NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES

CS 1004– Object Oriented Programing Lab

LAB Instructors: Saeeda Kanwal “saeeda.kanwal@nu.edu.pk”

Lab 01

Outline

Introduction To Object Oriented Programming

- Object oriented programming is a way of solving complex problems by breaking them into smaller problems using objects.
- Before Object Oriented Programming (commonly referred as OOP), programs were written in procedural language, they were nothing but a long list of instructions. On the other hand, the OOP is all about creating objects that can interact with each other, this makes it easier to develop programs in OOP

Headers File in- C and C++

In C++, all the header files may or may not end with the .h extension but in C, all the header files must necessarily begin with the.h extension.

- A header file in C/C++ contains:
- Function definitions
- Data type definitions

1. Standard library header files: These are the pre-existing header files already available in the C/C++ compiler.
2. User-defined header files: Header files starting #define can be designed by the user.

- ***#include<string.h>***
- ***#include<stdio.h>***
- ***#include<iostream>***
- ***#include<factorial.h>***

The #include is a preprocessor command that tells the compiler to include the contents

Using Namespace std

- Namespaces allow to group entities like classes, objects and functions under a name. This way the global scope can be divided in "sub-scopes", each one with its own name.

- ***An Example C++ Program***

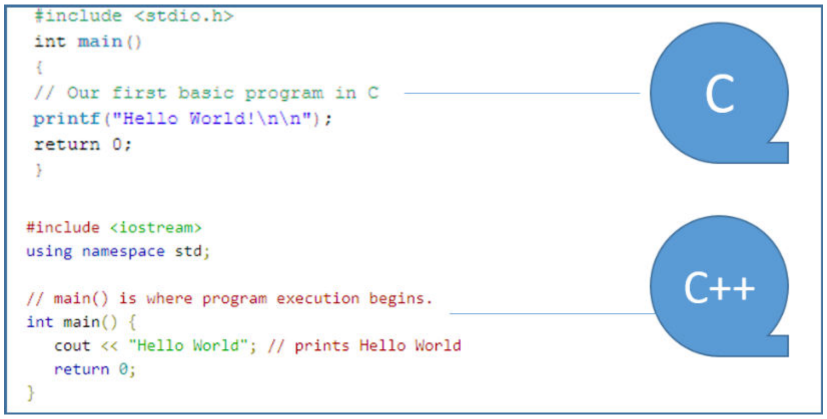
  - #include <iostream>
  - using namespace std;
  - namespace first {
  - int var = 5;
  - }
  - namespace second {
  - int double = 3.1416;
  - }
  - int main () {
  - first::var;
  - second::double();
  - return 0;
  - }

Basic I/O Function

- Every C program should necessarily contain the header file <stdio.h> which stands for standard input and output used to take input with the help of scanf() function and display the output using printf() function.
- C++ program should necessarily contain the header file <iostream> which stands for input and output stream used to take input with the help of "cin>>" function and display the output using "cout<<" function.

Difference between C vs C++

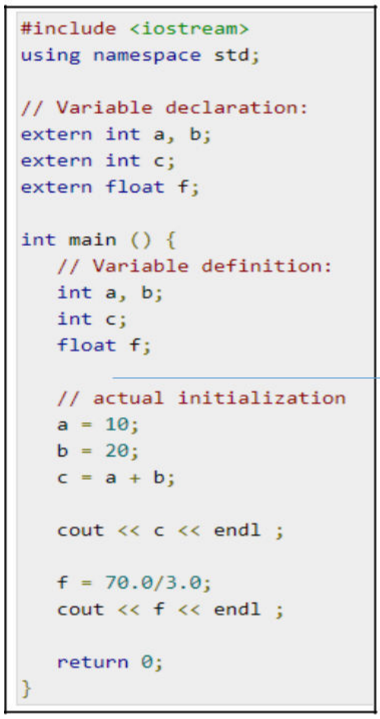
- The main difference between both these languages is C is a procedural programming language and does not support classes and objects, while C++ is a combination of both procedural and object-oriented programming languages



```
#include <stdio.h>
int main()
{
// Our first basic program in C
printf("Hello World!\n\n");
return 0;
}
```

