LAB5 : Arrays

Objectives:

1. To learn how to use C array as a counter.
2. To learn how to add an element to the array.
3. To learn how to delete an element from the array.
4. To learn how to declare two dimensional array.
5. To learn passing arrays as function parameters

Definitions:

Two Dimensional Array - An array with a multiple indexs.

Theory:

Multi-dimensional Arrays

An array is a collection of individual values, all of the same data type, stored in adjacent memory locations. Using the array name together with an integral valued index in square brackets refers to the individual values. Multi-dimensional arrays have additional indices for each additional dimension. The index indicates the position of the individual component value within the collection of values. The index is also called the subscript. In C, the first array element always has subscript 0. The second array element has subscript 1, etc. The base address of an array is its beginning address in memory.

Declaring a Multi-dimensional Array:

In order to declare an m-dimensional array, a C programmer will use the following syntax below.

```
DataType ArrayName[#of rows][#of columns];
```

The example below shows the declaration of a 2-dimensional integer array of size 3 x 3 with elements [0 - 2] [0 - 2].

```
#define MAXSIZE  3
......
int array[MAXSIZE][MAXSIZE];
```

A multi-dimensional array can be initialized during declaration by equating the array to a listing of the array's members in brackets. For example: (Note: Each dimension has its own { } brackets.)

```
int array[MAXSIZE][MAXSIZE] = {{2 , 4, 6}, {8, 10, 12}, {14, 16, 18}};
```

Passing Arrays as Function Parameters

In C, arrays are always passed by reference by default. Whenever an array is passed as a parameter, its base address is sent to the called function.

Generally, functions that work with arrays require 2 items of information as actual parameters: the beginning address of the array (in memory), and the number of elements to process in the array.

For example (Function PrintArray):

```
void PrintArray(int Array[], int ArraySize)
{
   for (int i = 0; i <ArraySize; i++)
      printf("array[%d] = %d\n" ,i,Array[i]);

}
```

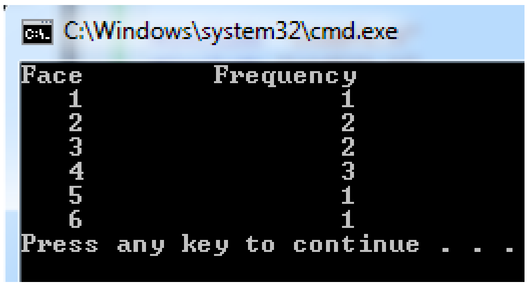
Use of Const

Since arrays are passed to functions by reference, the values contained in the array will be changed permanently. In order to prevent an array that is used as an actual parameter from being unintentionally changed by a function, you can place **const** in the function heading and prototype.

NOTE: If the values of the array need to be changed in the function do not use **const**.

Assignment:

1) Complete the following program below by filling in the blanks.



```
Face            Frequency
   1                    1
   2                    2
   3                    2
   4                    3
   5                    1
   6                    1
Press any key to continue . . .
```

```
#include "stdafx.h"
#include "stdlib.h"
#include "time.h"
#include "conio.h"
```

```
#define SIZE 7
/* function main begins program execution */

void main()
{
   int face; /* random die value 1 - 6 */
   int roll; /* roll counter */
   int frequency[ SIZE ] = _____________; /* clear counts */
   srand( time( NULL ) ); /* seed random-number generator */

   //roll a die 10 times
   for ( roll = 1; roll <= ______; roll++ ) {
      face = _________________; //get a random value between 1 and 6
      _________________;  // increases the appropriate frequency counter
depending on the value of face
   }
   printf( "%s%17s\n", "Face", "Frequency" );
   /* output frequency elements 1-6 in tabular format */
   for ( face = 1; face < __________________; face++ ) {
      printf( "%4d%17d\n", face, __________________ );
   }
}
```

Sol:

```
#include "stdafx.h"
#include "stdlib.h"
#include "time.h"
#include "conio.h"
#define SIZE 7
/* function main begins program execution */

void main()
{
   int face; /* random die value 1 - 6 */
   int roll; /* roll counter */
   int frequency[ SIZE ] = { 0 }; /* clear counts */
   srand( time( NULL ) ); /* seed random-number generator */

   for ( roll = 1; roll <= 10; roll++ ) {
      face = 1 + rand() % 6;
      ++frequency[ face ]; /* replaces 26-line switch of Fig. 5.8 */
   } /* end for */
   printf( "%s%17s\n", "Face", "Frequency" );
   /* output frequency elements 1-6 in tabular format */
   for ( face = 1; face < SIZE; face++ ) {
      printf( "%4d%17d\n", face, frequency[ face ] );
   }
}
```

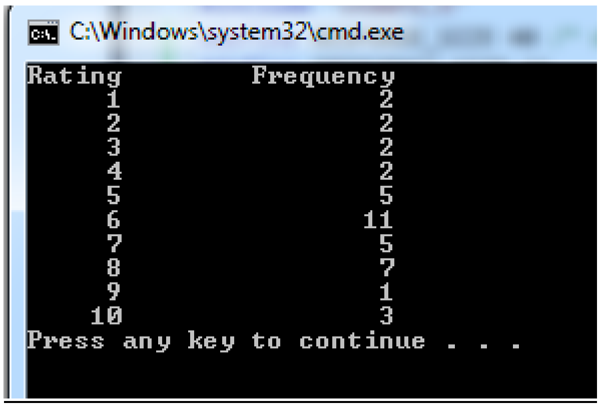
2) Using array to summarize survey results

40 Students were asked to rate the quality of the food in the student cafeteria on a scale of 1 to 10 (1 means bad and 10 means excellent). Place the 40 responses in an integer array and summarize the results of the survey.

- Summarize the number of responses of each type (1 through 10)
- The array responses is a 40-element array of students' responses.
- The array frequency is a 11-element array to count the number of occurrences of each response.

responses array elements are = {1, 2, 6, 4, 8, 5, 9, 7, 8, 10, 1, 6, 3, 8, 6, 10, 3, 8, 2, 7, 6, 5, 7, 6, 8, 6, 7, 5, 6, 6, 5, 6, 7, 5, 6, 4, 8, 6, 8, 10}

Hint: initialize frequency counter to zero. Initialize responses array elements



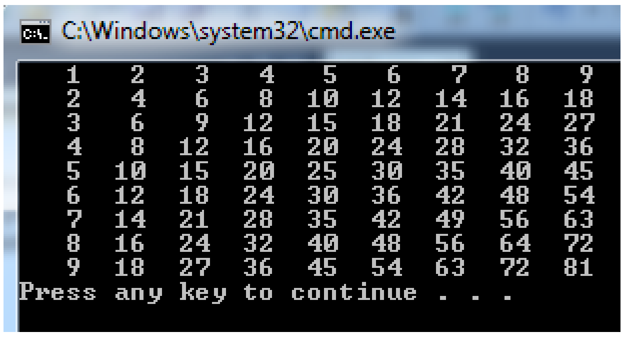
```
Rating          Frequency
     1                  2
     2                  2
     3                  2
     4                  2
     5                  5
     6                 11
     7                  5
     8                  7
     9                  1
    10                  3
Press any key to continue . . .
```

Sol:

```
#include "stdafx.h"
#define RESPONSE_SIZE 40 /* define array sizes */
#define FREQUENCY_SIZE 11
void main()
{
  int answer; /* counter to loop through 40 responses */
  int rating; /* counter to loop through frequencies 1-10 */

  /* initialize frequency counters to 0 */
  int frequency[ FREQUENCY_SIZE ] = { 0 };

  /* place the survey responses in the responses array */
  int responses[ RESPONSE_SIZE ] = { 1, 2, 6, 4, 8, 5, 9, 7, 8, 10,
      1, 6, 3, 8, 6, 10, 3, 8, 2, 7, 6, 5, 7, 6, 8, 6, 7, 5, 6, 6,
      5, 6, 7, 5, 6, 4, 8, 6, 8, 10 };
```

```
   /* for each answer, select value of an element of array responses
         and use that value as subscript in array frequency to
         determine element to increment */
   for ( answer = 0; answer < RESPONSE_SIZE; answer++ ) {
      ++frequency[ responses [ answer ] ];
   }
   printf( "%s%17s\n", "Rating", "Frequency" );

   for ( rating = 1; rating < FREQUENCY_SIZE; rating++ ) {
      printf( "%6d%17d\n", rating, frequency[ rating ] );
   } /* end for */

} /* end main */
```

3) Complete and run the following program. This program creates and prints the multiplication table for 1 to 9



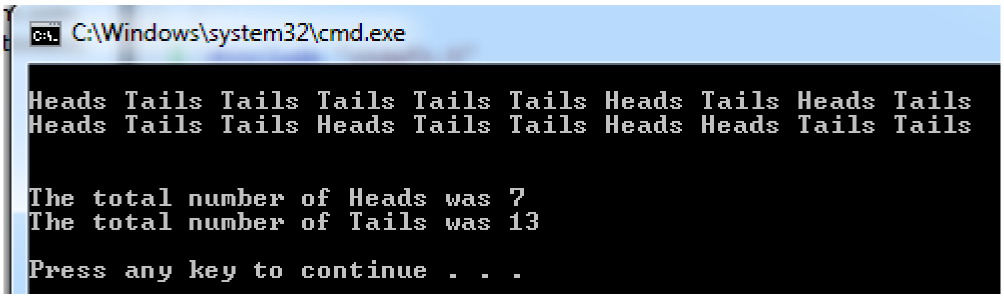
```
   1   2   3   4   5   6   7   8   9
   2   4   6   8  10  12  14  16  18
   3   6   9  12  15  18  21  24  27
   4   8  12  16  20  24  28  32  36
   5  10  15  20  25  30  35  40  45
   6  12  18  24  30  36  42  48  54
   7  14  21  28  35  42  49  56  63
   8  16  24  32  40  48  56  64  72
   9  18  27  36  45  54  63  72  81
Press any key to continue . . .
```

```
#include "stdafx.h"
 #include "stdafx.h"
#define MAX 10
void  main()
{
       int table[____][MAX],row,col;
       //create the table
       for (row = 0; row <= __________ ; row++)
       {
              for (col = _____; col <= ________; ______)
                     ______________ = row * col;
       }

       //print the table
       for (row = 1; row <= MAX - 1; row++)
       {
              for (_________; ____________ ; col++)
                     printf("%4d", _______________ );
              printf("\n");
       }
}
```

Sol:

```
#include "stdafx.h"
#define MAX 10
void  main()
{
	int table[MAX][MAX],row,col;
	//create the table
	for (row = 0; row <= MAX - 1; row++)
	{
		for (col = 0; col <= MAX -1; col++)
			table[row][col] = row * col;
	}

	//print the table
	for (row = 1; row <= MAX - 1; row++)
	{
		for (col = 1; col <= MAX -1; col++)
			printf("%4d", table[row][col] );
		printf("\n");
	}
}
```

4) Write a program that simulates coin tossing. For each toss of the coin the program should print Heads or Tails. Let the program toss the coin 20 times, and count the number of times each side of the coin appears. Print the results. (0 for tails and 1 for heads)


C:\Windows\system32\cmd.exe

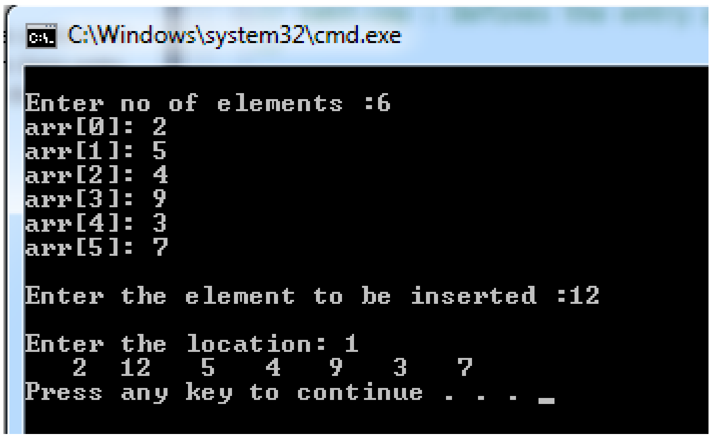
```
Heads Tails Tails Tails Tails Tails Heads Tails Heads Tails
Heads Tails Tails Heads Tails Tails Heads Heads Tails Tails


The total number of Heads was 7
The total number of Tails was 13

Press any key to continue . . .
```

Sol:

```
#include "stdafx.h"
 #include "time.h"
#include "stdlib.h"
#define MAX 20
void main()
{
   int loop;
   int headCount = 0;
   int tailCount = 0;
   int HorT[MAX];
```

```
  srand(time(NULL));

   for ( loop = 0; loop < MAX; loop++ )
   {
     if ( loop % 10 == 0 )
        printf( "\n" );

     HorT[loop] = rand() %2;

     if ( HorT[loop] == 0 )
        {
          printf( "Tails " );
          tailCount++;
        }

     else
     {
         printf( "Heads " );
         headCount++;
     }

   }

   printf( "\n\n\nThe total number of Heads was %d\n", headCount );
   printf( "The total number of Tails was %d\n", tailCount );
   printf("\n");

}
```

5) Complete and run the following program. This program insert an element in an array



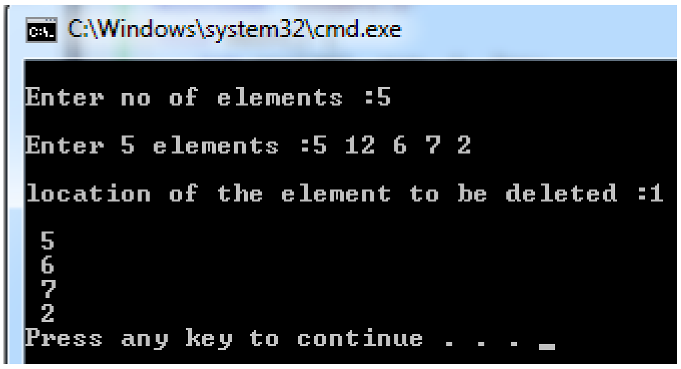
```
Enter no of elements :6
arr[0]: 2
arr[1]: 5
arr[2]: 4
arr[3]: 9
arr[4]: 3
arr[5]: 7

Enter the element to be inserted :12

Enter the location: 1
   2  12   5   4   9   3   7
Press any key to continue . . . _
```

```
#include "stdafx.h"
void main() {
   int arr[30], element, num, i, location;

   printf("\nEnter no of elements :");
   scanf_s("%d", &num);

   for (i = 0; i < num; i++) {
      printf("arr[%d]: ",i);
      scanf_s(___________, _____________); //read values to the array
   }

   printf("\nEnter the element to be inserted :");
   scanf_s("%d", &element);

   printf("\nEnter the location: ");
   scanf_s("%d", &location);

   //Create space at the specified location
   for (i = num; i >= location; i--) {
      arr[_____] = arr[_____]; //move array elements
   }

   ________________; //increment num by one
   arr[location] = element;

   //Print out the result of insertion
   for (i = 0; i < num; i++)
      printf("%4d", ______________); //print array elements
   printf("\n");

}
```

Sol:

```
#include "stdafx.h"
void main() {
   int arr[30], element, num, i, location;

   printf("\nEnter no of elements :");
   scanf_s("%d", &num);

   for (i = 0; i < num; i++) {
      printf("arr[%d]: ",i);
      scanf_s("%d", &arr[i]);
   }

   printf("\nEnter the element to be inserted :");
   scanf_s("%d", &element);

   printf("\nEnter the location: ");
   scanf_s("%d", &location);

   //Create space at the specified location
   for (i = num; i >= location; i--) {
      arr[i] = arr[i - 1];
   }

   num++;
   arr[location] = element;

   //Print out the result of insertion
   for (i = 0; i <=num; i++)
      printf("%4d", arr[i]);
   printf("\n");

}
```

6) Write a c program to delete an element from specified location from array

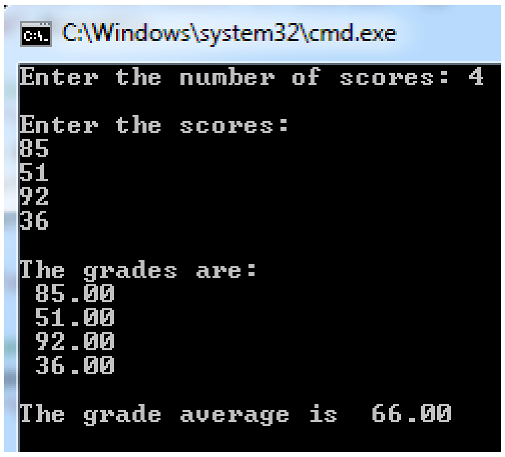


```
Enter no of elements :5

Enter 5 elements :5 12 6 7 2

location of the element to be deleted :1

 5
 6
 7
 2
Press any key to continue . . .
```

Sol:

```
#include "stdafx.h"
void main() {
   int arr[30], num, i, loc;

   printf("\nEnter no of elements :");
   scanf_s("%d", &num);

   //Read elements in an array
   printf("\nEnter %d elements :", num);
   for (i = 0; i < num; i++) {
      scanf_s("%d", &arr[i]);
   }

   //Read the location
   printf("\nlocation of the element to be deleted :");
   scanf_s("%d", &loc);

   loc++;
   /* loop for the deletion  */
   while (loc < num) {
      arr[loc - 1] = arr[loc];
      loc++;
   }
   num--;  // No of elements reduced by 1

   //Print Array
   for (i = 0; i <num; i++)
      printf("\n %d", arr[i]);
   printf("\n");
}
```

7) Complete the following program below by filling in the blanks.



```
Enter the number of scores: 4

Enter the scores:
85
51
92
36

The grades are:
 85.00
 51.00
 92.00
 36.00

The grade average is  66.00
```

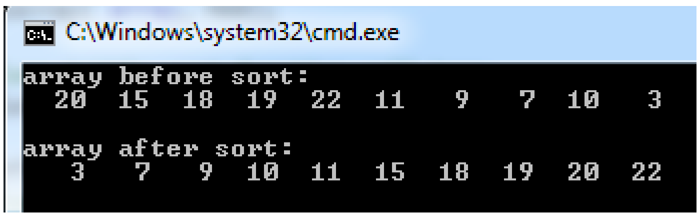
```
#include "stdafx.h"
#define MAXGRADES  5    // maximum number of grades
____________________________________ // function protoype

void main()
{
    int count;                    // actual number of grades
    int i;                        // loop counter
    float grade[ MAXGRADES ];   // array of size MAXGRADES
    float sumofgrades = 0.0;     // sum of array elements

    // prompt the user for the number of scores (less than MAXSCORE)
    printf( "Enter the number of scores: ");
    scanf_s("%d",&count);

    // read in the scores and store them in the array
    printf( "\nEnter the scores: \n");
    for (i = ____; i <= _______ ; i++) {
         scanf_s("%f",& _________ );       // read in grade array element
i
         sumofgrades+=grade[i];    // compute the sumofgrades here
    }
    printf("\n");

    // find and display the average of the scores
    printGrades(_______ , ________ );        // call function printGrades
    printf( "The grade average is  %.2f\n\n\n",sumofgrades / count );
 }

void printGrades(float num[], int size)
{
    printf( "The grades are:\n");
    for (int i = 0; i <= _____________ ; i++)
        printf("%6.2f\n",___________ );
    printf("\n");
}
```

