Swinburne University of Technology Sarawak
COS10009
Introduction to Programming
Semester 2, 2018

Custom Program

Due Date: 5 PM on Friday of week 12 (30 November 2018).

Submission Guidelines: Your program and report must be submitted as part of your portfolio in an A4 paper envelope with a facing sheet attached to the front of envelope. Late penalties will apply as described in the unit outline.

Deliverables: (X means required)

1	Facing sheet with your signature	X
2	All source files, data files, extra libraries, project file, and the executable file. The source C or project PRJ files (if any) ***must*** be named as *studentID_surname.C* or *.PRJ*	X
3	Printed design report, including user manual, description of program design and flowchart.	X
4	Screen capture of the program output. (optional)	-
5	Print-out of the source code.	X

In the design report,** write a short description of program design and brief user manual. Also, submit a **Flowchart** showing the logic in the "main(void)" function.*

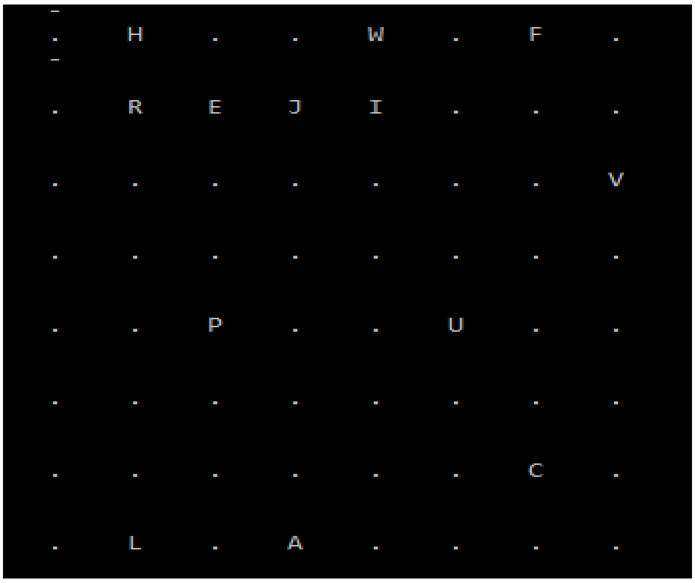
Topic: Travelling Salesman Problem

Background:
The traveling salesman problem (TSP) is an algorithmic problem tasked with finding the shortest route between a set of points and locations that must be visited. The salesman's goal is to keep the distance travelled as low as possible. Focused on optimization, TSP is often used in computer science to find the most efficient route for data to travel between various nodes. Applications include identifying network or hardware optimization methods. TSP has been studied for decades and several solutions have been theorized. The simplest solution is to try all possibilities, but this is also the most time consuming and expensive method. Many solutions use heuristics, which provides probability outcomes. However, the results are approximate and not always optimal. Hence, no general method of solution is known, and the problem is known as NP-hard.

Video:
https://www.youtube.com/watch?v=XBTiQ6IvgmM

https://www.youtube.com/watch?v=7B8Sx_nAxLk

Specification & Requirement:



```
_
.   H   .   .   W   .   F   .
_

.   R   E   J   I   .   .   .

.   .   .   .   .   .   .   V

.   .   .   .   .   .   .   .

.   .   P   .   .   U   .   .

.   .   .   .   .   .   .   .

.   .   .   .   .   .   C   .

.   L   .   A   .   .   .   .
```

- Develop a game with C that generates 8 by 8 grid (2-dimensional array).

- You are expected to design your own mechanism of generating the alphabets randomly at the beginning of the game. It should be documented in your design report to elaborate the details of your mechanism.

- You are allowed to design your own rules of playing the game, such as the size of the board, methods to position the alphabets, routing restriction, randomize starting point, provide hints or suggestions to players, scoring system, winning condition etc.

- Each element of array contains the following ***suggested*** attributes :
    - a variable of character
    - and any other attributes you think appropriate to your program

  Thus, the element can be described as "struct" of array in the C programming language.

- The record of previous players is stored in a plain text data file. The information of all the previous players should be displayed at the beginning of the game. User / player will also be prompted to key in his/her information, and that information will then be stored into the same text file.

- Input to the system is through a DOS command window.

