MSSE SOFTWARE, INC.

Test Plan for

GolfScore

Revision 1.1

Shamima Sultana

December 4, 2023

Contents

1.0 Introduction

1.1. Objective

This document outlines the test plan for the upcoming GolfScore Release 1.1. It provides comprehensive details regarding the scope of testing, testing methodologies, and exclusions. Additionally, it includes specifics about the testing schedule, planned tests, dependencies, necessary resources, testing tools, and performance metrics. It is imperative that any alterations in requirements or team structure are accurately reflected in this document.

The primary objective of this testing effort is to validate the software requirements specified in the GolfScore program, as outlined in Appendix A of this document. The plan will encompass a thorough examination of the system's functionalities to ensure adherence to the specified requirement .

1.2. Project Description

GolfScore is a software application designed for the generation of golf tournament results for players on each course. The program operates by taking an input text file, as detailed in the Software Requirements Specification (SRS). Subsequently, it generates three output text files, also outlined in the SRS, containing relevant information related to the golf tournament results

1.3. Process Tailoring

The GolfScore program operates independently, devoid of external dependencies. Consequently, the test plan is structured around Functional and Non-functional Testing within the framework of Design Verification and System Validation. Testing unfolds in distinct phases:

1. Entrance Test:
    - Objective: Verify the correct execution of the program and its ability to handle input parameter errors as outlined in the SRS.
    - Reference: Refer to Appendix C for a detailed description of Entrance Testing test cases. Appendix A contains the relevant details from the SRS.
2. Main Test:
    - Objective: Confirm the accuracy of program execution by assessing its ability to process input data in accordance with specifications, and to generate the required outputs. Additionally, scrutinize the program's handling of input and output errors for correctness.
    - Reference: Detailed Main Testing test cases are outlined in Appendix C.
3. Exit Test:
    - Objective: Validate whether the program has produced the required outputs and saved them in the correct format and location
    - Reference: Review Exit Testing test cases in Appendix C.
4. Regression Test:
    - Objective: Post identification and resolution of defects during testing, rerun all tests to ensure consistent and proper behavior.

The following references are used to produce this document :

- Software Requirements Specification for GolfScore, Revision 1, dated July 18, 2017.
- System Verification Test Plan for Advanced Color Module, Revision 2, dated February 22, 2000.

2.0 Assumptions/Dependencies

The development team is assumed to conduct unit testing and integration testing during the software development process. Customer validation testing is expected to be conducted collaboratively by field personnel and customers.

To align with the established schedule, the development team is required to make the program available by January 10, 2024.

3.0 Test Requirements

1. Entrance Tests:
    - The program is implemented in either C or C++,Java.
    - The program is designed to operate on a PC running Windows 7 or any later version.
    - The program functions as a stand-alone executable.
    - Execution from the command line prompt is supported by the program.
    - The program is launched with valid input parameters.
    - The program's compatibility with common development environments is verified.
    - It seamlessly handles standard input formats as specified in the Software Requirements Specification (SRS).
    - The program initiates without critical errors or crashes during the start-up process.
    - Environmental variables and dependencies required for the program are appropriately configured.
2. Main Tests:
    - The tournament must involve a specified number of golf courses, ranging from 1 to 5.
    - Each golfer is required to play every course exactly once.
    - The number of golfers participating in the tournament must fall within the range of 2 to 12.
    - Par values for holes on each course should be limited to 3, 4, or 5.
    - Scores earned by golfers for each hole played must be within the range of 0 to 6, inclusive.
    - The first section of records in the input file, specifically the course records, should exist and adhere to the prescribed format for each entry.
    - An identifiable delimiter record must mark the conclusion of course records.
    - A second set of records, the golfer records, should exist in the input file, with each entry conforming to the specified format.
    - Another delimiter record must signal the end of the input file.
    - The program should gracefully handle cases where the input file is empty or missing.
    - It should appropriately manage scenarios with irregular or unexpected input formats, providing meaningful error messages.
    - The program's processing time for varying input sizes should be within acceptable limits.
    - Robustness testing should be performed to evaluate the program's behavior under stress conditions, including large datasets and simultaneous user requests.
3. Exit Tests:
    - The program is expected to generate multiple reports corresponding to the specified options.

