# Educational Bowling Game Test Plan

## Test Plan Identifier

Educational Bowling Game Test Plan v1

## Introduction

This document describes what this master test plan is meant to achieve:

* To describe the extent of which the program will be tested, and how they will be conducted.
* To list constraints regarding budget and time.
* To list the environmental needs of which the tests will be conducted with.
* To ensure that the Bowling Game’s mechanics work as expected.

Only unit tests will be conducted.

## Test Items

The test item, BowlingGame.py, is the sole file that makes up the educational bowling game. Main.py will be used to test the forementioned file.

The Bowling Game will be tested on a computer running Windows 10. Since the game is written in Python, the latest stable versions of Python and Wing IDE should be installed as they are to be used for running both the game and the tests.

## Features to be tested

The features to be tested are as follows:

* As a player, score a total of zero for all 10 frames/20 rolls
* As a player, score only 2 pins per frame for all 10 frames (total score of 20)
* As a player, score a spare and receive a bonus
* As a player, score a strike and receive a bonus
* As a player, score a perfect game (all strikes, total score of 300)
* As a player, score only spares for the whole game (total score of 150)

## Features not to be tested

Since all the features listed above are essentially every feature that the game’s current version has to offer, no features will not be tested. Although it is expected that running the game and the tests on a different environment will provide the same results, so long as the environment can run Python.

## Approach

The tests will be conducted using Python’s unit testing framework called *unittest*, which supports test automation. Depending on a test case’s results, unittest will display on a console either a ‘.’ or a ‘E’ to indicate success or failure, and show the reasoning behind a test’s failure. Test results will be summarized in a one-page Word document once testing ends.

## Item pass/fail criteria

It is expected that both BowlingGame.py and Main.py will have no critical defects while the test cases are being conducted. Since the size of the code being tested is up to 50 lines long, which is easily maintainable should there be any bugs, 100% of the test cases must pass. Testing will fail if at least one test case fails.

## Suspension and resumption requirements

All testing should be suspended in case the testing halts due to an ‘AssertionError’, which prevents any further test cases to be conducted. This implies that the test failed, and thus will be mentioned in the summary report. Any errors that may arise should also result in the suspension of tests, as an error can be caused by a mere typo (syntax error).

Testing will resume once any errors are presumed fixed.

## Test deliverables

Deliverables include:

* This document
* The test results summarized in a one-page Word document
* The test summary document

## Testing tasks

* Prepare the test plan
* Create test cases and use PythonDoc
* Perform tests
* Diagnose any errors if present
* Summarize test results in a one-page Word document
* Prepare one-page summary report

## Environmental needs

Python and Wing IDE must be setup on a compatible system before the tests are conducted. If Wing IDE proves to be inconvenient, any other IDE that supports Python debugging should be fine.

## Responsibilities

The author of this document bears all responsibility regarding the entire testing process.

## Staffing and training needs

There is only one person working on testing, and that is the author of this document. Experience with Python, or any other similar programming language is required for testing and debugging the game.

## Schedule

After the completion of this test plan, testing and debugging will commence, and it must be completed by the 1st of October, 2021. The tester is free to create test cases, test, and debug game at their own pace.

## Risks and contingencies

Risks include:

* Tester complacency
* Tester falls ill for a long period
* Unfamiliarity with Python
* Test cases are re-evaluated by a professional, which can increase the workload

A contingency should these risks become real include extending the completion date by an extra week, or asking for time off of work.

## Approvals

The author of this document will approve whether the entire testing project is finished or not.