**CP1401 SP1 2018 Assessment 2  
Implementing the Tropical Golf Game**

# Task

You are to implement a golf game program as outlined in the following description. Use what you have created for Assignment One, making any changes or improvements

**See the last section of this document for sample output showing how the game   
is played.**

# Program Features:

**Short description**:

* You are required to create a Python 3 program for a game of Tropical Golf. The player will be attempting to get the ball in the hole using the least amount of swings.
* The rules and gameplay for the program are as follows:
  + The game asks for the players name, and then welcomes the player and displays the menu
  + The main menu displays 3 options “Instructions”, “Play game and quit”
  + If the player chooses “I” the instructions are displayed
  + If the player chooses “P” the game is played:
    - The ball starts 230m away from the hole
    - The player chooses one of three clubs, each of which hits a different average distance
    - Every hit travels directly towards the hole in a straight line (this is a simple golf game, we’re only playing in 2D)
    - Once the ball is in the hole the player is told how many swings it took and whether they are over or under par
      * ‘Par’ is the estimate of how many swings a game should take for an average player. Par for this game is 5.
    - The player may use as many swings as is necessary to get the ball in the hole
  + After every menu choice OTHER than ‘Q’ the menu is displayed again.
  + Once the player enters “Q” the game thanks the player by name and says goodbye
* **Additional Functionality**
  + Your main function is to maintain a list of past scores
    - This list should be created empty before the menu is displayed
  + Once the game has been played the users score (total number of shots taken) will be returned, and appended to the list.
  + When the user quits you will display a list of past scores, using the example below as a guide for your formatting

If the list of scores was [5,8,2,4,7,5] then the output would be

Round 1 : 5 shots. On par.

Round 2 : 8 shots. 3 over par.

Round 3 : 2 shots. 3 under par.

Round 4 : 4 shots. 1 under par.

Round 5 : 7 shots. 2 over par.

Round 6 : 5 shots. On par.

**Detailed instructions:**

Ensure that your program has the following features:

1. Ask the player’s name and welcome them to the game using their name.
2. Show the game menu:

(I)nstructions

(P)lay golf

(Q)uit

The user will enter the first letter of each option to choose it.

1. If the user chooses (I) then the following message is to be displayed:|

This is a simple golf game in which each hole is 230m game away with par 5. You are able to choose from 3 clubs, the Driver, Iron or Putter. The Driver will hit around 100m, the Iron around 30m and the Putter around 10m. The putter is best used very close to the hole.

After the instructions are displayed the game menu will be presented again.

1. If the player chooses (Q) then the following message is to be displayed:

Farewell and thanks for playing <user name>.

1. After displaying the instructions, or playing a round of golf, the program is to return to the menu and loop until the user chooses to quit
2. If the user chooses (P) then a game of golf will begin as described below

**Playing the game:**

* The player starts 230m from the hole
* For each swing, the player chooses a club and then the program generates the distance hit for each shot, updating the distance to the hole accordingly. After each swing the distance the ball travelled, and the distance remaining is displayed along with the current score using something like:

Your shot went 103m. You are 127m from the hole, after 1 shot/s.

* The player has three clubs to choose from for each shot. Each club hits the ball a set average distance, but the actual distance randomly varies between 80% and 120% of the average. You will need to generate a random number between 80 and 120 to do this.
* The clubs and their average distances are:
  + Driver: 100m (actual distance will be a random number between 80 and 120)
  + Iron: 30m
  + Putter: 10m\*

**\*When the ball is inside 10m and the putter is used, the shot will be between 80% and 120% of the distance to the hole, not the club’s average distance. The minimum distance the putter can hit is 1m (no 0m hits). All distances are whole numbers.**

* The user will enter the first letter of a club to choose it (i.e. ‘**I**’ or ‘**i**’ for the Iron)
* If an invalid club is chosen, this is counted as an air swing (missed the ball) and the number of shots increases by one, but the ball doesn’t move.