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

// main() is where program execution begins.
int main() {
   cout << "Hello World"; // prints Hello World
   return 0;
}
```

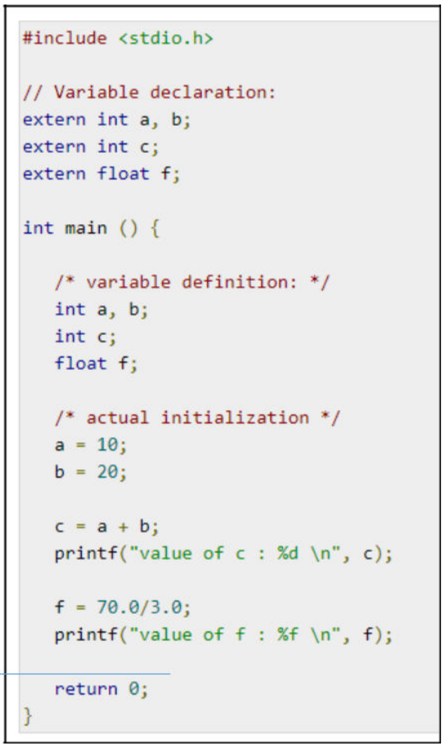
Variable Declaration in C vs C++



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

// Variable declaration:
extern int a, b;
extern int c;
extern float f;

int main () {
   // Variable definition:
   int a, b;
   int c;
   float f;

   // actual initialization
   a = 10;
   b = 20;
   c = a + b;

   cout << c << endl ;

   f = 70.0/3.0;
   cout << f << endl ;

   return 0;
}
```

C++

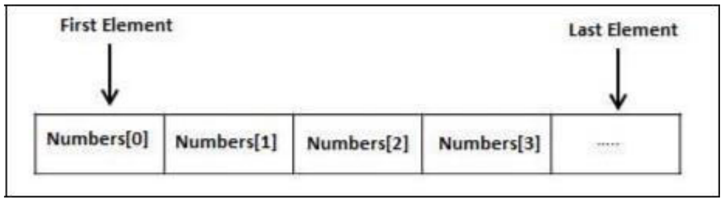
C



```
#include <stdio.h>

// Variable declaration:
extern int a, b;
extern int c;
extern float f;

int main () {

   /* variable definition: */
   int a, b;
   int c;
   float f;

   /* actual initialization */
   a = 10;
   b = 20;

   c = a + b;
   printf("value of c : %d \n", c);

   f = 70.0/3.0;
   printf("value of f : %f \n", f);

   return 0;
}
```

Array

- Arrays a kind of data structure that can store a fixed-size sequential collection of elements of the same type.



Types

- One-dimensional array – Vectors
- Two-dimensional array – Matrix

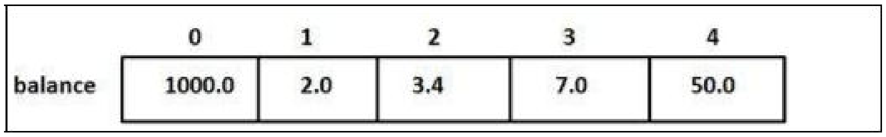
Declaring Arrays

```
type arrayName [arraySize];
```

```
double balance[10];
```