- The produced reports should be stored as text files in the designated output directory. In the absence of a specified directory, they should be saved in the same directory as the input file, with the file extension ".rep."
- Upon request, the tournament ranking report should include a comprehensive list of all golfers in the specified format. The list should be arranged in descending order based on the final score and saved with the output filename "trank.rep."
- If requested, the golfer report should feature an alphabetical list of all golfers based on their last names, following the specified format. The report should be saved with the output filename "golfer.rep."
- If requested, the course report should include a section for each Golf Course listed in the input Course Records, adhering to the specified format. The report should be saved individually for each course, using appropriate filenames.
- Reports should be generated efficiently, and their content should accurately reflect the tournament results.
- The program should handle situations where the output directory is read-only or inaccessible, providing appropriate error messages.
- - It should gracefully manage cases where specific reports are not requested, ensuring no unnecessary files are generated.
- The program should perform file-saving operations securely, preventing data corruption or loss during the report generation process.
- Validation checks should be in place to ensure that generated filenames are valid and conform to file naming conventions.

4.0 Test Tools

To facilitate the testing process, the following testing tools are essential:

- Defect reporting and tracking software
- Installation media for various Windows versions beyond 2000, including but not limited to XP, Vista, 7, 8, 8.1, and 10.
- Automated testing frameworks for efficient and systematic test execution.
- Performance testing tools to assess the program's responsiveness and scalability under different conditions.
- Security testing tools to identify vulnerabilities and ensure robust protection against potential threats.
- Code coverage tools for analyzing the extent to which the program's code is exercised during testing.
- Version control systems to manage and track changes in the software codebase.
- Load testing tools to evaluate the program's performance under different levels of user activity.
- Compatibility testing tools to verify the program's functionality across diverse hardware and software configurations.
- Documentation tools to streamline the creation and maintenance of test cases, plans, and results.
- Network simulation tools to emulate various network conditions and assess the program's behavior in different environments.
- Data generation tools to create diverse datasets for comprehensive testing scenarios.
- Test management tools for effective planning, execution, and tracking of the entire testing process.

5.0 Resource Requirements

The following resource would be required:

- GolfScore Program version 1.1
- Three PCs with the capability to host virtual machines
- Virtualization software for managing virtual environments
- Three personnel assigned to the Test Group, each with a commitment of at least 70% of their working hours dedicated to this testing effort. Refer to Appendix A for specific details regarding Test Group personnel.
- Defect reporting and tracking software for efficient management of identified issues.
- Installation media for multiple Windows versions beyond 2000 (e.g., XP, Vista, 7, 8, 8.1, & 10) to ensure comprehensive compatibility testing.
- Automated testing tools to enhance efficiency in test execution and result analysis.

6.0 Test Schedule

No.	Test	Start	Finish
1	Test Development	22.12.2023	10.01.2024
2	Program Availability	04.01.2024	--
3	Entrance Testing	05.01.2024	15.01.2024
4	Main Testing	16.01.2024	30.01.2024
5	Exit Testing	31.01.2024	5.02.2024
6	Regression Testing	6.02.2024	10.02.2024

See Appendix B for details.

7.0 Risks/Mitigation

The absence of a program designed to enforce conformity to a structured input format significantly elevates the risk of encountering errors within the input data

8.0 Metrics

The following metrics data will be collected. Some will be collected prior to, and some after product shipment.

Prior to shipment:

Effort expended during DVT, SVT and Regression

# of defects uncovered during DVT, SVT and Regression, and development phase each defect is attributable to

Test tracking S-Curve

PTR S-Curve

After shipment:

# of defects uncovered and development phase each defect is attributable to

Size of software

Appendix A – Detailed Resource Requirements

No.	Test	No of Personnel	No of Hours
1	Test Development	3	70
2	Entrance Testing	2	30
3	Main Testing	4	90
4	Exit Testing	4	30
5	Regression Testing	4	50

- PCs that are capable of hosting virtual machines are required such that the program can be tested on multiple versions of Windows.
- A virtualization software is required such that multiple versions of Windows can be installed to test the program.

