

Lab. 03

Double Pointer and 2-D Arrays

OBJECTIVE:

Things that will be covered in today's lab:

- Double Pointers
- Dynamic Two Dimensional Array

THEORY:

Double Pointer: A double pointer is a pointer to pointer. Normally, a pointer contains the address of a variable. When we define a pointer to a pointer, the first pointer contains the address of the second pointer, which points to the location that contains the actual value as shown below:



In C++, you declare a double pointer variable by using two asterisks (**) between the data type and the variable name. The general syntax to declare a pointer variable is as follows:

```
datatype ** identifier;
```

```
int var=3000;
int *ptr;
int **pptr;

ptr = &var;           // assigns the address of var to ptr
pptr = &ptr;          // assigns the address of ptr to pptr

cout << var << endl;
cout << *ptr << endl; // pointer points to var
cout << **pptr << endl; // Pointer points of ptr
```

Dynamic 2-D Array: First, we will allocate memory for an array which contains a set of pointers. Next, we will allocate memory for each array which is pointed by pointers. The de-allocation of memory is done in the reverse order of memory allocation

```
int **Array = 0;
Array = newint *[ROWS]; //memory allocated for elements of
// row.
for(int i=0; i<ROWS; i++)//memory allocated for each col.
Array[i] = newint[COLUMNS];

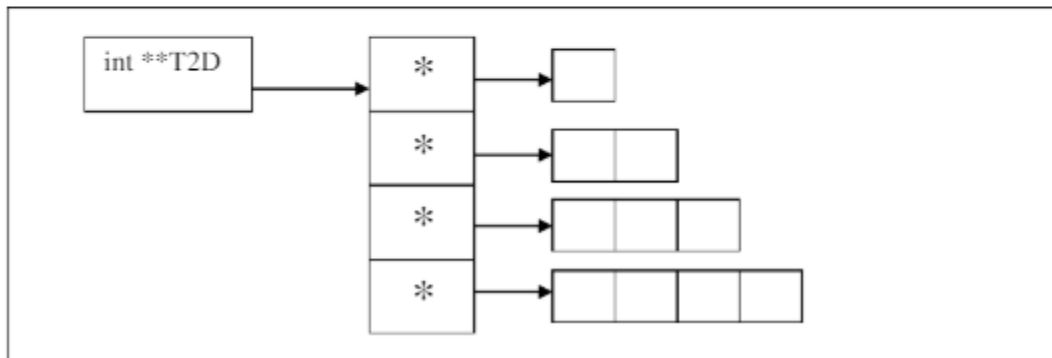
for(int i=0; i<ROWS; i++)//free the allocated memory
delete [] Array[i] ;
delete [] Array ;
```

Exercise 1:

A double pointer is used for declaring two dimensional arrays dynamically. For example,

```
int **p;
p=newint *[rows];
for(int i=0; i<rows; i++)
p[i]=newint [cols];
```

We want to implement a **triangular 2D** array in which each row has one column greater than the previous one. i.e., the first row has one column, the second one has two columns, and the third one has three columns and so on. You have to take the total number of rows from the user. Following is an example of a triangular 2D array with four rows.



Lab Manual of Programming Fundamentals

Write following functions with given prototype:

a. `void AddColumns (int * &, int size)`

This function takes a single pointer by reference and dynamically allocates memory to it. You will call this function in main to allocate number of columns to each row turn by turn.

```
int main( ){  
    int **p;  
    p=newint *[rows];  
    for(int i=0; i<rows; i++)  
        //call function...  
}
```

b. `void RowWiseInput(int *, int size)`

This function simply takes a pointer as an argument and takes input in it from the user. The second argument is the size of 1D array pointed by pointer.

c. `void RowWisePrint(int *, int size)`

This function takes a pointer as argument and prints its contents. The second argument is the size of 1D array pointed by pointer.

d. `int main()`

In main function you have to do the following tasks:

- 1) Ask the user to enter the number of rows.
- 2) Declare a 2D array.
- 3) Allocate memory for its columns in this array using **AddColumns** Function defined above.
- 4) Take input in the 2D array using **RowWiseInput** function.
- 5) Print the 2D array using **RowWisePrint** function.
- 6) You may use loops in main.
- 7) No global variables are allowed.

Exercise 2:

(Airplane Seating Assignment) Write a program that can be used to assign seats for a commercial airplane. The airplane has 13 rows, with six seats in each row. Rows 1 and 2 are first class, rows 3 through 7 are business class, and rows 8 through 13 are economy class. Your program must prompt the user to enter the following information:

- Ticket type (first class, business class, or economy class)
- Desired seat

Output the seating plan in the following form:

	A	B	C	D	E	F
Row 1	*	*	X	*	X	X
Row 2	*	X	*	X	*	X
Row 3	*	*	X	X	*	X
Row 4	X	*	X	*	X	X
Row 5	*	X	*	X	*	*
Row 6	*	X	*	*	*	X
Row 7	X	*	*	*	X	X
Row 8	*	X	*	X	X	*
Row 9	X	*	X	X	*	X
Row 10	*	X	*	X	X	X
Row 11	*	*	X	*	X	*
Row 12	*	*	X	X	*	X
Row 13	*	*	*	*	X	*

Here, * indicates that the seat is available; X indicates that the seat is occupied. Make this a menu-driven program; show the user's choices and allow the user to make the appropriate choices.

Post Lab:

Write a C++ program to take the transpose of a matrix. A transpose of the matrix, is the matrix obtained by interchanging the rows and columns of the original matrix. Consider following examples:

1. Given Matrix

1	2	3
4	5	6
7	8	9

Transpose:

1	4	7
2	5	8
3	6	9

2. Given Matrix

1	2
3	4
5	6

Transpose:

1	3	5
2	4	6

A matrix is represented by a 2D array. Use a double pointer to declare this 2D array dynamically. The number of rows and columns should be taken as input from the user. Your code should be generic, i.e., it should work for any number of rows and columns.

You will have to create a new matrix of reversed size. For example, if the input matrix is of size (2x3), the new matrix should be of a size (3x2). Create the new matrix dynamically.