SCHOOL OF COMPUTING
COMMON INFOCOMM TECHNOLOGY PROGRAMME
DIPLOMA IN APPLIED AI & ANALYTICS
DIPLOMA IN INFOCOMM SECURITY MANAGEMENT
DIPLOMA IN INFORMATION TECHNOLOGY

ST0502 Fundamentals of Programming

2020/2021 SEMESTER-1
PRACTICAL ASSIGNMENT (CA2)

Instructions and Guidelines:

1. The assignment should be submitted by **10 August 2020 (Mon), 9:00am**. You are required to submit a softcopy of source codes in BlackBoard. Please indicate your Name (as in SAS2), Class and Admission Number in your submission.

2. This is an individual assignment.

3. The assignment will account for **30%** of this module.

4. You must use JavaScript to develop the application.

5. The interview will be conducted during the lessons in week 17-18 (from **10 August to 21 August 2020)**. You are expected to explain the program logic and modify the program during the interview. **If you are absent from the interview without a valid Leave of Absence approved by SP, you will be awarded zero mark for the assignment.**

6. **No marks will be awarded**, if the work is copied or you have allowed others to copy your work. This is a very serious offence of plagiarism committed by all involved. Please refer the clause in RED below regarding plagiarism.

7. **50% of marks** to be deducted for submission of assignment within **ONE** calendar day after the deadline. **No marks shall** be awarded for assignments submitted **more than one day** after the deadline.

Warning: Plagiarism means passing off as one's own the ideas, works, writings, etc., which belong to another person. In accordance with this definition, you are committing plagiarism if you copy the work of another person and turning it in as your own, even if you would have the permission of that person. Plagiarism is a serious offence and disciplinary action will be taken against you. If you are guilty of plagiarism, you may fail all modules in the semester, or even be liable for expulsion.

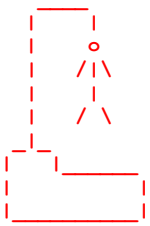
Quiz Application

1. Objective

The objective of this assignment is for the student to create a Hangman Game.

2. Overview

Hangman is a classic word game in which a computer randomly selects a word that the player must guess by supplying each of its letter: for each incorrect guess, a part of a stick figure of a hanged man is drawn. The player loses when the complete hanged man is drawn as illustrated below:



3. Sample Output

The screenshots below illustrate the application when executed. You are encouraged to design your own text-based UI.

```
-= Welcome to Hangman =-

Please enter your name: Jane

Word 1 / 10

_ _ _ _ _ _ _ _ _

A B C D E F G H I J K L M
N O P Q R S T U V W X Y Z

Jane's guess (Enter 9 for lifelines or 0 to pass): E

Sorry. E is not a part of the word.

  _|_
 |   |_____
 |         |
 |_________|

_ _ _ _ _ _ _ _ _

A B C D   F G H I J K L M
N O P Q R S T U V W X Y Z

Jane's guess (Enter 9 for lifelines or 0 to pass): O
```

```
Good job! O is one of the letters!

_ O _ _ _ O _ _ _

A B C D   F G H I J K L M
N   P Q R S T U V W X Y Z

Jane's guess (Enter 9 for lifelines or 0 to pass): I

Sorry. I is not a part of the word.


   ____
  |    |
  |
  |
  |
  |
 _|_
|   |_____
|         |
|_________|



_ O _ _ _ O _ _ _

A B C D E F G H   J K L M
N   P Q R S T U V W X Y Z

Jane's guess (Enter 9 for lifelines or 0 to pass): A

Good job! A is one of the letters!

_ O _ _ _ O A _ _

  B C D   F G H   J K L M
N   P Q R S T U V W X Y Z
```

4. Basic Requirements

- There must be at least 40 words in a pool.
- All words must be stored in an array.
- The program randomly selects 10 words from the pool for each gameplay.
- The game is over either when the user wins by correctly guessing all 10 words or loses when all parts of the hangman is drawn.
- There must be at least 3 lifelines for each gameplay. They are (1) show all vowels, (2) show definition of the word, and (3) allow user to score and move on to the next word. Note that player will not be allowed to use lifeline (1) if all vowels are already shown and player is given only 1 of each lifeline per gameplay.
- The program must be able to track the scores of each gameplay. The player scores when he/she guesses a word correctly.

- When the game is over, the result screen that shows the player's score must be displayed. A congratulation message should be displayed if the player manages to guess all words correctly.

5. Basic Program Requirements:

You are to develop the application using the object oriented concept learnt in the module.

Your application should consist of at least 2 of these classes i.e.

- ***Word*** class to define each word in the game (along with other attribute(s) such as its definition).
- ***WordCollection*** class to represent a collection (pool) of all words. All required functions, such as a function to randomly select words from the pool for each gameplay, a function to check if the word is guessed correctly by player, a function to keep track of the score, etc. should also be included here.
- Feel free to add any other classes you feel are needed in your program.

6. Advanced Features

Marks will be awarded for advanced features, such as, but not limited to the following:

- All words are stored in an external file (e.g. CSV file or TXT file) or database.
- There are categories of words for player to choose at the beginning of each game. For example, IT Related, Geography, Biology, etc. The game then randomly picks the words from the selected category from the word pool.
- The player can choose to display statistics such as his/her average score, highest score or lowest score, elapse time, etc.
- The program is able to keep track of player's scores (the scores and player info must be stored in an external file or database).
- Any other features advanced features ie no in basic requirements above.

Please keep in mind that all these are advanced features. The main bulk of marks are allocated to the completion of a workable program that meets the minimum requirements.

You should try to fulfil the minimum requirements before you attempt the advanced features.

7. Assessment Guidelines

Marking criteria will be based on student's ability to demonstrate program requirements and explanation during Q&A interview in the following areas :

Assessement Criteria	Maximum marks allocated
Program functionalities - Execute basic requirements mentioned above correctly	35
Program design: - Correct and efficient usage of classes and programming constructs - Appropriate method decomposition - Propoer use of Arrays and Functions - Appropriate validations - Code efficiency Program readability: - Meaningful identifiers - Meaningful comments and indentation in source documentation	45
Innovation and creativity or any advanced features	20
Total	100

-- End --