Programming Fundamentals

Lab Manual - Week 12

Introduction

Welcome Back to your favorite Programming Lab students. In this lab manual, we shall work together to learn and implement new programming concepts.

Skill: Learning the use of 2D Arrays to store data

Let's do some coding.

Introduction

Students, you can think of a 2D Array as an array of arrays. These arrays are often referred to as matrix where you can store data in rows and columns.

Consider the following task for better understanding.

Task 0

1(WP): Create a program that stores the following data in your program.

	Red	Black	Brown	Blue	Gray
Suzuki	10	7	12	10	4
Toyota	18	11	15	17	2
Nissan	23	19	12	16	14
BMW	7	12	16	0	2
Audi	3	5	6	2	1

We can use the Parallel Arrays to store this data. However, this gets even more complex as our data and its different attributes increase.

```
{
    string color[5] = {"Red", "Black", "Brown", "Blue", "Grey"};
    int suzukiQuantity[5] = {10, 7, 12, 10, 4};
    int toyotaQuantity[5] = {18, 11, 15, 17, 2};
    int nissanQuantity[5] = {23, 19, 12, 16, 14};
    int bmwQuantity[5] = {7, 12, 16, 0, 2};
    int audiQuantity[5] = {3, 5, 6, 2, 1};
}
```

Skill: Learning the use of 2D Arrays to store data

Therefore, we can use the 2D array to store this data.

```
int carData[5][5] = {
    {10, 7, 12, 10, 4},
    {18, 11, 15, 17, 2},
    {23, 19, 12, 16, 14},
    {7, 12, 16, 0, 2},
    {3, 5, 6, 2, 1}
};
```

Task 02(WP):

Write a program that passes this **carData** array into a function and prints all the values in matrix form.

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

void printCars(int cars[][5], int rowSize);

main()
{
    const int rowSize = 5;
    const int colSize = 5;
    int cars[rowSize][colSize] = {{10, 7, 12, 10, 4
                                   {18, 11, 15, 17, 2},
                                   {23, 19, 12, 16, 14},
                                   {7, 12, 16, 0, 2},
                                   {3, 5, 6, 2, 1}};

    printCars(cars, rowSize);
}

void printCars(int cars[][5], int rowSize)
{
    for(int row = 0; row < rowSize; row++)
    {
        for(int col = 0; col < 5; col++)
        {
            cout << cars[row][col] << "\t";
        }
        cout << endl;
    }
}
```

Skill: Learning the use of 2D Arrays to store data

```
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 12\Lab Tasks>Task.exe
10      7       12      10      4
18      11      15      17      2
23      19      12      16      14
7       12      16      0       2
3       5       6       2       1
```

Note: Important thing to note here is:

```
void printCars(int cars[][5], int rowSize);
```

when passing a 2D array to a function, you need to always specify the size of the second dimension (columns) in order for the function to correctly iterate through the elements.
This information is crucial for the function to properly access the elements within each row. Without specifying the column size, the function wouldn't know how far to move in memory from one element to another within a row. Therefore, providing the column size is essential for the function to iterate through the 2D array correctly.
Therefore, it is better to make the colSize variable global and use it main as well as in the required function.

Task 03(WP):

A. Pass the 2D array to the function and Print only the Toyota Blue cars available in the company.

B. Pass the 2D array to the function and return total number of “Red” cars in company

C. Pass the 2D array to the function and return total number of “Nissan” cars in company

D. Write a program that passes the 2D array and color as a parameter into a function named as “converter” and returns the sum total number of cars of that color available.

E. Pass the 2D array to the function and Print the matrix but convert the rows into column and vise versa

Task 04(OP): (printSum)

Skill: Learning the use of 2D Arrays to store data

Programming Fundamentals

Lab Manual - Week 12

Write a program that reads a user defined matrix with column size always 3 and passes that 2D array along with its rowSize to a function named **printSum** that prints the sum of elements of that matrix.

Input	Output
Enter row size: 3 Enter the elements of the matrix: Enter element at position [0][0]: 2 Enter element at position [0][1]: 4 Enter element at position [0][2]: 5 Enter element at position [1][0]: 6 Enter element at position [1][1]: 7 Enter element at position [1][2]: 1 Enter element at position [2][0]: 0 Enter element at position [2][1]: 1 Enter element at position [2][2]: 2	The sum of elements in the matrix is: 28

