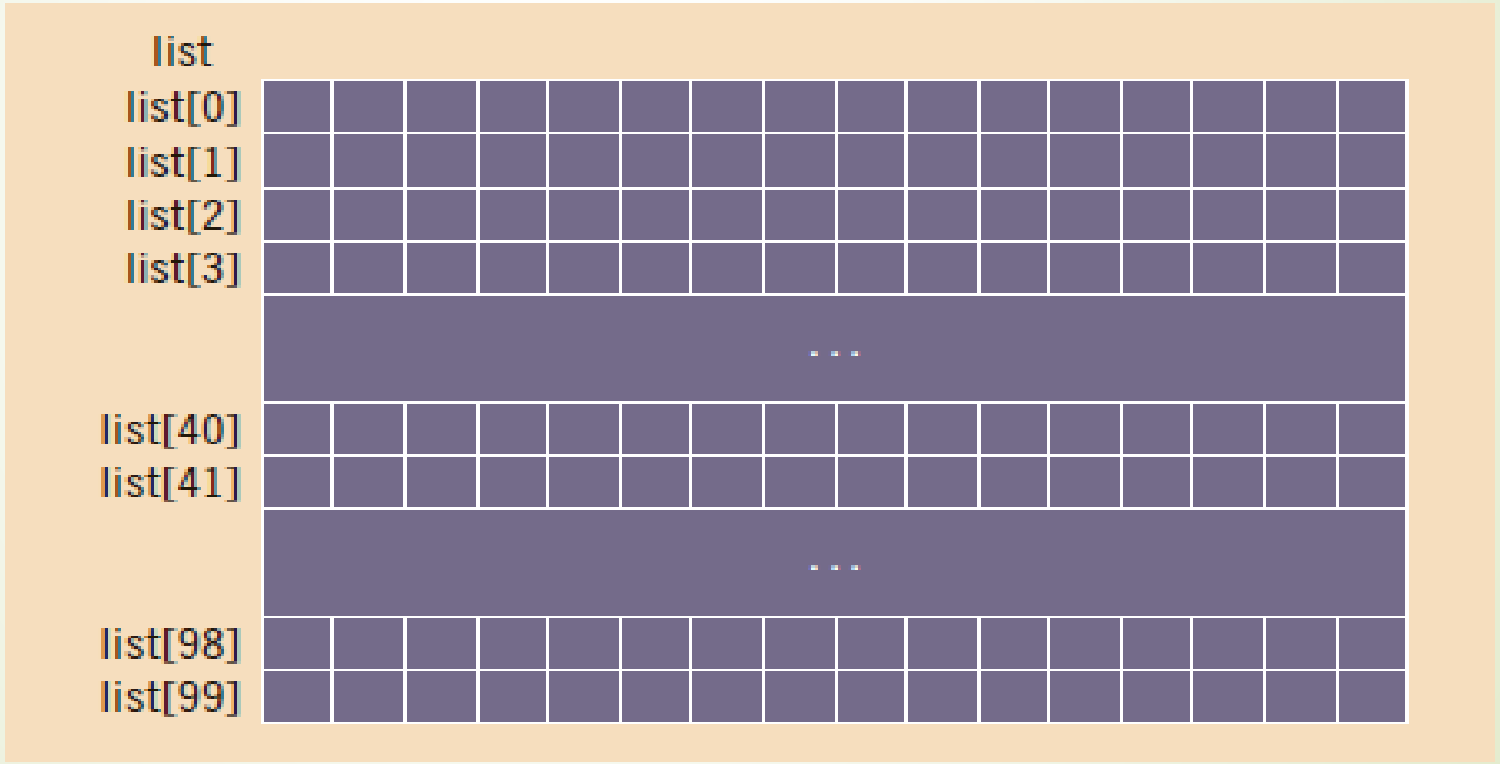
Programming Fundamentals

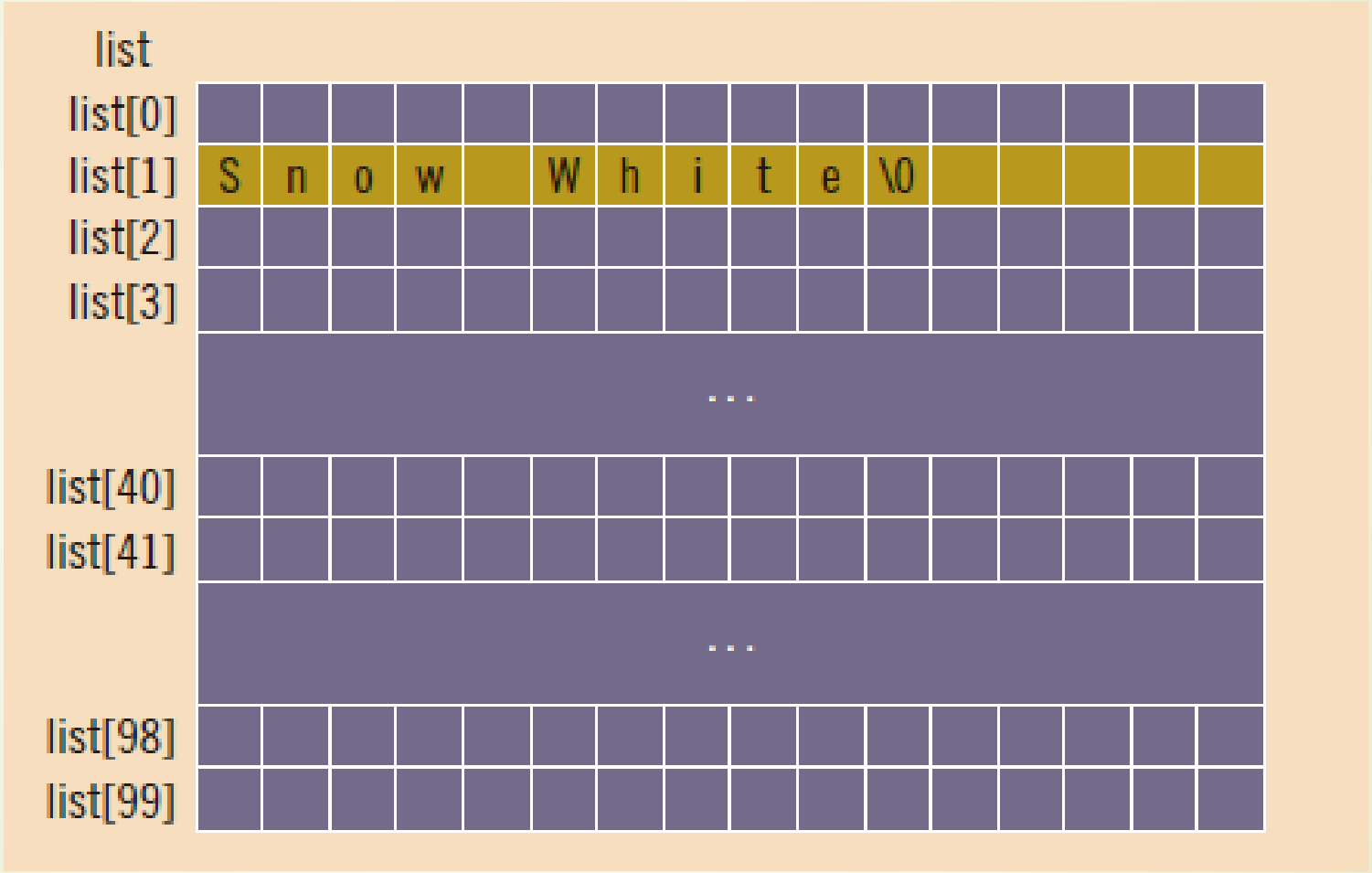
Aamina Batool

2-D Character Arrays

- `char list[100][16];`



```
strcpy(list[1], "Snow White");
```



Inputting/outputting char 2-d arrays

- Suppose that you want to read and store data in **`list`** and that there is one entry per line.
- The following **`for`** loop accomplishes this task:

```
for (int j = 0; j < 100; j++)
    cin.get(list[j], 16);
```

