FIT1045/FIT1053 Algorithmic Problem Solving – Assignment 1 (5%).

Due: 23:55:00, Friday 13th April, 2018.

Submission Procedure

1. In the file that you submit, **DO NOT** put your name, student ID or any other information that can reveal your identity. Peer marking will be done following a *double-blind* policy where your identity and markers' identities are not disclosed to anyone else except the teaching team.
2. Submit your file named `scrabble1.py` containing your solution to Moodle. Make sure that your assignment is not in "Draft" mode. You need to click "Submit" to successfully submit the assignment.

Important Notes:

1. Please ensure that you have read and understood the university's policies on plagiarism and collusion available at `http://www.monash.edu.au/students/policies/academic-integrity.html`. You will be required to agree to these policies when you submit your assignment. The assignments will be checked for plagiarism using an advanced plagiarism detector and the students will be interviewed by tutors to demonstrate the understanding of their code. Last year, many students were detected by the plagiarism detector and almost all got zero mark for the assignment and, as a result, failed the unit. "Helping" others is **NOT** okay. Please do not share your solutions with others. If someone asks you for help, ask them to visit us during consultation hours for help.
2. Your program will be checked against a number of test cases. Do not forget to include comments in your code explaining your algorithm. If your implementations have bugs, you may still get some marks based on how close your algorithm is to the correct algorithm.
3. You must implement your own function to *search* an item in a list. Do not use built-in Python functions such as "`if item in aList`" to search for an item in the list.
4. Your submission will be assessed by your lab demonstrator as well as two randomly chosen peers. The peers' assessment will be checked for consistency by your demonstrator.

Marks: This assignment has a total of 100 marks and contributes to 5% of your final mark. Late submission will have 10% off the total assignment marks per day (including weekends) deducted from your assignment mark, i.e., 10 marks per day. So, if you are 1 day late, you will lose 10 marks. Assignments submitted 7 days after the due date will normally not be accepted.

Marking Guide:

Task 1: 70 marks

(a) Code readability (Non-trivial comments where necessary and meaningful variable names) – 5 marks

(b) Code decomposition – 5 marks

(c) Properly reading the data from `dictionary.txt` – 5 marks

(d) Correctly checking that the word exists in the dictionary – 10 marks

(e) Correctly checking that the word contains only English letters – 15 marks

(f) Correctly checking that the word can be made using the tiles – 15 marks

(g) Correctly checking if the word is valid or not using the above rules – 5 marks

(h) Repeatedly asking the user to enter a word until they enter a valid word or `***` and then printing a message accordingly – 10 marks

Task 2: 30 marks

(a) Code readability (Non-trivial comments where necessary and meaningful variable names) – 5 marks

(b) Code decomposition – 5 marks

(c) Correctly computing and printing the word with the highest score of the word (if such a word exists) – 20 marks

Background

You will be building a single player Scrabble game this semester. In Assignment 1, you will need to complete some basic tasks that you will be using in Assignment 2 to complete the Scrabble game.

You have been provided with four files on Moodle named `dictionary.txt`, `scores.txt`, `scrabble1.py` and `tiles.txt`. You will need these files to complete this assignment.

The file `dictionary.txt` contains a list of words found in an online dictionary. This dictionary will be used to determine whether the word entered by the player is a valid English word or not, i.e., if the entered word exists in the dictionary, it is considered a valid English word otherwise not.

Each English letter has a score assigned to it. The scores for all the letters can be found in the file `scores.txt`. The contents of `scores.txt` are shown below.

```
A 1
B 3
C 5
D 3
E 1
F 5
G 4
H 3
I 1
J 10
K 8
L 3
M 5
N 3
O 2
P 5
Q 20
R 3
S 3
T 2
U 1
V 10
W 12
X 16
Y 8
Z 20
```

The second column in the above file shows the score for the letter in the first column, e.g., A has score 1, B has score 3 and Z has score 20. The score of a word is the sum of the scores of all the letters in the word. For example, the score of word `SWEET` is $3 + 12 + 1 + 1 + 2 = 19$.

You will implement your solution in the file `scrabble1.py` which already contains some code to get you started on the assignment. When you run the program `scrabble1.py`, it prints 7 letters (called tiles in Scrabble) and score for each tile is printed below it. The program uses the file `tiles.txt` to fetch the tiles and print it to the user. The program asks you whether you want the tiles to be presented in a random order or a fixed order. If you enter `N`, you will be presented first 7 tiles from `tiles.txt` (i.e., you get the same tiles every time you run the program and enter `N`). You must enter `Y` if you want random tiles, i.e., in a different order each time you run the program.

Below are two examples of the output generated by the provided code in `scrabble1.py`.

```
Do you want to use random tiles (enter Y or N): N

Tiles : B  S  N  O  E  U  T
Scores: 3  3  3  2  1  1  2
```

```
Do you want to use random tiles (enter Y or N): Y

Tiles : O  X  B  B  S  Q  I
Scores: 2  16 3  3  3  20 1
```

You may use `Y` to test your program on different set of tiles each time the program is run. You may use `N` if you need the same tiles every time (e.g., for debugging your program). You must ensure that all the files (`scrabble1.py`, `scores.txt`, `dictionary.txt` and `tiles.txt`) are present in the same folder otherwise the program `scrabble1.py` may not run properly.

You are required to complete the following tasks in this assignment.

Task 1: Check validity of the entered word 70 Marks

After the program prints the 7 tiles and their scores, you will need to ask the user to enter a word. Your program then should check if the word entered by the user is valid or not. A word must satisfy all of the following to be considered a valid word.

1. First, your program must ensure that the word must consist of only English letters. E.g., the word "SWE3T" is invalid because it contains a number. Similarly, "Hello World" is also invalid because it contains a whitespace. If the word does not consist of only the English letters, your program must print a message "Only use English letters.".

2. If the above rule is satisfied, your program must ensure that the the word exists in the dictionary `dictionary.txt`. Note that the user may enter the word in lowercase or uppercase letters or a combination of the two. Your program will need to convert the word into uppercase letters. If the word does not exist in the dictionary, you should print a message "I have never heard of this word". Note that the dictionary does not contain **all** English words. For example, the word `LATE` is not in the dictionary and, therefore, it is not considered a valid word in this game.

3. If the word satisfies both of the above rules, your program must check whether the word can be made using the tiles that the user have. E.g., if the tiles are `T Y S E U W I` then the word `SWEET` is invalid because the tiles contain only one `E`. If the word cannot be made using the tiles, you should print a message "This word cannot be made using the tiles".

If the entered word satisfies all of the above rules, it is considered a valid word. Note that the user input should be treated as case-insensitive, e.g., `SWEET` and `SwEeT` are to be considered the same. Your program should repeatedly ask the user to enter a word until the user either enters a valid word or enters `***` indicating that the user has given up and does not want to enter a word. If the user enters a valid word, you should print "Cool, this is a valid word". If the user enters `***`, you should print "Better luck next time!!!".

Task 2: Finding the word with highest score 30 Marks

After the user has entered a valid word, you must compute and print the score of the word (if the user had entered `***`, you must not display the score). Finally, you must find a valid word with the highest score and display this to the user along with its score. Recall that the score of a word is the total score of all the letters in the word where score of each letter is given in `scores.txt`. If there is no valid word that can be made using the tiles, you must print "No word can be made using the tiles".

Output Samples

Below are some samples of the output your program should generate.

```
Do you want to use random tiles (enter Y or N): N

Tiles : B  S  N  O  E  U  T
Scores: 3  3  3  2  1  1  2

Enter a word: N3T
Only use English letters!!!

Enter a word: NetK
I have never heard of this word.

Enter a word: SEEN
This word cannot be made using your tiles.

Enter a word: SeNT
Cool, this is a valid word.
Score for the word SENT is: 9

The word BONUS is the word with highest score. Its score is 12
```

```
Do you want to use random tiles (enter Y or N): N

Tiles : B  S  N  O  E  U  T
Scores: 3  3  3  2  1  1  2

Enter a word: NEAT
```

```
This word cannot be made using your tiles.

Enter a word: ***
Better luck next time!!!

The word BONUS is the word with highest score. Its score is 12
```

```
Do you want to use random tiles (enter Y or N): Y

Tiles : R  T  C  E  S  B  R
Scores: 3  2  5  1  3  3  3

Enter a word: RATING
I have never heard of this word.

Enter a word: RATE
This word cannot be made using your tiles.

Enter a word: BETS
I have never heard of this word.

Enter a word: BEST
Cool, this is a valid word.
Score for the word BEST is: 9

The word CREST is the word with highest score. Its score is 14
```

```
Do you want to use random tiles (enter Y or N): Y

Tiles : W  E  I  I  C  A  N
Scores: 12 1  1  1  5  1  3

Enter a word: I CAN
Only use English letters!!!

Enter a word: CANW
I have never heard of this word.

Enter a word: CANDY
This word cannot be made using your tiles.

Enter a word: CAN
Cool, this is a valid word.
Score for the word CAN is: 9

The word WINCE is the word with highest score. Its score is 22
```

```
Do you want to use random tiles (enter Y or N): Y

Tiles : H  K  R  C  C  B  Z
Scores: 3  8  3  5  5  3  20

Enter a word: ***
Better luck next time!!!

No word can be made using these tiles
```