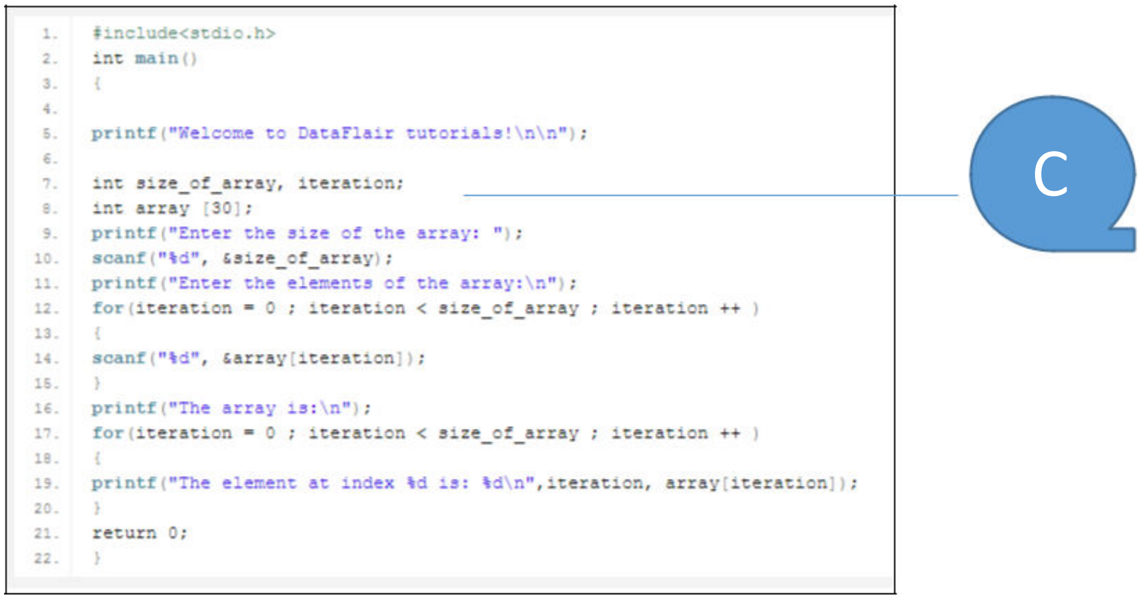
Initializing Arrays

```
double balance[5] = {1000.0, 2.0, 3.4, 7.0, 50.0};
```

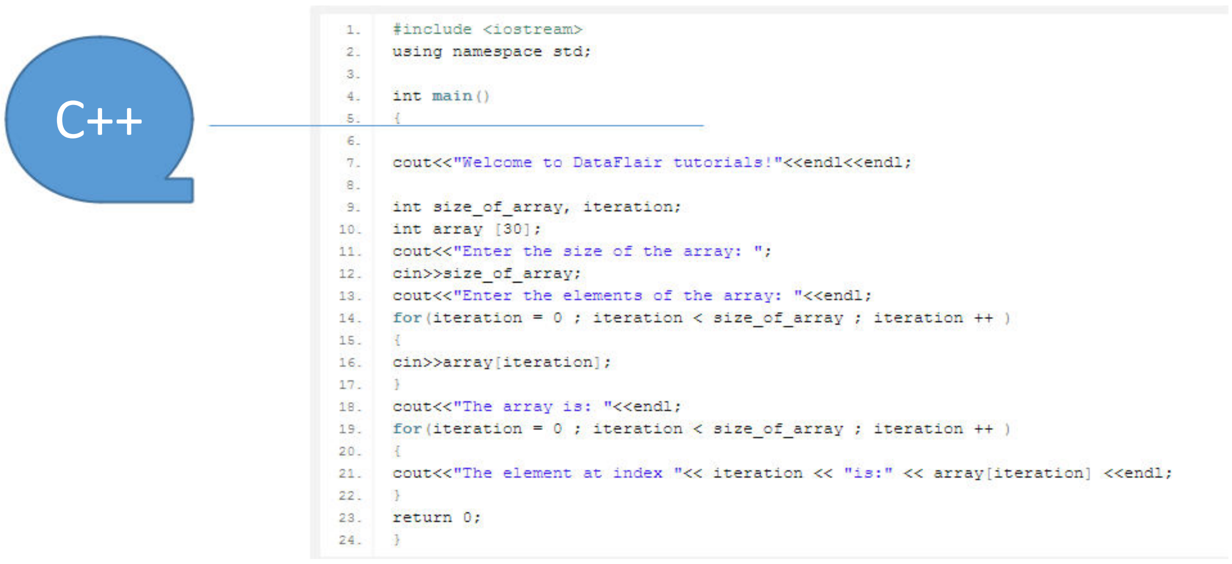


Accessing Array- C vs C++

double salary = balance [3]
Output = 7.0



```c
#include<stdio.h>
int main()
{

printf("Welcome to DataFlair tutorials!\n\n");

int size_of_array, iteration;
int array [30];
printf("Enter the size of the array: ");
scanf("%d", &size_of_array);
printf("Enter the elements of the array:\n");
for(iteration = 0 ; iteration < size_of_array ; iteration ++ )
{
scanf("%d", &array[iteration]);
}
printf("The array is:\n");
for(iteration = 0 ; iteration < size_of_array ; iteration ++ )
{
printf("The element at index %d is: %d\n",iteration, array[iteration]);
}
return 0;
}
```



