Course Name : Data Structure Instructor : Professor Seyed Mehdi Vahidipour

Task : Implement the maze game(maze)

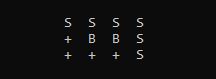
Project goal : Implementing the game, building and solving it by stack and queue.

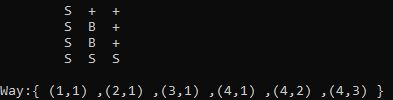
Author : Armin Heidari

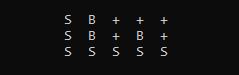
**Spring 2017**

Part B :

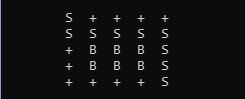
The first step is to create five different maze samples :

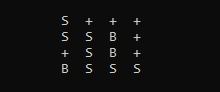
Sample one :

Sample two:

Sample three :

Sample four:



Sample five :

Description :

In the first step, we specify the number of rows and columns of the maze .

( For example in example one row = 3 And column = 4 ) .

In the second step, we specify the closed points ( enter the coordinates of each block ).

In the last step, we select the maze solution option .

\* Tips :

In general, we add two rows and columns to the maze in order to make the margins of the matrix act as unacceptable places .

Closed houses with letters B Are displayed .

We leave the first and last house of the maze empty .

The program method is that the program checks the number of rows and column If the number of rows is more, he chooses his priority based on moving first to the right and then to the left, and if there are more columns, vice versa .

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sample name | **Row x   column** | **Number of available routes** | **The length of the shortest path** | **The length of the longest path** |
| *Example one* | 3 x 4 | 2 | 5 | 5 |
| *Example two* | 4 x 3 | 2 | 5 | 5 |
| *Example three* | 3 x 5 | 2 | 6 | 10 |
| *Sample four* | 5 x 5 | 21 | 7 | 16 |
| *Example five* | 4 x 4 | 7 | 6 | 11 |

In the next step, the priorities change first downwards and then to the right ( in order to move in a zigzag pattern ).

Part C :

|  |  |  |
| --- | --- | --- |
| The name of the movement model | **Number of neighbors** | **round clock /counter clockwise** |
| *A* ( Downward priority ) | 4 | round clock |
| *B* ( Priority to move down the next dimension right ) | 8 | round clock |
| *C* ( Right move priority ) | 4 | counter clockwise |
| *D* ( Priority to move right down ) | 8 | counter clockwise |

Part D :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sample name** | **The name of the movement model** | **The length of the first path found** | **Time required to find the first route** | **Maximum stack size used** |
| *Example one* | A | 5 | 8 | 4 |
| *Example two* | A | 5 | 7 | 3 |
| *Example three* | A | 6 | 10 | 5 |
| *Sample four* | A | 9 | 12 | 5 |
| *Example five* | A | 6 | 10 | 4 |
| *Example one* | B | 5 | 8 | 3 |
| *Example two* | B | 5 | 8 | 4 |
| *Example three* | B | 6 | 10 | 4 |
| *Sample four* | B | 8 | 13 | 5 |
| *Example five* | B | 6 | 8 | 2 |
| *Example one* | C | 5 | 7 | 2 |
| *Example two* | C | 5 | 8 | 3 |
| *Example three* | C | 6 | 8 | 3 |
| *Sample four* | C | 8 | 12 | 4 |
| *Example five* | C | 6 | 9 | 3 |
| *Example one* | D | 5 | 8 | 3 |
| *Example two* | D | 5 | 8 | 3 |
| *Example three* | D | 6 | 11 | 5 |
| *Sample four* | D | 8 | 12 | 5 |
| *Example five* | D | 6 | 9 | 4 |

 Part H :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sample name** | **The name of the movement model** | **The length of the first path found** | **Time required to find the first route** | **The maximum queue size that is used** |
| *Example one* | A | 5 | 15 | 5 |
| *Example two* | A | 5 | 17 | 5 |
| *Example three* | A | 6 | 1 2 | 4 |
| *Sample four* | A | 9 | 12 | 6 |
| *Example five* | A | 6 | 1 3 | 3 |
| *Example one* | B | 5 | 20 | 4 |
| *Example two* | B | 5 | 8 1 | 5 |
| *Example three* | B | 6 | 1 4 | 5 |
| *Sample four* | B | 8 | 2 3 | 6 |
| *Example five* | B | 6 | 18 | 2 |
| *Example one* | C | 5 | 10 | 3 |
| *Example two* | C | 5 | 15 | 5 |
| *Example three* | C | 6 | 14 | 4 |
| *Sample four* | C | 8 | 1 3 | 6 |
| *Example five* | C | 6 | 17 | 6 |
| *Example one* | D | 5 | 11 | 6 |
| *Example two* | D | 5 | 18 | 6 |
| *Example three* | D | 6 | 1 3 | 7 |
| *Sample four* | D | 8 | 1 5 | 6 |
| *Example five* | D | 6 | 10 | 5 |