Project

Start Assignment

- Due Thursday by 23:59
- Points 100
- Submitting a file upload
- Attempts 0
- Allowed Attempts 2

OVERVIEW

In this individual project you have the opportunity to demonstrate your ability to create a test plan, design a suite of unit tests, debug and refactor an application, create Python documentation, use git and report on the findings.

Weighting: 50%

Learning Outcomes

This assessment contributes to the assessment of the following course learning outcomes:

- LO1: Recommend and apply a range of software quality assurance techniques to assess correctness of systems
- LO2: Plan, design, execute and manage testing activities using manual and automated techniques to assess the software quality.
- LO4: Apply programme maintenance techniques, including bug fixing and testing, optimisation and refactoring to ensure software efficiency and improve the software performance.
- LO5: Organise source code by using source and version control tools for collaboration and manage changes to course code over time.
- LO6: Produce and revise accurate and clear technical and user documentation to assist users and developers to use the software product.

Conditions

- It is recommended that you spend up to 4 days – around 32 hours to complete this assessment.
- You have access to all course materials and any other resources you wish to use as you work on this assessment.
- You must complete this assessment on your own. You can ask a tutor to clarify the instructions and for advice, but they cannot assist you in completing the tasks required – you must carry out the tasks yourself!

Success Criteria

This written assessment contributes to 50% of the final grade.

Tasks	Marks available	Type assessment
Test Plan	20	Achievement-based
Unit Test Design	20	Achievement-based
Debugging Implementation	15	Achievement-based
Code Refactoring	15	Achievement-based
Version Control	10	Achievement-based
Documentation	10	Achievement-based
Summary Report	10	Achievement-based

The total marks available for this assessment is 100.

To be successful in this task you must obtain a minimum of 50% of the total available marks. You will be marked according to the rubrics in the Marking Form. Please take the time to study these as they contain information on how you will be assessed.

You are allowed a maximum of two attempts. The maximum percentage to be awarded on a second assessment attempt is 50%. Even if you score higher than 50% on a second attempt, you will only be awarded 50%.

INSTRUCTIONS

Scenario

You work for a software development company that specialises in educational software for high school age students. The project is developing a 10pin bowling game prototype that can be used to teach a variety of subjects. The source code has been developed and your role is to test this prototype using the provided source code.

Your course tutor will provide you with code for the back end of the program. There is not yet any GUI, and no system for receiving input data from files or a database. Once the backend has been tested, these parts will be developed. The bowling game source code file shows example usage of the system. Activities for this project assessment are:

- Create a Test Plan
- Design a suite of unit tests
- Debug and refactor an application
- Create Pythondoc and associated documentation
- Use git
- Report findings

Business Rules

How the program works is largely defined by the business rules developed with the client at the beginning of the project.

Rules of Play

Each game of bowling consists of ten frames. In each frame, the bowler will have two chances to knock down as many pins as possible with their bowling ball. In games with more than one bowler, as is common, every bowler will take their frame in a predetermined order before the next frame begins. If a bowler knocks down all ten pins with their first ball, he is awarded a strike. If the bowler knocks down all 10 pins with the two balls of a frame, it is known as a spare. Bonus points are awarded for both a strike and a spare. The bonus points awarded depend on what is scored in the next 2 balls (for a strike) or 1 ball (for a spare). If the bowler knocks down all 10 pins in the tenth frame, the bowler is allowed to throw 3 balls for that frame. This allows for a potential of 12 strikes in a single game, and a maximum score of 300 points, a perfect game.

Scoring

In general, one point is scored for each pin that is knocked over. For example, if a player bowls over three pins with the first shot, then six with the second, the player would receive a total of nine points for that frame. If a player knocks down nine pins with the first shot, but misses with the second, the player would also score nine. When a player fails to knock down all ten pins after their second ball it is known as an open frame. In the event that all ten pins are knocked over by a player in a single frame, bonuses are awarded.

A ten-pin bowling score sheet showing how a strike is scored: Bowlstrike.png

Strike: When all ten pins are knocked down with the first ball (called a strike and typically rendered as an “X” on a score sheet), a player is awarded ten points, plus a bonus of whatever is scored with the next two balls. In this way, the points scored for the two balls after the strike are counted twice.

Frame 1, ball 1: 10 pins (strike) Frame 2, ball 1: 3 pins
Frame 2, ball 2: 6 pins

The total score from these throws is:

Frame one: 10 + (3 + 6) = 19 Frame two: 3 + 6 = 9
TOTAL = 28

A player who scores multiple strikes in succession would score like so:

Frame 1, ball 1: 10 pins (strike) Frame 2, ball 1: 10 pins (strike) Frame 3, ball 1: 4 pins
Frame 3, ball 2: 2 pins

The score from these throws are:

Frame one: 10 + (10 + 4) = 24 Frame two: 10 + (4 + 2) = 16 Frame three: 4 + 2 = 6
TOTAL = 46

The most points that can be scored in a single frame are 30 points (10 for the original strike, plus strikes in the two subsequent frames).

A player who bowls a strike in the tenth (final) frame is awarded two extra balls allowing the playing to gain bonus points. If both these balls also result in strikes, a total of 30 points (10 + 10 + 10) is awarded for the frame. These bonus points do not count on their own; they only count as the bonus for the strike.

Your Tasks

- Create a plan to test the software. (LO2)
- Your plan should include unit testing as described below.
- Design a suite of unit tests for the software. (LO2)
- Test the software using the unit tests. If you identify any bugs, fix them. (LO2, LO4)
- Each time you make a change, make a git commit with appropriate comments. (LO5, LO6)
- Make sure the program is completely documented using PythonDoc comments and generate PythonDoc documentation. (LO6)
- Write a one page report summarising all of the above. (LO2, LO6)

Deliverables

Please submit the following for assessment:

- Your test plan
- A zip of the entire project folder (including your fixed program and your test suite)
- Your generated Pythondoc documentation
- Your report, including:
- Identifying instances of refactoring, and why they improve the program.
- A link to your git repository

SUBMISSION

Submission Checklist

Before you submit your work in Canvas, make sure you have completed all the tasks in this checklist:

Task
You have read through this document including all the task requirements
Created test plan
Created test cases
Tested software
Fixed errors discovered through testing
Refactored where appropriate
Committed your changes to Git report
Produced a reported detailing process and results

Submission Instructions

Once you have completed the research, the required report, submit your assessment as follows:

1. Verify that all items have been completed using the submission checklist above.
2. Read the declaration below.
3. Upload your assessment documentation and submit your work.

Declaration

The work presented in this assessment is to the best of my knowledge original, except as acknowledged in the text, and the material has not already been submitted, either in whole or in part, for any academic award at this or any other tertiary institution. I promise not to share this assessment in part or whole with any other student at Whitecliffe or outside this campus.

A3 Rubric

Criteria	Ratings				Pts
Test Planning Bowling game test plan, Test strategy, Risk assessment, Test approach	**20 to >17.0 pts** **Comprehensive test plan with detailed risk assessment and clear methodology for bowling game features.**	**17 to >12.0 pts** **Good test plan with adequate risk consideration. Minor gaps in approach.**	**12 to >7.0 pts** **Basic test plan meeting minimum requirements. Limited risk assessment.**	**7 to >0 pts** **Poor or inadequate test plan.**	20 pts
Unit Test Design Test case design, Coverage of game rules, Test data selection	**20 to >17.0 pts** **Complete unit test suite with excellent coverage of all bowling rules and scoring scenarios. Thoughtful test data selection.**	**17 to >12.0 pts** **Good test suite with adequate coverage. Some gaps in edge cases or test data.**	**12 to >7.0 pts** **Basic test suite with limited coverage. Minimal consideration of test data.**	**7 to >0 pts** **Poor test suite design or coverage.**	20 pts
Debugging Implementation Bug identification, Fix implementation, Issue documentation	**15 to >13.0 pts** **All bugs systematically identified and fixed with comprehensive documentation of issues and solutions.**	**13 to >10.0 pts** **Most bugs identified and fixed with good documentation. Minor issues remain.**	**10 to >6.0 pts** **Basic debugging with some documentation. Several issues unresolved.**	**6 to >0 pts** **Poor debugging effort or documentation.**	15 pts
Code Refactoring Code optimization, Refactoring justification, Improvement evidence	**15 to >13.0 pts** **Excellent refactoring with clear justification and measurable improvements. Follows best practices.**	**13 to >10.0 pts** **Good refactoring with reasonable justification. Some improvements evident.**	**10 to >6.0 pts** **Basic refactoring with limited justification or improvement.**	**6 to >0 pts** **Poor refactoring effort or justification.**	15 pts
Version Control Git usage, Commit frequency, Commit message quality	**10 to >9.0 pts** **Excellent Git usage with appropriate commit frequency and detailed, descriptive commit messages.**	**9 to >7.0 pts** **Good Git usage with reasonable commit frequency. Some commit messages lack detail.**	**7 to >5.0 pts** **Basic Git usage meeting minimum requirements. Inconsistent commit messages.**	**5 to >0 pts** **Poor Git usage or commit quality.**	10 pts
Documentation Pythondoc comments, API documentation, Generated documentation quality	**10 to >9.0 pts** **Comprehensive Pythondoc documentation with detailed API docs and excellent generated output.**	**9 to >7.0 pts** **Good documentation with adequate API coverage. Some minor gaps.**	**7 to >5.0 pts** **Basic documentation meeting minimum requirements. Limited API coverage.**	**5 to >0 pts** **Poor or inadequate documentation.**	10 pts

Criteria	Ratings				Pts
Summary Report Findings summary, Testing approach, Recommendations	**10 to >9.0 pts** **Professional report with thorough analysis, clear methodology description, and actionable recommendations**	**9 to >7.0 pts** **Good report with adequate analysis and methodology. Some gaps in recommendations**	**7 to >5.0 pts** **Basic report meeting minimum requirements. Limited analysis or recommendations**	**5 to >0 pts** **Poor or incomplete report**	10 pts
					Total Points: 100