College of Computer Studies

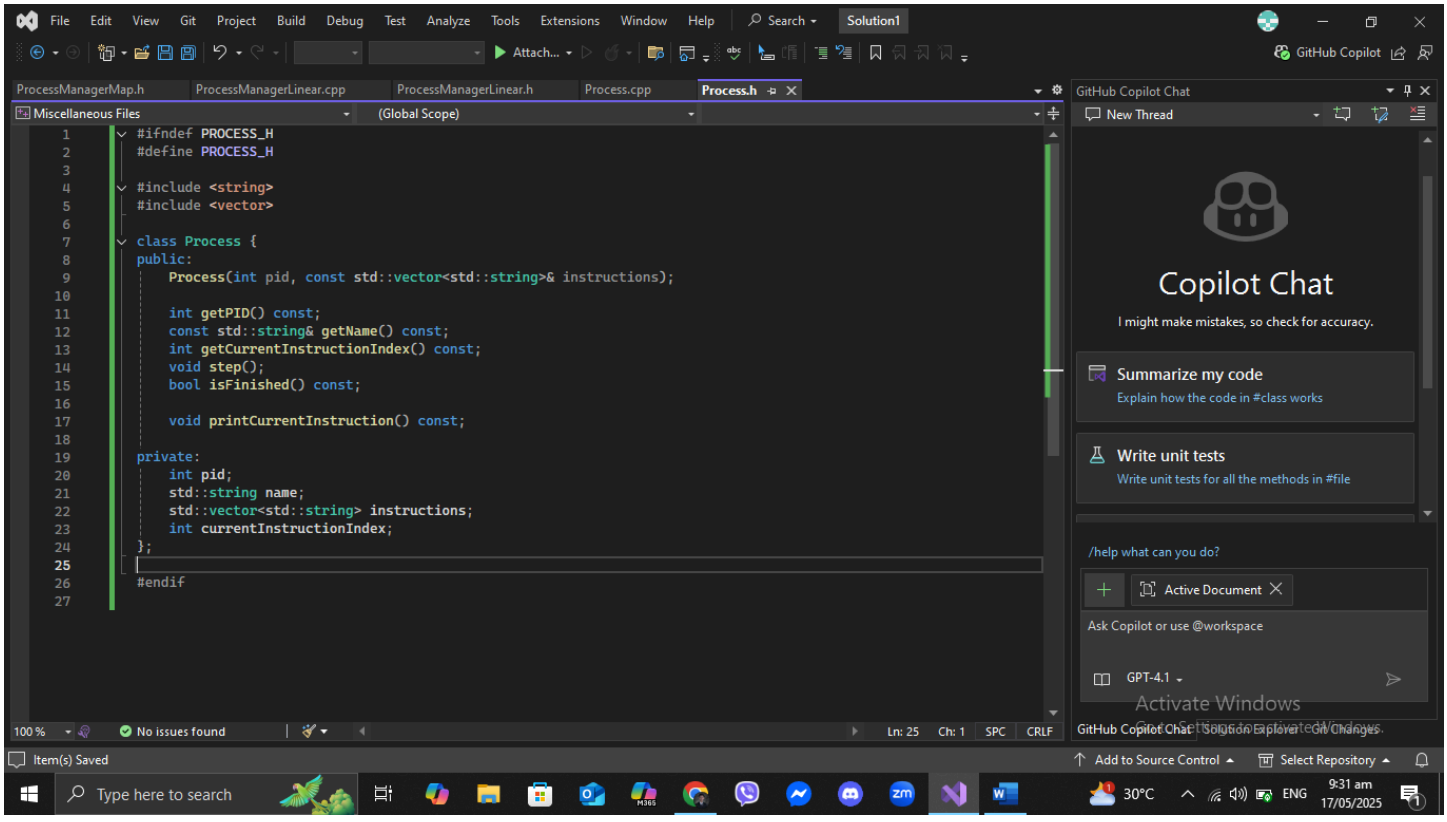
DEPARTMENT OF
SOFTWARE TECHNOLOGY

CSOPESY Seatwork
Instructor: Neil Patrick Del Gallego, Ph.D.

Use Calibri Font Size 11 for texts.

NAME:	**Daniel Gavrie Clemente**
SECTION:	**S18**
DATE OF SEATWORK	**May 17, 2025**

1.



