

**Programming 1 (PRG1)**

Year 1 (2019/20), Semester 1

**SCHOOL OF INFOCOMM TECHNOLOGY**

Diploma in Financial Informatics

Diploma in Cyber Security & Forensics

Diploma in Information Technology

Diploma in Common ICT Programme

# ASSIGNMENT

**Due on 5 August 2019 (Monday), 8.30 am**

**Weightage:** 30% of Module

**Individual/Team/Both:** Individual

**Format:** Programming and Presentation

Basic Requirement (70%)

Additional Requirement (30%)

**Penalty for late submission:**

* + 10% per day from the due date.
  + NO submission shall be entertained after 7 calendar days of the due date.

There are a total of 12 pages (including this page) in this handout.

|  |
| --- |
| ***WARNING***  ***If a student is found to have submitted work not done by him/her, he/she will not be awarded any marks for this assignment. Disciplinary action will also be taken.***  ***Similar action will be taken for the student who allows other student(s) to copy his/her work.*** |

**1. OBJECTIVE**

This assignment assesses the student’s ability to apply relevant programming concepts to develop a simple application using Python programming language.

**2. BACKGROUND**

Tom wants to develop a maze game*.*

**3. SCOPE**

A simple maze is to be developed for users to play. You are assigned to develop a simple Python program to demonstrate the features of the game.

A data file containing an initial maze diagram is shown in Figure 1 below, where X represents the walls of the maze, O represents the open paths in the maze, A represents the start position of the game and B represents the exit/end position in the maze. You are to read the data from the data file, process it and display the information accordingly.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | XXXXXXXX | | XOOOXOAX | | XOXOXOXX | | XOXOXOOX | | XOXOXXOX | | XOXOXOOX | | XOXOOOXX | | XBXXXXXX | |  | |

## Figure 1 – content of data file maze.csv

The assignment consists of “**Basic Requirements**” and “**Advanced Requirements**” as described in sections 4 and 5 respectively. You are advised to complete the basic requirements BEFORE proceeding with the advanced requirements.

For this assignment, you are expected to:

* understand the problem completely and plan your program layout before you start coding
* develop the solution for each task by using functions
* functions developed should be as generic as possible - values used in functions should be passed in as the function parameters
* implement and test each feature as it is developed
* do all the possible data validations

**4. BASIC REQUIREMENTS**

The application should provide the following **basic** features:

* **Display main menu** (and allow for repetition)

When the program is run, it should display the main menu as shown in Figure 2. When a user enters an option from 1 to 6, the program will process the option accordingly. After the option has been processed, the program will display the main menu again and the process is repeated until the user chooses to exit.

**Main menu**

1. **Read and load maze from file**
2. **View maze**
3. **Play maze game**
4. **Configure current maze**
5. **Export maze to file**
6. **Create new maze**
7. **Play maze using SenseHAT**
8. **View Leaderboard**
9. **Exit Maze**

**Enter your option:**

|  |
| --- |
|  |

## Figure 2 - Main Menu

* **Read and load maze from file**

This feature allows the application to read the maze diagram from the data file and store it in a nested list. It also stores the start (A) location and the end (B) location. The application should prompt the user for the name of the data file. The number of records/lines read is then displayed as shown in Figure 3:

|  |
| --- |
|  |

## Figure 3 – Read and load maze from file

* **View maze**

This feature allows the user to view the stored maze from the list created, the start and end location done in option 1 as shown in Figure 4:

|  |
| --- |
|  |

## Figure 4 – View Maze

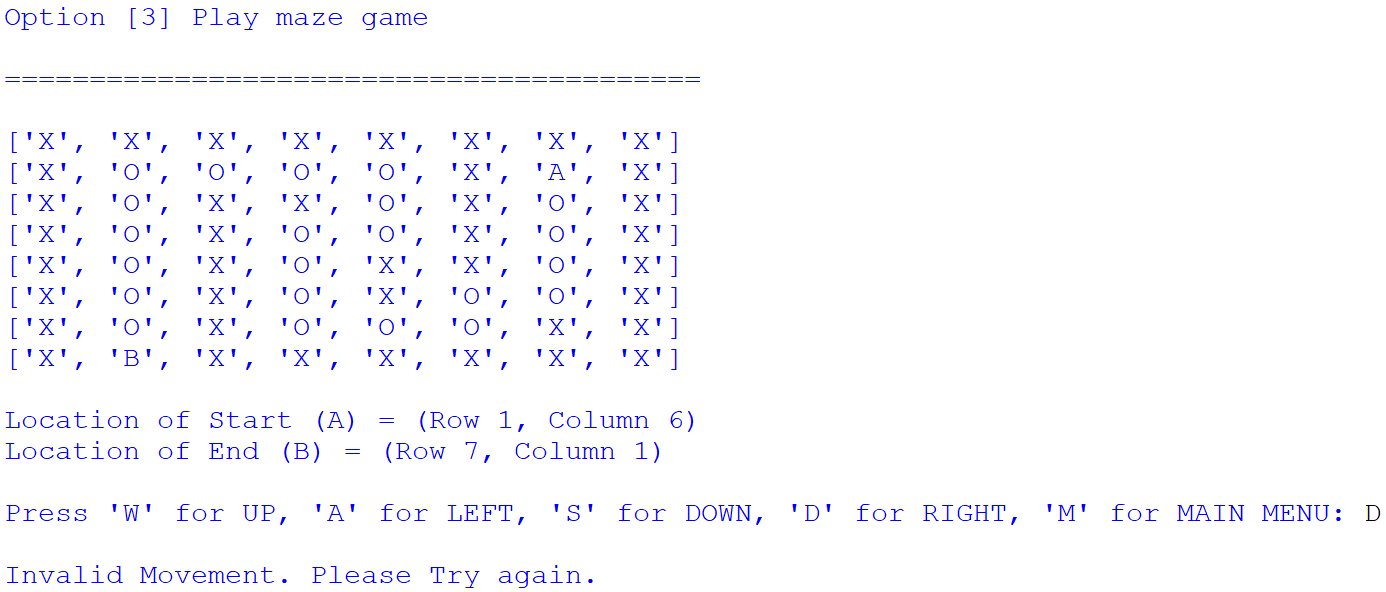
The code for this option should not read the maze from the file again.

* **Play maze game**

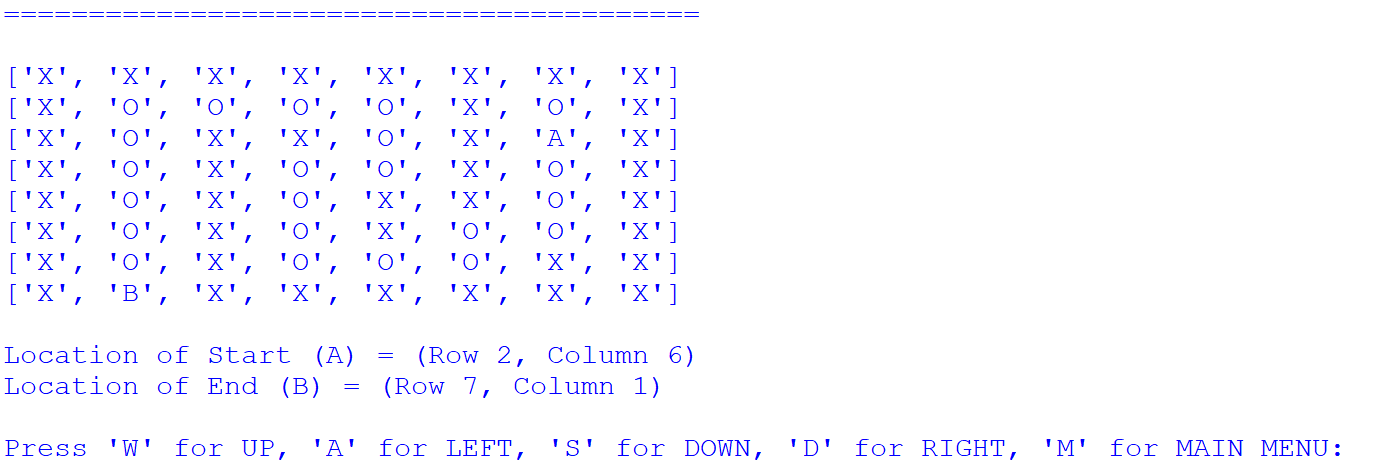
This option allows the user to play the maze game. It displays the maze from the list created, the start and end location as was done in Option 2 and the user can enters one of the following characters: W,A,S,D or M for moving UP, LEFT, DOWN, RIGHT or MAIN MENU respectively, the program will process the option accordingly if it is a valid option i.e. if the user enters W and there is a path for going up, then the program will move the START (A) up one space, otherwise it stays in the current location and a message that the move is invalid will be displayed. After the option has been processed, the program will display the maze again showing the new position of the start point and end point for the maze and the process is repeated until the user chooses to exit to the main menu.

