Test Requirements/Test Cases for
Golf Score

1.1
02/17/23

Moti Begna, Kalven Schraut, Rohit Bagda
Team Koala Bears

Document Revision History

Rev	Date	Author	Change Description
1.0	5 Feb 2022	Moti B, Kalven S, Rohit B	Initial release of this document.
1.1	31 March 2022	Moti B, Kalven S, Rohit B	Updates to existing test cases and additional tests are added. List of changes can be found in the Test Plan / Case Revisions for 1.1 document.

Table of Contents

1. Test Requirements

1.1 Purpose

The purpose for this Test Requirements document is to list all the explicit and implicit requirements inferred from the Golf Score SRS document which serves as a test basis for the test cases in section 2.

1.2 Explicit Requirements

The explicit requirements were obtained from the Software Requirements Specification Document of the Golf Score Software Requirements document.

1.3 Implicit Requirements

The implicit requirements were obtained from inferences made based on our understanding of the SRS. That is to say, we derived them from our own rationale of the domain.

1.4 Requirements Matrix

Requirement number	Requirement name	Relevant section(s) of SRS	Relevant Test Cases	Notes
1	**Input Parameters**			
1-A	GolfScore shall parse the first field as options	2.2	1	The SS does not restrict which options can be imputed. The handled options are -c, -t, -g, -h.
1-B	GolfScore shall parse the second field as filename	2.2	3	
1-C	GolfScore shall parse the third field as output-directory	2.2	4	
1-D	When a field after the filename is not provided, GolfScore shall use the directory of the filename	2.2	2	
1-E	When the -h option is provided, help information shall be displayed and all other options and fields will be ignored. Version number of the software should also be displayed	2.2	1	
2	**Input Parameter Errors**			
2-A	When a specified filename does not exist, GolfScore shall report a file not found message.	2.6.1	3	
2-B	When a field following the filename is provided, GolfScore shall validate if it is an existing directory	2.6.1	5	
2-D	When no valid options are provided (-h, -c, -t, -g or some combination), a invalid option message shall be displayed along with the help information	IMP	32	The program requires some kind of flag otherwise it won't know what to do so based on how a typical CLI behaves in the fashion we added this implicit requirement.
3	**Input Data Errors**			
3-A	When par values are not 3, 4, or 5, GolfScore shall report an invalid par value error	2.6.2	6	

Requirement number	Requirement name	Relevant section(s) of SRS	Relevant Test Cases	Notes
3-B	When a golfer has two or more records for the same golf course, GolfScore shall report a duplicate record message and discard any after the first.	2.6.2	7	
3-C	When stroke count is a non numeric integer the program will display an appropriate error message.	2.6.2	8	
3-D	When course record line length is not 38, GolfScore shall report an invalid course row length error.	IMP	9	Assumptions made where each column is a 1 character width wide and that there can be no more or no less otherwise processing will be incorrect.
3-E	When the golf record line length is not 48, GolfScore shall report an invalid record length error.	IMP	10	See notes in 3-D also to note that this also restricts the stroke count to only single digits
4	**Errors on Output**			
4-A	When the output report already exists in the output-directory, GolfScore shall prompt the user if they want to overwrite the existing file.	2.6.3	12	
4-B	When a user responds with N to the file overwrite, GolfScore shall not overwrite the existing file.	2.6.3	13	
4-C	When a user responds with Y to the file overwrite, GolfScore shall overwrite the existing file.	2.6.3	14	
5	**Tournament Ranking Report (TRR) Output**	2.5.1		
5-A	TRR must contain name, score, tournament score and standing for each golfer.	2.5.1	15	
5-B	Each Golfer's name in TRR must be in columns 0-19.	2.5.1 (imp)	15	
5-C	Each golfer's score for the course (1-5) must be in columns 20-24 respectively	2.5.1 (imp)	15	
5-D	Each golfer's tournament score must be in column 25-27	2.5.1 (imp)	15	Score will be read from left to right so single digit scores will have empty spaces in columns 26-27 and double digit scores will have an empty space in column 27.
5-E	Each golfer's tournament standing must be in column 28-29	2.5.1 (imp)	15	
5-F	TRR must be sorted in descending order of final score.	2.5.1	16	
5-G	For golfers with equal scores a secondary sort must be applied to the TRR to sort the tied golfers data based on alphabetical order of last name.	2.5.1	17	Incomplete requirement - alphabetical order for first name or last name not specified. Assumption made is alphabetical order of last name.
5-H	The TRR output file must be called trank.rep	2.5.1	18	
5-I	The TRR output file must be published in the output directory	2.5.1	19	
6	**Golfer Report (GR) Output**			
6-A	GolfScore shall output the GR in the same format as the TRR	2.5.2	20	
6-B	GolfScore shall output the same data as TRR	2.5.2	20	

