Final Project Rubric for Foundations of Programming in Python

---

Overview

- **Total Points:** 100
- **Deliverables:**
  - **A) Code Implementation (70 points)**
  - **B) Documentation (30 points)**

A) Code Implementation (70 Points)

1. Functionality (40 Points)

Criteria	Points
Complete Feature Implementation	
- All specified features are correctly implemented and functional	**20**
- Most features are implemented; minor issues present	15
- Some features are missing or not working correctly	10
- Major features are missing or not functional	5
- Project does not meet the basic functionality requirements	0
User Interface Functionality	
- GUI is fully functional and user-friendly	**10**
- GUI is mostly functional; minor usability issues	7
- GUI has significant usability issues	4
- GUI is minimal or not functional	0
Data Persistence (if applicable)	
- Data saving and loading features work correctly	**5**
- Data persistence features have minor issues	3
- Data persistence features are not working	0
Input Validation and Error Handling	
- Program handles invalid inputs gracefully without crashing	**5**
- Some invalid inputs cause errors or crashes	3
- No input validation; program crashes on invalid input	0

2. Code Quality and Best Practices (20 Points)

Criteria	Points
Code Organization	
- Code is well-organized with appropriate functions and/or classes	**10**
- Code is moderately organized; some functions/classes used	7
- Code is poorly organized; minimal use of functions/classes	4
- Code is unorganized; no functions/classes used	0
Programming Concepts Usage	
- Appropriate use of data structures, control flow, and algorithms	**5**
- Basic use of programming concepts; could be improved	3
- Inappropriate or incorrect use of programming concepts	0
Code Efficiency	
- Code is efficient with no unnecessary complexity	**5**
- Code has some inefficiencies or unnecessary complexity	3
- Code is inefficient and overly complex	0

3. Comments and Internal Documentation (10 Points)

Criteria	Points
Commenting	
- All code sections are appropriately commented	**5**
- Most code sections are commented; some comments missing	3
- Few comments present; many sections lack explanations	1
- No comments in the code	0
Readability	
- Code is easy to read; proper indentation and naming conventions used	**5**
- Code is readable; minor issues with formatting or naming	3
- Code is difficult to read; poor formatting and naming	1
- Code is unreadable due to poor formatting	0

B) Documentation (30 Points)

1. Formatting and Structure (10 Points)

Criteria	Points
Title Page	

- Includes a separate title page with project title, student's name, date, and period	**2**
- Title page is missing one element	1
- Title page is missing or lacks multiple required elements	0
Purpose	
- Clearly explains the purpose of the activity in two sentences	**2**
- Purpose is unclear or too brief/lengthy	1
- Purpose is missing	0
Procedures and Explanation	
- Organized and detailed explanation of procedures followed	**2**
- Explanation is incomplete or lacks organization	1
- Procedures are not explained	0
Pictures of Code for Each Explained Part	
- Includes code snippets/screenshots for each explained part	**2**
- Some code snippets/screenshots are missing	1
- No code snippets/screenshots provided	0
Results and Outputs with Screenshots	
- Includes screenshots of results and outputs	**2**
- Some screenshots are missing or incomplete	1
- No screenshots of results provided	0

2. Detailed Explanation of Code Implementation (10 Points)

Criteria	Points
Comprehensiveness	
- Thoroughly explains how the code works, step by step	**5**
- Explains most parts; minor details missing	3
- Explanation is incomplete; significant details missing	1
- Explanation is minimal or not provided	0
Programming Concepts Explanation	
- Clearly explains which programming concepts are used and how	**5**
- Explains some concepts; lacks depth	3
- Minimal explanation of programming concepts	1
- No explanation of programming concepts	0

3. Conclusion (10 Points)

Criteria	Points
Results Discussion	
- Clearly states what the results were	**3**
- Results are mentioned but lack clarity	2
- Results are not discussed	0
Error Analysis	
- Identifies any errors encountered and discusses them	**3**
- Mentions errors but lacks detail	2
- Does not discuss errors	0
Improvements	
- Provides thoughtful insights on how to improve the project	**4**
- Suggestions for improvement are vague or minimal	2
- Does not provide any suggestions for improvement	0

Total Score Calculation

- **Code Implementation:** _____ / 70 Points
- **Documentation:** _____ / 30 Points
- **Extra Credit:** _____ / 10 Points
- **Late Penalty:** _____ Points Deducted
- **Final Score:** _____ / 100 Points

Detailed Breakdown of Documentation Requirements

Title Page (Separate Page)

- **Must Include:**
  - **Title** of the project.
  - **Student's Name**
  - **Date**
  - **Period**

Purpose

- **Requirements:**
  - Explain the purpose of the activity in **two sentences**.
  - Should provide a clear and concise overview of what the project aims to achieve.

Procedures and Explanation

- **Requirements:**

    - Provide an organized and detailed step-by-step explanation of how the project was developed.
    - Discuss the approach taken to solve the problem.
    - Include explanations of key functions, classes, or modules used.

Pictures of Code for Each Explained Part

- **Requirements:**
    - Include code snippets or screenshots corresponding to each part explained in the procedures.
    - Ensure that the code images are clear and readable.
    - Label each code snippet appropriately.

Results and Outputs Including Screenshots

- **Requirements:**
    - Present screenshots of the program in action.
    - Include examples of different functionalities (e.g., adding a task, answering a quiz question).
    - Ensure screenshots are clear and illustrate the key features.

Conclusion

- **Requirements:**
    - **What were the results?**
        - Summarize the outcomes of the project.
        - Discuss whether the project met its objectives.
    - **What was the error?**
        - Identify any issues or bugs encountered.
        - Explain how they were addressed or remain unresolved.
    - **How would you improve?**
        - Provide insights into potential enhancements.
        - Reflect on what could be done differently in future iterations.