- This program must first prompt the user (in the command window) for the next action to be carried out. (Do provide appropriate menu of actions). The user should be able to terminate the program at any time.

- Output should be printed on DOS window to show the latest layout of the board.

- The program codes MUST be in functions. DO NOT write all the code in the "main()" function.

- Must NOT use global variable and goto in the program.

- Graphical window interface can be used as an alternative (optional)

- This C program is recommended to be written using Quincy 2005 / CodeBlocks. Build options must include strict ANSI/ISO compliance, C99 support.

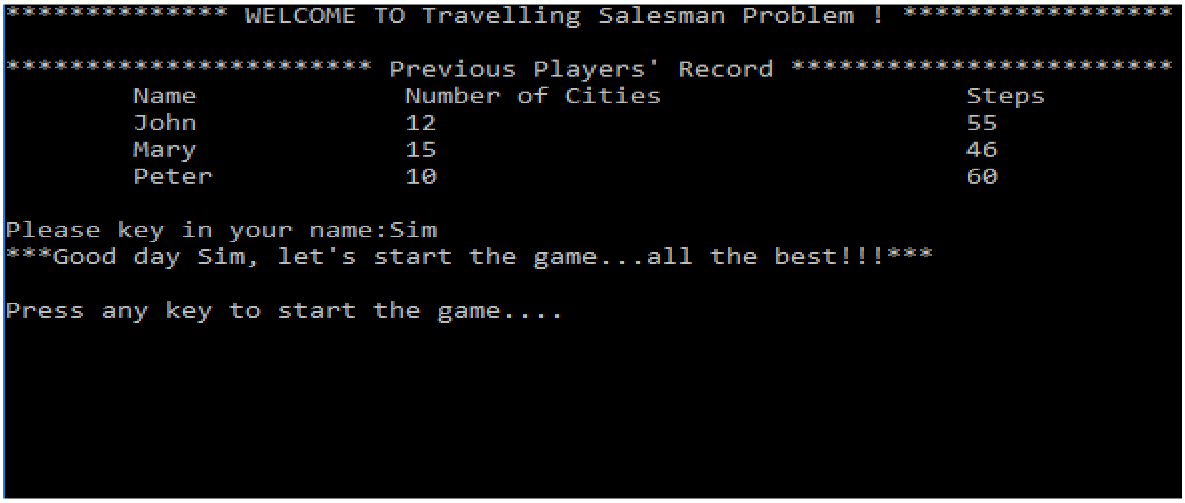
Procedure of Program Development

Use an iterative process (spiral or prototype SDLC) to design and implement a solution to your program. That means