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

int main()
{

cout<<"Welcome to DataFlair tutorials!"<<endl<<endl;

int size_of_array, iteration;
int array [30];
cout<<"Enter the size of the array: ";
cin>>size_of_array;
cout<<"Enter the elements of the array: "<<endl;
for(iteration = 0 ; iteration < size_of_array ; iteration ++ )
{
cin>>array[iteration];
}
cout<<"The array is: "<<endl;
for(iteration = 0 ; iteration < size_of_array ; iteration ++ )
{
cout<<"The element at index "<< iteration << "is:" << array[iteration] <<endl;
}
return 0;
}
```

POINTER

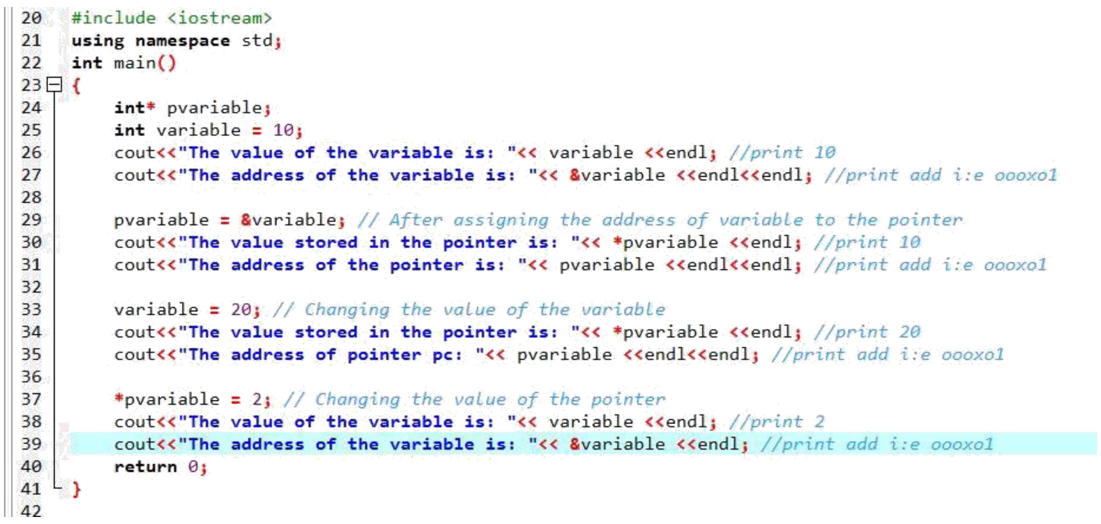
- Pointer in C and C++ is nothing but a way to access a variable by storing its memory location. In programming terminology, A pointer is simply a variable that stores the memory location of another variable.
- Pointers hold data and its reference.

int number = 8;
Memory location = 4572

We can do it in 2 ways:

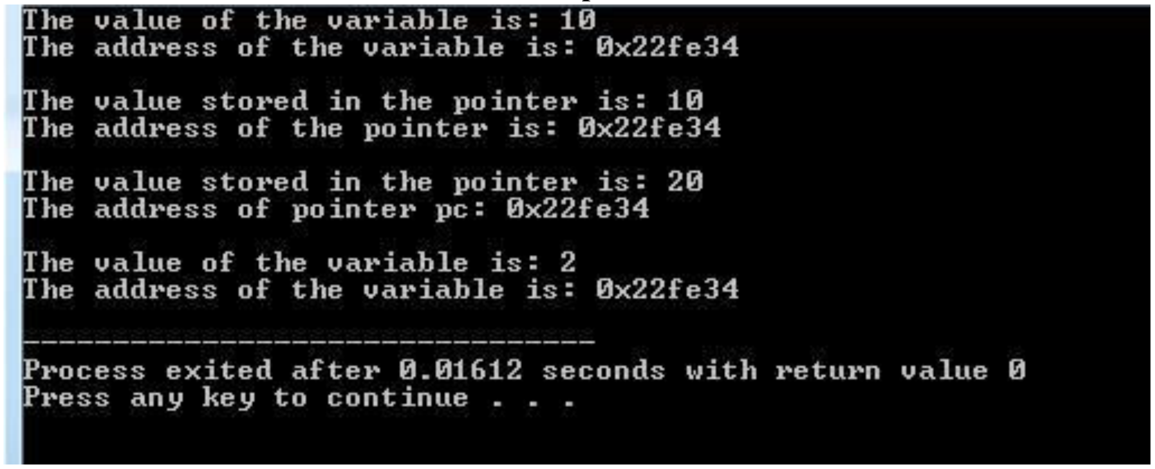
- data_type *pointer: Here, the dereference operator is placed just before the identifier.
- data_type* pointer: Here, the dereference operator is placed after the data type of the pointer.

Program as Example