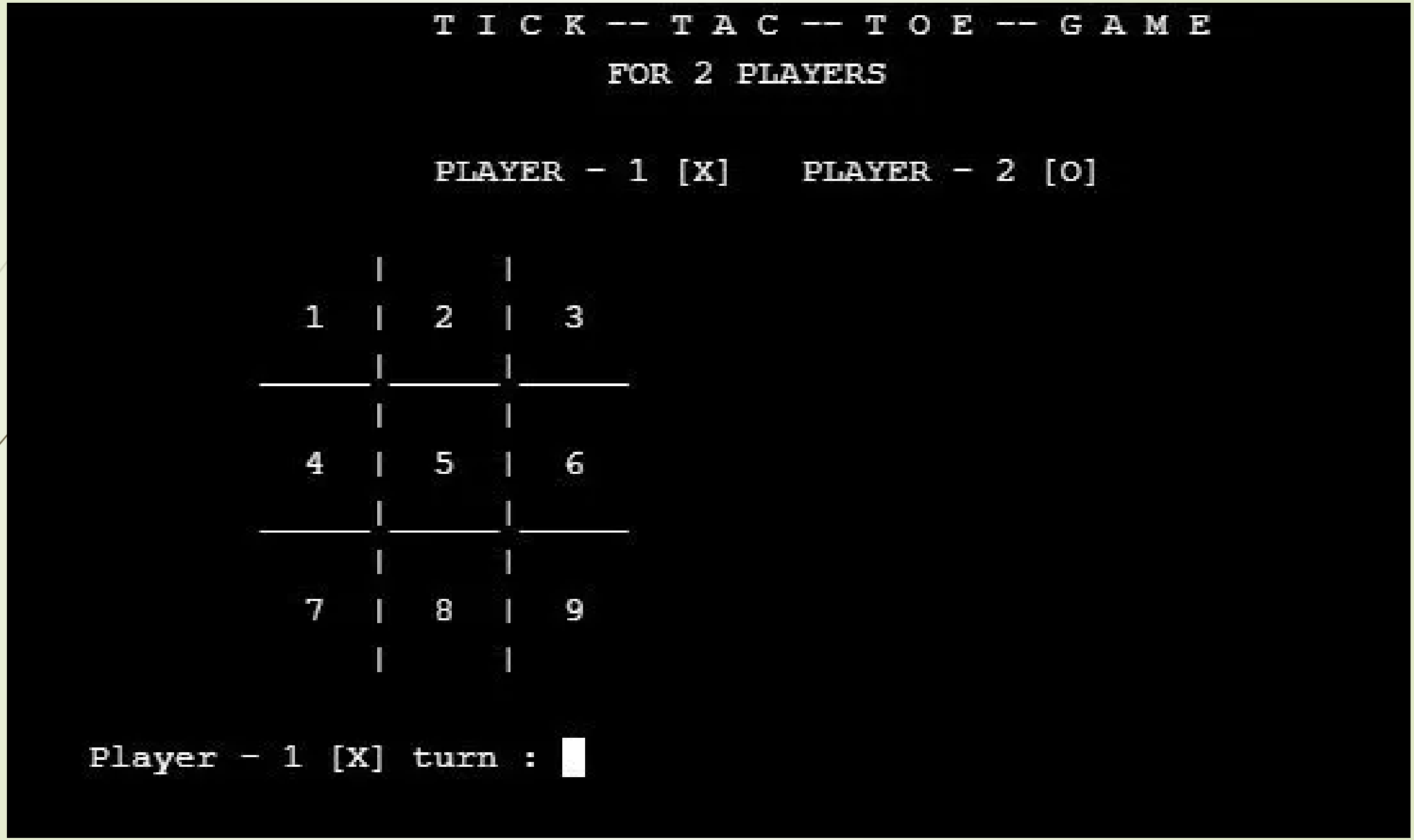
- The following **`for`** loop outputs the string in each row:

```
for (int j = 0; j < 100; j++)
    cout << list[j] << endl;
```

- You can also use other string functions (such as **`strcmp`** and **`strlen`**) and **`for`** loops to manipulate **`list`**.

Another way to initialize

```
char colour[4][10] = { "Blue", "Red", "Orange” , “Yellow” };
```


T I C K -- T A C -- T O E -- G A M E
FOR 2 PLAYERS
PLAYER - 1 [X] PLAYER - 2 [O]
1 | 2 | 3
4 | 5 | 6
7 | 8 | 9
Player - 1 [X] turn :

Not a c-string always! E.g. Tic Tac Toe

- `char board[3][3] =`

```
{{'1','2','3'},
 {'4','5','6'},
{'7','8','9'}};
```

Reading from files

Data in the input file:

30

40

50

60

70

Task: Read these integers from the file one by one (you'd need an integer variable) and save into another file.

Reading from files

- Data in the input file:

Hello World!

We gotta find a new way of greeting the world :3

Eh, Who cares!

Tasks:

a) Read these lines one by one and save them in an array. You'd need a 2d character array to save these c-strings.

b) Once you've read the whole file and saved it in the array, now write the data from this 2-D character array into an output file (in the same format as in the input file)

Hint: use getline function to read (also remember that aggregate input/output operations are allowed on character arrays)

Reading from files

- Data in the input file:

3 45 67 90

40 6 1 3

Tasks:

a) **Read the data from the file and save in two separate 1-D arrays.**

b) **Write the data from 2 arrays of part a in a single output file in the same format as the input file.**

c) **Read the data from the file and save in a 2-D array.**

d) **Write the data from 2D array of part c in a single output file in the same format as the input file.**

Passing Two-Dimensional Arrays as Parameters to Functions

- Two-dimensional arrays can be passed as parameters to a function
- By default, arrays are passed by reference
- The base address, that is, the address of the first component of the actual parameter is passed to the formal parameter

