****

**Green University of Bangladesh**

**Department of Computer Science and Engineering (CSE)**

**Faculty of Sciences and Engineering**

**Semester: (Spring, Year: 2023), B.Sc. in CSE (Day)**

**KSA Assignment-1**

**Course Title: Algorithms Lab**

**Course Code: CSE-206 Section: DC**

**Student Details**

|  |  |  |
| --- | --- | --- |
| **Name** | | **ID** |
|  | Md. Sohanur Rahman | 213902106 |

**Lab Date : 21.03.2023**

**Submission Date : 30.03.2023**

**Course Teacher’s Name : Md. Sultanul Islam Ovi**

**[For Teachers use only: Don’t Write Anything inside this box]**

|  |
| --- |
| **Lab Report Status**  **Marks: ………………………………… Signature: .....................**  **Comments: .............................................. Date: ..............................** |

Problem Statement:

1. In this assignment, your task will be to check the grid as Lo Shu Magic Square. Lo Shu Magic Square uses the formula of n(n2+1)/2 = 15, (if n is 3, it means it's a 3 x 3 grid). Thus, all rows, all columns, and all diagonals will have the sum of 15.
2. If it’s a grid of 3 x 3, then the grid contains the numbers 1 through 9. If it’s a grid of 4 x 4, then the grid contains the numbers 1 through 16. **Check for both cases.**
3. In this assignment, your task will be to use a two-dimensional array to simulate a magic square. Write a function/method that accepts a two-dimensional array as an argument, and determines whether the array is a Lo Shu Magic Square or not.

Explanation:

At first take the matrix input from user. Then add the diagonal value of the matrix. After adding diagonal value add the value of every rows and compare that with the diagonal sum. If they are equal then continue this process. Similarly continue this process for the columns.

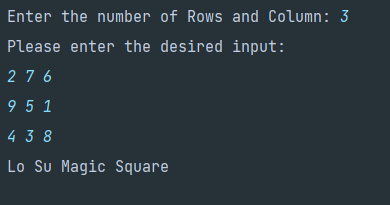
------More Explanation is given inside the code-------

Code:

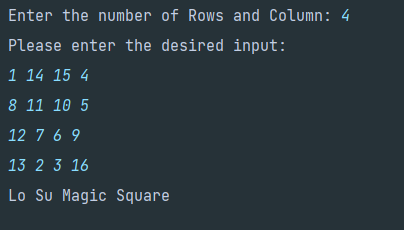
*package* Algorithms\_Lab.KSA\_1;  
  
*import* java.util.Scanner;  
  
*public class* MagicSquare {  
 *static int n*;  
 *public static void* magicSquare(*int*[][] magic){  
 *int* Diagonal,Row,Column;  
 *//Sum of Diagonal.* Diagonal=0;  
 *for*(*int* i=0; i<*n*;i++){  
 Diagonal=Diagonal+magic[i][i];  
 }  
 *int* flag=0;  
 *//Sum of Rows.  
 for*(*int* i=0;i<*n*;i++){  
 Row=0;  
 *for*(*int* j=0;j<*n*;j++){  
 Row=Row+magic[i][j];  
 }  
 *if*(Row==Diagonal){  
 flag=1;  
 }  
 *else*{  
 *break*;  
 }  
 } *//if the sum of diagonal and rows and column are same  
 //then continue the next step.  
 //Sum of Column.  
 if*(flag==1){  
 *for*(*int* i=0;i<*n*;i++){  
 Column=0;  
 *for*(*int* j=0;j<*n*;j++){  
 Column=Column+magic[j][i];  
 }  
 *if*(Column==Diagonal){  
 flag=1;  
 }  
 *else*{  
 flag=0;  
 }  
 }  
 }  
 *//if flag =1 that's mean the magic square condition is satisfied.  
 if*(flag==1){  
 System.***out***.println("Lo Su Magic Square");  
 }  
 *else*{  
 System.***out***.println("Not Magic Square");  
 }  
 }  
 *public static void* main(String[] args) {  
 Scanner scan=*new* Scanner(System.***in***);  
 System.***out***.print("Enter the number of Rows and Column: ");  
 *n*=scan.nextInt();  
 *int*[][] Magic=*new int*[*n*][*n*];  
 *//Take input from user.* System.***out***.println("Please enter the desired input: ");  
 *for*(*int* i=0;i<*n*;i++){  
 *for*(*int* j=0;j<*n*;j++){  
 Magic[i][j]=scan.nextInt();  
 }  
 }  
 *//Call the function/method.  
 magicSquare*(Magic);  
 }  
}

