**Task 1: *Design***

First, please read the following two papers (attached to the email):

*Studying the Effect of AI Code Generators on Supporting Novice Learners in Introductory Programming*

*LLM-Based Code Generators in Introductory Programming: Patterns, Practices, and Learning Outcomes*

*Towards More Effective AI-Assisted Programming: A Systematic Design Exploration to Improve Visual Studio IntelliCode’s User Experience*

Second, please install Github Copilot on your VS Code and try to play around with its code generation capabilities. You need to get your Github Education Pack to use Copilot for free.

AI Code Generators like Github Copilot are based on OpenAI Codex and can generate functioning code. Our own research has showed that they have the potential to reduce the barrier to introductory programming for novice learners. These LLMs generate code in blocks with no further information about the generated code on top of it.

We want you to design, and sketch three completely different code generation techniques that would (i) support complete beginners, (ii) not overwhelm the novice user, (iii) support learning and skill development, and (iv) support tinkering and verification of AI-generated code.

For your sketches, please try to be minimal: either draw on paper and take a picture from your sketches, or if you want to show transitions like a storyboard, you might also use PowerPoint.

Try to use from the ideas proposed in the attached papers.

**Task 2: *Programming***

Write a TypeScript (don’t use plain JavaScript) program that produces a simple drag-and-drop puzzle[[1]](#footnote-1) from the following JSON object.

Features:

* Each block should be draggable and could be connected to other blocks on the left.
* Each block could potentially have an input in which the user is able to manually add code. you don’t need color coding or anything for the text that the user writes here.
* Your canvas should allow blocks to be connected from the left, and snap into an invisible grid for indentations.
* You are free to use any third-party library if they would help you develop this program faster
* Your code should check if the puzzle was fixed, and then display a pop-up message indicating how long it took for them to work on the task.
* You should commit your code into github, and then use Vercel[[2]](#footnote-2) to deploy your node application.
* Bonus activity: use the code-gen-starter.ts file to generate Parsons problems for each generated code.

1. Starts with a randomized puzzle:

Graphical user interface, text, application, chat or text message

Description automatically generated

2. The user must drag the puzzle pieces to get to the following state:

Diagram

Description automatically generated with medium confidence

3. And then manually fill in the blanks like this:

Graphical user interface

Description automatically generated

Here’s the JSON file that you could use. You are free to use any other template, but this is a good starter.

{

  "lines": [

    {

      "indentations": 0,

      "tokens": [

        {

          "text": "count",

          "type": "user-defined-variable"

        },

        {

          "text": " = ",

          "type": "operator"

        },

        {

          "text": "{input}"

        }

      ]

    },

    {

      "indentations": 0,

      "tokens": [

        {

          "text": "while",

          "type": "keyword"

        },

        {

          "text": "count",

          "type": "user-defined-variable"

        },

        {

          "text": " < ",

          "type": "operator"

        },

        {

          "text": "{input}"

        },

        {

          "text": ":",

          "type": "operator"

        }

      ]

    },

    {

      "indentations": 1,

      "tokens": [

        {

          "text": "print",

          "type": "built-in-function"

        },

        {

          "text": "(",

          "type": "operator"

        },

        {

          "text": ")",

          "type": "operator"

        },

        {

          "type": "{input}"

        }

      ]

    },

    {

      "indentations": 1,

      "tokens": [

        {

          "text": "count",

          "type": "user-defined-variable"

        },

        {

          "text": " = ",

          "type": "operator"

        },

        {

          "text": "count",

          "type": "user-defined-variable"

        },

        {

          "text": " + ",

          "type": "operator"

        },

        {

          "text": "{input}"

        }

      ]

    }

  ]

}

1. These puzzles are called Parsons problems. This particular variant of Parsons problems are called Faded Parsons problems. You can see this paper to learn more about them: *Improving Instruction of Programming Patterns with Faded Parsons Problems* (CHI’21) [https://dl.acm.org/doi/abs/10.1145/3411764.3445228] [↑](#footnote-ref-1)
2. A free account is required to deploy your project on Vercel: https://vercel.com/ [↑](#footnote-ref-2)