Two-Dimensional Arrays

- When declaring a two-dimensional array as a formal parameter
  - Can omit size of first dimension, but not the second
- Number of columns must be specified

Code to print array

Print

```
for (row = 0; row < NUMBER_OF_ROWS; row++)
{
    for (col = 0; col < NUMBER_OF_COLUMNS; col++)
        cout << setw(5) << matrix[row][col] << " ";

    cout << endl;
}
```

```
const int NUMBER_OF_ROWS = 6;
const int NUMBER_OF_COLUMNS = 5;
```

Consider the following definition of the function `printMatrix`:

```
void printMatrix(int matrix[][NUMBER_OF_COLUMNS],
                 int noOfRows)
{
    for (int row = 0; row < noOfRows; row++)
    {
        for (int col = 0; col < NUMBER_OF_COLUMNS; col++)
            cout << setw(5) << matrix[row][col] << " ";

        cout << endl;
    }
}
```

Code to sum rows

Sum by Row

```
  //Sum of each individual row
for (row = 0; row < NUMBER_OF_ROWS; row++)
{
     sum = 0;
     for (col = 0; col < NUMBER_OF_COLUMNS; col++)
          sum = sum + matrix[row][col];

     cout << "Sum of row " << row + 1 << " = " << sum << endl;
}
```

Largest Element in Each Row and Each Column

```
  //Largest element in each row
for (row = 0; row < NUMBER_OF_ROWS; row++)
{
    largest = matrix[row][0]; //Assume that the first element
                              //of the row is the largest.
    for (col = 1; col < NUMBER_OF_COLUMNS; col++)
        if (largest < matrix[row][col])
            largest = matrix[row][col];

    cout << "The largest element in row " << row + 1 << " = "
         << largest << endl;
}

  //Largest element in each column
for (col = 0; col < NUMBER_OF_COLUMNS; col++)
{
    largest = matrix[0][col]; //Assume that the first element
                              //of the column is the largest.
    for (row = 1; row < NUMBER_OF_ROWS; row++)
        if (largest < matrix[row][col])
            largest = matrix[row][col];

    cout << "The largest element in column " << col + 1
         << " = " << largest << endl;
}
```

```
void largestInRows(int matrix[][NUMBER_OF_COLUMNS],int noOfRows)
//Largest element in each row

void sumRows(int matrix[][NUMBER_OF_COLUMNS],int noOfRows);
//Sum of each individual row
```

```
const int NUMBER_OF_ROWS = 6;
const int NUMBER_OF_COLUMNS = 5;
void printMatrix(int matrix[][NUMBER_OF_COLUMNS], int NUMBER_OF_ROWS);
void sumRows(int matrix[][NUMBER_OF_COLUMNS], int NUMBER_OF_ROWS);
void largestInRows(int matrix[][NUMBER_OF_COLUMNS], int NUMBER_OF_ROWS);
int main()
{int board[NUMBER_OF_ROWS][NUMBER_OF_COLUMNS]=
                    {{17, 8, 24, 10, 28},
                    {9, 20, 16, 55, 90},
                    {25, 45, 35, 8, 78},
                    {5, 0, 96, 45, 38},
                    {76, 30, 8, 14, 28},
                    {9, 60, 55, 62, 10}};
printMatrix(board, NUMBER_OF_ROWS);
cout << endl;
sumRows(board, NUMBER_OF_ROWS);
cout << endl;
largestInRows(board, NUMBER_OF_ROWS);
return 0; }
```

Output

Sample Run:

```
    17      8     24     10     28
     9     20     16     55     90
    25     45     35      8     78
     5      0     96     45     38
    76     30      8     14     28
     9     60     55     62     10

Sum of row 1 = 87
Sum of row 2 = 190
Sum of row 3 = 191
Sum of row 4 = 184
Sum of row 5 = 156
Sum of row 6 = 196

The largest element of row 1 = 28
The largest element of row 2 = 90
The largest element of row 3 = 78
The largest element of row 4 = 96
The largest element of row 5 = 76
The largest element of row 6 = 62
```

Exercises

- C++ Program to store temperature of two different cities for a week and display it.
- Find Column/Row wise max, min, average, sum.
- Sort the array Row/Column wise.
- Write a program for adding two matrices of size 2x2, take input from the user.
- Write a program for multiplying two matrices of size 2x2, take input from the user.
- Write a program to find transpose of a 3x3 matrix and a 3x2 matrix.
- Write a program to find inverse of a 2x2 matrix.

References

1. C++ Programming: From Problem Analysis to Program Design, Third Edition
2. https://www.just.edu.jo/~yahya-t/cs115/