Requirement number	Requirement name	Relevant section(s) of SRS	Relevant Test Cases	Notes
6-C	GolfScore shall sort the GR data by golfer's last name for the output report	2.5.2	21	
6-D	The GR output file must be called golfer.rep	2.5.2	22	
6-E	The GR output file must be published in the output directory.	2.5.2	23	
7	**Course Report (CR) Output**			
7-A	GolfScore shall output the CR with a section for each course	2.5.3	24	
7-B	GolfScore shall output the golfer's name, score per hole and course score per course section.	2.5.3	24	
7-C	GolfScore shall output in the CR per course section per golfer their name in columns 0-19.	2.5.3 (imp)	24	
7-D	GolfScore shall output in the CR per course section per golfer's stroke count per hole in columns 20-37	2.5.3 (imp)	24	
7-E	GolfScore shall output in the CR per course section per golfer's course their course score in columns 48-40	2.5.3	24	
7-F	GolfScore shall sort the output of CR per course by descending of each golfer's course score.	2.5.3	25	
7-G	For golfers with equal scores a secondary sort must be applied to the CR to sort the tied golfers data based on alphabetical order of last name per course section	2.5.3 (imp)	26	
7-H	GolfScore shall output the CR in a file named course.rep	2.5.3	27	
7-I	GolfScore shall output the CR in the output directory		28	
8	**Program Functionality**			
8-A	GolfScore shall calculate the hole score for each golfer from the hole's par.	2.3.2	29	
8-B	GolfScore shall award score of 0 for over par	2.3.2	29	
8-C	GolfScore shall award score of 1 for par	2.3.2	29	
8-D	GolfScore shall award score of 2 for 1 under par	2.3.2	29	
8-E	GolfScore shall award score of 4 for 2 under par	2.3.2	29	
8-F	GolfScore shall award score of 6 for 3 or more under par	2.3.2	29	
8-G	GolfScore shall calculate a course score per golfer by taking the sum of the scores of all the holes in a course.	2.3.1	29	
8-H	GolfScore shall calculate a tournament score per golfer by taking the sum of the course scores (8-G).	2.3.1	30	
8-I	GolfScore shall calculate a tournament standing for the golfers by sorting the tournament score in descending order and assigning the standing based on the position starting at 1	2.3.1	31	
9	**Non-Functional**			
9-A	When running Java –version on the system it is greater than 8	1.3	0	
9-B	When executed golf score shall finish within 1 minute	4	0-30	Every test case run will need to finish within the 1 minute threshold to meet this requirement.

1.5 Reference Documents

- Software Requirements Specification/ Design Document GolfScore Revision 1.0 by Neil Bitzengolfer

1.6 Definitions and Acronyms

- TRR - Tournament Ranking Report
- GR - Golfer Report
- CR - Course Report

2. Test Cases

2.1 Description

The test cases in this document cover system level testing for the Golf Score Application. In each test case, specific requirement number(s) will be stated in the purpose that corresponds to the requirements matrix. Test cases are grouped in test modules based on the requirement matrix sections. Also in each test case, any artifacts generated will be cleaned up at the end.

2.2 Test Case Modules

○ *2.2.1 Input Parameters*

Test Case ID	**Test Case Title**
0	Java Version
Purpose:	Verify requirements 9-A
Owner:	Team Koala Bears
	Expected Results: Java version running is greater than or equal to version 8
Test Data:	N/A
Test Tools:	N/A
Dependencies:	N/A
Initialization:	N/A
	Description: Running Java –version to get the version.