Invalid club selection = air swing :(

Your shot went 0m. You are 230m from the hole, after 1 shot/s

* + Note the similarities to the output from a successful swing
* The ball cannot be a negative distance to the hole. Whether it is in front of or behind the hole, it is still a positive distance to the hole. Most programming languages (including Python) have an absolute value function that you can use to help with this.
* Play proceeds until the ball is in the hole (distance to the hole is zero), and then the program informs the user of their score.
* The players score is the number of shots taken to get the ball in the hole. The final output will display the number of shots taken and how this relates to par. If their score is less than 5 (par for this hole) is “under par”, equal to 5 is called “par”, and more than 5 is “over par”. See sample output for exact outputs.
  + Clunk... After 10 hits, the ball is in the hole! Disappointing. You are 5 over par.
  + Clunk... After 5 hits, the ball is in the hole! And that’s par.
  + Clunk... After 3 hits, the ball is in the hole! Congratulations, you are 2 under par.
* If the player has done a previous round (i.e. this is the second time the game is played), the game should then display the total score with an appropriate message, and then show the game menu again.

# Implementation:

You may show your assignment to your tutor during practical time to get comments or suggestions. It is important to note that you can only get help from staff in practical time after your prac work is finished.

You must submit a zipped file with the following:

* Python file - .py
* Brief reflection - .doc

**Python file:**You are to divide your solution into functions, following the principles shown in class. These may involve one for each menu option (other than Quit) as well as functions for parts of the program (e.g. calculating the current player balance could be implemented as a function).

**Brief reflection:**

You also need to write a word document (around 1 page in length) describing the changes you made to your program between Assessment 1 and 2. Quite often things that seem straight-forward on writing pseudocode end up being more complicated upon implementation.

Answer the following questions in your reflection, with one paragraph for each:

* One item I found challenging
* One item I changed between algorithm and implementation
* One extension that I would make to my program if I had more time
* An overall description of your experience coding this assignment

As this is a reflection you are welcome to use I in your reflection.

# General Principles:

In this assignment, you will be focusing on basic planning and implementation using selections, repetition, and functions. Lists may be used where they are feasible.

Use the techniques and patterns that you have learned and seen demonstrated in class.

* Coding standards:
  + Variables:
    - Wherever possible, variables should be set at the top of the function
    - Although Python does not use constants you may set variables to values that will remain the same for the life of the program. These should be named in all caps
      * You should be able to modify these variables to adjust the various values of items such as PAR, STARTING\_DISTANCE, etc. This is an important aspect of this assignment, so consider carefully how to use constants. Remember to use these variables everywhere you can.
      * You will need to decide the best position to place these variables.
    - You should AVOID the use of global variables unless they are needed in multiple places in the application
      * In other words, you should pass the values to any functions that need them and any constants only needed in one function should be ONLY in that function
  + Identifier Names
    - Variables should be named appropriately using either camelCase or under\_scores
    - No variables should be a single character such as “x” or “y” unless they are used in a loop as an index. Variable names should be meaningful, and describe their use and/or purpose
    - Any list variable should be named with plurals to show multiple data is held (e.g. prices versus price)
    - Function names should begin with a verb (they are active), for example getValue() is better than value(). Good verbs include – ‘get’, ‘display’, ‘calculate’, ‘check’ etc (not all these are good for this problem...)
  + Comments:
    - You will need to add at LEAST one comment for each function that you define including main
    - The comment should either be placed directly above or below the function header
    - The comment should describe: (a) the purpose of the function, (b) any inputs and (c) any returned values
    - Additional comments should be placed at any position where the functionality of the program may be unclear
    - All code MUST be properly indented
  + Error handling:
    - Note that menu choice and other character selection should handle upper and lower case letters.
      * Look into the upper() or lower() functions in Python
    - You also must make sure that any functions that get input from the user do error checking to ensure that the input is within the expected range (greater than zero, etc).
  + Output
    - Make your output display as close as possible to the given examples, especially formatting and layout.
    - Use Google to find out how to display tabs and other special characters using Python 3, or how to display things at a set width
  + Python Functions
    - Check out <https://docs.python.org/3/> and look up:
      * ‘randint’ to see how to generate a random number in Python
      * ‘abs’ to see how to use Pythons absolute value function
      * ‘upper’ or ‘lower’ to see case conversion functions

# Submission:

You must submit a zipped file with the following:

* Python file - .py
* Brief reflection - .doc

Please name the file like: **FirstnameLastnameA2.py**

e.g. if your name were Miles Davis, the filename would be MilesDavisA1.py

Submit your file by uploading it on LearnJCU under Assessment.

# Due:

Submit your assignment bythe date and time specified on LearnJCU.