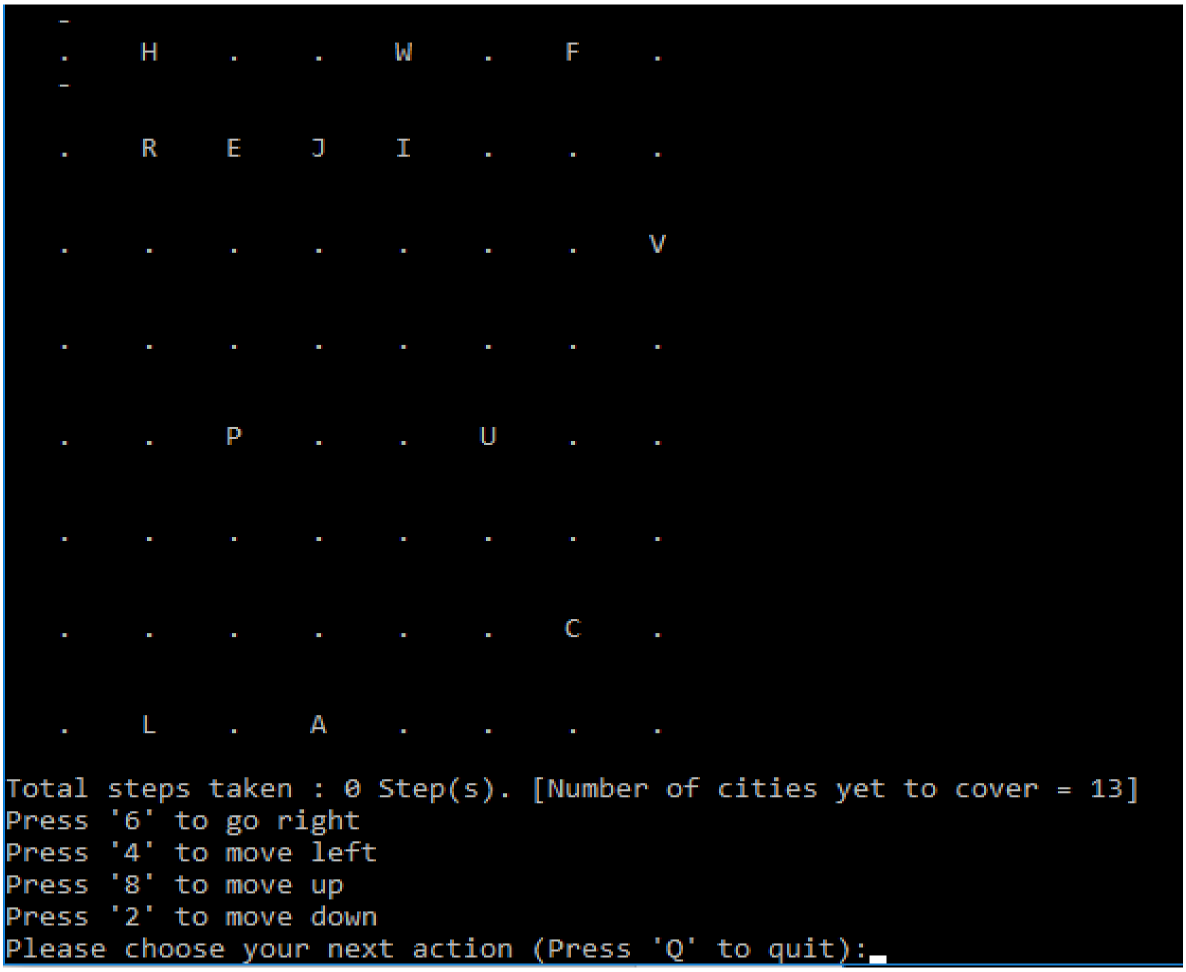
1. write a small program to do a little bit of the problem
    a. compile and run the small program
2. next, add a function or two
    a. compile and run the improved program
3. repeat step 2 until you are finished

(Kindly refer the last page of this document for the assessment details)

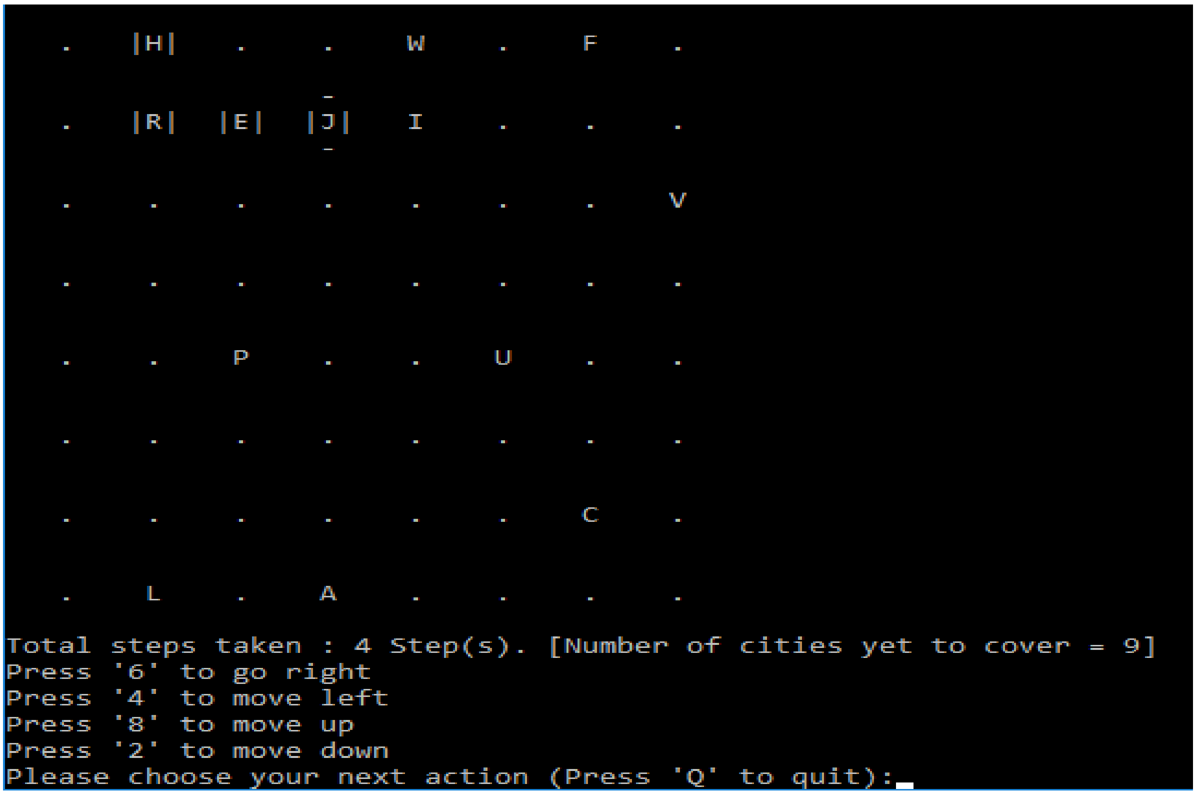
Snapshots:



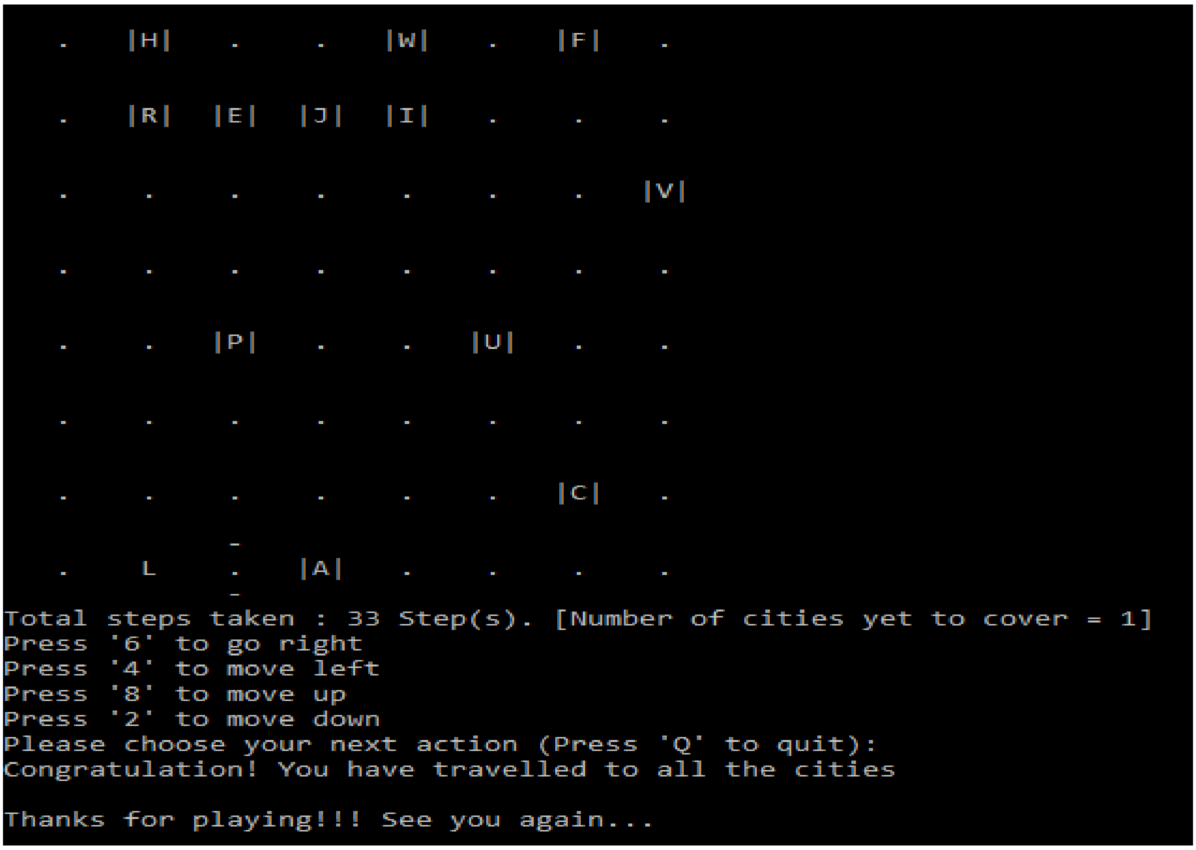
```
*************** WELCOME TO Travelling Salesman Problem ! ******************

*********************** Previous Players' Record ************************
        Name              Number of Cities                   Steps
        John              12                                 55
        Mary              15                                 46
        Peter             10                                 60

Please key in your name:Sim
***Good day Sim, let's start the game...all the best!!!***

Press any key to start the game....
```



```
   -
   .    H    .    .    W    .    F    .
   -

   .    R    E    J    I    .    .    .


   .    .    .    .    .    .    .    V


   .    .    .    .    .    .    .    .


   .    .    P    .    .    U    .    .


   .    .    .    .    .    .    .    .


   .    .    .    .    .    .    C    .


   .    L    .    A    .    .    .    .
Total steps taken : 0 Step(s). [Number of cities yet to cover = 13]
Press '6' to go right
Press '4' to move left
Press '8' to move up
Press '2' to move down
Please choose your next action (Press 'Q' to quit):
```



```
   .   |H|   .    .    W    .    F    .

                  -
   .   |R|  |E|  |J|   I    .    .    .
                  -

   .    .    .    .    .    .    .    V


   .    .    .    .    .    .    .    .


   .    .    P    .    .    U    .    .


   .    .    .    .    .    .    .    .


   .    .    .    .    .    .    C    .


   .    L    .    A    .    .    .    .
Total steps taken : 4 Step(s). [Number of cities yet to cover = 9]
Press '6' to go right
Press '4' to move left
Press '8' to move up
Press '2' to move down
Please choose your next action (Press 'Q' to quit):_
```



```
   .   |H|   .    .   |W|   .   |F|   .


   .   |R|  |E|  |J|  |I|   .    .    .


   .    .    .    .    .    .    .   |V|


   .    .    .    .    .    .    .    .


   .    .   |P|   .    .   |U|   .    .


   .    .    .    .    .    .    .    .


   .    .    .    .    .    .   |C|   .

             -
   .    L    .   |A|   .    .    .    .
             -
Total steps taken : 33 Step(s). [Number of cities yet to cover = 1]
Press '6' to go right
Press '4' to move left
Press '8' to move up
Press '2' to move down
Please choose your next action (Press 'Q' to quit):
Congratulation! You have travelled to all the cities

Thanks for playing!!! See you again...
```