Description of Attributes:

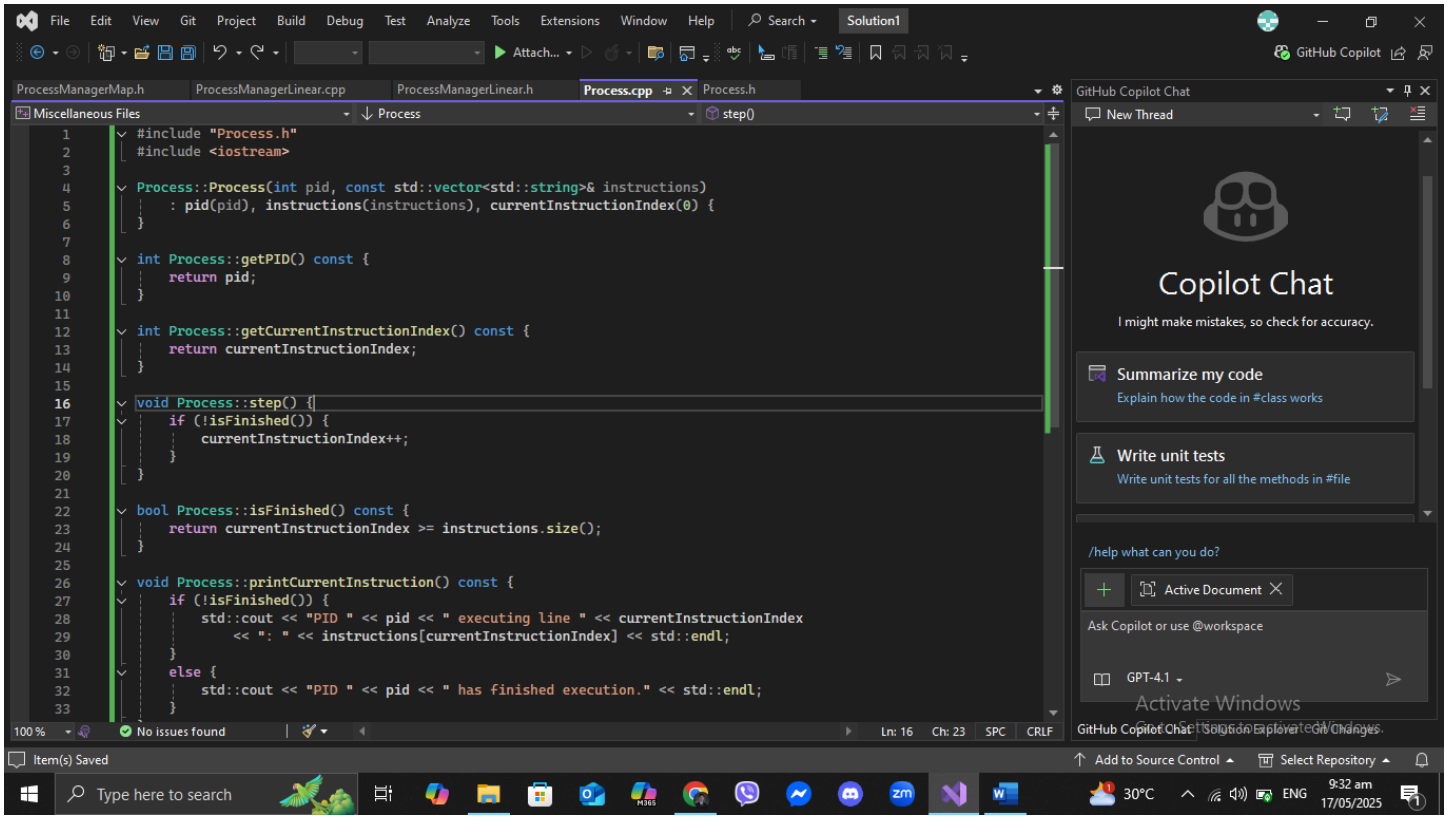
- pid: Unique process ID
- name: Optional process name
- instructions: A list of instruction lines

- currentInstructionIndex: Tracks the line being executed

Key Functions:

- Constructor initializes the process with an ID and its instructions.
- step() advances the process to the next instruction.
- isFinished() checks if the process completed all instructions
- printCurrentInstruction() prints the instruction currently being executed.

