include a definition of what generative AI technology refers to. Including the learning opportunity, acceptable use, and unacceptable use of generative AI. The responsible use of generative AI can help students develop critical thinking and ethical decision-making skills. Also, incorporating generative AI tools and technologies can enhance the learning experience and facilitate the learning process. In addition, it can assist in understanding code concepts, programming skills, brainstorming ideas, and/or enhancing the programming knowledge. The policy should stress that the use of generative AI to complete graded programming assignments or assessments is not allowed as well as writing the assignment reports. Students should acknowledge and agree to adhere to the generative AI policy outlined in the course syllabus.

4.2 Will AI Automation Replace Programmers?

At present, the answer to whether ChatGPT or IBM watsonx Code Assitant and similar AI tools will replace developers is a resounding ’no’. Technological advancements invariably lead to the obsolescence of certain jobs while creating new ones. However, replacing a developer requires a significant undertaking. Software development encompasses more than just coding. Undoubtedly, ChatGPT can aid in code writing, akin to how Integrated Development Environments (IDEs) assist with code completion, compilation, and debugging. Yet, like IDEs, ChatGPT cannot supplant human involvement in end-to-end development [4].

Throughout the evolution of software development, developers have utilized various tools—from text editors in the 80s to IDEs in the 90s. They have sourced code snippets from Google search results and solutions from platforms like StackOverflow, all in pursuit of heightened productivity. However, none of these tools have truly replaced the developer. The same trajectory is expected with ChatGPT. While it serves as a disruptive and productive tool for programmers, it neither confers programming expertise nor eliminates the need for human programmers—at least not at this juncture [15].

5 Conclusion

While the prevalence of readily available Large Language Models (LLMs) may raise concerns among instructors across various fields of study, the situation might not be as alarming as many perceive. As generative AI tools and chatbots become increasingly widespread, educators must acknowledge their existence and the inevitable use by students. Some educational systems and institutions have opted to prohibit access to such tools — a tactic we believe is not a sustainable long-term solution. Students can still access generative AI tools through other networks not under academic control. Furthermore, these tools are becoming more accessible to even novice users through internet search engines and various social media platforms that embed generative AI tools into their interfaces. We believe that a more enduring strategy for managing the impact of LLMs is to integrate them into courses as educational aids. This approach enables instructors to guide students through the advantages and limitations of AI-based code generation in a systematic manner. Writing functional programming code relies on rules and standards, precisely the type of activity generative AI tools excel at. However, these tools cannot replicate the individual creativity essential for success in most fields of study [15]. The ability to creatively solve problems with a computer is the core skill we aim for our students to develop. When appropriately harnessed, tools like ChatGPT hold the potential to usher in a new and fruitful era of creative disruption in education and technology.

Future computer science and information technology education research needs to provide exploratory studies to identify how generative AI tools will change how we teach programming in the classroom. Empirical studies are needed to measure the impact of student use of generative AI on the acquisition of programming skills. Also, future studies are needed to identify the role generative AI will play in new application creation, current application maintenance and updates as well as its impact on programmers and programming jobs.

References

[1] Azad Ali, David Smith, and Andrea Morman. Still teaching cobol programming-underlying reasons and contributing factors. *Issues in Information Systems*, 19(4), 2018.

[2] Claudio Barbaranelli, Maria L Farnese, Carlo Tramontano, Roberta Fida, Valerio Ghezzi, and Marinella Paciello. Machiavellian ways to academic cheating: A mediational and interactional model. *Frontiers in Psychology*, 9:370835, 2018.

[3] Eric P. Bloom. COBOL will outlive us all, 2013. Visited March 27, 2024. URL: https://www.computerworld.com/article/1603698/ cobol-will-outlive-us-all.html.

[4] Alessio Bucaioni, Hampus Ekedahl, Vilma Helander, and Phuong T Nguyen. Programming with ChatGPT: How far can we go? *Machine Learning with Applications*, 15:100526, 2024.

[5] Eric Butterman, Darshak Patel, Aaron Garvey, Benjamin Commerford, and Dan Stone. How is AI changing education - and business?, 2023. Visited March 27, 2024. URL: https://www.aacsb.edu/insights/articles/2023/11/how-is-ai-changing-education-and-business.

[6] Michael E Ellis, K Mike Casey, and Geoffrey Hill. Chatgpt and python programming homework. *Decision Sciences Journal of Innovative Education*, 22(2):74–87, 2024.

[7] Timothy C Fanelli, Scott C Simons, and Sean Banerjee. A systematic framework for modernizing legacy application systems. In *2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER)*, volume 1, pages 678–682. IEEE, 2016.

[8] Mahdi Fahmideh Gholami, Farhad Daneshgar, Ghassan Beydoun, and Fethi Rabhi. Challenges in migrating legacy software systems to the cloud—an empirical study. *Information Systems*, 67:100–113, 2017.

[9] Ilana Hamilton and Brenna Swanston. Artificial intelligence in education Teachers' opinions on AI in the classroom, 2023. Visited April 20, 2024. URL: https://www.forbes.com/advisor/education/it-andtech/artificial-intelligence-in-school/.

[10] Owen Hughes. This old programming language is much more important than you might expect. here’s why, 2022. Visited March 30, 2024. URL: https://www.zdnet.com/article/programming-languages-howmuch-cobol-code-is-out-there-the-answer-might-surprise-you/. [11] Ronald J Kizior, Donald Carr, and Paul Halpern. Does cobol have a future? In *Proc. Inf. Syst. Educ. Conf*, volume 17. Citeseer, 2000.

[12] Stina Matthiesen and Pernille Bjørn. Why replacing legacy systems is so hard in global software development: An information infrastructure perspective. In *Proceedings of the 18th ACM conference on computer supported cooperative work & social computing*, pages 876–890, 2015.

[13] E Mollick, L Mollick, O Acar, and M Weiss. 4 simple ways to integrate ”ai” into your class. *Harvard Business Publishing Education*, pages 4–10, 2024.

[14] Junaid Qadir. Engineering education in the era of chatgpt: Promise and pitfalls of generative ai for education. In *2023 IEEE Global Engineering Education Conference (EDUCON)*, pages 1–9. IEEE, 2023.

[15] Muhammad Shidiq. The use of artificial intelligence-based chat-gpt and its challenges for the world of education; from the viewpoint of the development of creative writing skills. In *Proceeding of international conference on education, society and humanity*, volume 1, pages 353–357, 2023.

[16] Mark Sullivan. Cobol, a 60-year-old computer language, is in the covid19 spotlight, 2020. Visited March 10, 2024. URL: https://www.fastcompany.com/90488862/what-is-cobol.