Test Case ID	**Test Case Title**
1	Help
Purpose:	Verify requirements 1-A and 1-E
Owner:	Team Koala Bears
	Expected Results: Help information is displayed to the console/stdout
Test Data:	N/A
Test Tools:	N/A
Dependencies:	N/A
Initialization:	N/A
	Description: Test will start by running the executable with the -h option for example, golf -h. and assert the expected results.

Test Case ID	**Test Case Title**

2	No Output Directory so Filename Directory is Used
Purpose:	Verify requirement 1-D
Owner:	Team Koala Bears
	Expected Results: Report file is written in supplied same directory as input file
Test Data:	~/golf-data/GolfScoreTestHappyPath.txt
Test Tools:	N/A
Dependencies:	N/A
Initialization:	N/A
	Description: Prior to running the golf program, the test will assert that course.rep does not exist in ~/gold-data. Next the test will run the executable, for instance golf -c ~/golf-data/GolfScoreTestHappyPath.txt, Next assert the course.rep file exists in ~/golf-data.

Test Case ID	**Test Case Title**
3	File not found
Purpose:	Verify requirement 2-A and 1-B
Owner:	Team Koala Bears
	Expected Results: Error displayed about file not found
Test Data:	N/A
Test Tools:	N/A
Dependencies:	N/A
Initialization:	N/A
	Description: Test will start by running the executable with a file path that does not exist and assert the expected results

Test Case ID	**Test Case Title**
4	Output Directory is Used to Write Reports In
Purpose:	Verify requirement 1-C
Owner:	Team Koala Bears
	Expected Results: Report file is written in supplied output directory
Test Data:	~/golf-data/GolfScoreTestHappyPath.txt
Test Tools:	N/A

Dependencies: N/A

Initialization: N/A

Description: Prior to running the golf program, the test will assert that course.rep does not exist in the output directory /home/kalvens/golf-score. Next the test will start by running the executable, for instance golf -c ~/golf-data/GolfScoreTestHappyPath.txt ~/golf-score. Then assert course.rep is a file that exists in the directory.

○ *2.2.2 Input Parameter Errors*

Test Case ID **Test Case Title**
5 Throws Error if Invalid Output Directory is Provided

Purpose: Verify requirement 2-B

Owner: Team Koala Bears

Expected Results: Error stating invalid output directory

Test Data: ~/golf-data/GolfScoreTestHappyPath.txt

Test Tools: N/A

Dependencies: N/A

Initialization: N/A

Description: Run the executable with an invalid output directory on the system and assert the expected results

○ *2.2.3 Input Data Errors*

Test Case ID **Test Case Title**
6 Throws Error if Invalid Par Value in Course Record

Purpose: Verify requirement 3-A

Owner: Team Koala Bears

Expected Results: Error stating invalid par value

Test Data: ~/golf-data/GolfScoreInvalidPars.txt

Test Tools: N/A

Dependencies: N/A

Initialization: N/A

Description: Run the executable with ~/golf-data/GolfScoreInvalidPars.txt as the input file and assert the expected results

Test Case ID **Test Case Title**
7 Displays Warning if Duplicate Golfer for the Same Course

Purpose: Verify requirement 3-B

Owner: Team Koala Bears

Expected Results: Warning message is displayed about duplicate golfer for the same course

Test Data: ~/golf-data/GolfScoreDuplicateGolferForCourse.txt

Test Tools: N/A

Dependencies: N/A

Initialization: N/A

Description: Run the executable with ~/golf-data/GolfScoreDuplicateGolferForCourse.txt as the input file and assert the expected results

Test Case ID **Test Case Title**
8 Throws Error if Non Numeric Course Par

Purpose: Verify requirement 3-C

Owner: Team Koala Bears

Expected Results: An Error is thrown and displayed if a for a non numeric par in a course record.