Submissions received after this date will incur late penalties as described in the subject outline.

# Integrity:

The work you submit for this assignment must be your own. You are allowed to discuss the assignment with other students and get assistance from your peers, but you may not do anyone else’s work for them and you may not get anyone else to do any part of your work. A good rule to follow when offering assistance is that you can describe how code works but you should not be giving code to others. Programs that are detected to be too similar to another student’s work will be dealt with promptly according to University procedures for handling plagiarism.

If you require assistance with the assignment, please ask **general** questions on the Slack site for it@jcu (see subject outline for details), or get **specific** assistance with your own work by talking with your lecturer or tutor.

|  |
| --- |
| **NB: A quick note about planning and seeking assistance:**  You should plan to start your assessment no less than two - three weeks before the due date. As you become more proficient with coding this length of time will shorten, but in the beginning stages you will need time to become accustomed to “thinking like a programmer”.  Planning ahead will allow you time to seek assistance with parts that may be confusing to you. You will not know what these items are until you start to complete the assessment. |

# Marking Scheme:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Exemplary (9 – 10)** | **Good (7 – 8)** | **Satisfactory (5 – 6)** | **Limited (2 – 4)** | **Poor (0 – 1)** |
| **Correct use of Python syntax** | No syntax errors that prevent the program running as required. | Exhibits aspects of exemplary (left) and satisfactory (right) | Any errors in code syntax are small (mistyped names, or functions). | Exhibits aspects of satisfactory (left) and poor (right) | Code submitted does not run, and required significant work to be runnable |
| **Correct use of Python comments** | All comments are correctly inserted, clear, and appropriate | Most comments are correctly inserted and clear, but some comments are not. | No comments, or all comments are unclear or incorrect. |
| **Effective and correct use of variables in the implemented solution** | All variables are correctly used. All variable names are clear and meaningful  No inappropriate use of global or constant variables. | Most variables are correctly created and used  Most names are clear and appropriate | All variables are inconsistently used, and/or poorly named |
| **Effective and correct use of functions in the implemented solution** | Functions are all appropriate and correctly used  All names are valid and meaningful  No unnecessary inputs or outputs  All necessary inputs and outputs are correct | Most functions are appropriate and correctly used  Most names are valid and meaningful  Almost no unnecessary inputs or outputs  All necessary inputs and outputs are correct | All functions are inappropriate or incorrectly used  No names are valid or meaningful  Many unnecessary inputs or outputs  Some necessary inputs and outputs are incorrect |
| **Effective and correct use of branches in the implemented solution.** | The branches identified are appropriate and correctly used  All branches are efficient and well designed | Most branches identified are appropriate and correctly used  Most branches are efficient and well designed | Almost all branches identified are inappropriate or incorrectly used, or there is almost no use of branches  All branches are inefficient or poorly designed |
| **Effective and correct use of loops in the implemented solution.** | All loops used are appropriate and correctly used  All loops exit when appropriate | Most loops used are appropriate and correctly used  Most loops exit when appropriate | Very few loops used are appropriate and correctly used, or almost no loops are used  No (or very few) loops exit when appropriate |
| **Thoughtful reflection on the implementation process** | Reflection answers all the questions, and shows evidence of thoughtful responses. Shows improvement in student's ability to think systematically. | Reflection answers most/all of the questions, but it not as considered as exemplary. Shows some improvement in systematic thinking. | Shows very little sign of reflection (or missing reflection document entirely). Show little improvement in systematic thinking or planning. |

Example of game play. Game output is in **Bold**.

**What is your name?**

Bob

**Welcome Bob.**

**Let's play golf, CP1401 style!**

**(I)nstructions**

**(P)lay round**

**(Q)uit**

U

**Invalid menu choice.**

**Let's play golf, CP1401 style!**

**Golf!**

**(I)nstructions**

**(P)lay round**

**(Q)uit**

I

**This is a simple golf game in which each hole is 230m game away with par 5. You are able to choose from 3 clubs, the Driver, Iron or Putter. The Driver will hit around 100m, the Iron around 30m and the Putter around 10m. The putter is best used very close to the hole.**