Output:

For 3x3 matrix:



For 4x4 matrix:



Problem Statement:

1. In this assignment, your first task will be the grid addition. Starting with any number on the top row, make your way to the bottom, adding the numbers as you go.
2. Each time you go to the next row, you can move straight down, or one place to the left or right. What is the highest total you can make? What is the lowest?
3. Check for 6 x 6 and 5 x 5 matrices.

Explanation:

This code computes the maximum and minimum sums for each cell in the grid by considering the maximum and minimum sums in the three cells in the previous row that can be reached from the current cell. It then finds the maximum and minimum sum in the last row to determine the highest and lowest totals, respectively.

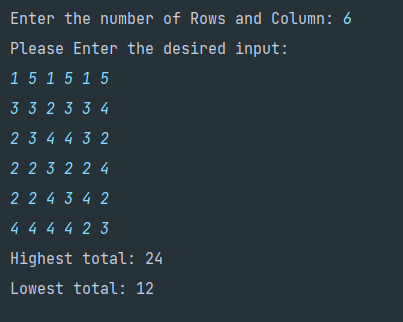
-----More explanation is given inside the code. -----

Code:

*package* Algorithms\_Lab.KSA\_1;  
  
*import* java.util.Scanner;  
  
*public class* GridSum {  
 *static int*[][] *matrix*;  
 *public static void* gridSum(*int*[][] matrix){  
 *int* rows=matrix.length;  
 *int* columns=matrix[0].length;  
 *// Create two-dimensional arrays to store the highest and lowest totals  
 int*[][] highestTotal = *new int*[rows][columns];  
 *int*[][] lowestTotal = *new int*[rows][columns];  
  
*// Initialize the first row of the highest and lowest total arrays  
 for* (*int* j = 0; j < columns; j++) {  
 highestTotal[0][j] = matrix[0][j];  
 lowestTotal[0][j] = matrix[0][j];  
 }  
  
*// Iterate through the remaining rows to determine the highest and lowest totals  
 for* (*int* i = 1; i < rows; i++) {  
 *for* (*int* j = 0; j < columns; j++) {  
 *// Determine the highest and lowest totals for the current cell  
 int* max = Integer.***MIN\_VALUE***;  
 *int* min = Integer.***MAX\_VALUE***;  
 *for* (*int* k = -1; k <= 1; k++) {  
 *if* (j+k >= 0 && j+k < columns) {  
 max = Math.*max*(max, highestTotal[i-1][j+k] + matrix[i][j]);  
 min = Math.*min*(min, lowestTotal[i-1][j+k] + matrix[i][j]);  
 }  
 }  
 highestTotal[i][j] = max;  
 lowestTotal[i][j] = min;  
 }  
 }  
  
*// The highest total will be in the last row of the highestTotal array  
 int* highest = Integer.***MIN\_VALUE***;  
 *for* (*int* j = 0; j < columns; j++) {  
 highest = Math.*max*(highest, highestTotal[rows-1][j]);  
 }  
  
*// The lowest total will be in the last row of the lowestTotal array  
 int* lowest = Integer.***MAX\_VALUE***;  
 *for* (*int* j = 0; j < columns; j++) {  
 lowest = Math.*min*(lowest, lowestTotal[rows-1][j]);  
 }  
  
 System.***out***.println("Highest total: " + highest);  
 System.***out***.println("Lowest total: " + lowest);  
 }  
  
 *public static void* main(String[] args) {  
 Scanner scan=*new* Scanner(System.***in***);  
 *//Take the matrix size from user.* System.***out***.print("Enter the number of Rows and Column: ");  
 *int* n=scan.nextInt();  
 *//Take input from user.* System.***out***.println("Please Enter the desired input: ");  
 *matrix*=*new int*[n][n];  
 *for*(*int* i=0;i<n;i++){  
 *for*(*int* j=0;j<n;j++){  
 *matrix*[i][j]=scan.nextInt();  
 }  
 }  
 *gridSum*(*matrix*);  
 }  
}

Output:

For 6x6 matrix:



For 5x5 matrix:

