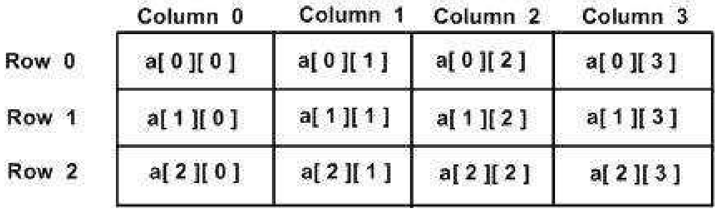
CSE 115L: Programming Language I Lab (Section: 06)

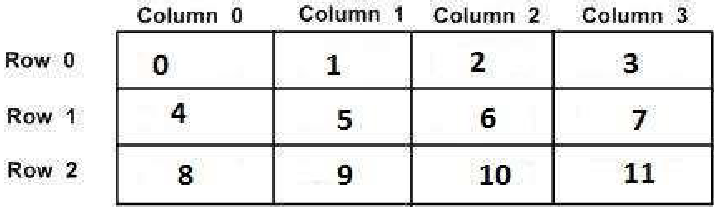
Spring 2020

Lab-07 (2D Arrays)

Memory Representation



	Column 0	Column 1	Column 2	Column 3
Row 0	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 1	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 2	a[2][0]	a[2][1]	a[2][2]	a[2][3]



	Column 0	Column 1	Column 2	Column 3
Row 0	0	1	2	3
Row 1	4	5	6	7
Row 2	8	9	10	11

To access value 1 we write a[0][1]
To access value 11 we write a[2][3]

Example:

```
printf("value in row-0, column-1: %d", a[0][1]);
//this will output 1
```

Basic Syntax

DataType arrayName [row][column];

How to declare the 2D arrays:-

```
int a[3][4];
```

In the declaration above row=3 and column=4 OR

```
int a[3][4] = {
                {0, 1, 2, 3} , /* row 0 */
                {4, 5, 6, 7} , /* row 1 */
                {8, 9, 10, 11} /* row 2 */
                };
```

OR

The above statement is same as:

```
int a[3][4] = {0,1,2,3,4,5,6,7,8,9,10,11};
```

Example: Declaring and accessing the elements of a two-dimensional array

```
#include <stdio.h>
#include <math.h>

int main()
{
        int A[100][100], i, j, rows, columns;
        printf("Number of rows: ");
        scanf("%d", &rows);

        printf("Number of columns: ");
        scanf("%d", &columns);

        for(i=0;i<rows;i++)
        {
        for(j=0;j<columns;j++)
        {
        printf("A[%d][%d]: ",i, j);
        scanf("%d",&A[i][j]);
        }
        }

        printf("Values in array A:\n");

        for(i=0;i<rows;i++)
        {
        for(j=0;j<columns;j++)
        {
        printf("\t%d",A[i][j]);
        }
        printf("\n");
        }

        return 0;
}
```

Bubble Sort:

Bubble sort is a simple sorting algorithm. This sorting algorithm is a comparison-based algorithm in which each pair of adjacent elements is compared and the elements are swapped if they are not in order.

Example:

14	33	27	35	10

Bubble sort starts with very first two elements, comparing them to check which one is greater:

14	33	27	35	10

In this case, value 33 is greater than 14, so it is already in sorted locations. Next, we compare 33 with 27.

14	33	27	35	10

We find that 27 is smaller than 33 and these two values must be swapped.

14	33	27	35	10

The new array should look like this -

14	27	33	35	10

Next we compare 33 and 35. We find that both are in already sorted positions.

14	27	33	35	10

Then we move to the next two values, 35 and 10.

14	27	33	35	10

We know then that 10 is smaller than 35. Hence they are not sorted.

14	27	33	35	10

We swap these values. We find that we have reached the end of the array. After one iteration, the array should look like this -

14	27	33	10	35

To be precise, we are now showing how the array should look like after each iteration. After the second iteration, it should look like this -

14	27	10	33	35

Notice that after each iteration, at least one value moves at the end.

14	10	27	33	35

And when there's no swap required, bubble sorts learns that an array is completely sorted.

10	14	27	33	35

```
#include <stdio.h>

int bubbleSort(int list[], int size) {
   int temp;
   int i,j,k;

   int swapped;

   // loop through all numbers
   for(i = 0; i < size-1; i++)
   {
            swapped = 0;

            // loop through numbers falling ahead
            for(j = 0; j < size-1-i; j++)
            {
            printf("Items compared: [ %d, %d ] ", list[j],list[j+1]);

            // check if next number is lesser than current no
            //   swap the numbers.
            //  (Bubble up the highest number)

            if(list[j] > list[j+1])
            {
            temp = list[j];
            list[j] = list[j+1];
            list[j+1] = temp;

            swapped = 1;
            printf(" => swapped [%d, %d]\n", list[j],list[j+1]);
            }
            else
            {
            printf(" => not swapped\n");
            }

            }

            // if no number was swapped that means
            //   array is sorted now, break the loop.
            if(!swapped) {
            break;
            }
```

```
        printf("\nAfter Iteration #%d: ",(i+1));
        for(k=0; k<size; k++)
        {
        printf("%d ", list[k]);
        }
        printf("\n\n");

  }
        printf("\nThe numbers arranged in ascending order are given below \n");

        for(i=0; i<size; i++)
        {
        printf("%d ", list[i]);
        }
}

int main(void)
{
        int size, i;
        printf("Enter the number of element in the array: \n");
        scanf("%d", &size);

        int arr [size];
        printf("Enter the elements:\n");

        for (i = 0; i < size; i++)
        scanf("%d", &arr[i]);

        printf("\n");

        bubbleSort(arr, size);

        return 0;
}
```