

# National University of Computer and Emerging Sciences



## Lab Manual # 8

### Programming Fundamentals

#### (Section BCS-G)

Course Instructor	Mr.Razi Udd Din
Lab Instructor(s)	Miss.Sonia Mr. Muhammad Naveed
Section	BCS-G
Semester	Fall 2021

Department of Computer Science

FAST-NU, Lahore, Pakistan

## Objectives

The objectives of this lab are to cover the following:

- To learn what is array
- Accessing Array Elements
- Array Initialization
- Nested Loop
- Array with Function

## **Array:**

**An array allows you to store and work with multiple values of the same data type.**

The variables you have worked with so far are designed to hold only one value at a time. Each of the variable definitions in causes only enough memory to be reserved to hold one value of the specified data type.

```
int count;    Enough memory for 1 int
              12314

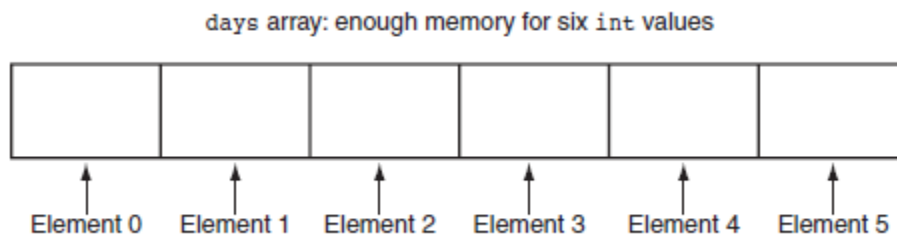
float price;  Enough memory for 1 float
              56.981

char letter;  Enough memory for 1 char
              A
```

An array works like a variable that can store a group of values, all of the same type. The values are stored together in consecutive memory locations. Here is a definition of an array of integers:

```
int days[6];
```

The name of this array is days. The number inside the brackets is the array's size declarator. It indicates the number of elements, or values, the array can hold. The day's array can store six elements, each one an integer.



An array's size declarator must be a constant integer expression with a value greater than zero. It can be either a literal, as in the previous example, or a named constant, as shown in the following:

```
const int NUM_DAYS = 6;  
int days[NUM_DAYS];
```

**Arrays of any data type can be defined. The following are all valid array definitions:**

```
float temperatures[100];      // Array of 100 floats  
string names[10];             // Array of 10 string objects  
long units[50];               // Array of 50 long integers  
double sizes[1200];           // Array of 1200 doubles
```

## Memory Requirements of Arrays

Array Definition	Number of Elements	Size of Each Element	Size of the Array
<code>char letters[25];</code>	25	1 byte	25 bytes
<code>short rings[100];</code>	100	2 bytes	200 bytes
<code>int miles[84];</code>	84	4 bytes	336 bytes
<code>float temp[12];</code>	12	4 bytes	48 bytes
<code>double distance[1000];</code>	1000	8 bytes	8000 bytes

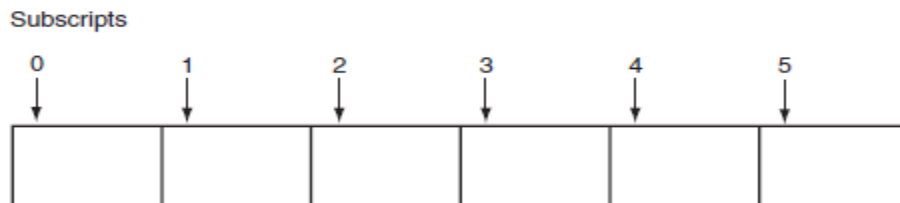
## Accessing Array Elements

The individual elements of an array are assigned unique subscripts. These subscripts are used to access the elements.

Consider array

```
int hours[6] ;
```

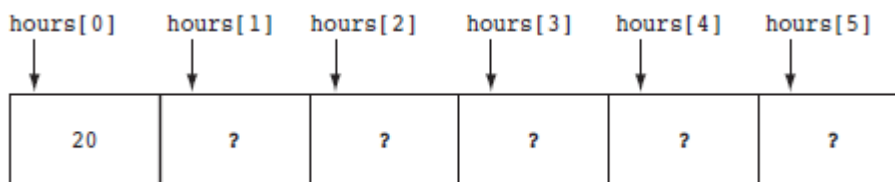
Even though an entire array has only one name, the elements may be accessed and used as individual variables. This is possible because each element is assigned a number known as a **subscript**. A subscript is used as an index to pinpoint a specific element within an array. The first element is assigned the subscript 0, the second element is assigned 1, and so forth. The six elements in the array `hours` would have the subscripts 0 through 5.



Each element in the `hours` array, when accessed by its subscript, can be used as a short variable. Here is an example of a statement that stores the number 20 in the first element of the array:

```
hours[0] = 20;
```

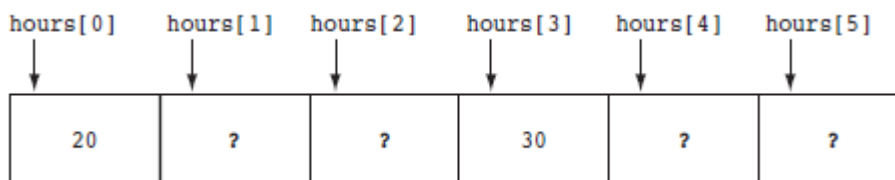
The contents of the array `hours` after the statement assigns 20 to `hours[0]`.