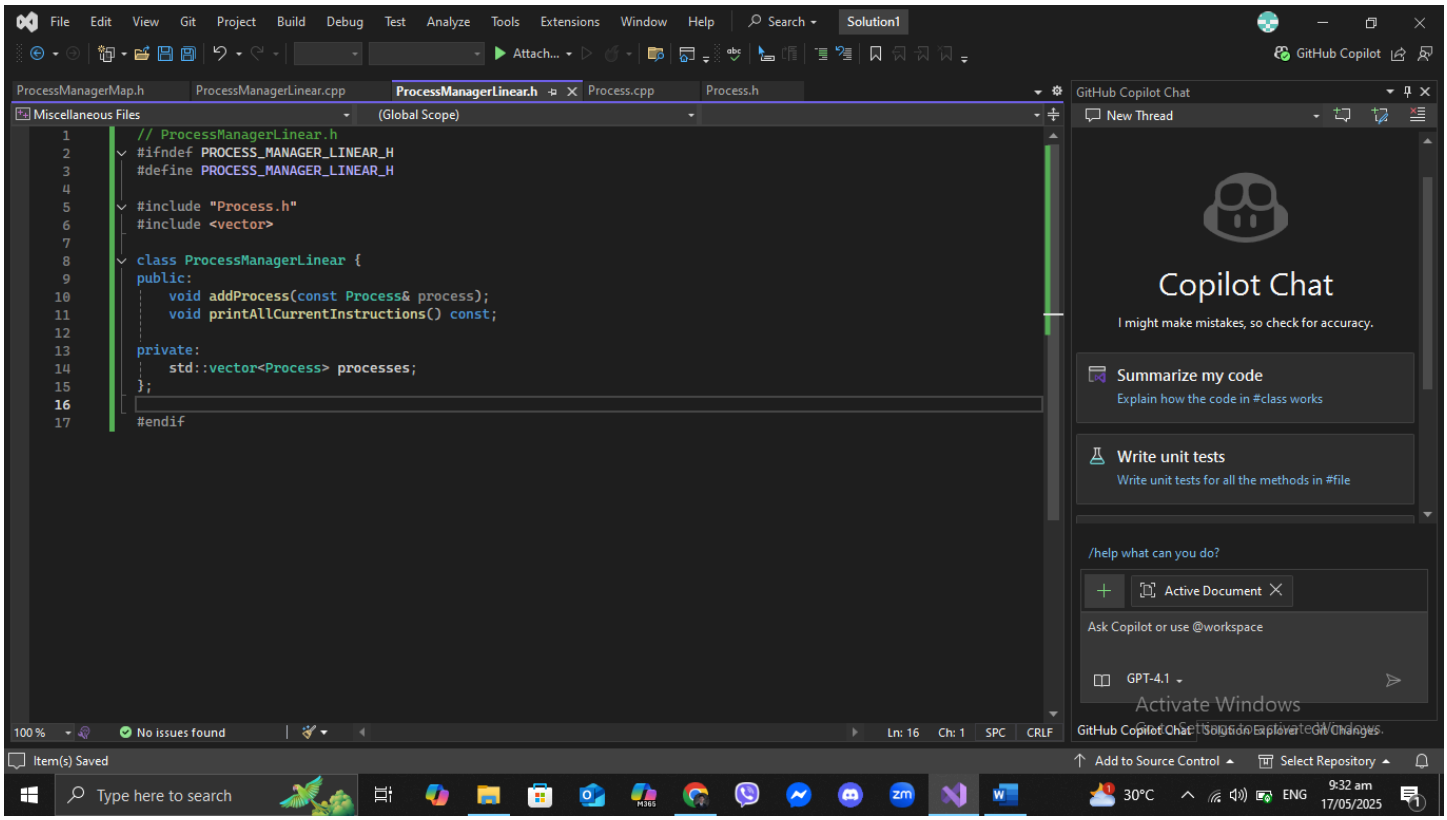
2.