```
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 12\Lab Tasks>Task4.exe
Enter row size: 3
Enter the elements of the matrix:
Enter element at position [0][0]: 2
Enter element at position [0][1]: 4
Enter element at position [0][2]: 5
Enter element at position [1][0]: 6
Enter element at position [1][1]: 7
Enter element at position [1][2]: 1
Enter element at position [2][0]: 0
Enter element at position [2][1]: 1
Enter element at position [2][2]: 2
The sum of elements in the matrix is: 28
```

Task 05(OP): (Identity Matrix)

Write a program that reads a 3x3 matrix and check whether the matrix is an identity matrix or not.

Note: The identity matrix of size n is the n × n square matrix with ones on the main
diagonal and zeros elsewhere.

Skill: Learning the use of 2D Arrays to store data

```
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 12\Lab Tasks>Task5.exe
Enter the elements of the matrix:
Enter element at position [0][0]: 1
Enter element at position [0][1]: 0
Enter element at position [0][2]: 1
Enter element at position [1][0]: 0
Enter element at position [1][1]: 1
Enter element at position [1][2]: 0
Enter element at position [2][0]: 0
Enter element at position [2][1]: 0
Enter element at position [2][2]: 1
The matrix you entered is:
1       0       1
0       1       0
0       0       1
The entered matrix is NOT an identity matrix.

G:\Semesters\Programming Fundamentals (Fall 2023)\Week 12\Lab Tasks>Task5.exe
Enter the elements of the matrix:
Enter element at position [0][0]: 1
Enter element at position [0][1]: 0
Enter element at position [0][2]: 0
Enter element at position [1][0]: 0
Enter element at position [1][1]: 1
Enter element at position [1][2]: 0
Enter element at position [2][0]: 0
Enter element at position [2][1]: 0
Enter element at position [2][2]: 1
The matrix you entered is:
1       0       0
0       1       0
0       0       1
The entered matrix is an identity matrix.
```

Task 06(CP): (BattleShip)

Remember the game Battleship? Ships are floating in a matrix. You have to fire torpedoes at their suspected coordinates, to try and hit them.

Create a function that takes a 2D array and coordinates as a string. If the coordinate contains only water " . ", return "splash" and if the coordinate contains a ship " * ", return "BOOM".

Hardcode the Battleship configuration in the 5x5 2D array as given in the test case.

Test Cases:

Skill: Learning the use of 2D Arrays to store data

[[" ", " ", " ", "*", "*"], [" ", "*", " ", " ", " "], [" ", "*", " ", " ", " "], [" ", "*", " ", " ", " "], [" ", " ", "*", "*", " "],]	fire("A1") ➡ "splash" fire("A4") ➡ "BOOM" fire("D2") ➡ "BOOM"
Enter coordinate to fire torpedo(e.g. , A1, B3, E5): A1 Result : splash	

```
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 12\Lab Tasks>Task6.exe
Enter coordinates to fire torpedo (e.g., A1, B3, E5): A1
Result: splash

G:\Semesters\Programming Fundamentals (Fall 2023)\Week 12\Lab Tasks>Task6.exe
Enter coordinates to fire torpedo (e.g., A1, B3, E5): A4
Result: BOOM

G:\Semesters\Programming Fundamentals (Fall 2023)\Week 12\Lab Tasks>Task6.exe
Enter coordinates to fire torpedo (e.g., A1, B3, E5): D2
Result: BOOM
```

Task 07(CP): (Football)

In (American) Football, a team can score if they manage to kick a ball through the goal (i.e. above the crossbar and between the uprights).

Create a function that returns true if the ball 0 goes through the goal. Initialize the 2D array named **field** in the main function with 7 rows and 16 columns.

[[" # # "], [" # # "], [" # # "], [" ######## "], [" # 0 "], [" # "], [" # "]]	isGoalScored(field) ➡ false
[[" # # "], [" # 0 # "],	isGoalScored(field) ➡ True

Skill: Learning the use of 2D Arrays to store data

[" # # "], [" ######## "], [" # "], [" # "], [" # "]]	

Task08(CP): (Largest Column First)

Implement the method largestColumnFirst(). This method takes a two-dimensional array of integers M, finds the column with the largest sum, and switches it with the first column in M.

For example:

$$\begin{matrix} 6 & 7 & \mathbf{9} & 4 & 8 \\ 3 & 2 & \mathbf{7} & 4 & 1 \\ 9 & 4 & \mathbf{5} & 8 & 3 \end{matrix} \Rightarrow \begin{matrix} \mathbf{9} & 7 & 6 & 4 & 8 \\ \mathbf{7} & 2 & 3 & 4 & 1 \\ \mathbf{5} & 4 & 9 & 8 & 3 \end{matrix}$$

Hint: Once you find the largest-sum column, you will need to swap its entries with the first column, element-by-element.
Always set the column size to 5.

Skill: Learning the use of 2D Arrays to store data

