Python - Turtlympics - Program

Overview

Now that you have planned how your maze will be constructed, it is time to implement it by programming it in Python. Use the IPO charts and sketches you have created to program an amazing maze.

Remember, the categories you are competing for are:

- Best Theme
- Best Aesthetics
- Most Complex

Project Criteria

Your task is to create a program that creates a maze for the Turtlympics. Your program must satisfy the following criteria:

- **Define** and **use** at least 2 functions that can accept parameters.
- **Define** and **use** at least 2 functions that can return a value.
- In total you should **define** and **use** at least 4 functions when creating your maze.
- Use **docstrings at the top of the program and for each function** you have created to effectively document your code.
- The maze must have a definite **start** and **end** point defined.
- Program a turtle navigating the maze from start to finish.

Outcome Criteria

- **2 Demonstrate the ability to translate algorithms into working programs.**
    - **2.1** Describe the purpose of key parts of a programming development environment.
    - **2.3** Use the key components of a programming development environment to write and run programs.
    - **2.4** Convert algorithms into code while:
        - **2.4.1** maintaining an IPO structure
        - **2.4.2** including documentation
        - **2.4.3** using appropriate data types
        - **2.4.4** using variables and constants correctly
        - **2.4.5** supplying data using literals and input commands
        - **2.4.6** using appropriate operators
        - **2.4.7** displaying results clearly
    - **2.5** Test programs using appropriate data.
    - **2.6** Revise programs based on testing.
- **3 Analyze program results and debug errors.**
    - **3.1** Compare program output with the intended algorithm.
    - **3.2** Use debugging techniques to correct:
        - **3.2.1** run-time and syntax errors
        - **3.2.2** logic errors
    - **3.3** Modify algorithms or code to improve accuracy.

Mastery Rubric

	5 – Mastery of Skill	**4 – Excellence in Skill**	**3 – Proficiency in Skill**	**2 – Attempting Skill**	**1 – Starting to Attempt Skill**	**0 – Refusal to Engage with Skill**
2.1 Use a programming IDE.	Independently and efficiently uses an IDE to manage files, write code, run programs, and resolve issues without assistance.	Uses an IDE confidently to write, run, and manage the project with minimal guidance.	Uses an IDE to write and run code, but may rely on guidance for some features.	Uses an IDE inconsistently or incorrectly; basic functionality is partially used.	Shows minimal ability to use an IDE.	Does not use an IDE.
2.3 Use the features of an IDE.	Effectively uses editor features, terminal, and file explorer together to develop, test, and debug the program.	Uses most IDE features correctly and consistently.	Uses basic IDE features (editor and run tools) with limited exploration.	Attempts to use IDE features, but with frequent errors or confusion.	Rarely uses IDE features correctly.	Does not use IDE features.
2.4 Convert your algorithm into code.	Faithfully and efficiently converts the planned algorithm into working code that maintains IPO structure, includes clear docstrings, uses appropriate data types, variables, operators, and clearly outputs results.	Successfully converts the algorithm into code with correct IPO structure and documentation, with only minor issues.	Converts the algorithm into working code, but IPO structure or documentation may be incomplete.	Code partially reflects the algorithm; IPO structure or documentation is weak or inconsistent.	Code shows little connection to the planned algorithm.	No meaningful code produced.
2.5 Test your program using appropriate data.	Systematically tests the program using varied inputs, including edge cases, and clearly identifies expected vs actual results.	Tests the program using appropriate data and identifies issues.	Performs basic testing with limited data.	Attempts testing, but data choice or results are unclear.	Very little evidence of testing.	No testing performed.
2.6 Revise programs based on testing.	Makes thoughtful, well-documented revisions that clearly improve correctness, readability, or performance.	Revises the program based on test results with clear improvements.	Makes some revisions after testing, though improvements may be limited.	Attempts revisions, but changes are minimal or ineffective.	Shows little evidence of revision.	No revisions made.
3.1 Compare program output with the intended output.	Accurately and clearly compares actual output with intended behavior, identifying discrepancies and causes.	Correctly compares output to the intended result.	Basic comparison between output and expectation is shown.	Comparison is incomplete or unclear.	Very limited comparison evident.	No comparison made.
3.2 Use debugging techniques to correct errors.	Independently identifies and fixes syntax, run-time, and logic errors using appropriate debugging strategies and tools.	Correctly fixes most errors using debugging techniques with minimal guidance.	Fixes basic errors, often with guidance.	Attempts to debug, but struggles to identify or fix errors.	Shows minimal debugging ability.	No attempt to debug.
3.3 Modify algorithms or code to improve accuracy.	Thoughtfully refines algorithms or code structure to improve accuracy, efficiency, or clarity beyond basic requirements.	Makes clear modifications that improve accuracy or correctness.	Makes some changes to improve accuracy.	Attempts improvements, but changes are limited or unfocused.	Shows little understanding of how to improve accuracy.	No modifications made.

Submission

The following are *required* submissions:

- Python file titled `firstname_lastinitial_turtlympics.py`.
    - *For example, Mr. Forsyth's file would be named* `Nathan_F_turtlympics.py`
    - *Please **DO NOT zip** your python file.*
- Reference log including AI usage and all other links to websites you have used throughout the project.

*To be considered for the maze competition, your assignment **MUST** be handed in **ON TIME!***