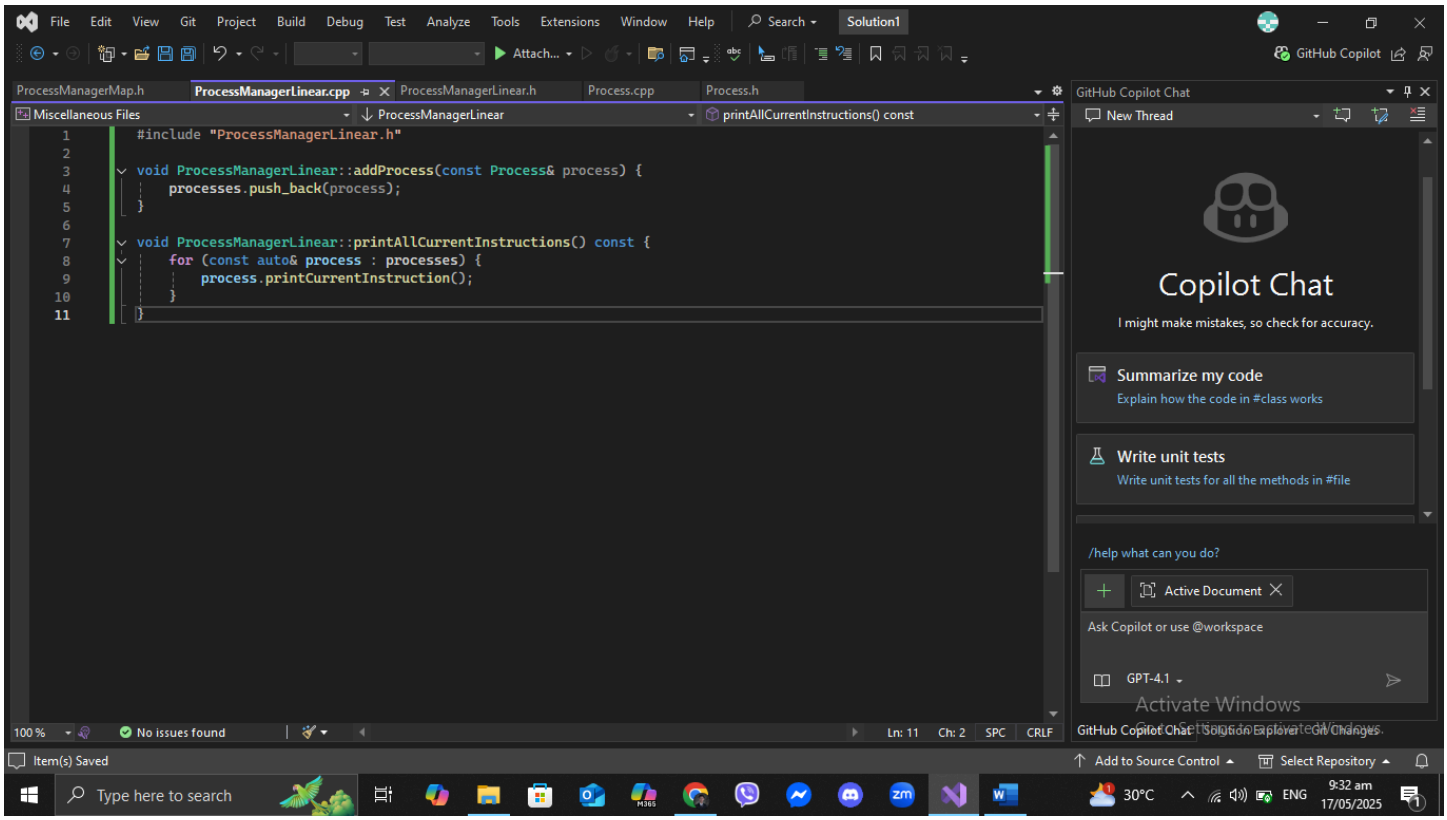


Explanation:

- **step():** Moves the process forward by incrementing the instruction index, simulating execution of the next line.
- **isFinished():** Returns true if the process has executed all instructions.
- **printCurrentInstruction():** Prints the current instruction being executed, along with the process ID and line number. If all instructions have been executed, it prints a finished message.

3.