```
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 12\Lab Tasks>Task8.exe
Enter row size: 3
Enter the elements of the matrix:
Enter element at position [0][0]: 6
Enter element at position [0][1]: 7
Enter element at position [0][2]: 9
Enter element at position [0][3]: 4
Enter element at position [0][4]: 8
Enter element at position [1][0]: 3
Enter element at position [1][1]: 2
Enter element at position [1][2]: 7
Enter element at position [1][3]: 4
Enter element at position [1][4]: 1
Enter element at position [2][0]: 9
Enter element at position [2][1]: 4
Enter element at position [2][2]: 5
Enter element at position [2][3]: 8
Enter element at position [2][4]: 3

Original Matrix:
6 7 9 4 8
3 2 7 4 1
9 4 5 8 3

Matrix after largest column moved to first:
9 7 6 4 8
7 2 3 4 1
5 4 9 8 3
```

Task 09(CP): (Count Identical Arrays)

Create a function that checks the nx3 2D array and returns a count of the total number of identical rows.

Input	Output
[[0, 0, 0], [0, 1, 2], [0, 0, 0], [2, 1, 0]]	countIdenticalArrays(arr, 4) ➞ 2
[[0, 1, 0],	countIdenticalArrays(arr, 4) ➞ 0

Skill: Learning the use of 2D Arrays to store data

[0, 1, 2], [0, 2, 0], [2, 1, 0]]	
[[0, 1, 2], [0, 1, 2], [0, 1, 2], [2, 1, 0]]	countIdenticalArrays(arr, 4) ➡ 3

```
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 12\Lab Tasks>Task8.exe
Enter the number of rows for the array: 4
Enter the elements of the array:
Enter element at position [0][0]: 0
Enter element at position [0][1]: 0
Enter element at position [0][2]: 0
Enter element at position [1][0]: 0
Enter element at position [1][1]: 1
Enter element at position [1][2]: 2
Enter element at position [2][0]: 0
Enter element at position [2][1]: 0
Enter element at position [2][2]: 0
Enter element at position [3][0]: 2
Enter element at position [3][1]: 1
Enter element at position [3][2]: 0
The count of identical rows in the array is: 2