```
#include <iostream>
using namespace std;
int main()
{
    int* pvariable;
    int variable = 10;
    cout<<"The value of the variable is: "<< variable <<endl; //print 10
    cout<<"The address of the variable is: "<< &variable <<endl<<endl; //print add i:e oooxo1

    pvariable = &variable; // After assigning the address of variable to the pointer
    cout<<"The value stored in the pointer is: "<< *pvariable <<endl; //print 10
    cout<<"The address of the pointer is: "<< pvariable <<endl<<endl; //print add i:e oooxo1

    variable = 20; // Changing the value of the variable
    cout<<"The value stored in the pointer is: "<< *pvariable <<endl; //print 20
    cout<<"The address of pointer pc: "<< pvariable <<endl<<endl; //print add i:e oooxo1

    *pvariable = 2; // Changing the value of the pointer
    cout<<"The value of the variable is: "<< variable <<endl; //print 2
    cout<<"The address of the variable is: "<< &variable <<endl; //print add i:e oooxo1
    return 0;
}
```

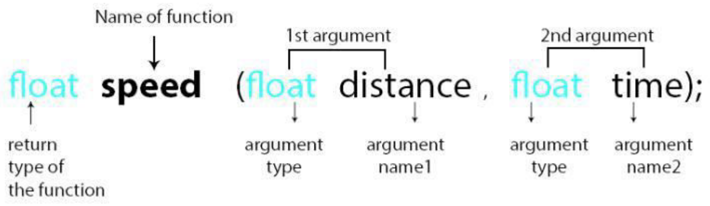
Output



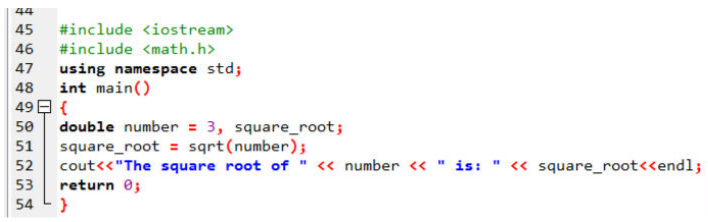
```
The value of the variable is: 10
The address of the variable is: 0x22fe34

The value stored in the pointer is: 10
The address of the pointer is: 0x22fe34

The value stored in the pointer is: 20
The address of pointer pc: 0x22fe34

The value of the variable is: 2
The address of the variable is: 0x22fe34

--------------------------------
Process exited after 0.01612 seconds with return value 0
Press any key to continue . . .
```

Functions

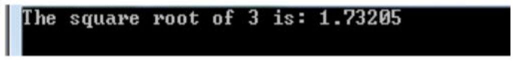
- **Functions give you the provision to divide your code into fragments to reduce code complexity.**



Program as example



```
#include <iostream>
#include <math.h>
using namespace std;
int main()
{
double number = 3, square_root;
square_root = sqrt(number);
cout<<"The square root of " << number << " is: " << square_root<<endl;
return 0;
}
```



```
The square root of 3 is: 1.73205
```

Activity

First Task:

- Write the program that will print your name and roll number
- Write the program of adding two numbers.
    - The numbers must be given by user during run time.

Second Task:

- Write a program that can print first 10 numbers.
- Write a program that can print up to n.
    - The number n must be given by user on run time.
- Write a program that can print from a to b.
    - The number a and b must be given by user on run time.

Third Task:

- Write a program having following functionality using functions.
    - Addition
    - Subtraction
    - Multiplication
    - Division
- Hint. You must use different function for each functionality**.**