The following statement stores the integer 30 in `hours[3]`.

```
hours[3] = 30;
```

The contents of the array after the previous statement executes:



## Inputting and Outputting Array Contents

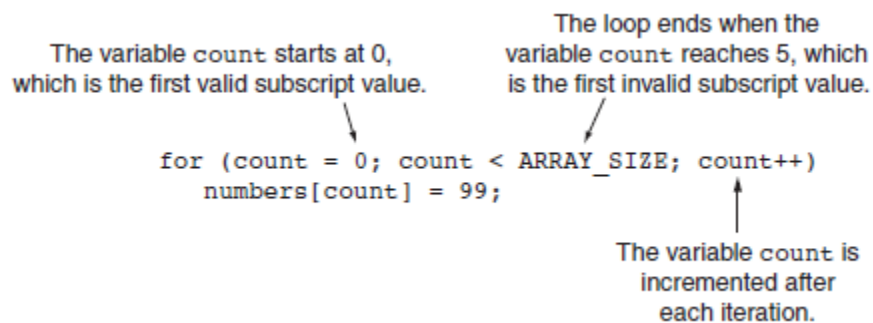
Array elements may be used with the cin and cout objects like any other variable. Program shows the array hours being used to store and display values entered by the user.

```
const int ARRAY_SIZE = 5;
```

```
int numbers[ARRAY_SIZE];
```

```
for (int count = 0; count < ARRAY_SIZE; count++)  
    numbers[count] = 99;
```

This code first defines a constant int named ARRAY\_SIZE and initializes it with the value 5. Then it defines an int array named numbers, using ARRAY\_SIZE as the size declarator. As a result, the numbers array will have five elements. The for loop uses a counter variable named count. This loop will iterate five times, and during the loop iterations the count variable will take on the values 0 through 4.



## Sample Program

```
#include <iostream>
using namespace std;
int main()
{
    const int NUM_EMPLOYEES = 6;    // Number of employees
    int hours[NUM_EMPLOYEES];      // Each employee's hours
    int count;                     // Loop counter

    // Input the hours worked.
    for (count = 0; count < NUM_EMPLOYEES; count++)
    {
        cout << "Enter the hours worked by employee " << (count + 1) << ": ";
        cin >> hours[count];
    }

    // Display the contents of the array.
    cout << "The hours you entered are:";
    for (count = 0; count < NUM_EMPLOYEES; count++)
        cout << " " << hours[count];
    cout << endl;
    return 0;
}
```

## Output

Enter the hours worked by employee 1: **20** [Enter]

Enter the hours worked by employee 2: **12** [Enter]

Enter the hours worked by employee 3: **40** [Enter]

Enter the hours worked by employee 4: **30** [Enter]

Enter the hours worked by employee 5: **30** [Enter]

Enter the hours worked by employee 6: **15** [Enter]

The hours you entered are: 20 12 40 30 30 15

## Arrays as Function Arguments

To pass an array as an argument to a function, pass the name of the array.

Quite often you'll want to write functions that process the data in arrays. For example, functions could be written to put values in an array, display an array's contents on the screen, total all of an array's elements, or calculate their average. Usually, such functions accept an array as an argument.

This program demonstrates the showValues function being used to display the contents of two arrays.

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

void showValues(int [], int);      // Function prototype
int main()
{
    const int SIZE1 = 8; // Size of set1 array
    const int SIZE2 = 5; // Size of set2 array
    int set1[SIZE1] = {5, 10, 15, 20, 25, 30, 35, 40};
    int set2[SIZE2] = {2, 4, 6, 8, 10};

    // Pass set1 to showValues.
    showValues(set1, SIZE1);

    // Pass set2 to showValues.
    showValues(set2, SIZE2);
    return 0;
}

void showValues(int nums[], int size)
{
    for (int index = 0; index < size; index++)
        cout << nums[index] << " ";
    cout << endl;
}
```

### Program Output:

```
5 10 15 20 25 30 35 40
2 4 6 8 10
```

## Problems

### Question#1

Write a program that lets the user enter 10 values into an array. The program should then display the largest and smallest values stored in the array.

### Question#2

Write a program that inputs salary and grade passes to the function by reference. The function adds the 50% bonus if the grade is greater than 15. It adds 25% bonus if the grade is 15 or less and calculate the total salary return back to the main function.

### Question#3

Write a program that takes 10 element in array passing as a parameter to function. The function find out the unique elements from array.

### Question#4

Write a program that takes 10 element in array passing as a parameter to function as well as a number N. The function find the number of pairs of integers in a given array of integers whose sum is equal to a specified number N entered by the user. If you find that pair then show the pair with corresponding indexes.

### Question#5

Write a function that will receive N as argument and print a hollow triangle as given in examples below:

** *	*** ** *	**** * * ** *	***** * * * * ** *	***** * * * * * * ** *
N = 2	N=3	N=4	N=5	N=6

These are just sample examples. Your program must work for any input.

**END**