Test Data: ~/golf-data/GolfScoreNonNumericPar.txt

Test Tools: N/A

Dependencies: N/A

Initialization: N/A

Description: Run the executable with ~/golf-data/GolfScoreNonNumericPar.txt as the input file and assert the expected results

Test Case ID **Test Case Title**
9 Throws Error if Invalid Course Record Column Length

Purpose: Verify requirement 3-D

Owner: Team Koala Bears

Expected Results: An Error is thrown for invalid course record length

Test Data: ~/golf-data/GolfScoreInvalidCourseColumnLength.txt

Test Tools: N/A

Dependencies: N/A

Initialization: N/A

Description: Run the executable with ~/golf-data/GolfScoreInvalidCourseColumnLength.txt as the input file and assert the expected results

Test Case ID **Test Case Title**
10 Throws Error if Invalid Golf RecordColumn Length

Purpose: Verify requirement 3-E

Owner: Team Koala Bears

Expected Results: An Error is thrown for invalid golf record length

Test Data: ~/golf-data/GolfScoreInvalidGolfColumnLength.txt

Test Tools: N/A

Dependencies: N/A

Initialization: N/A

Description: Run the executable with ~/golf-data/GolfScoreInvalidGolfColumnLength.txt as the input file and assert the expected results

○ *2.2.4 Errors on Output*

Test Case ID **Test Case Title**
12 Prompts User to overwrite or not a pre-existing file.

Purpose: Verify requirement 4-A

Owner: Team Koala Bears

Expected Results: Message comes up with Y/N asking if wanting to overwrite existing file
Test Data: ~/golf-data/GolfScoreTestHappyPath.txt

Test Tools: N/A

Dependencies: N/A

Initialization: ~/golf-data/course.rep Exists

Description: Run the executable with -c option and ~/golf-data/GolfScoreInvalidGolfColumnLength.txt twice and expect a second run to ask if you want to overwrite.

Test Case ID **Test Case Title**
13 Files if overwritten if user types Y when prompted

Purpose: Verify requirement 4-B

Owner: Team Koala Bears

Test Data: Expected Results: Assert new file is written and overwrites existing file
~/golf-data/GolfScoreTestHappyPath.txt

Test Tools: N/A

Dependencies: N/A

Initialization: ~/golf-data/course.rep Exists

Description: Run the executable with -c option and ~/golf-data/GolfScoreInvalidGolfColumnLength.txt as the input file and assert the expected results

Test Case ID **Test Case Title**
14 File is not overwritten if user types N when prompted

Purpose: Verify requirement 4-C

Owner: Team Koala Bears

Test Data: Expected Results: Assert that the original file is the same and wasn't overwritten
~/golf-data/GolfScoreTestHappyPath.txt

Test Tools: N/A

Dependencies: N/A

Initialization: ~/golf-data/course.rep Exists

Description: Run the executable with -c option and ~/golf-data/GolfScoreInvalidGolfColumnLength.txt as the input file and assert the expected results

○ *2.2.5 Tournament Ranking Report (TRR) Output*

Test Case ID **Test Case Title**
15 Happy Path for TRR

Purpose: Verify requirements 5-A, 5-B, 5-C, 5-C, 5-D, 5-E

Owner: Team Koala Bears

Expected Results: Golfer's name in TRR will be contained in columns 0-19, their score for each course (1-5) will be in columns 20-24, their tournament score will be in column 25-27, and their tournament standing will be in column 28-29.

Test Data: ~/golf-data/GolfScoreTestHappyPath.txt

Test Tools: N/A

Dependencies: N/A

Initialization: N/A

Description: First, a valid input data file will be passed alongside specific CLI arguments in order to output a file containing Tournament Ranking data. The arguments will look something like this:
>golf -t c:\GolfScoreTestHappyPath.txt c:\output\
The test will then validate that file in the following order:
1. Assert that the golfers name is contained in columns 0-19
2. Assert that the golfers score for each course is contained in columns 20-24
3. Assert that the golfers tournament score is contained in column 25-27
4. Assert that the golfers tournament standing is contained in column 28-29

Test Case ID **Test Case Title**
16 Validate correct list ordering in the TRR

Purpose: Verify requirement 5-F

Owner: Team Koala Bears

Expected Results: The order of golfers in the TRR will be listed in descending order based on their final score.

Test Data: ~/golf-data/GolfScoreTestHappyPath.txt

Test Tools: N/A

Dependencies: N/A

Initialization: N/A

Description: First, a valid input data file will be passed alongside specific CLI arguments in order to output a file containing Tournament Ranking data. The arguments will look something like this:
>golf -t c:\GolfScoreTestHappyPath.txt c:\output\
The test will then validate that each entry (golfer) in the file is listed in descending order based on their final score.