**For each shot, you may use a driver, iron or a putter – each shot will vary in distance.**

**The average distance each club can hit is:**

**Driver = 100m**

**Iron = 30m**

**Putter = 10m**

**Golf!**

**(I)nstructions**

**(P)lay round**

**(Q)uit**

P

**This hole is a 230m par 5.**

**You are 230m from the hole, after 0 shot /s.**

**Club selection: press D for driver, I for Iron, P for Putter.**

**Choose club:** d

**Your shot went 103m**.

**You are 127m from the hole, after 1 shot/s**

**Club selection: press D for driver, I for Iron, P for Putter.**

**Choose club:** d

**Your shot went 95m**

**You are 32m from the hole, after 2 shot/s**

**Club selection: press D for driver, I for Iron, P for Putter.**

**Choose club:** i

**Your shot went 32m.**

**Clunk…After 3 hits your ball is in the hole!**

**Congratulations. You are 2 under par for this hole.**

**Your overall score is 3 and you are 2 under par after 1 hole.**

**Golf!**

**(I)nstructions**

**(P)lay round**

**(Q)uit**

P

**This hole is a 230m par 5.**

**You are 230m from the hole, after 0 shot /s.**

**Club selection: press D for driver, I for Iron, P for Putter.**

**Choose club:** a

**Invalid club selection – air swing :(**

**Your shot went 0m**.

**You are 230m from the hole, after 1 shot/s**

**Club selection: press D for driver, I for Iron, P for Putter.**

**Choose club:** i

**Your shot went 30m**

**You are 200m from the hole, after 2 shot/s**

**Club selection: press D for driver, I for Iron, P for Putter.**

**Choose club:** p

**Your shot went 12m.**

**You are 188m from the hole, after 3 shot/s**

**Club selection: press D for driver, I for Iron, P for Putter.**

**Choose club:** d

**Your shot went 104m.**

**You are 84m from the hole, after 4 shot/s**

**Club selection: press D for driver, I for Iron, P for Putter.**

**Choose club:** d

**Your shot went 90m.**

**You are 6m from the hole, after 5 shot/s**

**Club selection: press D for driver, I for Iron, P for Putter.**

**Choose club:** i

**Your shot went 26m.**

**You are 20m from the hole, after 6 shot/s**

**Club selection: press D for driver, I for Iron, P for Putter.**

**Choose club:** p

**Your shot went 9m.**

**You are 11m from the hole, after 8 shot/s**

**Club selection: press D for driver, I for Iron, P for Putter.**

**Choose club:** p

**Your shot went 9m.**

**You are 2m from the hole, after 6 shot/s**

**Club selection: press D for driver, I for Iron, P for Putter.**

**Choose club:** p

**Your shot went 1m.**

**You are 1m from the hole, after 8 shot/s**

**Club selection: press D for driver, I for Iron, P for Putter.**

**Your shot went 1m.**

**Clunk…After 10 hits your ball is in the hole!**

**Disappointing. You are 5 over par for this hole.**

**Your overall score is 18 and you are 3 over par after 2 holes.**

**Golf!**

**(I)nstructions**

**(P)lay round**

**(Q)uit**

P

**This hole is a 230m par 5.**

**You are 230m from the hole, after 0 shot /s.**

**Club selection: press D for driver, I for Iron, P for Putter.**

**Choose club:** d

**Your shot went 92m**.

**You are 138m from the hole, after 1 shot/s**

**Club selection: press D for driver, I for Iron, P for Putter.**

**Choose club:** i

**Your shot went 25m**

**You are 113m from the hole, after 2 shot/s**

**Club selection: press D for driver, I for Iron, P for Putter.**

**Choose club:** d

**Your shot went 106m.**

**You are 7m from the hole, after 3 shot/s**

**Club selection: press D for driver, I for Iron, P for Putter.**

**Choose club:** p

**Your shot went 6m.**

**You are 1m from the hole, after 4 shot/s**

**Club selection: press D for driver, I for Iron, P for Putter.**

**Choose club:** p

**Your shot went 1m.**

**Clunk…After 5 hits your ball is in the hole!**

**And that's par.**

**Your overall score is 18 and you are 3 over par after 3 holes.**

**Golf!**

**(I)nstructions**

**(P)lay round**

**(Q)uit**

Q

**Thanks for playing Bob.**

**Round 1 : 3 shots. 2 under par.**

**Round 2 : 10 shots. 5 over par.**

**Round 3 : 5 shots. On par.**