Task01 (2D array basics)

Write a program to declare an array of `N x M` size where N and M equals 2 and 3 respectively, perform following operations:

a) Initialize an array to following:

```
 5    2   11
245  22    0
```

And print 2d array without loop.

b) Take user input N x M times, store in array and print (use loop).

c) Print value at 1st row 3rd column.

d) Change value at 2nd row 2nd column to 20.

e) Print ODD/EVEN VALUES stored in array.

f) Print values which are on ODD columns (e.g. 1st, 3rd, 5th column values) ONLY.

g) Take input from user and find whether it is present in array or not. Design function as bool searchVal(int arr[][], int size_arr, int input), that will return true/false.

h) Design a function as int greaterThanValue(int arr[][], int size_arr, int input), that will return number of values greater than input.

i) Design a function as int lessThanValue(int arr[][], int size_arr, int input), that will return number of values less than input.

Solution

```
#include<iostream>
using namespace std;

bool search_value(int a[2][3], int input){
            for(int i=0;i<2;i++){
            for(int j=0;j<3;j++){
                                    if(a[i][j] == input){
                                    return true;
                                    }}}
            }
int greater_than_input(int a[][3], int input){
           int count=0;
           for(int i=0;i<2;i++){
            for(int j=0;j<3;j++){
                                    if(a[i][j] > input){
                                    count++;
                                          }}}
           return count;
           }
int less_than_input(int a[][3], int input){
           int count=0;
           for(int i=0;i<2;i++){
            for(int j=0;j<3;j++){
                                    if(a[i][j] < input){
                                    count++;
                                          }}}
                return count;
           }
int main(){
           //Task a
           cout<<"Task A"<<endl;
           int array[][3]={{1,2,3},
                           {4,5,6}};

           cout<<array[0][0]<<" "<<array[0][1]<<" "<<array[0][2]<<" ";
           cout<<endl;
           cout<<array[1][0]<<" "<<array[1][1]<<" "<<array[1][2]<<" ";

           //Task b
           cout<<"\nTask B"<<endl;
           int arr2[2][3]={};
           cout<<"\nEnter 6 numbers to store: "<<endl;
           for(int i=0;i<2;i++){
           for(int j=0;j<3;j++){
                                    int num;
                                    cout<<"Enter number: ";
                                    cin>>num;
                                    arr2[i][j] = num;
                                    }
                                    }
```

```
    for(int i=0;i<2;i++){
    for(int j=0;j<3;j++){
                          cout<<arr2[i][j]<<" ";
                          }
                          cout<<endl;
                          }

    //Task c
    cout<<"\nTask C"<<endl;
    cout<<"Value at first row, third column: "<<arr2[0][2]<<endl;

    //Task d
    cout<<"\nTask D"<<endl;
    arr2[1][1] = 20;
    cout<<"Array Updated successfully!!!"<<endl;

    //Task e
 cout<<"\nTask E"<<endl;
 cout<<"Even Numbers in an array"<<endl;
 for(int i=0;i<2;i++){
    for(int j=0;j<3;j++){
                          if(arr2[i][j] % 2 ==0){
                          cout<<arr2[i][j]<<" ";
                          }}}

    //Task f
    cout<<"\n\nTask F"<<endl;
    cout<<"Values on ODD Columns"<<endl;
 for(int i=0;i<2;i++){
    for(int j=0;j<3;j++){
                          if(j % 2 ==0){
                          cout<<arr2[i][j]<<" ";
                          }
    }
    cout<<endl;
    }

    //Task g
    cout<<"\nTask G"<<endl;
    int input;
    cout<<"Enter value you want to search: ";
    cin>>input;

    if(search_value(arr2,input)){

cout<<"Value found"<<endl;

}
 else{
 cout<<"Value not found"<<endl;
    }
```

```
            //Task h
            cout<<"\nTask H"<<endl;
            int num1;
            cout<<"Enter number: ";
            cin>>num1;
            int x = greater_than_input(arr2,num1);
            cout<<"There are "<<x<<" values greater than input"<<endl;

            //Task i
            cout<<"\nTask I"<<endl;
            int num2;
            cout<<"Enter number: ";
            cin>>num2;
            int y = less_than_input(arr2,num2);
            cout<<"There are "<<y<<" values less than input"<<endl;



 cout<<endl;
 system("PAUSE");
 return 0;
}
```