Sol:

```
#include "stdafx.h"
#define MAXGRADES  5    // maximum number of grades
```

```
void printGrades(float num[], int size); // function protoype

void main()
{
    int count;                     // actual number of grades
    int i;                         // loop counter
    float grade[ MAXGRADES ];    // array of size MAXGRADES
    float sumofgrades = 0.0;      // sum of array elements

    // prompt the user for the number of scores (less than MAXSCORE)
    printf( "Enter the number of scores: ");
    scanf_s("%d",&count);

    // read in the scores and store them in the array
    printf( "\nEnter the scores: \n");
    for (i = 0; i <= count-1; i++) {
         scanf_s("%f",& grade[i]);        // read in grade array element i
         sumofgrades+=grade[i];    // compute the sumofgrades here
    }
    printf("\n");

    // find and display the average of the scores
    printGrades(grade, count);          // call function printGrades
    printf( "The grade average is  %.2f\n\n\n",sumofgrades / count );
 }

void printGrades(float num[], int size)
{
    printf( "The grades are:\n");
    for (int i = 0; i <= size - 1; i++)
        printf("%6.2f\n",num[i]);
    printf("\n");
}
```

8) Complete the following code. (Read comments for assistance.)


C:\Windows\system32\cmd.exe

```
array before sort:
  20   15   18   19   22   11    9    7   10    3

array after sort:
   3    7    9   10   11   15   18   19   20   22
```

```
#include "stdafx.h"
#define MAX 10
void bubbleSort(int array[], int size);  // bubble sort function prototype
void swap(_________________);      // helper function prototype of bubbleSort
___________________________________; // print array function prototype

void main()
{

    // Declare an array called value of size MAX with
    //   elements: 20, 15, 18, 19, 22, 11, 9, 7, 10, 3
    _________________________________________________________;
    printf("array before sort: \n");
    // Place print array function call here.
    ________________________________________________________;

    // Place bubble sort function call here.
    ________________________________________________________;

    printf("\n\narray after sort: \n");
    // Place print array function call here.
    ________________________________________________________;
       printf("\n\n\n");
}

void bubbleSort(int array[], int size)
{
    for (int pass = 1; pass <= size - 1; pass++)
        for (int i = 0; i <= size - 2 ; i++)
            if (array[i] > array[i + 1])
               swap(array[i], array[i + 1]);
}

void swap (_____________________)
{
    int temp;
    temp = ____________;
    _________ = _____________;
    ___________ = temp;
}

void printArray( int array[], int size)
{
    for (int i = 0; i <= ______________; i++)
       printf("%4d",_________________) ;
}
```

Sol:

```
#include "stdafx.h"
#define MAX 10
void bubbleSort(int array[], int size);       // bubble sort
void swap(int &x, int &y);     // helper function of bubbleSort
void printArray( int array[], int size); // print array

void main()
{
    // Declare an array called value of size MAX with
    //   elements: 20, 15, 18, 19, 22, 11, 9, 7, 10, 3
       int array[MAX]={20,15,18,19,22,11,9,7,10,3};

    printf("array before sort: \n");
    // Place print array function call here.
       printArray( array, MAX);

    // Place bubble sort function call here.
       bubbleSort(array, MAX);

    printf("\n\narray after sort: \n");
    // Place print array function call here.
       printArray( array, MAX);
       printf("\n\n\n");
}

void bubbleSort(int array[], int size)
{
    for (int pass = 1; pass <= size - 1; pass++)
        for (int i = 0; i <= size - 2 ; i++)
            if (array[i] > array[i + 1])
               swap(array[i], array[i + 1]);
}

void swap (int &x, int &y)
{
    int temp;
    temp = x;
    x = y;
    y = temp;
}

void printArray( int array[], int size)
{
    for (int i = 0; i <= size - 1; i++)
       printf("%4d",array[i]) ;
}
```