Test Case ID **Test Case Title**
17 Validate correct list ordering in the TRR w/ a tied final score

Purpose: Verify requirement 5-G

Owner: Team Koala Bears

Expected Results: Golfer with name “colfer aame aaaaaaaa” is listed first followed by golfer with name “golfer name bbbbbbbb” followed by “bolfer name zaaaaaaa”

Test Data: ~/golf-data/PlayersTournamentScoreTied.txt

Test Tools: N/A

Dependencies: N/A

Initialization: N/A

Description: First, a valid input data file will be passed alongside specific CLI arguments in order to output a file containing Tournament Ranking data. The arguments will look something like this:

>golf -t c:\PlayersTournamentScoreTied.txt c:\output\

The test will then validate that the entries with identical final scores will be listed in alphabetical order by their last name.

Test Case ID	**Test Case Title**
18	Valid TRR filename
-	

Purpose: Verify requirement 5-H

Owner: Team Koala Bears

Expected Results: The output file is named trank.rep

Test Data: ~/golf-data/GolfScoreTestHappyPath.txt

Test Tools: N/A

Dependencies: N/A

Initialization: N/A

Description: First, a valid input data file will be passed alongside specific CLI arguments in order to output a file containing Tournament Ranking data. The arguments will look something like this:

>golf -t c:\GolfScoreTestHappyPath.txt c:\output\

The program will run, and the test will then validate that the output file is named trank.rep

Test Case ID	**Test Case Title**
19	Valid TRR output directory

Purpose: Verify requirement 5-I

Owner: Team Koala Bears

Expected Results: The TRR is outputted to c:\output\

Test Data: ~/golf-data/GolfScoreTestHappyPath.txt

Test Tools: N/A

Dependencies: N/A

Initialization: N/A

Description: First, a valid input data file will be passed alongside specific CLI arguments in order to output a file containing Tournament Ranking data. The arguments will look something like this:

>golf -t c:\GolfScoreTestHappyPath.txt c:\output\

The program will run, and the test will then validate that TRR exists in the c:\output\ directory

○ *2.2.6 Golfer Report (GR) Output*

Test Case ID	**Test Case Title**
20	Happy Path for GR

Purpose: Verify requirements 6-A, 6-B

Owner: Team Koala Bears

Expected Results: Golfer's name in GR will be contained in columns 0-19, their score for each course (1-5) will be in columns 20-24, their tournament score will be in column 25-27, and their tournament standing will be in column 28-29.

Test Data: ~/golf-data/GolfScoreTestHappyPath.txt

Test Tools: N/A

Dependencies: N/A

Initialization: N/A

Description: First, a valid input data file will be passed alongside specific CLI arguments in order to output a file containing Tournament Ranking data. The arguments will look something like this:

>golf -g c:\GolfScoreTestHappyPath.txt c:\output\

The test will then validate that file in the following order:

1. Assert that the golfers name is contained in columns 0-19
2. Assert that the golfers score for each course is contained in columns 20-24
3. Assert that the golfers tournament score is contained in column 25-27
4. Assert that the golfers tournament standing is contained in column 28-29

Test Case ID	**Test Case Title**
21	Validate correct list ordering in the GR

Purpose: Verify requirement 6-C

Owner: Team Koala Bears

Expected Results: The order of golfers in the TRR will be listed alphabetically.

Test Data: ~/golf-data/PlayersTournamentScoreTied.txt

Test Tools: N/A

Dependencies: N/A

Initialization: N/A

Description: First, a valid input data file will be passed alongside specific CLI arguments in order to output a file containing Tournament Ranking data. The arguments will look something like this:

>golf -g c:\GolfScoreTestHappyPath.txt c:\output\

The test will then validate that each entry (golfer) in the file is listed in alphabetical order.

Test Case ID	**Test Case Title**
22	Valid GR filename

Purpose: Verify requirement 6-D

Owner: Team Koala Bears

Expected Results: The output file is named golfer.rep

Test Data: ~/golf-data/GolfScoreTestHappyPath.txt

Test Tools: N/A

Dependencies: N/A

Initialization: N/A

Description: First, a valid input data file will be passed alongside specific CLI arguments in order to output a file containing Tournament Ranking data. The arguments will look something like this:

>golf -g c:\GolfScoreTestHappyPath.txt c:\output\

The program will run, and the test will then validate that the output file is named golfer.rep

Test Case ID **Test Case Title**
23 Valid GR output directory

Purpose: Verify requirement 6-E

Owner: Team Koala Bears

Expected Results: The GR is outputted to c:\output\

Test Data: ~/golf-data/GolfScoreTestHappyPath.txt

Test Tools: N/A

Dependencies: N/A

Initialization: N/A

Description: First, a valid input data file will be passed alongside specific CLI arguments in order to output a file containing Tournament Ranking data. The arguments will look something like this:

>golf -g c:\GolfScoreTestHappyPath.txt c:\output\

The program will run, and the test will then validate that GR exists in the c:\output\ directory

○ *2.2.7 Course Report (CR) Output*

Test Case ID **Test Case Title**
24 Happy Path for CR

Purpose: Verify requirements 7-A, 7-B, 7-C, 7-C, 7-D, 7-E

Owner: Team Koala Bears

Expected Results: Golfer's name in TRR will be contained in columns 0-19, their stroke count per course section per hole will be in columns 20-39, and their course score per course section will be in column 40-42.

Test Data: ~/golf-data/GolfScoreTestHappyPath.txt

Test Tools: N/A

Dependencies: N/A

Initialization: N/A

Description: First, a valid input data file will be passed alongside specific CLI arguments in order to output a file containing Tournament Ranking data. The arguments will look something like this:

>golf -c c:\GolfScoreTestHappyPath.txt c:\output\

The test will then validate that file in the following order:

1. Assert that the golfer's name in the CR will be contained in columns 0-19 per course section
2. Assert that the golfers stroke count per hole will be in columns 20-39 per course section
3. Assert that the golfers course score will be in column 40-42 per course section

Test Case ID **Test Case Title**
25 Validate correct list ordering in the CR

Purpose: Verify requirement 7-F

Owner: Team Koala Bears

Expected Results: The order of golfers in the TRR will be listed in descending order based on their course score per course section.

Test Data: ~/golf-data/GolfScoreTestHappyPath.txt

Test Tools: N/A

Dependencies: N/A

Initialization: N/A

Description: First, a valid input data file will be passed alongside specific CLI arguments in order to output a file containing Tournament Ranking data. The arguments will look something like this:

>golf -c c:\GolfScoreTestHappyPath.txt c:\output\

The test will then validate that each entry in the file is listed in descending order based on their course score per course section.

Test Case ID **Test Case Title**
26 Validate correct list ordering in the CR w/ a tied course score

Purpose: Verify requirement 7-G

Owner: Team Koala Bears

Expected Results: Golfer with the name "golfer name 20000001" is listed above the golfer with name "golfer name 20000002".

Test Data: ~/golf-data/PlayersTournamentScoreTied.txt

Test Tools: N/A

Dependencies: N/A

Initialization: N/A

Description: First, a valid input data file will be passed alongside specific CLI arguments in order to output a file containing Tournament Ranking data. The arguments will look something like this:
>golf -c c:\PlayersTournamentScoreTied.txt c:\output\
The test will then validate that the entries with identical course scores will be listed in alphabetical order by their last name per course section.

Test Case ID **Test Case Title**
27 Valid CR filename

Purpose: Verify requirement 7-H

Owner: Team Koala Bears

Expected Results: The output file is named course.rep

Test Data: ~/golf-data/GolfScoreTestHappyPath.txt

Test Tools: N/A

Dependencies: N/A

Initialization: N/A

Description: First, a valid input data file will be passed alongside specific CLI arguments in order to output a file containing Tournament Ranking data. The arguments will look something like this:
>golf -c c:\GolfScoreTestHappyPath.txt c:\output\
The program will run, and the test will then validate that the output file is named course.rep

Test Case ID **Test Case Title**
28 Valid CR output directory

Purpose: Verify requirement 7-I

Owner: Team Koala Bears

Expected Results: The CRis outputted to c:\output\

Test Data: ~/golf-data/GolfScoreTestHappyPath.txt

Test Tools: N/A

Dependencies: N/A

Initialization: N/A

Description: First, a valid input data file will be passed alongside specific CLI arguments in order to output a file containing Tournament Ranking data. The arguments will look something like this:

>golf -c c:\GolfScoreTestHappyPath.txt c:\output\
The program will run, and the test will then validate that CR exists in the c:\output\ directory

○ *2.2.8 Program Functionality*

Test Case ID	**Test Case Title**
29	

Purpose: Verify requirement 8-A, 8-B, 8-C, 8-D, 8-E, 8-F, 8-G

Owner: Team Koala Bears

Expected Results: golfer.rep file is generated with “golfer name 20000001” receives a final score of 0 for course id a
golfer.rep file is generated with “golfer name 20000002” receives a final score of 18 for course id a
golfer.rep file is generated with “golfer name 20000003” receives a final score of 36 for course id a
golfer.rep file is generated with “golfer name 20000004” receives a final score of 72 for course id a
golfer.rep file is generated with “golfer name 20000005” receives a final score of 108 for course id a

Test Data: ~/golf-data/ProgramFunctionalityScoreIsCorrectlyCalculated.txt

Test Tools: N/A

Dependencies: N/A

Initialization: N/A

Description: This test verifies that the golf score calculates the score for each hole correctly based on the par score and the stroke count.
- golfer 20000001 has a over par stroke count for each hole receives a score of 0 for each hole
- golfer 20000002 has a par stroke count for each hole receives a score of 1 for each hole
- golfer 20000003 has a 1 under par stroke count for each hole receives a score of 2 for each hole
- golfer 20000004 has a 2 under par stroke count for each hole receives a score of 4 for each hole
- golfer 20000005 has a 3 under par stroke count for each hole receives a score of 6 for each hole

Test Case ID	**Test Case Title**
30	Validate final tournament scores

Purpose: Verify requirement 8-H

Owner: Team Koala Bears

Expected Results: golfer.rep file is generated with “golfer name 20000001” receiving a tournament score of 108
golfer.rep file is generated with “golfer name 20000002” receives a a tournament score of 18
golfer.rep file is generated with “golfer name 20000003” receives a a tournament score of 54
golfer.rep file is generated with “golfer name 20000004” receives a a tournament score of 90
golfer.rep file is generated with “golfer name 20000005” receives a a tournament score of 180

Test Data: ~/golf-data/ProgramFunctionalityTournamentScoreIsCorrectlyCalculated.txt

Test Tools: N/A

Dependencies: N/A

Initialization: N/A

Description: This test verifies that the golf score calculates the score for each hole correctly based on the par score and the stroke count.
- golfer 20000001 has a score of 108
- golfer 20000002 has a score of 18
    - golfer 20000003 has a score of 54
    - golfer 20000004 has a score of 90
- golfer 20000005 has a score of 180

Test Case ID **Test Case Title**
31 Validate final tournament rankings

Purpose: Verify requirement 8-I

Owner: Team Koala Bears

Expected Results: golfer.rep file is generated with “golfer name 20000005” is ranked 1st in the tournament
golfer.rep file is generated with “golfer name 20000001” is ranked 2nd in the tournament
golfer.rep file is generated with “golfer name 20000004” is ranked 3rd in the tournament
golfer.rep file is generated with “golfer name 20000003” is ranked 4th in the tournament
golfer.rep file is generated with “golfer name 20000002” is ranked 5th in the tournament

Test Data: ~/golf-data/ProgramFunctionalityTournamentScoreIsCorrectlyCalculated.txt

Test Tools: N/A

Dependencies: N/A

Initialization: N/A

Description: This test verifies that the golf score calculates the score for each hole correctly based on the par score and the stroke count.
- golfer 20000002 has a final standing of 1
    - golfer 20000003 has a final standing of 2
    - golfer 20000004 has a final standing of 3
` - golfer 20000001 has a final standing of 4
    - golfer 20000005 has a final standing of 5

Test Case ID	**Test Case Title**
32	No valid option
Purpose:	Verify requirements 2-D
Owner:	Team Koala Bears
Expected Results:	Stating invalid option and then state the options
Test Data:	N/A
Test Tools:	N/A
Dependencies:	N/A
Initialization:	N/A
Description:	If any option other than the expected options are shown help information is displayed