Output

```
Task A
1 2 3
4 5 6
Task B

Enter 6 numbers to store:
Enter number: 1
Enter number: 3
Enter number: 5
Enter number: 7
Enter number: 9
Enter number: 11
1 3 5
7 9 11

Task C
Value at first row, third column: 5

Task D
Array Updated successfully!!!

Task E
Even Numbers in an array
20

Task F
Values on ODD Columns
1 5
7 11

Task G
Enter value you want to search: 6
Value not found

Task H
Enter number: 9
There are 2 values greater than input

Task I
Enter number: 10
There are 4 values less than input
```

Tas02 (2d Arrays Medium)

Create 2d array of integer size (N x M) and perform following operations:

a) Initialize 2d array randomly in range between 1 and 10 (inclusive). Use following code to generate random number between 1 and 10:

```
#include <stdlib.h>
#include <time.h>

srand (time(NULL));
int randNum = rand() % 10 + 1;
```

b) Ask user which row and column, wants to print, design function as: void printArr(int arr[][], int size_arr, int row, int col).
Given 2d array, you have to print specific row and column which user wants.

Input	Output
arr = 1 0 0 0 1 0 , row = 0, col = -1 0 0 1	Print first row complete. 1 0 0
arr = 1 0 0 0 1 0 , row = -1, col = 2 0 0 1	Print 3rd column complete. 0 0 1
arr = 1 0 0 0 1 0 , row = 0, col = 2 0 0 1	Print value at 1st row and 3rd column. 0

c) Download task (diagonal difference) and perform.

d) Initialize random matrix of N x M and define function to extract and return only boundaries (e.g. 1st row, 1st column, last row, last column)

Input	Output
arr = 55 30 0 74 225 178 41 68 98 33 200 120 130 63 88 160	55 30 0 74 225 68 98 120 130 63 88 160

e) Modify part d and design function to extract and return only center part of matrix. Center part of matrix is defined as all the values except boundaries.

Input	Output
arr = 55 30 0 74 225 178 41 68 98 33 200 120 130 63 88 160	178 41 33 200

Solution

```
#include <iostream>
using namespace std;


void printArr(int arr[][3], int row, int col)
{
    if (row >= 0 && col >= 0) {
        cout << "Value at (" << row << "," << col << ") = " << arr[row][col] <<
endl;
        return;
    }

    if (row >= 0 && col == -1) {
        cout << "Row " << row << ": ";
        for (int j = 0; j < 3; j++) {
            cout << arr[row][j] << " ";
        }
        cout << endl;
        return;
    }

    if (col >= 0 && row == -1) {
        cout << "Column " << col << ":" << endl;
        for (int i = 0; i < 3; i++) {
            cout << arr[i][col] << endl;
        }
        return;
    }
}

void printBoundaries(int arr[][4])
{
    cout << "\nBoundary Elements:\n";

    for (int i = 0; i < 4; i++) {
        for (int j = 0; j < 4; j++) {
                                if (
                i == 0 || i == 4 - 1 || j == 0 || j == 4 - 1) {
```

```
                cout << arr[i][j] << " ";
            }
        }
        cout << endl;
    }
}


void printCenter(int arr[][4])
{
    cout << "\nCenter Elements:\n";

    for (int i = 0; i < 4; i++) {
        for (int j = 0; j < 4; j++) {


            if (
                i == 0 || i == 4 - 1 || j == 0 || j == 4 - 1) {
                continue;
            }

            cout << arr[i][j] << " ";
        }
        cout << endl;
    }
}

int main()
{
    // Task (a)
    int arrA[3][3] = {
        {1, 0, 0},
        {0, 1, 0},
        {0, 0, 1}
    };

    cout << "Task (a):\n";
    int r, c;
    cout << "Enter row (-1 for skip): ";
    cin >> r;
    cout << "Enter col (-1 for skip): ";
    cin >> c;

    printArr(arrA, r, c);

    // Task (c)
    int arrB[4][4] = {
        {55, 30,  0, 74},
        {225,178, 41, 68},
        {98, 33, 200,120},
        {130,63,  88,160}
    };
```

```
    printBoundaries(arrB);

            // Task (d)
    printCenter(arrB);


cout<<endl;
system("PAUSE");
return 0;
}
```

Output

```
Task (a):
Enter row (-1 for skip): 2
Enter col (-1 for skip): -1
Row 2: 0 0 1

Boundary Elements:
55 30 0 74
225 68
98 120
130 63 88 160

Center Elements:

178 41
33 200
```