G:\Semesters\Programming Fundamentals (Fall 2023)\Week 12\Lab Tasks>Task8.exe
Enter the number of rows for the array: 4
Enter the elements of the array:
Enter element at position [0][0]: 0
Enter element at position [0][1]: 1
Enter element at position [0][2]: 0
Enter element at position [1][0]: 0
Enter element at position [1][1]: 1
Enter element at position [1][2]: 2
Enter element at position [2][0]: 0
Enter element at position [2][1]: 2
Enter element at position [2][2]: 0
Enter element at position [3][0]: 2
Enter element at position [3][1]: 1
Enter element at position [3][2]: 0
The count of identical rows in the array is: 0
```

Skill: Learning the use of 2D Arrays to store data

Programming Fundamentals

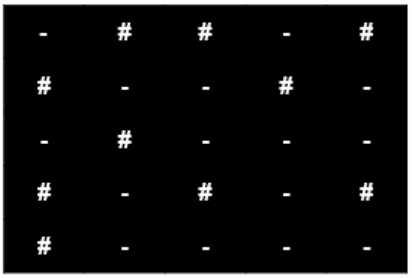
Lab Manual - Week 12

Task 10(CP): (Galaxy)

Given a 2D array (5x5) of some suspended blocks (represented as hashtags), Create three functions:

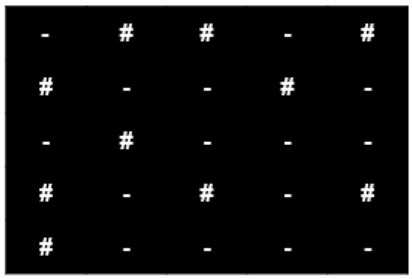
1. DisplayWorld()
2. SetGravityStatus(bool status)
3. TimeTick(int times)

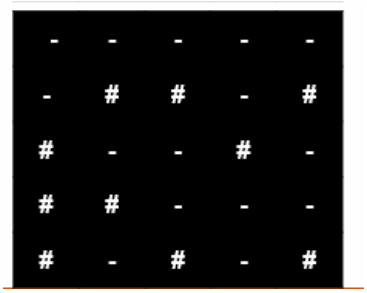
At the moment, we are assuming the last row is hard ground and the block stays on it when it hits the ground. Other blocks can pile on each other.



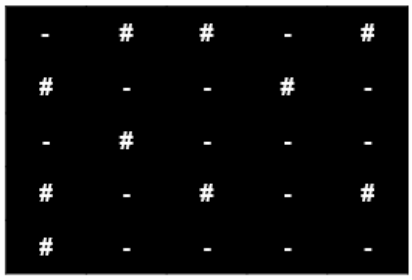
setGravityStatus(bool status) function changes the gravity of the world but will not make any changes in the array or the display.

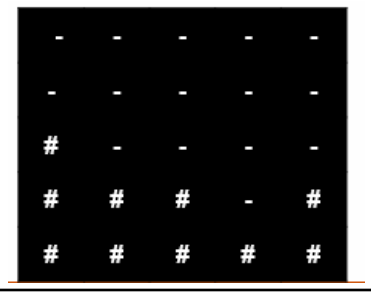
TimeTick(1) will change the world by one tick. We are assuming after tick all blocks fall one level down if possible as shown in figure below.





TimeTick(3) will change the world by 3 ticks as shown below.





Skill: Learning the use of 2D Arrays to store data

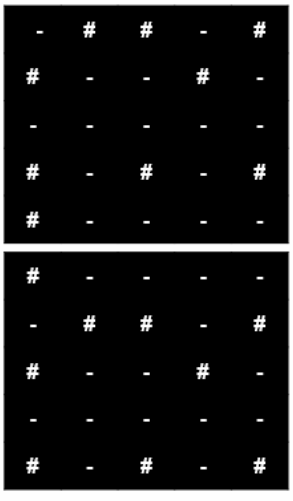
Starter Code is given below.

```
char objects[5][5] = {
    {'-', '#', '#', '-', '#'},
    {'#', '-', '-', '#', '-'},
    {'-', '#', '-', '-', '-'},
    {'#', '-', '#', '-', '#'},
    {'#', '-', '-', '-', '-'}
    };
bool gravity = false;
main(
{
    displayWorld();
    setGravityStatus(true);
    timeTick(3);
    displayWorld();
}
```

Task 10(CP) (Galaxy-BlackHole Extended):

Now, you need to change the block activity but assume the last row is not solid ground. Instead, it is black hole that ports everything back to the first row when it tries to get out of it. The last row will only act as black hole if the flag variable isBlackHole set to true. In case, its value is false the behaviour of the program will remain same as it was previously with solid ground.

```
char objects[5][5] = {
    {'-', '#', '#', '-', '#'},
    {'#', '-', '-', '#', '-'},
    {'-', '#', '-', '-', '-'},
    {'#', '-', '#', '-', '#'},
    {'#', '-', '-', '-', '-'}
    };
bool gravity = false;
bool isBlackHole = true;
main(){
    displayWorld();
    setGravityStatus(true);
    timeTick(1);
    displayWorld();
}
```

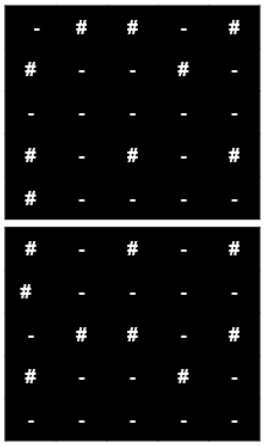


Skill: Learning the use of 2D Arrays to store data

Programming Fundamentals

Lab Manual - Week 12

```
char objects[5][5] = {
    {'-', '#', '#', '-', '#'},
    {'#', '-', '-', '#', '-'},
    {'-', '#', '-', '-', '-'},
    {'#', '-', '#', '-', '#'},
    {'#', '-', '-', '-', '-'}
    };
bool gravity = false;
bool isBlackHole = true;
main(){
    displayWorld();
    setGravityStatus(true);
    timeTick(2);
    displayWorld();
}
```



Good Luck and Best Wishes !!

Happy Coding ahead :)

Skill: Learning the use of 2D Arrays to store data