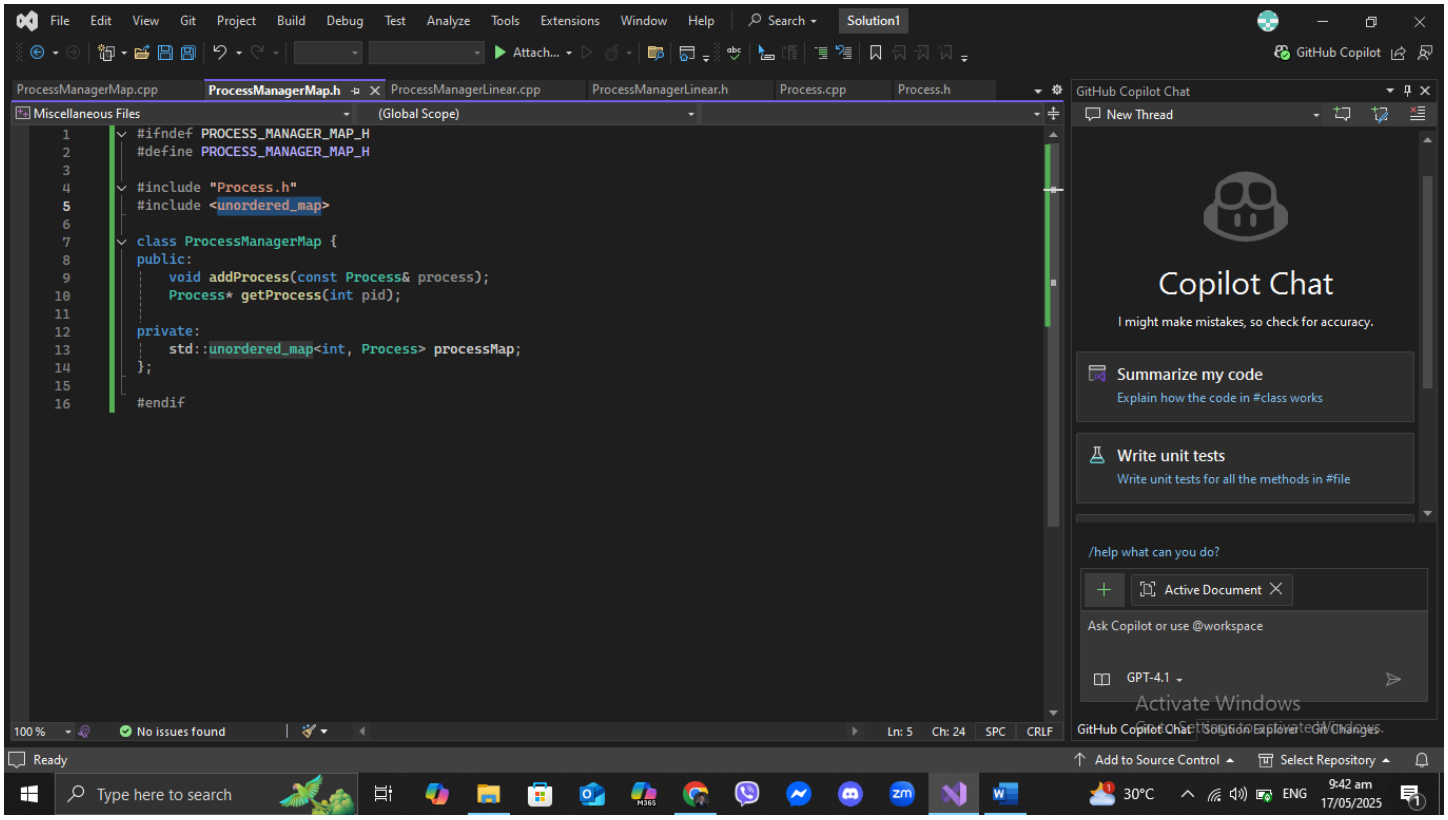


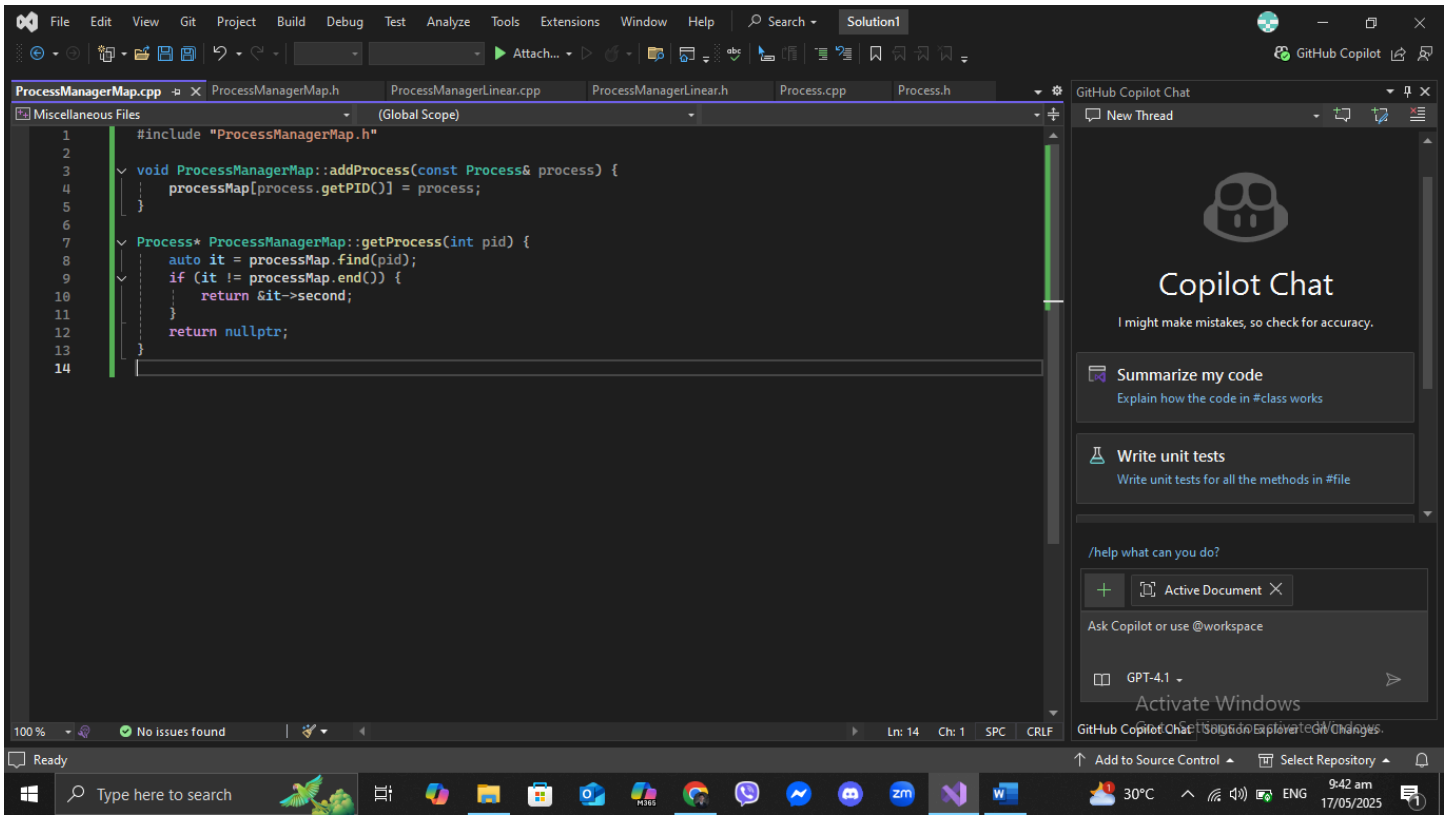
Explanation:

- **Purpose:** Manage multiple processes and allow iterating over them sequentially.
- **Data Structure:** Uses a std::vector<Process> to store processes.

- **addProcess():** Adds a new process to the vector.
- **printAllCurrentInstructions():** Iterates over all processes and prints each process's current instruction.
- **Traversal:** Since it uses a vector, iterating through all processes is done in linear time O(N), where N is the number of processes.

4.





Explanation:

- **Purpose:** Store processes in a hash map for fast retrieval by pid.
- **Data Structure:** std::unordered_map<int, Process> maps pid (key) to Process object (value).
- **addProcess():** Inserts or updates the process associated with the given PID.
- **getProcess():** Retrieves a pointer to the process with the given PID in **constant time O(1)**. Returns nullptr if no such process exists.
- **Benefit:** This is more efficient when you want to directly access a process without iterating over the entire list.

References:

https://en.cppreference.com/w/
https://www.w3schools.com/cpp/default.asp
https://cplusplus.com/reference/