COS10009
Semester 2, 2018
Marking Scheme

Student ID: ________________________ **Student Name:** ____________________

Assessment of this custom program will be part of the portfolio interview. Each interview will last about 10-15 minutes.

Pass			Credit			Distinction			High Distinction			
50	55	59	60	65	69	70	75	79	80	85	90	100
▪ Design report consists of required components ▪ Flowchart is correct, corresponds to logic of main() function ▪ Use functions and parameters ▪ Adequate comments to explain the code ▪ C code compiles and links, executable runs, does not crash			In addition to including the criteria required for Pass, the custom program must: ▪ Use functions, parameters, pointers and struct, ▪ Meet most of the requirements ▪ Read and write external text/ data file ▪ Program handles errors gracefully, allowing the user to continue.			In addition to including the criteria required for Credit, the design report must: ▪ Have good description of core program functionality and how it works The custom program must: ▪ Meet all the requirements ▪ Code written with proper indentation that helps document program structure ▪ Bugs free			In addition to including the criteria required for Distinction, the software solution must have: ▪ Code with extensively comments that provide meaningful insights into the code being documented ▪ Additional feature other than the specified requirements. ▪ The additional feature must be thoroughly documented in the design report ▪ The design report must demonstrate good communication skills, and present a well thought out program design			

Evidence of plagiarism: