**PROJECT Traversing a Maze - Using Methods in Java 100 points**

**Objective** To type a simple Java program, execute ( run ) the program for some particular values, observe the output and then modify the program.

***PROJECT DESCRIPTION***

Type, compile and run the basic Java program that is shown in **Figure 1** , which follows.

Then compile and run your program, observe the output then modify the program.

***Information About This Project***

For this project, we will create an application that traverses a rectangular grid - like maze, as shown in **Figure 1** below. The maze has a start entrance position and an exit or finishing position.

**Figure 1**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
| *start* → |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  | ↓  *finish* |  |  |

One way to program a maze structure is to envision the maze as comprised of cells or zones. This maze has none cells or zones.

**Figure 2**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
| *start* → |  | zone 1 | zone 2 | zone 3 |  |
|  |  | zone 4 | zone 5 | zone 6 |  |
|  |  | zone 7 | zone 8 | zone 9 |  |
|  |  |  |  | ↓  *finish* |  |

The optimal solution ( minimal movements ) to the above maze would be:

|  |  |  |  |
| --- | --- | --- | --- |
| ***cell start position*** | ***movement*** | ***action*** | ***cell end position*** |
|  |  |  |  |
| *start* | *move 1* | *move right* | *zone 1* |
| *zone 1* | *move 2* | *move right* | *zone 2* |
| *zone 2* | *move 3* | *move down* | *zone 5* |
| *zone 5* | *move 4* | *move right* | *zone 6* |
| *zone 6* | *move 5* | *move down* | *zone 9* |
| *zone 9* | *move 6* | *move down* | *finish* |

***Steps to Complete This Project***

**STEP 1**  **Open NetBeans**

Open NetBeans and create a Java project with the following details.

For Project Name include **Lab7**

For the Main Class include **lab7.Maze**

In your **Code** window, shown below, copy in the program code shown in **Figure 3** below, in the appropriate places, except substitute your own name in place of Sammy Student.

**PROJECT Traversing a Maze - Using Methods in Java**

**Figure 3 Source Code for the Traversing a Maze Program**

|  |
| --- |
| **import java.util.Scanner;**  **//Sammy Student, Programmer**  **public class Maze**  **{**  **static Scanner *sc* = new Scanner(System.*in*);**  **// maze movements**  **static char *myMove* = '\0';**  **// cell position**  **static int *currentCell* = 0;**  **static int *score* = 0;**  **static boolean *advance* = true;**  **static boolean *checkThis* = false;**  **public static void main(String args[])**  **{**  **// the local variables declared and initialized**  **char answer = 'Y';**    ***displayMenu*();**    **while(answer == 'Y' || answer == 'y')**  **{**  ***displayMovement*();**  ***makeYourMove*();**  ***checkThis* = *checkYourMove*();**  ***mazeStatus*();**    **System.*out*.println("move again(Y or N)?");**  **answer = *sc*.next().charAt(0);**    **}**  **System.*out*.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");**  **}// end main() method**  **static void displayMenu()**  **{**  **System.*out*.println("");**  **System.*out*.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");**  **System.*out*.println("----The Maze Strategy---");**  **System.*out*.println("");**  **}// end method** |

**PROJECT Traversing a Maze - Using Methods in Java**

**Figure 3 Source Code for the Traversing a Maze Program ( continued )**

|  |
| --- |
| **static void displayMovement()**  **{**  **if(*currentCell* == 0)**  **{**  **System.*out*.println("You have entered the maze!!");**  **System.*out*.println("There is no turning back!!");**  ***currentCell* = 1;**  ***mazeStatus*();**  ***advance* = true;**  **}**  **System.*out*.println("make your move (W, A, S, D)");**  **System.*out*.println("W = up, A = left, S = down, D = right)");**  **}// end method**  **static void makeYourMove()**  **{**  ***myMove* = *sc*.next().charAt(0);**    **switch(*myMove*)**  **{**  **case 'W': { *MoveUp*(); break; }**  **case 'A': { *MoveLeft*(); break; }**  **case 'S': { *MoveDown*(); break; }**  **case 'D': { *MoveRight*(); break; }**  **}**  **// end program menu**  **}// end method**  **static boolean checkYourMove()**  **{**  **if(*currentCell* == 1 && *advance* == true)**  **{**  **if (*myMove* == 'W')**  **{**  ***advance* = false;**  **System.*out*.println("try again");**  **return *advance*;**  **}**  **if (*myMove* == 'A')**  **{**  ***advance* = false;**  **System.*out*.println("SORRY, there is no return");**  **return *advance*;**  **}** |

**PROJECT Traversing a Maze - Using Methods in Java**

**Figure 3 Source Code for the Traversing a Maze Program ( continued )**

|  |
| --- |
| **if (*myMove* == 'D')**  **{**  ***currentCell* = 2;**  ***advance* = true;**  **System.*out*.println("continue through the maze");**  **return *advance*;**  **}**  **if (*myMove* == 'S')**  **{**  ***advance* = false;**  **System.*out*.println("continue through the maze");**  **return *advance*;**  **}**  **}**  **if(*currentCell* == 2 && *advance* == true)**  **{**  **if (*myMove* == 'W')**  **{**  ***advance* = false;**  **System.*out*.println("try again");**  **return *advance*;**  **}**  **if (*myMove* == 'A')**  **{**  ***advance* = false;**  **System.*out*.println("SORRY, there is no return");**  **return *advance*;**  **}**  **if (*myMove* == 'D')**  **{**  ***advance* = false;**  **System.*out*.println("continue through the maze");**  **return *advance*;**  **}**  **if (*myMove* == 'S')**  **{**  ***currentCell* = 5;**  ***advance* = true;**  **System.*out*.println("continue through the maze");**  **return *advance*;**  **}**  **}**  **return *advance*;**  **// end program menu**  **}// end method** |

**PROJECT Traversing a Maze - Using Methods in Java**

**Figure 3 Source Code for the Traversing a Maze Program ( continued )**

|  |
| --- |
| **static void MoveLeft()**  **{**  **System.*out*.println("you moved to the left");**    **}//end method**  **static void MoveRight()**  **{**  **System.*out*.println("you moved to the right");**    **}//end method**  **static void MoveUp()**  **{**  **System.*out*.println("you moved up (forward)");**    **}//end method**  **static void MoveDown()**  **{**  **System.*out*.println("you moved down (downward)");**    **}//end method**  **static void mazeStatus()**  **{**  **System.*out*.println("current position: cell " + *currentCell*);**  **}//end method**  **}// end class** |

**STEP 2 Build, Compile and Run the Program**

From the NetBeans Run menu select Run Project (Lab6) to run your app.

**STEP 3 Test the Program**

Once you have successfully compiled your program, review the output **Console** window of NetBeans.

Enter the sample information shown in **Figure 4** that follows.

**PROJECT Traversing a Maze - Using Methods in Java**

**Figure 4 Initial Test Run**

|  |
| --- |
| \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ----The Maze Strategy---  You have entered the maze!!  There is no turning back!!  current position: cell 1  make your move (W, A, S, D)  W = up, A = left, S = down, D = right)  D  you moved to the right  continue through the maze  current position: cell 2  move again(Y or N)?  Y  make your move (W, A, S, D)  W = up, A = left, S = down, D = right)  S  you moved down (downward)  continue through the maze  current position: cell 5  move again(Y or N)?  N  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* |

**STEP 5 Modify the Program**

Supplement the checkYourMove() method by including the remaining **if** statements that allow the maze participant to escape the maze.

**if(*currentCell* == 5 && *advance* == true)**

**{**

**. . .**

**}**

**if(*currentCell* == 6 && *advance* == true)**

**{**

**. . .**

**}**

**if(*currentCell* == 9 && *advance* == true)**

**{**

**. . .**

**}**

**PROJECT Traversing a Maze - Using Methods in Java**

**EXTRA CREDIT:** For extra credit you can utilize the score variable to assign a score to each move. Also for extra credit you can track the number of moves that the user makes in their escape of the maze.

**STEP 6 Submit Your Project**

Once you have determined that your modified program is correctly displaying the maze path solution, complete the submission process as follows:

Open MS Word and type a heading for a new document that includes your full name, course number, lab number and date.

Within the document paste in a snapshot of your modified code. Label your snapshot of your modified run with a reasonable description.

After your snapshot, paste in your finished source code as well copied in from your NetBeans editor.

Submit your MS Word document to Blackboard when complete.