|  |
| --- |
|  |

## Figure 5a – Play maze game



**Figure 5b – Play maze game: invalid move**

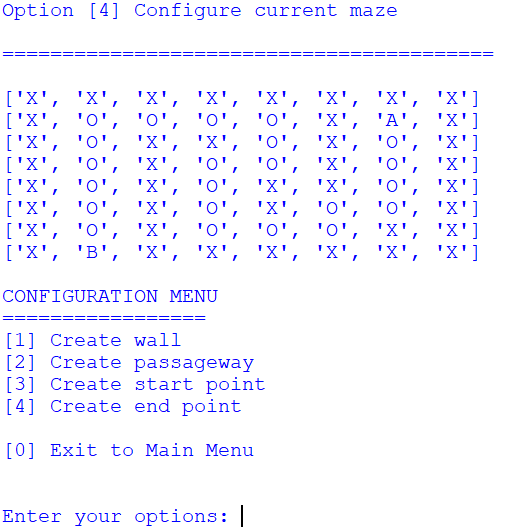


**Figure 5c – Play maze game: update in A for valid move**

* **Configure current maze**

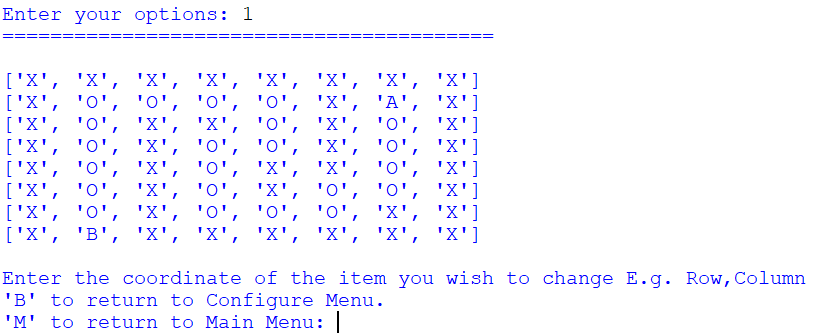
This feature allows the user to change the configuration of the current maze by changing some of the walls and paths, the start and end locations. The sample outputs show one way this can be done. You are free to design your method for configuring the maze.

Sample outputs are shown in Figure 6a and 6b.



## Figure 6a – Configure current maze: Configuration Menu

Sample outputs are shown in Figure 6a and 6b.

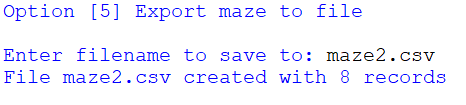


## Figure 6b – Configure current maze: Option 1

* **Export maze to file**

This feature allows the user to export the current maze to an existing or a new csv file. The code should overwrite the file if it exists.

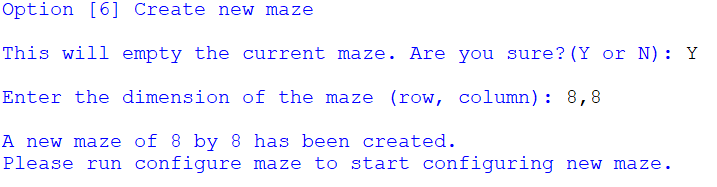
Sample output is shown in Figure 7.



## Figure 7 – Export maze to file

* **Create new maze**

This feature allows the user to create a new empty maze and it also clears the current maze. The user can determine the size of the new maze. The user will then have to use option [4] to configure the new maze.



## Figure 8 – Create new maze

* **Program documentation**

The program should have sufficient comments, which includes your name, class, date, overall description of what the program does, as well as the description of the functions.

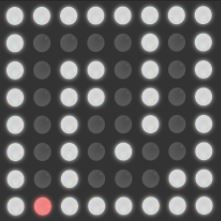
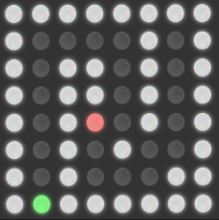
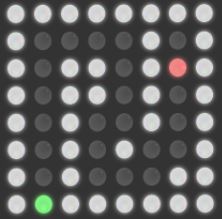
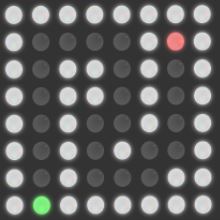
**5. ADVANCED REQUIREMENTS**

* **Program validation -- 10 marks**

ADD appropriate validation for the basic requirements of the program.

* **Play maze using SenseHAT -- 10 marks**

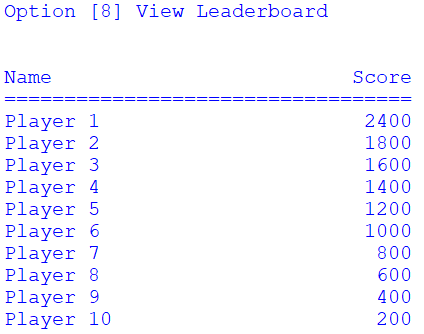
This option will load the maze on the SenseHAT and allows user to play the game by using the joystick of the SenseHAT. Figure 9 shows the gameplay on the SenseHAT, where the start location is red and the end location is green, walls are light grey and path is not lighted.



## Figure 9 – Play maze using SenseHAT

* **View Leaderboard -- 5 marks**

You are to modify your code to keep track of the number of moves taken by the user to complete the maze i.e. moves from start location to end location. This option allows the user to view the leaderboard of the game that displays the 10 top ranking players of the maze game.



## Figure 10 – View Leaderboard

**Note:**

* ***You are expected to follow naming conventions introduced in this module.***
* ***You are encouraged to implement all the basic features before you implement the advanced features.***
* ***You should think carefully what input is required for each option if there is any.***
* ***You are allowed to customize your own output for the advanced features.***
* ***You are required to present your solution to your tutor. Your tutor may ask you questions to verify and assess your understanding of your work. Your tutor may ask you to make some changes to your program to handle another similar feature.***
* ***NO MARKS will be awarded for the advanced features if all the basic features have NOT been fully implemented (and fully working).***
* ***Marks will be deducted if you are not able to show your understanding of the program, both basic and advanced features (if applicable), during the presentation.***

**Reference:**

* **Sense HAT emulator: https://trinket.io/sense-hat**

**6. DELIVERABLES**

* Name the file **"S10009999A\_Assignment.py"** where **"S10009999A"** is your student ID.
* Submit your program into **MeL > Assignment > Assignment Submission** by **5 August 2019 8.30am.**
* Demonstrate your application to your tutor during your PRG1 lessons (and other timeslots scheduled by tutor) starting **5 August 2019.**

**7. ASSESSMENT**

This assignment constitutes 30% of this module.

Performance Criteria for grading the assignment is as described below. Marks awarded will be based on **program code** as well as student’s degree of understanding of work done as assessed during the **presentation**.

### A Grade

* Program implements the Basic Requirements with input validation successfully
* Program implements the Advanced Requirements successfully
* Program demonstrates good design with the correct use of functions
* Program complete with good documentation
* Program has been tested adequately
* Program is coded with good application of fundamental concepts
* Excellent demonstration of program and showing excellent understanding of work done during presentation

### B Grade

* Program implements the Basic Requirements with input validation successfully
* Program implements the Advanced Requirements with partial success
* Program demonstrates good design with the correct use of functions
* Program complete with good documentation
* Program has been tested adequately
* Program is coded with good application of fundamental concepts
* Good demonstration of program and showing good understanding of work done during presentation

### C Grade

* Program implements the Basic Requirements with input validation successfully
* Program demonstrates good design with the use of functions
* Program complete with some documentation
* Program has been tested adequately
* Some demonstration of program and showing some understanding of work done during presentation

### D Grade

* Program implements the Basic Requirementssuccessfully
* Program complete with some documentation
* Program has been tested adequately
* Able to answer some questions during presentation

Validation is required so that if there are more than one employees whose name contains what is entered, the user is asked to choose the correct employee.