Appendix B – Detailed Test Schedule

No.	Test	Start	Finish
1	Test Development	22.12.2023	10.01.2024
2	Program Availability	04.01.2024	--
3	Entrance Testing	05.01.2024	15.01.2024
4	Main Testing	16.01.2024	30.01.2024
5	Exit Testing	31.01.2024	5.02.2024
6	Regression Testing	6.02.2024	10.02.2024

No.	Test	Dependencies
1	Test Development	3 PCs 3 Personnel
2	Program Availability	GolfScore Program
3	Entrance Testing	2 PCs 2 Personnel Virtualization Software
4	Main Testing	4 PCs 4 Personnel Virtualization Software
5	Exit Testing	4 PCs 4 Personnel Virtualization Software
6	Regression Testing	4 PCs 4 Personnel Virtualization Software

Appendix C – Test Cases

Test No.	Test Case	Test Type
1	The program shall be written in C or C++	Non-functional
2	The program shall run on a PC Windows 2000,XP, Vista,7,8,10	Non-functional
3	The program shall operate as a stand-alone executable.	Non-functional
4	The program should execute from the command line prompt.	Non-functional
5	The program shall accept the command line option "-ctg."	Functional
6	The program shall accept the command line option "-c."	Functional
7	The program shall accept the command line option "-t."	Functional
8	The program shall accept the command line option "-g."	Functional
9	The program shall accept the command line options "-c -t -g."	Functional
10	The command line option "-k" shall prompt an "unrecognizable options" message.	Functional
11	The command line option "-j" shall prompt an "unrecognizable options" message.	Functional
12	The command line option "-kj" shall prompt an "unrecognizable options" message.	Functional
13	The command line option "-ckj" shall prompt a message stating "unrecognizable options."	Functional
14	Specifying a non-existing input filename shall result in an "input parameter error" being displayed.	Functional
15	Specifying a non-existing output directory shall result in an "input parameter error" being displayed.	Functional
16	The command line option "-g" shall be accepted and display help information.	Functional
17	The program called as "golf -ctg in.txt golfout" with an existing and valid "in.txt" file and an existing "golfout" folder shall be accepted.	Functional
18	If the program is called as "golf -ctg in.txt golfout dis" where "in.txt" exists and is valid, and the folder "golfout" also exists, it shall be accepted.	Functional
19	If the program is called as "golf -ctg in.txt golfout" where "in.txt" exists and is valid, but the folder "golfout" does not exist, it shall display an "input parameter error."	Functional
20	If the program is called as "golf -ctg in.txt golfout" and "in.txt" does not exist, it shall display an "input parameter error."	Functional
21	The number of golf courses "1,5" shall be accepted.	Functional
22	The number of golf courses "-5,0,6" shall result in an error	Functional
23	Multiple records for a golfer on the same golf course shall be accepted. A message shall be displayed to indicate this, and the program shall use the first record while continuing the processing.	Functional

24	The number of golfers “0,1” shall return an error	Functional
25	The number of golfers “2,12” shall be accepted	Functional
26	The number of golfers “13” shall return an error	Functional
27	Par for hole “2,6” shall return an error	Functional
28	Par for holes "3,4,5" shall be accepted.	Functional
29	A golfer score per hole of "-1,7" shall trigger an error.	Functional
30	A golfer score per hole of "0" shall be accepted.	Functional
31	Input data containing non-numeric values where numeric data is expected shall result in an error.	Functional
32	Input data featuring numeric values where non-numeric data is anticipated shall result in an error.	Functional
33	Input data violating delimiter constraints shall trigger an error.	Functional
34	An input file without course records shall return an error.	Functional
35	An input file lacking golfer records shall result in an error.	Functional
36	When invoked with the "-ctg" option, the program generates "trank.rep," "golfer.rep," and "course.rep." If any of these files exist, the user shall be prompted to decide whether to overwrite them or not.	Functional
37	When invoked with the command line option "-c," the program generates an output file named "course.rep." If the file already exists, the user shall be prompted to decide whether to overwrite it or not.	Functional
38	When executed with the command line option "-t," the program generates an output file named "trank.rep." If the file already exists, the user shall be prompted with a message indicating the existing file and asked whether to overwrite it or not.	Functional
39	When invoked with the command line option "-g," the program generates an output file named "golfer.rep." If the file already exists, the user shall be prompted to decide whether to overwrite it or not.	Functional
40	If the program encounters insufficient permissions preventing the saving of the output, it shall display an error.	Functional