**iostream** is the header file which contains all the functions of program like cout, cin etc.

and #**include** tells the preprocessor to **include** these header file in the program.

**include** <**iostream**.h> #**include** is known as a preprocessor directive,which is used to load files

**conio**. **h** is a C header file used mostly by MS-DOS compilers to provide console input/output. ... This header declares several useful library functions for performing "console input and output" from a program.

In **C++** it is optional to type "**return 0**;" at the end of the main function and the compiler includes it automatically.

When you **return 0** or **return** EXIT\_SUCCESS , you tell the caller (OS in case of main) that the status of your program was successful and it could **do** what it was supposed to **do**.

In C and C++ **int main**(**void**) means that the function takes NO arguments. C++ does the same even if you simply write **int main**() , which says the function does not take any argument, and if you dont put **void** in it, it'll be the same, but in C there's a little difference in **int main**() and **int main**(**void**) .

**int main**(): **main** is a function **with a** special characteristic that the program execution always start from **main**. So the function **main** needs arguments and a return type. These **int** and **void** are its return type. **Void** means it will not return any value, which is also okay.

n C and C++ programs the main function is of type int and therefore it should **return** an integer value. The **return** value of the main function is considered the "Exit Status" of the application. On most operating systems **returning 0** is a success status like saying "The program worked fine". ... In the end of main function.

**Using namespace std:**

So when we run a program to **print something**, “using namespace std” says if you find something that is not declared in the current scope go and check std.

So now you have the answer why both statements

#include <iostream>  
using namespace std;

are used. It is because computer needs to know the code for the cout, cin functionalities and it needs to know which namespace they are defined.

#include<iostream>

using namespace std;

int main()

{

cout<<"\*\*\*\*\*\*\*\*width of data type keyword\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

cout<<"Size of integer\_int\t="<<sizeof(int)<<" bytes"<<endl;

cout<<"Size of character\_char\t="<<sizeof(char)<<" bytes"<<endl;

cout<<"Size of float\t="<<sizeof(float)<<" bytes"<<endl;

cout<<"Size of double\t="<<sizeof(double)<<" bytes"<<endl;

cout<<"Size of string\t="<<sizeof(string)<<" bytes"<<endl;

cout<<"Size of valueless\_void\t="<<sizeof(void)<<" bytes"<<endl;

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//swapping two number

#include<iostream>

using namespace std;

int main()

{

int x,y,temp; //local variable initialize

cout<<"Enter the value of x and y"<<endl; //print the statement

cin>>x; //get the user value

cin>>y;

cout<<"swapping before x ="<<x<<endl;

cout<<"swapping before y ="<<y<<endl;

temp=x; //temp variable initialize x

x=y; //x initialize the value of y

y=temp; //y is initialize temp while temp==x.

cout<<"after swapping :"<<endl;

cout<<"x ="<<x<<endl;

cout<<"y ="<<y<<endl;

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#include<iostream>

using namespace std;

int g;

int main() //global variable,will hold their value throughout the life-time of our program.

{

int a,b; //local variable

a=10,b=20;

g=a+b;

cout<<g;

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#include<iostream>

using namespace std;

int main()

{

int a=4; //initialze the vlaue

int b (5); // also declare the value in this scope

int result;

cout<<"welcome to c++"<<endl;

result=a+b;

cout<<"result\t="<<result<<endl;

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#include<iostream>

#include<string>

using namespace std;

int main()

{

cout<<"Cin with Strings\n"<<endl;

string mystr;

cout<<"What is you Name:\t";

getline (cin,mystr);

cout<<"Welcome\t"<<mystr<<"\n";

cout<<"what is your favorite team ?\t"<<"\n";

getline(cin,mystr);

cout<<"I Like\t"<<mystr<<"\ttoo\n";

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#include<iostream>

#include<string>

using namespace std;

int main()

{

cout<<"Cin with Strings\n"<<endl;

string mystr; //string variable mystr

cout<<"What is your Name:\t";

getline (cin,mystr); //from user input, "Zeeshan Ali".

cout<<"Welcome\t"<<mystr<<"\n"; //Welcome <<Zeeshan Ali.

cout<<"what is your favorite team ?\t"<<"\n"; //print the statement.

getline(cin,mystr); //cin, get the user answer,in string form.

cout<<"I Like\t"<<mystr<<"\ttoo\n"; //I like "string name,favourite name" too.

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#include<iostream>

#include<string>

#include<sstream>

using namespace std;

int main()

{

string mystr;

float price=0;

int quantity=0;

cout<<"Enter Price:";

getline(cin,mystr);

stringstream(mystr)>>price;

cout<<"Enter Quantity:";

getline(cin,mystr);

stringstream(mystr)>>quantity;

cout<<"total Price:"<<price\*quantity<<endl;

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#include<iostream>

#include<string>

#include<sstream>

using namespace std;

int main()

{

int n;

cout<<"Enter the Strarting Number :"<<endl;

cin>>n;

while(n >0) //if vale is true ,n>0 while condition will be executed.

{

cout<<n<<","; //if we write endl at the end of statement then each digit will show on new line.

n=n-1; //decrement operator.

}

cout<<"FIRE!\n";

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#include<iostream>

using namespace std;

//#include<string>

//#include<sstream>

using namespace std;

int main()

{

int n;

for (n=20;n>10;n--)

{

cout<<n<<",";

}

cout<<"FIRE!";

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#include<iostream>

using namespace std;

int main()

{

for (int n=11;n>0;n--) //for loop condition.

{

if(n==2)

continue; //for skip the value in loop condition we use continue statement.

if(n==4)

continue; //for skip the value in loop condition we use continue statement.

if(n==6)

continue; //for skip the value in loop condition we use continue statement.

if(n==8)

continue; //for skip the value in loop condition we use continue statement.

if(n==10 )

continue; //for skip the value in loop condition we use continue statement.

cout<<n<<",";

}

cout<<"Odd Numbers:";

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Function program Addition of three numbers:

#include<iostream>

using namespace std;

int addition(int a,int b,int c)

{

int n;

n=a+b+c;

return (n);

}

int main()

{

int z;

z =addition (5,5,10);

cout<<"The Result is "<<z;

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//Function example

#include<iostream>

using namespace std;

int subtraction(int a,int b)

{

int r;

r=a-b;

return (r);

}

int main()

{

int x=5,y=3, z;

z=subtraction(7,2);

cout<<"The 1st Result = "<<z<<endl; //subtraction z value

cout<<"The 2nd Result ="<<subtraction (x,y)<<endl; //subtract x into y.

cout<<"The 3rd Result ="<<5\*(z)<<endl; //multiply the result z

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//more than one returning

#include<iostream>

using namespace std;

void previous\_next(int x,int& previous,int& next)

{

previous=x-1;

next=x+1;

}

main()

{

int x=100 ,v,w;

previous\_next(x,v,w);

cout<<"Previous\t"<<v<<endl;

cout<<"next \t"<<w<<endl;

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//more than one returning

#include<iostream>

using namespace std;

int divide (int a,int b=2)

{

int r;

r=a/b;

return (r);

}

int main()

{

int n;

n=divide(12); // 12/b=2

cout<<n<<endl;

cout<<divide (20)<<endl; // 20/b=2

cout<<divide (20,4)<<endl; //in this statement 20/4=5 will show and ignore the argument value of b.

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#include<iostream>

#include<string>

using namespace std;

int main()

{

string mystring="This is string";

string mystring\_1 ("This is also my string");

cout<<mystring<<endl;

cout<<mystring\_1;

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## Defining Constants

There are two simple ways in C++ to define constants −

* Using **#define** preprocessor.
* Using **const** keyword.

## The #define Preprocessor

Following is the form to use #define preprocessor to define a constant −

#define identifier value

Following example explains it in detail −

[Live Demo](http://tpcg.io/N6xxDP)

#include <iostream>

using namespace std;

#define LENGTH 10

#define WIDTH 5

#define NEWLINE '\n'

int main() {

int area;

area = LENGTH \* WIDTH;

cout << area;

cout << NEWLINE;

return 0;

}

When the above code is compiled and executed, it produces the following result −

50

## The const Keyword

You can use **const** prefix to declare constants with a specific type as follows −

const type variable = value;

Following example explains it in detail −

#include <iostream>

using namespace std;

int main() {

const int LENGTH = 10;

const int WIDTH = 5;

const char NEWLINE = '\n';

int area;

area = LENGTH \* WIDTH;

cout << area;

cout << NEWLINE;

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Simple Math program**

#include<iostream>

#include<cmath>

using namespace std;

int main(){

cout<<sqrt(64)<<"\n"; //square root of 64 is 8

cout<<round(2.6)<<"\n";

cout<<log(2)<<"\n";

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//this program test that if the Amer is greater than Ammara age and displays the results.

#include<iostream>

using namespace std;

main()

{

double amount=0,discount=0,netpayable=0;

cout<<"please enter the amount of the bill";

cin>>amount;

//test the conditions and claculate net payable

if(amount>5000)

{

//calculate amount at 15% discount

discount=amount\*15.0/100;

netpayable=amount-discount;

cout<<"The discount at the rate of 15% is Rupees="<<discount<<endl;

cout<<"The netpayable amount of 15% is Rupees="<<netpayable<<endl;

}

else

{//calculate amount at the rate 10% discount

discount=amount\*10.0/100;

netpayable=amount-discount;

cout<<"The discount at the rate of 10% is Rupees="<<discount<<endl;

cout<<"The netpayable amount of 10% is Rupees="<<netpayable<<endl;

}

return 0;

}

#include<iostream> //simple addition program

using namespace std;

int main(){

int x=10,y=20;

cout << "addition is ="<< x+y<<endl;

}

#include<iostream>

using namespace std;

int main(){

int x=42,y=20;

cout << "addition is ="<< x+y <<endl; //output is 62

cout << "Division is ="<< x/y <<endl; //

cout << "subtraction is ="<< x-y <<endl;

cout << "reminder is ="<< x%y <<endl;

}

#include<iostream>

using namespace std;

int main(){

int x=20,y=19;

if(x>y) //if statement program//

{

cout<<"x is greater than y"<< endl; //output will be 1st, if condition true.

}

if(x<y)

{

cout<<"x is less then y"<<endl;

}

}

#include<iostream>

using namespace std; //output will be blank due to compiler is not understand

int main()

{

int x=20,y=19;

if(x==y)

{

cout<<"x is greater than y"<< endl;

}

if(x<y)

{

cout<<"x is less then y"<<endl; //condition is true,x is less then y printed.

}

}

#include<iostream>

using namespace std; // if else program ,analyse the if and else program//

int main(){

int x=100,y=101;

if(x>y){

cout<<"x is Greater than y"<<endl; //true,x is greater than y.

}else

{

cout<<"y is Greater than x"<<endl;

}

return 0;

}

#include<iostream>

using namespace std;

int main(){

int x=20,y=30,c=30;

if(x>y && x> c){ // x is not greater than y and c.//& and OR operator is used

cout<<"x is Greater than y and c"<<endl;

}else

{

cout<<"x is not Greater than y and c"<<endl; //ture ,printed

}

}

#include<iostream> //preprocessor directive input/output stream

using namespace std; //standard library built infunction

int main() //starting the program body

{ //block of the program

int x=20,y=30,c=10; //integer inilization

if(x>y || x> c){ //OR logical operator is used in this program

cout<<"x is Greater than y and c "<<endl; //output true, x is greater than y and c

}else

{

cout<<"x is not Greater than y and c "<<endl;

}

}

#include<iostream>

using namespace std;

int main(){

int rollnumber=579;

switch(rollnumber) //switch is used to avoid the repetition of if/else statements

case 579: // in case condition “:” is always is used

cout<<"Rollnumber is 579 "<<endl;

// break is used ,if the condition is true it will stop and goes into switch statement//

}

#include<iostream>

using namespace std;

int main(){

int rollnumber=101;

switch(rollnumber) **//switch is used to avoid the repition of if/else statements**

case 101:

cout<<"Rollnumber is 101 "<<endl; // if rollnumber=101,it will execute and break

break; //if value is true then stop ,no need to check the condition one by one

case 102:

cout<<"Rollnumber is 102"<<endl;

break;

case 103:

cout<<"Rollnumber is 103"<<endl;

break;

default:

cout<<"Roll number is not exist"<<endl;

break;

// break is used ,if the condition is true it will stop and goes into switch statement//

}

#include<iostream>

using namespace std;

int main(){

int i=1; //initial condition of i//

while(i<10) //loop statement 1 to 9//

{ // for print the value of 1 to 9 in sequence not printed 10//

cout<<i<<endl;

i++; //i=i+1//increment operator

}

return 0;

}

#include<iostream>

using namespace std;

int main(){

int i=1;

while(i<10) //if we write while(i<=10),it will printed 1 to 10 value.

{ // for print the value of 1 to 9 in sequence not printed 10//

cout<<i<<endl;

i++; //i=i+1//increment operator

}

cout<<"out of loop condition"<<endl;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#include<iostream>

using namespace std;

int main(){

int i=1;

while(i>=10) //false condition,print “out of loop condition”

{ // for print the value of 1 to 9 in sequence not printed 10//

cout<<i<<endl;

i++; //i=i+1//increment operator

}

cout<<"out of loop condition"<<endl;

}

#include<iostream>

using namespace std;

int main(){

int i=101;

do // check the value of i ,again check till 101

{

cout<<i<<endl;

i++; //i=i+1//increment operator

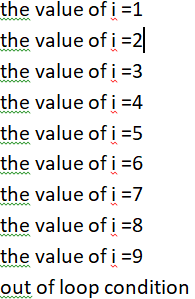
}

While (i<=50) ; //print the value of while condition//

cout<<"out of loop condition"<<endl;

}

**For loop condition**

#include<iostream>

using namespace std;

int main(){

//if ,for(int i=1 ; i<=10 ; i++) output printed 1to10//

for(int i=1;i<10;i++) //for loop is used to avoid the long term condition of while and do while condition//

{

cout<<"the value of i ="<< i <<endl;

}

cout<<"out of loop condition"<<endl;

}

Simple array program

#include<iostream>

using namespace std;

int main(){

int myarray[]= {10,20,30,40}; //array condition,if we want to write integer value in array then we wrie only this condition myarray[]={1,2,3,4,.....infinite} //

cout<<myarray[]<<endl; //output will be 0 because index value is not assigned//

-----------------------------------------------------------------------------------------------------------------------------------------

#include<iostream>

using namespace std;

int main(){

//index of 10 is assigned 0,20 of index is 1,30 of index is 2,40 of index is 3

int myarray[4]= {10,20,30,40}; //array condition,if we want to write integer value in array then we wrie only this condition myarray[]={1,2,3,4,.....infinite} //

cout<<myarray[3]<<endl; //output will be printed 40 due to [4] index value is assigned in myarray[3] //

array new program condition

#include<iostream>

using namespace std;

int main(){

int myarray[]= {10,20,30,40};

myarray[3]=50; //new value is assigned in [3],index of 3=50

myarray[2]=60; //new value is assigned in [2],indexof 2=60//

cout<<myarray[3]<<endl; // array condition will be printed on out=50//

cout<<myarray[2]<<endl; // array condition will be printed on out=50//

}

#include<iostream>

using namespace std;

int main(){

int myarray[4]= {10,20,30,1000};

for(int i = 0; i<4 ; i++)

{

cout<<"array values are ="<< myarray[i]<<endl; //array values are printed in myarray[i],(10,20,30,1000)

}

}

Covered /////////Function lecture

#include<iostream>

using namespace std;

main()

{

int myarray[3][2]={{10,20}, //2D array program//

{30,40},

{50,60}

};

for(int row=0;row<3;row++)

{

for (int col=0;col<2;col++)

{

cout<<myarray[row][col]<<" ";

}

cout<<endl;

}

}

#include<iostream>

using namespace std;

//prototype print method program

void print(){

cout<<"i am a print method"<<endl;

}

int main() {

print();

print();

print();

print();

}

#include<iostream>

using namespace std;

void print(int x); //memory address operator//

int main()

{

int y=10;

cout<<&y<<endl;

}

//constructor is used to initialize the value.

#include<iostream>

#include<string>

using namespace std;

class myclass

{

public:

myclass();

};

myclass::myclass(){

cout<<"i'am constructor"<<endl;

}

int main(){

myclass m;

}

//desconstructor is used to destroy the garbage value,in this method memeory space is decreases and program is worked in efficient way.

#include<iostream>

#include<string>

using namespace std;

class myclass

{

public:

myclass();

~myclass();

};

myclass::myclass(){

cout<<"i am a constructor"<<endl;

}

myclass::~myclass(){

cout<<"i am desconstructor"<<endl;

}

int main(){

myclass m;

}

W3school program data

Variables are containers for storing data values.

In C++, there are different **types** of variables (defined with different keywords), for example:

* int - stores integers (whole numbers), without decimals, such as 123 or -123
* double - stores **floating point** numbers, with decimals, such as 19.99 or -19.99
* char - stores single characters, such as 'a' or 'B'. Char values are surrounded by single quotes
* string - stores text, such as "Hello World". String values are surrounded by double quotes
* bool - stores values with two states: true or false

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

int myNum = 5;               // Integer (whole number without decimals)  
double myFloatNum = 5.99;    // Floating point number (with decimals)  
char myLetter = 'D';         // Character  
string myText = "Hello";     // String (text)  
bool myBoolean = true;       // Boolean (true or false)

#include<iostream>

using namespace std;

int main()

{

int myage=27;

cout<<"i am "<<myage<<"years old";

return 0;

}

C++ Identifiers

All C++ **variables** must be **identified** with **unique names**.

These unique names are called **identifiers**.

Identifiers can be short names (like x and y) or more descriptive names (age, sum, totalVolume).

**Note:** It is recommended to use descriptive names in order to create understandable and maintainable code:

Example

// Good  
int minutesPerHour = 60;  
  
// OK, but not so easy to understand what **m** actually is  
int m = 60;

[Run example »](https://www.w3schools.com/cpp/showcpp.asp?filename=demo_variables_identifiers)

The general rules for constructing names for variables (unique identifiers) are:

* Names can contain letters, digits and underscores
* Names must begin with a letter or an underscore (\_)
* Names are case sensitive (myVar and myvar are different variables)
* Names cannot contain whitespaces or special characters like !, #, %, etc.
* Reserved words (like C++ keywords, such as int) cannot be used as names

**Constant:**

When you do not want others (or yourself) to override existing variable values, use the const keyword (this will declare the variable as "constant", which means **unchangeable and read-only**):

You have already learned that cout is used to output (print) values. Now we will use cin to get user input.

cin is a predefined variable that reads data from the keyboard with the extraction operator (>>).

cout is pronounced "see-out". Used for **output**, and uses the ***insertion operator*** (<<)

cin is pronounced "see-in". Used for **input**, and uses the extraction operator (>>)

As explained in the [Variables](https://www.w3schools.com/cpp/cpp_variables.asp) chapter, a variable in C++ must be a specified data type:

### Example

int myNum = 5;               // Integer (whole number)  
float myFloatNum = 5.99;     // Floating point number  
double myDoubleNum = 9.98;   // Floating point number  
char myLetter = 'D';         // Character  
bool myBoolean = true;       // Boolean  
string myText = "Hello";     // String

The data type specifies the size and type of information the variable will store:

|  |  |  |
| --- | --- | --- |
| **Data Type** | **Size** | **Description** |
| int | 4 bytes | Stores whole numbers, without decimals |
| float | 4 bytes | Stores fractional numbers, containing one or more decimals. Sufficient for storing 7 decimal digits |
| double | 8 bytes | Stores fractional numbers, containing one or more decimals. Sufficient for storing 15 decimal digits |
| boolean | 1 byte | Stores true or false values |
| char | 1 byte | Stores a single character/letter/number, or ASCII values |

float vs. double

The **precision** of a floating point value indicates how many digits the value can have after the decimal point. The precision of float is only six or seven decimal digits, while double variables have a precision of about 15 digits. Therefore it is safer to use double for most calculations.

A list of all assignment operators:

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Example** | **Same As** | **Try it** |
| = | x = 5 | x = 5 | [Try it »](https://www.w3schools.com/cpp/showcpp.asp?filename=demo_oper_ass1) |
| += | x += 3 | x = x + 3 | [Try it »](https://www.w3schools.com/cpp/showcpp.asp?filename=demo_oper_ass2) |
| -= | x -= 3 | x = x - 3 | [Try it »](https://www.w3schools.com/cpp/showcpp.asp?filename=demo_oper_ass3) |
| \*= | x \*= 3 | x = x \* 3 | [Try it »](https://www.w3schools.com/cpp/showcpp.asp?filename=demo_oper_ass4) |
| /= | x /= 3 | x = x / 3 | [Try it »](https://www.w3schools.com/cpp/showcpp.asp?filename=demo_oper_ass5) |
| %= | x %= 3 | x = x % 3 | [Try it »](https://www.w3schools.com/cpp/showcpp.asp?filename=demo_oper_ass6) |
| &= | x &= 3 | x = x & 3 | [Try it »](https://www.w3schools.com/cpp/showcpp.asp?filename=demo_oper_ass7) |
| |= | x |= 3 | x = x | 3 | [Try it »](https://www.w3schools.com/cpp/showcpp.asp?filename=demo_oper_ass8) |
| ^= | x ^= 3 | x = x ^ 3 | [Try it »](https://www.w3schools.com/cpp/showcpp.asp?filename=demo_oper_ass9) |
| >>= | x >>= 3 | x = x >> 3 | [Try it »](https://www.w3schools.com/cpp/showcpp.asp?filename=demo_oper_ass10) |
| <<= | x <<= 3 | x = x << 3 |  |

## C++ Strings

Strings are used for storing text.

A string variable contains a collection of characters surrounded by double quotes:

### Example

Create a variable of type string and assign it a value:

string greeting = "Hello";

To use strings, you must include an additional header file in the source code, the <string> library:

### Example

// Include the string library  
#include <string>  
  
// Create a string variable  
string greeting = "Hello";

## String Concatenation

The + operator can be used between strings to add them together to make a new string. This is called **concatenation**:

### Example

string firstName = "John ";  
string lastName = "Doe";  
string fullName = firstName + lastName;  
cout << fullName;

[Run example »](https://www.w3schools.com/cpp/showcpp.asp?filename=demo_strings_concat)

In the example above, we added a space after firstName to create a space between John and Doe on output. However, you could also add a space with quotes (" " or ' '):

### Example

string firstName = "John";  
string lastName = "Doe";  
string fullName = firstName + " " + lastName;  
cout << fullName;

[Run example »](https://www.w3schools.com/cpp/showcpp.asp?filename=demo_strings_concat2)

## Append

A string in C++ is actually an object, which contain functions that can perform certain operations on strings. For example, you can also concatenate strings with the append() function:

### Example

string firstName = "John ";  
string lastName = "Doe";  
string fullName = firstName.append(lastName);  
cout << fullName;

[Run example »](https://www.w3schools.com/cpp/showcpp.asp?filename=demo_strings_append)

It is up to you whether you want to use + or append(). The major difference between the two, is that the append() function is much faster. However, for testing and such, it might be easier to just use +.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//write a program that performs the arithmetic operation by using arithmetic operation and displays the result on screen// continued.29/3/2020

#include<iostream>

using namespace std;

int main()

{

float mm,inch;

mm=100.26;

inch=mm/25.4; // 1 inch=25.4mm //

cout<<"milimeter"<<mm<<endl;

cout<<"inches="<<inch<<endl;

return 0; }

//write a program that interchanges the values of two variables and displays the result on screen//

#include<iostream>

using namespace std;

int main()

{

int x=126 ,y=56,temp;

cout<<"value of x and y before interchanging \n";

cout<<"value of x = "<<x<<endl;

cout<<"value of y = "<<y<<endl;

temp=x; //

x=y;

y=temp;

cout<<"vlue of x and y after interchange \n";

cout<<"value of x = "<<x<<endl;

cout<<"value of y = "<<y<<endl;

return 0;

/\*write a program that assigns a value 3 to "int" type variable a,b,c

using multiple assingment statement and calculates the PRODUCT of these numbers and displays the result on screen\*/

#include<iostream>

using namespace std;

int main()

{

int a,b,c,p;

a=b=c=3; //compound assignment statement//

p=a\*b\*c; //product of 3//

cout<<"product of 3 ="<<p<<endl;

return 0;

}

/\*write a program that uses compound assignment statement and compound assignment operators and displays the result on screen\*/

#include<iostream>

using namespace std;

int main()

{

int a,m,d,r,s;

a=m=d=r=s=7; //compound assignment statement

a+=3; //a=3+7

s-=3; //s=3-7

m\*=3; //m=3\*7

d/=3; //d=3/7

r%=3; //r=3%7 // % modulus operator remainder result.

cout<<a<<endl;

cout<<s<<endl;

cout<<m<<endl; //10,4,21,2,1 output of the program

cout<<d<<endl;

cout<<r<<endl;

return 0;

}

/\*write a program that uses post increment operators and displays the result on screen\*/

#include<iostream> //preprocessor directive

using namespace std; //standard library function

int main()

{

int a,c;

a=10;

c=a++; //c=a,a=a+1postfix increment operator.

cout<<"value of a="<<a<<endl; //output is 11

cout<<"value of c="<<c<<endl; //output is 10

return 0;

}\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*write a program that uses post increment operators and displays the result on screen\*/

#include<iostream> //preprocessor directive

using namespace std; //standard library function

int main()

{

int a,b,res; //res=result

a=5;

b=10;

cout<<"value of a and b before evaluation"<<endl;

cout<<"value of a="<<a<<endl;

cout<<"value of b="<<b<<endl;

a+b++;

res=a+b;

cout<<"value of a and b after evaluation"<<endl;

cout<<"value of a="<<a<<endl;

cout<<"value of b="<<b<<endl;

cout<<"result is="<<res<<endl;

return 0;

}

int main()

{

cout<<"size of int in memory location="<<sizeof(int)<<endl;

cout<<"size of float in memory location="<<sizeof(float)<<endl;

cout<<"size of long in memory location="<<sizeof(long)<<endl;

cout<<"size of char in memory location="<<sizeof(char)<<endl;

cout<<"\nsize of double in memory location= "<<sizeof(double)<<endl;

cout<<"size of pak="<<sizeof("pak")<<endl;

cout<<"size of 120="<<sizeof(120)<<endl;

cout<<"size of 23.4="<<sizeof(23.4)<<endl;

return 0;

} size of int in memory location=4

size of float in memory location=4

size of long in memory location=4

size of char in memory location=1

size of double in memory location= 8

size of pak=4

size of 120=4

size of 23.4=8

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

{

float a=15.58971,b;

int n;

n=a;

b=a-n; consused.

cout<<"real number is ="<<a<<endl;

cout<<"integral part="<<n<<endl;

cout<<"fractional part="<<b<<endl;

return 0;

}

#include<iostream> //preprocessor directive

using namespace std; //standard library function

int main()

{

int f=98,c;

c=((5/9)\*(f-32));

cout<<" Celsius="<<c<<"\n";

return 0;

}

#include<iostream> //preprocessor directive

using namespace std; //standard library function

int main()

{

int n,m;

n=m=5;

n++;

++m;

cout<<"n="<<n<<endl<<"m="<<m; output, //n=6 ,m=6

return 0;

}

Area of square.The formula to compute the area of square is Area=height\*width.

#include<iostream> //preprocessor directive

using namespace std; //standard library function

int main()

{

int height,width,area;

height=15;

width=3;

area=height\*width; //Area=height\*width

cout<<"area of square="<<area<<endl;

return 0;

}\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//write a program that converts 45.6 centigrade tempearature into fahrenheit,using the formula F=9/5(C+32)and displays the temperature in Fahrenheit.

#include<iostream> //preprocessor directive

using namespace std; //standard library function

int main()

{

float c=45.6,f;

f=9/5.0\*(c+32); //write in this form of fahrenheit formula

cout<<"centigrade="<<c<<endl<<"fahrehheit="<<f<<endl; //output,centigrade=45.6fahrehheit=139.68

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#include<iostream> //preprocessor directive

using namespace std; //standard library function

int main()

{

int age,days,month;

age=27;

days=age\*365;

month=age\*12;

cout<<"age in days="<<days<<endl<<"age in month="<<month<<endl;

return 0;

} //age in days=9855,age in month=324

//display the value of integer variable and character variable.

#include<iostream> //preprocessor directive

using namespace std; //standard library function

int main()

{

int a=27;

char ch='Z'; //when we write one word "Z" it display Z and when we write name"zeeshan" it will be display the last word of name like n.

cout<<"value of integer variable="<<a<<endl;

cout<<"value of character variable="<<ch<<endl;

return 0;

}

#include<iostream> //preprocessor directive

using namespace std; //standard library function

int main()

{

float s,vi,t,a;

vi=15.5;

t=20.5;

s=((vi\*t)+1/2.0\*a\*(t\*t));

cout<<"distance covered="<<s<<endl; // 317.75 output //

return 0;

}

#include<iostream> //preprocessor directive

using namespace std; //standard library function

int main()

{

float km,miles;

cout<<"enter the miles=";

cin>>miles;

km=1.609\*miles;

cout<<"miles into kilometer="<<km<<endl;

return 0;

}\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#include<iostream> //preprocessor directive

using namespace std; //standard library function

main()

{

double pounds,kg;

kg=pounds\*0.453592;

cout<<"enter the value of pounds=";

cin>>pounds;

cout<<"pounds into kilograms="<<kg<<endl;

return 0;

}

//write a program that inputs principal amount,rate of interest and number of years and calculate the simple interest .

using namespace std; //standard library function

main()

{

double amt,rate,year,interest;

cout<<"enter the principal amount=";

cin>>amt;

cout<<"enter the rate of interest=";

cin>>rate;

cout<<"enter the total year=";

cin>>year;

interest=(amt\*rate\*year)/100;

cout<<"simple interest="<<interest<<endl;

return 0;

}

//A program that input marks obtained by a student in mcs 2nd semester and find out aggregate marks and percentage//

#include<iostream> //preprocessor directive

using namespace std; //standard library function

main()

{

float math,dbms,dld,oop,dsa,al;

float total,avg;

total=math+dbms+dld+oop+dsa+al; //To find the total is equal to summ of all subjects.

avg=total/6.0; //avg=total marks divided by no of subjects(6).

cout<<"enter the number of math="<<"\n";

cin>>math; //input data from user to enter the mathematics student number.

cout<<"enter the number of dbms="<<endl;

cin>>dbms;

cout<<"enter the number of dld="<<endl;

cin>>dld;

cout<<"enter the number of oop="<<endl;

cin>>oop;

cout<<"enter the number of dsa="<<endl;

cin>>dsa;

cout<<"enter the number of al="<<endl;

cin>>al;

cout<<"total marks in subjects="<<total<<endl;

cout<<"aggregate marks="<<avg<<endl;

return 0;

}

#include<iostream>

using namespace std;

int main()

{

float petrol,distance;

cout<<"enter the petrol in liters"<<endl;

cin>>petrol;

distance=40\*petrol; //bike covered one liter petrol in 40km.

cout<<"distance covered of car "<<distance<<endl;

return 0;}

#include<iostream>

#include<iomanip>

using namespace std;

main()

{

double n=78.88888;

cout<<setprecision(7)<<n<<endl;//setprecision manipulator is used to set the number of digits to be displayed after a decimal point.

cout<<setprecision(6)<<n<<endl;//the output value is rounded with the use of this manipulator.

cout<<setprecision(5)<<n<<endl;//setprecision(n)n indicates the number of digits to be displayed after the decimal point.

cout<<setprecision(4)<<n<<endl;

cout<<setprecision(3)<<n<<endl;

cout<<setprecision(2)<<n<<endl;

cout<<setprecision(1)<<n<<endl;

return 0;

}

#include<iostream>

#include<iomanip>

using namespace std;

main(){

int N;

cout<<"enter the integer value="<<endl;

cin>>N;

if(N%2==0)

cout<<"even value="<<endl;

if(N%2==1)

cout<<"odd value="<<endl;

return 0;

}

//even and odd number program//

#include<iostream>

#include<iomanip>

using namespace std;

main()

{

int n;

cout<<"enter the integer value="<<endl;

cin>>n;

if(n%2==0){ //The set of statement surrounded by curly braces{} is called compound statement.

cout<<"even value="<<endl;

}

if(n%2==1){

cout<<"odd value="<<endl;

}

return 0;

}

//inputs four numbers,dinds out the largest number and smallest number using simple if statement//

#include<iostream>

#include<iomanip>

using namespace std;

main()

{

int w,x,y,z,max,mini;

cout<<"Enter the first value ?"<<"\n";

cin>>w;

cout<<"Enter the second value ?"<<"\n";

cin>>x;

cout<<"Enter the third value ?"<<"\n";

cin>>y;

cout<<"Enter the fourth value ?"<<"\n";

cin>>z;

max=mini=w;

if(max<x) max=x;

if(max<y) max=y;

if(max<z) max=z;

if(mini>x) mini=x;

if(mini>y) mini=y;

if(mini>z) mini=z;

cout<<"The largest value is :"<<max<<endl;

cout<<"The smallest value is:"<<mini<<"\n";

return 0;

}

//inputs four numbers,dinds out the largest number and smallest number using simple if statement//

#include<iostream>

#include<iomanip>

using namespace std;

main()

{

int num1,num2;

cout<<"enter the first number=";

cin>>num1;

cout<<"enter the second number=";

cin>>num2;

if(num1\*num1==num2){

//The statement is in curly braces is called compound statement.

cout<<"second number is square of first number";

}

return 0;

}\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//number is positive or not using simple if statement//

#include<iostream>

#include<iomanip>

using namespace std;

main()

{

int num;

cout<<"enter the number=";

cin>>num;

if(num>0){

cout<<"number is POSITIVE\n";

}

if(num<0){

cout<<"number is NEGATIVE\n";

}

return 0;

}

//NUMBER IS ODD OR EVEN//

#include<iostream>

#include<iomanip>

using namespace std;

main()

{

int number;

cout<<"enter the number=";

cin>>number;

if(number%2==0)

cout<<number<<" number is even\n"; //4/2 ,remainder 0, number is even

else

cout<<number<<" number is odd\n"<<endl; //5/2 ,remainder 1 number is odd

cout<<"okay shani"<<endl;

return 0;}

/\*write a program that imputs salray and scale.

it adds 40% of salary increment if scale is greater than 16 .

otherwise,it adds 50% of the salary as increment.The program displays the net pay\*/

#include<iostream>

#include<iomanip>

using namespace std;

main()

{

float salary ,scale,incre ,netpay;

cout<<"enter the salary"<<endl;

cin>>salary;

cout<<"enter the scale"<<endl;

cin>>scale;

if(scale >16)

incre=salary\*40/100.0;

else

incre=salary\*50/100.0;

netpay=salary+incre;

cout<<"netpay="<<netpay<<endl;

return 0;

}

#include<iostream>

using namespace std;

main()

{

int d,q,dn,r;

cout<<"enter the divinded=";

cin>>dn;

cout<<"enter the divisior=";

cin>>d;

q=dn/d;

r=dn%d;

cout<<"quotient="<<q<<endl<<"reminder="<<r<<endl;

return 0;}

/\*write a program that inputs a character from user

and checks whether it is vowel or consonant,

using if-else statement or OR operator\*/

#include<iostream>

using namespace std;

main()

{

char ch;

cout<<"enter a single character?\n";

cin>>ch;

if

(ch=='a'||ch=='A'|| ch=='e'||ch=='E'||ch=='i'||ch=='I'||ch=='o'||ch=='O'||ch=='u'||ch=='U')

cout<<"number is vowel";

else

cout<<"number is consonent"

return 0;

}

#include<iostream>

using namespace std;

main()

{

cout<<"X\nXX\nXXX\nXXXX\nXXXXX\n"<<endl;

cout<<"XXXX\nXXX\nXX\nX\n";

return 0;

}

#include<iostream>

using namespace std;

main()

{

int i,j;

i=17;

j=5;

cout<<"sum="<<i+j<<"\n";

cout<<"product="<<i\*j<<"\n";

cout<<"difference"<<i-j<<"\n";

cout<<"quotient"<<i/j<<"\n";

cout<<"remainder"<<i%j<<"\n";

return 0;

}\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#include<iostream>

using namespace std;

main()

{

int x,y;

cout<<"Enter first number="<<endl; //cout is used to print the statement

cin>>x; //cin is used for getting input value from user.

cout<<"enter the second number="<<endl;

cin>>y;

if(x%y==0) //if remainder is 0,then first number is multiple of second otherwise else condition executed.

cout<<"first number is multiple of second\n";

else

cout<<"first number is not a multiple of second";

return 0;

}

/\* write a program that inputs two numbers and finds larger number using conditional operator\*/

#include<iostream>

using namespace std;

main()

{

int x,y;

cout<<"enter the first number=";

cin>>x;

cout<<"enter the second number=";

cin>>y;

(x>y) ? cout<<x<<"is greater" : cout<<y<<"is greater"; //conditional operator is used for making two-way decisions.

//alternative of if-else condition,three operands are used(?,:,:)//is called ternary operator.

return 0;

}

/\* write a program that inputs two numbers and finds larger number using conditional operator\*/

#include<iostream>

using namespace std;

main()

{

int x;

cout<<"enter the number=";

cin>>x;

//x%5==0,if remainder is 0,then this condition is executed means x is divisible by 5 otherwise not divisible by 5.

(x%5==0)?cout<<x<<"is divisible by 5":cout<<x<<"is not divisible by 5"; //conditional operator is used for making two-way decisions.//alternative of if-else condition,three operands are used**(?,:,:)**//is called ternary operator.

return 0;

}

/\* write a program that inputs two numbers and finds larger numbe using conditional operator\*/

#include<iostream>

using namespace std;

main()

{

int x,y,z;

cout<<"enter the first value=";

cin>>x;

cout<<"enter the second value=";

cin>>y;

cout<<"enter the third value=";

cin>>z;

if(x==y) //if x==y and if x==z ,print all values are equal

if(x==z)

cout<<"All values ar equal";

else //if x and y is different,print these value are different.

cout<<"these value are different";

else //if x and z is different,print these value are different.

cout<<"these value are different";

return 0;

}\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\* write a program that inputs two numbers and finds larger numbe using conditional operator\*/

#include<iostream>

using namespace std;

main()

{

int a,b,c,g; //g means greatest.

cout<<"enter the Ist value=";

cin>>a;

cout<<"enter the 2nd value=";

cin>>b;

cout<<"enter the 3rd value=";

cin>>c;

if(a>b) //if a>b,a>c; greatest b value otherwise greatest c.

if(a>c)

g=b; //greatest value is b.

else

g=c; //greatest value is c

if(b>c) //if b greater than c

g=b; //greatest assign b

else

g=c; //greatest assign c.

cout<<"greatest value is="<<g;

return 0;

}\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#include<iostream>

using namespace std;

main ()

{

float average,marks;

cout<<"Enter the obtained marks=";

cin>>marks;

average=marks\*100/1100.0; //average marks = obtained marks\*100/total marks.

cout<<"average marks="<<average<<endl;

if(average>=80) //if average marks is >80 or equal to 80 ,print A+//

cout<<"A+";

else if(average>=70) //if average marks is >70 or equal to 70 ,print A//

cout<<"A";

else if(average>=60) //if average marks is >60 (and or) equal to 80 ,print A+//

cout<<"B";

else if(average>=50) //if average marks is >50 (and or)equal to 50 ,print B//

cout<<"C";

else if(average>=40) //if average marks is >40 (and or) equal to 40 ,print D//

cout<<"D";

else if(average>=33) //if average marks is >33 (and or) equal to 30 ,print E//

cout<<"E";

else //if average marks <=33,fail/

cout<<"F"; //otherwise fail//

return 0;

}

//write a program that perform simple arithmetic calculations using "nested if-else"structure.

#include<iostream>

using namespace std;

main ()

{

float n,m;

char op;

cout<<"Enter the first value,arithmetic operator and second value=";

cin>>n>>op>>m; //get the user value ,n represent first value,op means arithmetic operator,m represent second value//

if(op=='+') //if operator is assigned (+) by user

cout<<n+m; //process addintion

if(op=='-') //if operator is assinged (-) by user ,process subtraction operation

cout<<n-m;

if(op=='\*') //if operator is assinged (\*) by user ,process multiplication operation

cout<<n\*m;

if(op=='/') //if operator is assinged (/) by user ,process divide operation

cout<<n/m;

else

cout<<"invalid operator"; //if result is 0 then print, invalid operator:

return 0;

}

//write a program that inputs the current reading and previous reading of electric meter.it finds out the total units consumed using nested 'if-else'structure and computes the electricity bill as follows:

//if units consumed are less than or equal to 300,then rate of electricity is Rs:12 per unit

//if units consumed are more than 300 and less than or equal to 400,then rate of electricity is Rs.14 per unit.

//if units consumed are more than 400,then rate of electricity is Rs.20 per unit.//

#include<iostream>

using namespace std;

main()

{

unsigned int previous,current,units,bill;

cout<<"Enter previous reading="; //print the statement of previous reading of bill

cin>>previous; // previous reading by user

cout<<"enter the current reading="; // print the statement of current reading

cin>>current; //enter the current reading of bill by users.

units=current-previous; //formula of electricity bill unit measurment

if(units>400) //if the units consumed is more than 400,rate of electricity should be Rs:20//

bill=units\*20;

else if(units>300) //if the units consumed >300,or less then or equal to 400,rate of electricity should be 14.

bill=units\*14;

else if(units<=300) //if the units consumed<=300,the rate of electricity should be Rs:12.

bill=units\*12;

cout<<"Total units="<<units<<endl;

cout<<"Electricity bill="<<bill;

return 0;

}\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//write a temperatur conversion program that gives the user the options

//for converting the fahrehheit to celsuus or celsius to fahrenheit using "switch" statement

//then carry out the conversion//

#include<iostream>

using namespace std;

main()

{

double c,f; //double keyword is used for temperature to fahrenheit conversion.

int op; //option point

cout<<"1-To convert Fahrenheit to celsius\n"; //print the statement of F to C.

cout<<"2-To convert celsius to Fahrenheit\n"; //print the statement of c to F.

cout<<"Enter option?"; //print the option of above statement

cin>>op; //get the user option

switch(op) //Switch is a control statement that allows a value to change control of execution.

{

case 1:

cout<<"Enter temperature in fahrenheit?";

cin>>f;

c=5.0/9.0\*(f-32);

cout<<"Temperature in celsius:"<<c;

break; //the break statement is used to exit from the body of the 'switch 'structure(or the loop structure)

case 2:

cout<<"Enter temperature in celsius?";

cin>>c;

f=9.0/5.0\*(c+32);

cout<<"Temperature in celsius:"<<c;

break;

default:

cout<<"Invalid option";

}

return 0;

}\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//write a temperatur conversion program that gives the user the options

//for converting the fahrehheit to celsuus or celsius to fahrenheit using "switch" statement

//then carry out the conversion//

#include<iostream>

using namespace std;

main()

{

int n;

cout<<"Enter an integer value[1-7]="; //print the value of integer

cin>>n; //get the value by user

switch(n) //Switch is a control statement that allows a value to change control of execution.

{

case 1:cout<<"Monday";break; //the break statement is used to exit from the body of the 'switch 'structure(or the loop structure)

case 2:cout<<"Tuesday";break;

case 3:cout<<"wednesday";break;

case 4:cout<<"Thrusday";break;

case 5:cout<<"Friday";break;

case 6:cout<<"saturday";break;

case 7:cout<<"sunday";break;

default:

cout<<"invalid number of day";

}

return 0;

}

#include<iostream>

using namespace std;

main()

{

float n,m;

char op;

cout<<"Enter first value,arithmetic operator and second value";

cin>>n>>op>>m;

switch(op) //Switch is a control statement that allows a value to change control of execution.

{

case'+':cout<<n+m;

break; //the break statement is used to exit from the body of the 'switch 'structure(or the loop structure)

case'-':cout<<n-m;

break;

case'\*':cout<<n\*m;

break;

case'/':if(m==0)

cout<<"division by zero\n";

else

cout<<n/m;

break;

default:

cout<<"invalid operator";

}

return 0;

}

#include<iostream>

using namespace std;

main()

{

char ch;

cout<<"Enter a single character\n";

cin>>ch;

switch(ch)

{

case 'a':

case 'A':

case 'e':

case 'E':

case 'i':

case 'I':

case 'o':

case 'O':

case 'u':

case 'U':

cout<<"it is a vowel";

break;

default:

cout<<"it is a constant";

}

return 0;

}

**Loop Structure**

#include<iostream>

using namespace std;

main()

{

int num;

num=1;

while(num<=15) //while(condition),the value should be less than 15 or equal to 15.

{

if(num%3==0) //for example 3\*5=15,remainder is 0 ,then number executed ,otherwise not.

cout<<num<<endl;

num=num+1; //increment the number.

}

return 0;

}

using namespace std;

main()

{

int square,cube,n;

n=2;

cout<<"No.\t square \t cube\n"; //print,No \t means new tab,print square and cube.

while(n<=7) //condition should be equal to 7 and less than seven.

{

cout<<n<<'\t'; //print n value like 2,3,4,5,6,7,

cout<<n\*n<<'\t'; //print n to power 2,like4,9,16,25,36,49.

cout<<n\*n\*n<<'\t'<<endl; //print n to power 2(2 to raise power3=8) and so on.

n++; //increment the vlaue of n until condition is equal to seven or less than seven.

}

return 0;

}

#include<iostream>

using namespace std;

main()

{

int num,start,end; //integer variable declared

cout<<"Enter the starting number"; //print the statement in "double coutes".

cin>>start; //get the any type of vale by the user.

cout<<"Enter the ending number"; //print the statement in"cdouble coutes"

cin>>end; //get the ending value, any type of value by the user.

num=start; //num is a keyword and is assigned start.

while(num<=end) //while condition num<=end means strating the number and end the number.

{

if(num%2==0) //if the number is divided by 2 is equal to zero (difference should be 2 in value) ,then print the number otherwise not.

cout<<num<<endl;

num++;

}

return 0;

}

#include<iostream>

using namespace std;

main()

{

int c=1,s=0;

while(c<=5) //in while condition we assign the limit should be less than 5 or equal to five.

{

cout<<c<<endl; //print the maximum limit value like 5

s=s+c;

c=c+1; //increment the result like 1+1=2,2+1=3,3+1=4,4+1=5.

}

cout<<"sum of five number is="<<s<<endl;

return 0;}

//write a program that inputs a number and displays the factorial of that number using the 'while'loop.

#include<iostream>

using namespace std;

main()

{

long fact,num,c; //fact,factorial(num,number)

fact=1;

c=1;

cout<<"enter any number value:"; //print the statement within "enter any number value"

cin>>num; //get the value by user.

while(c<=num)

{

fact=fact\*c;

c++;

}

cout<<"factorial of "<<num<<"="<<fact;

return 0;

}

//write a program that inputs 4 digit numbers and displays its digits on separate four lines.if we write12345 the output should be in vertical shape and separate.

#include<iostream>

using namespace std;

main()

{

long r,n,d=1000;

cout<<"enter 4 digit number:";

cin>>n;

while(n!=0)

{

r=n/d;

n=n%d;

d=d/10;

cout<<r<<endl;

}

return 0;

}

**Do-while program**

#include<iostream>

using namespace std;

main()

{

int num;

num=10;

do

{

cout<<num<<endl;

num=num-1;

}

while(num>=1);

return 0;

}

#include<iostream> //linbrary file,preprocessor directives

using namespace std; //standard input/output library file

main()

//main body of program.

{

int n,res; //int variable name (res means result)

for(n=1;n<=64;n+=n) //loop condition,n is a initial position;n should be < 64 and equal to 64 value;n should be incremented.

cout<<n<<"\t"; //Result will be 1,2,4,8,16,32,64 with the help of \t tab.

return 0;

}\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#include<iostream> //linbrary file,preprocessor directives

using namespace std; //standard input/output library file

main()

//main body of program.

{

int s,n,c;

cout<<"enter the value of n=";

cin>>n;

for (s=0,c=1;c<=n;c=c+1)

s=s+c\*c;

cout<<"sum of square="<<s;

return 0;

}\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#include<iostream>

using namespace std;

main()

{

int num,start,end; //int type variables

cout<<"Enter starting number";

cin>>start;

cout<<"Enter ending number";

cin>>end;

num=start; //num is assigned start value,if we not declare this line output will be started for 0,and so on.

while(num<=end)//while loop is a conditional loop structure.it is used to execute a statement or set of statements as long as given condition remains true.

{

if(num%2==0) //if the remainder is equal to zero ,then printed the number.

cout<<num<<endl;

num++; //increment the starting value from ending assigned value .

}

return 0;

}

#include<iostream>

using namespace std;

main()

{

int n;

//clrscr;

for (n=1;n<=10;n++) //for loop condition//initialization is equal to 1;the value of n should be < and equal to 10;assigned value should be incremented until the condition is equal to 10.

cout<<n<<endl; //value of n 1to10 is printed.

cout<<"OK";

return 0;

}\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#include<iostream>

using namespace std;

main()

{

int n,res;

for(n=1;n<=64;n+=n) //output will be displayed 1,2,4,8,16,32,64

cout<<n<<"\n"; //\n means new line

return 0;

}\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//write a program in c++ that inputs a number and displays its **TABLE** using the 'for' Loop.

#include<iostream>

using namespace std;

main()

{

int num,c; //int integer,num and c is int varialbes

cout<<"Enter a number"; //print the statement in double coutes "Enter a number"

cin>>num; //get any number of value from the user

for(c=1;c<=10;c++) //loop condition,value of character(c),initialization value is 1;condition should be less then or equal to 10;increment the vlaue

cout<<num<<”x”<<c<<"="<<num\*c<<endl;

//cout<<number is assigned by user<<multiplication symbol<<c should be in sequence(1 to 10)<<print equal value<<result of number\*character(1to10).

return 0;

}

**Table number and its length getting by user.**

#include<iostream>

using namespace std;

main()

{

int num,c,len; //int type variable num,c,len(length of requird)

cout<<"enter the number=";

cin>>num;

cout<<"enter the number of length="; //number of length getting by user ,like (1,2,3,4,5,6,7)

cin>>len;

for(c=1;c<=len;c++) //for loop condition initialization from 1,character<=length, getting by user;increment of character required by getting user.

cout<<num<<"x"<<c<<"="<<num\*c<<endl; //print the table format.

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#include<iostream>

using namespace std;

main()

{

unsigned long n,p;

for(n=1,p=1;n<=10;n++) //for loop condition

if(n%2==1) //for odd integer value ,if the remainder is equal to 1 ,like 2/5=r 1, then print the n.

{

cout<<n<<endl; //output of odd number like 1,3,5,7,9 w.r.t for loop condition.

p=p\*n; //formula for product of the odd integer value

}

cout<<"product of odd numbers="<<p; //product of the odd values.

return 0;

}

//display the even numbers between 1to15 in descending order using the "For"Loop statement.

#include<iostream>

using namespace std;

main()

{

int n;

for(n=15;n>=1;n--)

{

if(n%2==0)

cout<<n<<endl;

}

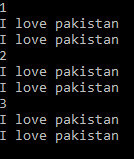
return 0;

}\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#include<iostream>

using namespace std;

main()

{

int u,i;

u=1;

while(u<=3)

{

cout<<u<<endl;

for(i=1;i<=2;i++)

cout<<"I love pakistan \n";

u=u+1;

}

return 0;

}\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Function:** it is recommended to partition or divide the problem into smaller solutions then are integrated to form a final solution.

For solving larger problems,the basic principle is “divide and conquer”. According to this principle,divide the problem into smaller pieces,so that each piece can be solved separately.

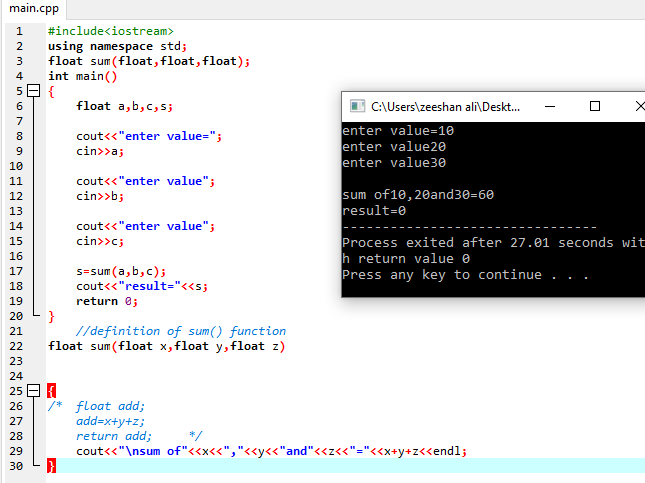
**Function:** A function is a piece of code,which is written to perform a specific task.

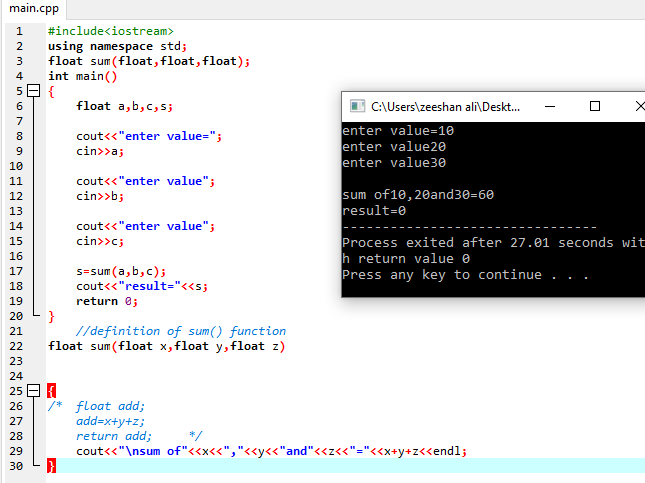
It is independent of the remaining code of the program.

A program may have many functions.each function has a unique name.The function is called for execution with the reference to its name.

A function can be called for execution by the main() function or any other function of the program.

Function provide structured programming approach.In structured programming, a large program is divided into smaller units.These small units are called functions or modules,or subprograms.



T**able program**

#include<iostream>

#include<iostream>

using namespace std;

void table(int t){

cin>>t;

for(int i=1;i<=10;i++){

cout<<t<<"\*"<<i<<"="<<t\*i<<endl;

}

}

int main(){

cout<<"enter the code"<<endl;

int n;

table(n);

}

**Function program**

#include<iostream>

using namespace std;

void display() //void means no return the value //,display only display the output // (parameter1,parameter 2)

{

cout<<"Hello world\n"; //body of function

}

int main()

{

display(); // function calling

return 0;

}

A function is a block of code which only runs when it is called.

You can pass data, known as parameters, into a function.

Functions are used to perform certain actions, and they are important for reusing code: Define the code once, and use it many times.

## Create a Function

C++ provides some pre-defined functions, **such as main(), which is used to** **execute code**. But you can also create your own functions to perform certain actions.

To create (often referred to as declare) a function, specify the name of the function, followed by parentheses **()**:

### Syntax

void myFunction() {  
  // code to be executed  
}

#### Example Explained

* myFunction() is the name of the function
* void means that the function **does not** have a return value. You will learn more about return values later in the next chapter
* inside the function (the body), add code that defines what the function should do

## Call a Function

Declared functions are not executed immediately. They are "saved for later use", and will be executed later, when they are called.

To call a function, write the function's name followed by two parentheses () and a semicolon ;

In the following example, myFunction() is used to print a text (the action), when it is called:

### Example

Inside main, call myFunction():

// Create a function  
void myFunction() {  
  cout << "I just got executed!";  
}  
  
int main() {  
  **myFunction();** // call the function  
  return 0;  
}  
  
// Outputs "I just got executed!"

[Run example »](https://www.w3schools.com/cpp/showcpp.asp?filename=demo_functions)

A function can be called multiple times:

### Example

void myFunction() {  
  cout << "I just got executed!\n";  
}  
  
int main() {  
  **myFunction();**  
  **myFunction();**  
  **myFunction();**  
  return 0;  
}  
  
// I just got executed!  
// I just got executed!  
// I just got executed!

[Run example »](https://www.w3schools.com/cpp/showcpp.asp?filename=demo_functions_multiple)

## Function Declaration and Definition

A C++ function consist of two parts:

* **Declaration:** the function's name, return type, and parameters (if any)
* **Definition:** the body of the function (code to be executed)

void **myFunction()** { // **declaration**  
  // the body of the function (**definition**)  
}

**Note:** If a user-defined function, such as myFunction() is declared after the main() function, **an error will occur**. It is because C++ works from top to bottom; which means that if the function is not declared above main(), the program is unaware of it:

### Example

int main() {  
  myFunction();  
  return 0;  
}  
  
void myFunction() {  
  cout << "I just got executed!";  
}  
  
// Error

[Run example »](https://www.w3schools.com/cpp/showcpp.asp?filename=demo_functions_err)

However, it is possible to separate the declaration and the definition of the function - for code optimization.

You will often see C++ programs that have function declaration above main(), and function definition below main(). This will make the code better organized and easier to read:

### Example

// **Function declaration**  
void myFunction();  
  
// The main method  
int main() {  
  myFunction();  // **call** the function  
  return 0;  
}  
  
// **Function definition**  
void myFunction() {  
  cout << "I just got executed!";  
}

[Run example »](https://www.w3schools.com/cpp/showcpp.asp?filename=demo_functions_opt)

## C++ Exercises

Top of Form

## Test Yourself With Exercises

## Exercise:

Create a function named myFunction and call it inside main().

void () {

cout << "I just got executed!";

}

int main() {

;

return 0;

}

## Parameters and Arguments

Information can be passed to functions as a parameter. Parameters act as variables inside the function.

Parameters are specified after the function name, inside the parentheses. You can add as many parameters as you want, just separate them with a comma:

### Syntax

void functionName(parameter1, parameter2, parameter3) {  
  // code to be executed  
}

The following example has a function that takes a string called **fname** as parameter. When the function is called, we pass along a first name, which is used inside the function to print the full name:

### Example

void myFunction(**string fname**) {  
  cout << fname << " Refsnes\n";  
}  
  
int main() {  
  myFunction(**"Liam"**);  
  myFunction(**"Jenny"**);  
  myFunction(**"Anja"**);

 return 0;  
}  
  
// Liam Refsnes  
// Jenny Refsnes  
// Anja Refsnes

[Run example »](https://www.w3schools.com/cpp/showcpp.asp?filename=demo_functions_param)

When a **parameter** is passed to the function, it is called an **argument**. So, from the example above: fname is a **parameter**, while Liam, Jenny and Anja are **arguments**.

## Function Overloading

With**function overloading**, multiple functions can have the same name with different parameters:

### Example

int myFunction(int x)  
float myFunction(float x)  
double myFunction(double x, double y)

Consider the following example, which have two functions that add numbers of different type:

### Example

int plusFuncInt(int x, int y) {  
  return x + y;  
}  
  
double plusFuncDouble(double x, double y) {  
  return x + y;  
}  
  
int main() {  
  int myNum1 = plusFuncInt(8, 5);  
  double myNum2 = plusFuncDouble(4.3, 6.26);  
  cout << "Int: " << myNum1 << "\n";  
  cout << "Double: " << myNum2;  
  return 0;  
}

[Run example »](https://www.w3schools.com/cpp/showcpp.asp?filename=demo_function_overloading)

Instead of defining two functions that should do the same thing, it is better to overload one.

In the example below, we overload the plusFunc function to work for both int and double:

### Example

int plusFunc(int x, int y) {  
  return x + y;  
}  
  
double plusFunc(double x, double y) {  
  return x + y;  
}  
  
int main() {  
  int myNum1 = plusFunc(8, 5);  
  double myNum2 = plusFunc(4.3, 6.26);  
  cout << "Int: " << myNum1 << "\n";  
  cout << "Double: " << myNum2;  
  return 0;  
}

A function is block of code which is used to perform a particular task, for example let’s say you are writing a large C++ program and in that program you want to do a particular task several number of times, like displaying value from 1 to 10, in order to do that you have to write few lines of code and you need to repeat these lines every time you display values. Another way of doing this is that you write these lines inside a function and call that function every time you want to display values. This would make you code simple, readable and reusable.

#### Syntax of Function

return\_type function\_name (parameter\_list)

{

//C++ Statements

}

Let’s take a simple example to understand this concept.

## A simple function example

#include <iostream>

using namespace std;

/\* This function adds two integer values

\* and returns the result

\*/int

sum(int num1, int num2){

int num3 = num1+num2; return num3;

}

int main(){

//Calling the function

cout<<sum(1,99);

return 0;

}

**Output:**

100

**The same program can be written like this:** Well, I am writing this program to let you understand an important term regarding functions, which is function declaration. Lets see the program first and then at the end of it we will discuss function declaration, definition and calling of function.

#include <iostream>

using namespace std;

//Function declaration

int sum(int,int);

//Main function

int main(){

//Calling the function

cout<<sum(1,99);

return 0;

}

/\* Function is defined after the main method

\*/

int sum(int num1, int num2){

int num3 = num1+num2;

return num3;

}

**Function Declaration:** You have seen that I have written the same program in two ways, in the first program I didn’t have any function declaration and in the second program I have function declaration at the beginning of the program. The thing is that when you define the function before the main() function in your program then you don’t need to do function declaration but if you are writing your function after the main() function like we did in the second program then you need to declare the function first, else you will get compilation error.

**syntax of function declaration:**

return\_type function\_name(parameter\_list);

**Note:** While providing parameter\_list you can avoid the parameter names, just like I did in the above example. I have given int sum(int,int); instead of int sum(int num1,int num2);.

**Function definition:** Writing the full body of function is known as defining a function.  
**syntax of function definition:**

return\_type function\_name(parameter\_list) {

//Statements inside function

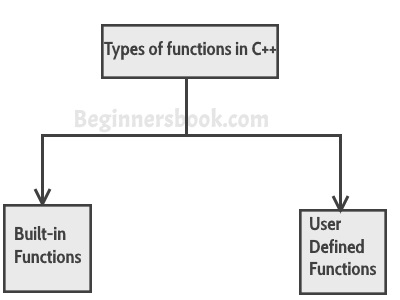
}

**Calling function:** We can call the function like this:

function\_name(parameters);

Now that we understood the **working of function**, lets see the types of function in C++

## Types of function

We have two types of function in C++:  


1) Built-in functions  
2) User-defined functions

### 1) Build-it functions

Built-in functions are also known as library functions. We need not to declare and define these functions as they are already written in the C++ libraries such as iostream, cmath etc. We can directly call them when we need.

#### Example: C++ built-in function example

Here we are using built-in function pow(x,y) which is x to the power y. This function is declared in cmath header file so we have included the file in our program using #include directive.

#include <iostream>

#include <cmath>

using namespace std;

int main(){

/\* Calling the built-in function

\* pow(x, y) which is x to the power y

\* We are directly calling this function

\*/

cout<<pow(2,5);

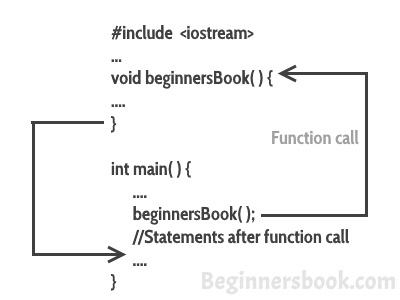
return 0;

}

**Output:**

32

### 2) User-defined functions

  
We have already seen user-defined functions, the example we have given at the beginning of this tutorial is an example of user-defined function. The functions that we declare and write in our programs are user-defined functions. Lets see another example of user-defined functions.

#### User-defined functions

#include <iostream>

#include <cmath>

using namespace std;

//Declaring the function sum

int sum(int,int);

int main(){

int x, y;

cout<<"enter first number: ";

cin>> x;

cout<<"enter second number: ";

cin>>y;

cout<<"Sum of these two :"<<sum(x,y);

return 0;

}

//Defining the function sum

int sum(int a, int b) {

int c = a+b;

return c;

}

**Output:**

enter first number: 22

enter second number: 19

Sum of these two :41

#include<iostream>

using namespace std; // standard library built-in library

void even\_odd(int x); //void is used for no return value ,name of function(even\_odd) & parameter int type

main()

{ int n; //int variable

cout<<"enter any integer value="; //body of program

cin>>n; //get the int value by user

even\_odd(n); //Definition of the even\_odd program//necessary for defining

}

void even\_odd(int x) //Definition of function

{ //calling the function program of above

if(x%2==0) //if the remainder is equal to 0 ,print Even number.

cout<<"Even Number";

if(x%2==1) //if the remainder is equal to 1,print odd number

cout<<"Odd Number";}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Program by defining a function “prime” that determines whether a number passed to the function is a prime number.use this function to determine and display all the prime numbers between 2 to 100.**

#include<iostream>

using namespace std; // standard library built-in library

//void is used for no return value ,an integer is said to be a prime if it is divisible by only 1 and itself e.g 2,3,5,7 are prime number & parameter int type

void prime(int );

main() //main() function begins by declaring the addition() function.the structure of function call statement is very similar to its declaration.

{

long i; //long variable

for(i=2;i<=100;i++) //loop condition i assign=2;i initilize 2 to100;increment of i

prime(i); // function is a prime function(and its parameter is long)

}

//definition of prime() function

void prime(int n) //(void) no return value ,variable name(parameter of int inilize n variable)

{

int d; //for loop condition calling a program of above mention.

for(d=2;d<n;d++)

if(n%d==0) //if the number is remainder is equal to 0,print the prime number

return; //also return the n value

cout<<n<<"\t"; //print n value with one tab space.

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#include<string>

using namespace std;

int main()

{

char question[]="Enter your first name:";

char greeting[]="HELLO,";

char yourname [80];

cout<<question;

cin>>yourname;

cout<<greeting<<yourname;

return 0;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\* write a program that accepts two integer values and passes these values to a function by reference.The swapped /exchanged value should be displayed in the calling function.\*/

/#include<iostream>

using namespace std; // standard library built-in library

//void is used for no return value

void swap(int&,int&); //& ampersand operator is used to get the address of int value from memeory

main()

{

int a,b; //integer variable of a,b...........

cout<<"enter the value of a=";

cin>>a;

cout<<"enter the value of b=";

cin>>b;

swap(a,b); //for the declaration of swaping the vlue of a and b ..definition

cout<<"values of a and b after swapping\n";

cout<<"value of a="<<a<<endl;

cout<<"value of b="<<b<<endl;

return 0;

}

//definition of the function

void swap(int& x,int& y) //address of the variable is assigned x and y,x variable is assigned the address of the value of a also y is assigned the address of the variable of b value.

{

int temp; //int variable of temperory

temp=x; // temp is assigned the value of x,while x is assigned the address of a to get the value by user.

x=y; //x is initialize of the value of y.

y=temp; //y is assignd temp ,while temp is assing is x as above second value'temp=x'.......

}

Bottom of Form

**"Return value from a function"**

**when a function completes its exectuion,it can return a single value to the calling function(or program.)**

**The returned value can be of any type EXCEPT string and array.**

**"RETURN STATEMENT" the return statement is used in the body of function to return a values as well as function.The general syntax of return statement is as follows:**

**retrun expression;**

**The expression may be an airthmetic expression or a single varialbe or a constant value.in case of arithmetic expression,it is evaluated first and then**

**its value is returned to the calling function.**

// Following program computes cube of a number by using a function..

#include<iostream>

using namespace std;

int cube(int); // integer variable is assigned function of cube (parameter of integer)

main(){

//int cube(int);//also declared in this scope

int x,y; //integer is assigned x and y

cout<<"Enter an integer value=";

cin>>x;

y=cube(x); // function 'cube' is called for execution through main() function.

cout<<"cube of "<<x<<"is=\t"<<y; //\t is used for spacing with respect to TAB keyboard button

//return 0;// also declared in this scope no error.

}

//definition of cube()function

int cube(int n) //The value of x is assigned by n in this definition

{

int c; //int variable c is assigned

c=n\*n\*n; //c is initialized cube (like n\*n\*n)

return c; //The return statement will transfer to control back to calling function along with value of c.

}

/\* write a program that gets three numbers and passes these numbers to a function.The function finds out the largest number and returns this number to the calling fucntion\*/

#include<iostream>

using namespace std;

int max(int,int,int); //function prototype

main() // in this case main() is the body of the function to declare the program very necessary

{

int a,b,c;

cout<<"enter the ist value=";

cin>>a;

cout<<"enter the 2nd value=";

cin>>b;

cout<<"enter the 3rd value=";

cin>>c;

cout<<"Largest number is :\t"<<max(a,b,c);

return 0;

}

//definition of function

int max(int d,int e,int f) //variable of int ,max is a function name (parameter of int assigned is d,int e and int f)

{

int mx=d; //integer variabel maximum max is initialize d.

if(e>mx) //if condition (e is assigned by int b of the above program,if e>mx

mx=e; //(mx is assigned e value)

if(f>mx) //f is assigned c in function of above program,f>mx while mx is assigned the value of ist value e.

mx=f; //mx is assigned the value of ist value e is assigned c

return mx; //return the maximum value of calling the funtion of main body program.

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

/\* write a program that gets three numbers and passes these numbers to a function.The function finds out the largest number and returns this number to the calling fucntion\*/

#include<iostream>

using namespace std;

int max(int,int,int); //function prototype

main() // in this case main() is the body of the function to declare the program very necessary

{

int a,b,c;

cout<<"enter the ist value=";

cin>>a;

cout<<"enter the 2nd value=";

cin>>b;

cout<<"enter the 3rd value=";

cin>>c;

cout<<"Largest number is :\t"<<max(a,b,c);

return 0;

}

//definition of function

int max(int d,int e,int f) //variable of int ,max is a function name (parameter of int assigned is d,int e and int f)

{

int mx=d; //integer variabel maximum max is initialize d.

if(e>mx) //if condition (e is assigned by int b of the above program,if e>mx

mx=e; // (mx is assigned e value)

if(f>mx) //f is assigned c in function of above program,f>mx while mx is assigned the value of ist value e.

mx=f; //mx is assigned the value of ist value e is assigned c

return mx; //return the maximum value of calling the funtion of main body program.

}

/\* write a program that gets three numbers and passes these numbers to a function.The function finds out the largest number and returns this number to the calling fucntion\*/

#include<iostream>

using namespace std;

int GCD(int,int); //function prototype

main() // in this case main() is the body of the function to declare the program very necessary

{

int a,b,cd;

cout<<"enter the ist integer value=";

cin>>a;

cout<<"enter the 2nd integer value=";

cin>>b;

cd=GCD(a,b); //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

cout<<"Greatest Common divisior is :\t"<<cd;

//return 0;

}

//definition of function

int GCD(int x,int y) //int variable is declared ,GCD is a function name (parameter of int assigned is x and int y)

{

int i, g=1;

//g=1; //g initilaize 1

for (i=2;i<=x;i++) //loop condition i is initialize is 2;value of i should be less then & equal to x,where x is assigned of a in the function program,i=i+1 increment operator

if(x%i==0 && y%i==0) //if the remainder i and y is 0 condition then g initilize i other than not.

g=i;

return g; //\*\*\*\*\*return the funtion of main body program.\*\*\*\*\*

}\_

#include<iostream>

using namespace std;

int qualitypoints(int);

main()

{

int avg; //for average marks int variable is used

cout<<"enter average marks=";

cin>>avg;

cout<<"Studetn quality points="<<qualitypoints(avg); //quality points of student either where he selected

cin>>avg;

// return 0;

}

// definition of qualitypoints()function

/\*A program by defining a function"qualitypoints" that takes average marks of a student ard returns the value as

assign 4 if average is between 90 & 100

assign 3 if average is between 70 & 89

assign 2 if average is between 50 & 69

assign 1 if average is between 40 & 49

assign 0 if average is lower than 40

\*/

int qualitypoints(int average)

{

int grade; //initilize the grade in integer variable

if(average>=90) //nested if else condition means multiple of if conditions within if

grade=4; //assign 4 **student quality points** if average is between 90 & 100

else if(average>=70) //assign 3 **student quality points** if average is between 70 & 89

grade=3;

else if(average>=50) //assign 2 **student quality points** if average is between 50 & 69

grade=2;

else if(average>=40) //assign 1 **student quality points** if average is between 40 & 49

grade=1;

else //assign 0 **student quality points** if average is lower than 40

grade =0;

return grade; //**assign the student quality points** value either whatever the average marks \*\*\*\*important\*\*\*\*

}

find the area of triangle the inputs base and heignt or a triangle .The function returns the result to the invoking point (calling funtion)

#include<iostream>

using namespace std;

float area\_triangle(int, int); //float variable is used point values

main() //it is the part of body of following function.

{

int b,h;

cout<<"Enter base of triangle=";

cin>>b;

cout<<"Enter the height of triangle=";

cin>>h;

int ans;

ans=area\_triangle(b,h); // to declare the function program area of triangle

cout<<"Area of trianlge=\t"<<ans;

}

//definition of area\_triangle() function

float area\_triangle(int base,int height)

{

return 0.5\*base\*height;

}

//write a function is\_prime that has an input parameter,i.e.num,and returns a value of 1 if number is prime.otherwise returns a value of 0.

#include<iostream>

using namespace std;

int is\_prime(int);

main()

{

// int is\_prime(int); //integer variable is also declared in this scope

int num;

cout<<"enter any number=";

cin>>num;

is\_prime(num)?cout<<"yes":cout<<"No"; //\*\*\*\*\*\*\*\*\*\*\*\*\*\*

}

//definition of prime number

int is\_prime(int n) //calling the fuction

{

int d=2; //initilize the integer d =2;

for(;d<=n/2;d++) //loop condition initilize 2;condition ;increment operator d=d+1;

if(n%d==0) //condition if remainder is equal to 0 then return 0,other wise return 1.

return 0;

return 1;

}

#include<iostream>

using namespace std;

int add (int,int);

int subtract (int,int);

int multiply (int,int);

int divide( int,int);

//void main(void)

int main()

{

int x,y;

int op; //declare option int variable

cout<<"Enter first number="; //print ist number on screen

cin>>x;

cout<<"Enter second number="; //print 2nd number on screen

cin>>y;

cout<<"1- Add Numbers\n"; // print the add,subtract,multiply & divide

cout<<"2- Subtract Numbers\n";

cout<<"3- Multiply Numbers\n";

cout<<"4- Divide Numbers\n";

cout<<"select your option/choice=";

cin>>op; // option declared

switch(op) //switch statement

{

case 1: cout<<x<<"+"<<y<<"="<<add(x,y);

break;

case 2: cout<<x<<"-"<<y<<"="<<subtract(x,y);

break;

case 3: cout<<x<<"\*"<<y<<"="<<multiply(x,y);

break;

case 4: cout<<x<<"/"<<y<<"="<<divide(x,y);

break;

default:

cout<<"Invalid Option";

}

return 0;

}

//definition of function Add(),Subtract(),Multiply()& Divide()

int add(int a,int b)

{

return a+b;

}

int subtract(int a,int b)

{

return a-b;

}

int multiply(int a,int b)

{

return a\*b;

}

int divide(int a,int b)

{

return a/b;

}

#include<iostream>

using namespace std;

void time(int=12,int=56,int=35);

main()

{

int a,b;

time();

time(11);

time(10,33);

time(10,45,58);

// return 0;

}

// definition of time() function

void time(int h,int m,int s)

{

cout<<h<<":"<<m<<":"<<s<<endl;

}

In command line interface environemtn such as DOS,the program is run on the computer by giving the executable file name.In some programs or commands,the parameters are also given at the command line.

The arguments that are passed to a program or command at the Dos prompt are called command line arguments.

For example,to format a new disk, the “format” command with disk drive as parameter is as under: c:\format A:

Program explanation: In this program,two arguments have been used in the main()function:

1. The first argument is always of “int” type.It is used to store the total number of command line arguments including executable file.
2. The second argument is an array of strings.The system creates an array of pointers to these strings.The first element of the array of string will be str[0],second element will be str[1] and so on. The arugments passing from DOSprompt with the program will be stored in these elements of array.The program file name with complete path will be stored in the second element,i.e str[1] and so on.

#include<iostream>

using namespace std;

main(int n,char \*str[])

{

cout<<"number of arguments passed:"<<n<<endl;

cout<<str[0]<<endl;

cout<<str[1]<<endl;

cout<<str[2]<<endl;

cout<<str[3]<<endl;

return 0;

}

#include<iostream>

using namespace std;

// template is a keyword which is used to define a function templae .it instructs the compiler that function template is defined.

//The keyword "class" is used to define a general data type.

//T means type ,specifies the name of general data type.Any valid identifier can be used as general data type.

// it indicates the return data type of function.it can be genereate data type or standard data type.

//function name is specifies the name of function template.

// it specifies the list of parameters that will be passed to the function.The type of parameters can be genereal data type or standard ddata type or mixed.

//it specifies the actual set of statements of the function.

template<class T>

T sum(T x,T y,T z)

{

return x+y+z;

}

main()

{

cout<<"sum of values:"<<sum(2,6,9)<<endl;

cout<<"sum of values:"<<sum(6.1,7.8,9.5)<<endl;

cout<<"sum of values:"<<sum(6,5,19)<<endl;

return 0;

}

/\* An array is a collection of data that holds fixed number of values of same type.  
while Pointers are used in C++ program to access the memory and manipulate the address.

An array is a consecutive group of memeory locations with the same name and data type. Teh memory locations in an array are called elements of the array.The toal number of elements in the array is called the size of array

The position number is called indezx or subscript.The index is written within square brackets[] with the name of array.The index must be an unsigned integer value.

in C++,first element of the array has an index value of 0.in an array with 'n' elements,the index values are 0,1,.....n-1.

The 'n-1' represents the index value of last element.The index of first element of array is called Lower Bound(LB) and the index of the last element of array is called its Upper Bound(UB).

Array are used to process a large amount of data of same data type.

\*/

#include<iostream>

using namespace std;

main()

{

int temp[10];

cout<<"Enter value in 1st element of temp=";

cin>>temp[0];

cout<<"Enter value in 2nd element of temp=";

cin>>temp[1];

cout<<"Enter value in 3rd element of temp=";

cin>>temp[2];

cout<<"Enter value in 4th element of temp=";

cin>>temp[3];

cout<<"Enter value in 5th element of temp=";

cin>>temp[4];

cout<<"values in array temp are:"<<endl;

cout<<temp[0]<<endl;

cout<<temp[1]<<endl;

cout<<temp[2]<<endl;

cout<<temp[3]<<endl;

cout<<temp[4]<<endl;

cout<<temp[5]<<endl;

}

/\*write a program that assigns values into the individual elements of an array and displays the values of the array\*/

/\*An array is a consecutive group of memory locations with the same name and data type.

The memory locationns in an array are called elements of the array.

The total number of elements in the array is called the size of array (or its length)\*/

#include<iostream>

//#include<array>

using namespace std;

main()

{ //arrays are used to process a large amount of data of same data type.

int t[5],i; //integer variable of array is assigned [0to4]

t[0]=2; //assign the value of arrary like t[0] is equal to 2.

t[1]=20; //assign the value of arrary like t[1] is equal to 20.

t[2]=15; //assign the value of arrary like t[2] is equal to 15.

t[3]=254; //assign the value of arrary like t[3] is equal to 254.

t[4]=5; //assign the value of arrary like t[4] is equal to 5.

//loop statement is used to access data of elements of an array by changing index value in the body of loop.

for (i=0;i<=4;i++) //for loop conditon (initilize the value is 0;value of i should be less than and equal to4;i=i+1)

{

cout<<"value in temp["<<i<<"]="<<t[i]<<endl; //print(value of temp[i]<<(t[i],means i is assign is 0,1,2,3,4 loop condition and it will continue 0 to 5)

}

return 0;

}

/\*An array is a consecutive group of memory locations with the same name and data type.

The memory locationns in an array are called elements of the array.

The total number of elements in the array is called the size of array (or its length)\*/

#include<iostream>

//#include<array>

using namespace std;

main()

{

int abc[15],i;

//Assign values to array "ABC"

for(i=0;i<=14;i++)

abc[i]=50; //assign the value of array w.r.t to program conditon

for(i=0;i<=14;i++)

cout<<abc[i]<<endl; //display the contents [15] while 15 is assign 50

}

/\***write a program that gets 5 integer values into an array**

**and then displays the values of the elements of array in reverse order**\*/

/\*An array is a consecutive group of memory locations with the same name and data type.

The memory locations in an array are called elements of the array.

The total number of elements in the array is called the size of array (or its length)\*/

#include<iostream>

//#include<array>

using namespace std;

main()

{

int arr[5]; //array is defined 5

int i; //i is integer variable

for(i=0;i<=4;i++) //for loop condition

{

cout<<"enter value in element:"<<i<<"="; //print the arrary loop condition [0 to 4]

cin>>arr[i]; //get the value by user

}

cout<<"output in reverse order"<<endl;

for(i=4;i>=0;i--) // loop condition in reverse order(i is initilized 4;condition in reverse ;i=i-1 decrement operator

{

cout<<arr[i]<<endl; // initilize the array arr[5]

}

return 0;

}

/\*write a program that input values into one-dimensional array and finds out the

total number of odd and even values entered into one-dimensional array.\*/

/\*An array is a consecutive group of memory locations with the same name and data type.

The memory locationns in an array are called elements of the array.

The total number of elements in the array is called the size of array (or its length)\*

#include<iostream>

using namespace std;

main()

{

int arr[10],i,odd,even; //integer variable arr[10],i,odd,even.

i=odd=even=0; //i=odd=even is initilized is 0.

while(i<=9) //while i should be less than and equal to 9,it means 0 to 10 value.

{

cout<<"enter value into element:"<<i<<"="; //print enter value into element<<also value of array is printed (like 0 to9)

cin>>arr[i]; // get the value of assigning arrary[10] by the user

(arr[i]%2==0) ? even++ : odd++; //unary operator (arry[i] value is divided 2)? if the remainder is equal to 0,even value otherwise odd : increment even and odd value from (0 to 9)

i++; //increment operator (0 to 9)

}

cout<<"\n even number="<<even; //count the vlue of even in program

cout<<"\n odd values="<<odd; //count the value of odd in program

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Zeeshan Ali\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/\*write a program that input values into one-dimensional array and finds out the

total number of odd and even values entered into one-dimensional array.\*/

/\*An array is a consecutive group of memory locations with the same name and data type.

The memory locationns in an array are called elements of the array.

The total number of elements in the array is called the size of array (or its length)\*/

#include<iostream>

using namespace std;

main()

{

int arr[5],i; //integer variable intilize arr[5],and i

i=0; //i is assign 0

/\*while condition to avoid the repetion of if elese condition

or nested if elese(i should be < and equal to 5 means ,value should be print 0 to 5\*/

while(i<=4)

{

//print the value of array "enter the value of int element<<array value 0,1,2,3,4,5

cout<<"enter the value of into element:"<<i<<"=";

cin>>arr[i]; //get the n user value of array 0 to 5 address.

i++; //increment operator value of to repeat the value 0 to 4.

}

cout<<"products of the values and the positions:"<<endl; //simple printed the statement

for(i=0;i<=4;i++) //loop condition

cout<<arr[i]<<"\*"<<i+1<<"="<<arr[i]\*(i+1)<<endl;

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Zeeshan ali\*\*\*\*\*\*\*\*\*\*\*\*

using namespace std;

main()

{

int arr[10] ,i=0;

while(i<=9)

{

cout<<"value of element"<<i<<"="; //print the statement and print loop condition 0to9

cin>>arr[i]; //get the user define value by user fro 0 to 9

if(arr[i]>30||arr[i]<=0) continue; //if the user define value is >30|| 0 then address of array will still constant,if value is 2,4,6,8,10 then continue

i++; //increment the operator from 0 to 30;

}

i=0;

cout<<"\n Element\t value\t Histogram\n";

while(i<=9)

{

cout<<i<<"\t"<<arr[i]<<"\t"; //\*\*\*\*\*\*\*\*

for(int c=1;c<=arr[i];c++)

cout<<"\*";

cout<<endl;

i++;

}

}

/\*write a program that initializes the vlues in one-dimensional array and computes the sum of odd assigned to the array. It also displays even vlues and their sum on the screeen.\*/

#include<iostream>

using namespace std;

main()

{

int arr[10]={14,16,34,43,77,34,21,12,33,30}; //arry value assigned

int i,sum; //integer variable i,sum

i=sum=0; //i and sum initialize=0

while(i<=9) //while (condition)

{

if(arr[i]%2==1) // array value from the given above is even or remainder is 0,then print value of a[i]

{

cout<<arr[i]<<endl; //array is print without "double quation marks"

sum=sum+arr[i]; //sum is also initilaize =0 ,0+ array value getting by user e.g 0+14=14

}

i++; //increment the loop 0 to9.

}

cout<<"\n sum ot even numbers="<<sum; //total sum value that are selected by user and while condition

}

/\*write a program that initializes the vlues in one-dimensional array and computes the sum of even assigned to the array. It also displays even vlues and their sum on the screeen.\*/

#include<iostream>

using namespace std;

main()

{

int arr[10]={14,16,34,43,77,34,21,12,33,30}; //arry value assigned 1to10 not strating by 0 because its assigned is 10

int i,sum; //integer variable i,sum

i=sum=0; //i and sum initialize=0

while(i<=9) //while (condition)

{

if(arr[i]%2==1) // array value from the given above is even or remainder is 0,then print value of a[i]

{

cout<<arr[i]<<endl; //array is print without "double quation marks"

sum=sum+arr[i]; //sum is also initilaize =0 ,0+ array value getting by user e.g 0+14=14

}

i++; //increment the loop 0 to9.

}

cout<<"\n sum ot even numbers="<<sum; //total sum value that are selected by user and while condition

}

#include<iostream>

using namespace std;

main()

{

long arr[]={2,6,3,5,4,7},i=0,f,n;

while(i<=5) //while(condition i should be and equal to 5)

{

f=1; // in this while condition factorial f is initialize=1

n=arr[i]; //no of factorial is assigned by n={array 2,6,3,5,4,7}

for(;n>=1;n--) // factorial is already assigned f=1;condition n should be > and equal to 1,decrement operator

f=f\*n; //f is long variable declare=f=1\*n,while n is initialize is arr[i]

cout<<"factorial of "<<arr[i]<<":"<<f<<endl; // print factorial of <<{2,6,3,5,4,7}<<:<<f=f\*n;

i++;

}

//return 0;

}

#include<iostream>

using namespace std;

main()

{

int arr[5],n,i,pos; //int variable declare arr[5],n,i and pos(position)

i=0; //i is initialize=0;

while(i<=4) //while(i<=4,i should be declared 0to5.

{

cout<<"enter value in element"<<i<<"?"; //print statement<<i means 0,1,2,3,4,5, with "enter value in element".

cin>>arr[i]; //get the user value assigned

i++; //increment operator

}

pos=0; //position pos is initialize=0;

cout<<"enter any value ="<<endl; //print the statement

cin>>n; //get the user value to assigned the position value.

i=0; //i is initialize=0

while(i<=4) //while condition (i should be< and equal to 4,0to4)

{

if(n==arr[i]) //if the user value is equal to array value by user .

{

pos=i+1; //0+1=1,0+2=2,0+3=3,0+4=4,0+5=5

break; //if the position is true ,not need to check all condition 0to4,break should be apply

}

i++; //inctrment operator i=i+1;

}

if(pos==0) //if (position==0) // print the statement value not found

{

cout<<"value not found"<<endl;

}

else //while print "value found at position"<<pos,address of position location also print

{

cout<<"value found at position="<<pos<<endl;

}

return 0;

}

//Confused//

/\*write a program that sorts one-dimensional array's values into descending order using bubble sort method.\*/

#include<iostream>

using namespace std;

main()

{

int array[10]={55,41,12,78,1,69,45,4,36,5}; //array value is assigned (1 to 10)

int u,i,temp;

cout<<"values of array before sorting\n";

for(i=0;i<=9;i++) //for loop condition

cout<<array[i]<<endl; // print array value is assigned {55.......5}

**//array sorting,The processing of arranging data of a list in a specific order is called sorting array.**

**// The order of data items in a list may be in ascending order or descending order.**

u=9;

while(u>=1)

{

i=0;

while(i<u)

{

if(array[i]<array[i+1])

{

temp=array[i];

array[i]=array[i+1];

array[i+1]=temp;

}

i++;

}

u--;

}

cout<<"\n values of array after sorting\n";

for(i=0;i<=9;i++)

cout<<array[i]<<"\t";}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*write a program that inputs data into a table and displays on the screen in tabular form\*/

#include<iostream>

using namespace std;

main()

{

int table[3][4] ,r ,c; //integer variable table [3 rows][4 column],row and column assign.

r=0; //r is initialize zero

while(r<=2) //while loop (condition , when r 0 = should < and equal to 2 it will stop the condition)

{

for(c=0;c<=3;c++) // for loop condition(column is initialize=0;column value should be <=and equal to 3,it will be check the cloumn condition,increment condition)

{

cout<<"enter value for row\t"<<r

<<"\tand column\t"<<c<<"\t"; //print the row and column statement

cin>>table[r][c]; //get the value by row and column with 3 row and 4 column

}

r++; // increment row 0 to 2 == to 3 times

}

cout<<"\n values in tabular form"<<endl;

for(r=0;r<=2;r++) //for loop condition //for row (r is initialize 0;r<=2;increment operaotr

{

for(c=0;c<=3;c++) //for loop condition//column is initilize =0;column should be less then and equal to 3;increment column 0 to 3 step by step

cout<<table[r][c]<<"\t"; //get the value in table form row and column

cout<<endl; // each value should be in new line,concept will be cleared while executed the program.

}

}

//searching Maximum and Minimum values from table

#include<iostream>

using namespace std;

main()

{

int table[3][4];

int r,c,max,min,mr\_r,mx\_c,mn\_r,mn\_c;

r=0;

while(r<=2)

{

for(c=0;c<=3;c++)

{

cout<<"enter value for row"<<r

<<"and column"<<c<<"?";

cin>>table[r][c];

}

r++;

}

//finding maximum & minimum values

max=min=table[0][0];

for(r=0;r<=2;r++)

for(c=0;c<=3;c++)

{

if(max>table[r][c])

{

max=table[r][c];

mx\_r=r; //error

mx\_c=c;

}

if(min>table[r][c])

{

min=table[r][c];

mn\_r=r;

mc\_c=c; //error

}

}

cout<<"\n Maximum value"<<max<<endl;

cout<<"Location["<<mx\_r<<"]["<<mx\_c<<"]\n"; //error

cout<<"\nMinimum value"<<min<<endl;

cout<<"Location["<<mn\_r<<"]["<<mn\_c<<"]\n";

}

/\*write a program that initializes values in a table and searches a value from it.

if value found,then it displays the value and its location otherwise it displays message

"value not found".It also displays the values of the table in tabular form.\*/

#include<iostream>

using namespace std;

main()

{

int table[3][4]={{22,33,2,8},

{5,87,15,33},

{23,65,33,23}};

int r,c,lr,lc,val,check=0;

cout<<"Enter a value to search ?";

cin>>val;

//searching a value from table

for(r=0;r<=2;r++)

for(c=0;c<=3;c++)

{

if(val==table[r][c])

{

lr=r;

lc=c;

check=1;

}

}

// display values of table in tabular form

for(r=0;r<=2;r++)

{

for(c=0;c<=3;c++)

cout<<table[r][c]<<"\t";

cout<<endl;

}

if(check==0)

cout<<"\n value not found"<<endl;

else

cout<<"\n value found at location "<<[lr][lc]; //error

}

/\*write a program that initialize values in a one-dimensional array.

it passes the array as argument to a function which coumputes the sum of the array.

The function should return the calculated sum to the calling function\*/

#include<iostream>

using namespace std;

int sum(int []);

main()

{

int arr[5]={4,6,3,4,7},i;

cout<<"values of array\n";

i=0;

for(;i<=4;i++) //loop condition(initialize value i=0;condition ;increment)

cout<<arr[i]<<endl;

/\*

arr[5]=4+6+3+4+7;

sum=sum+arr[5];

cout<<"\n sum of array="<<sum;

\*/

cout<<"\n sum of array ="<<sum(arr);

}

//definition of sum() function

int sum(int x[])

{

int s=0;

for(int i=0;i<=4;i++)

s=s+x[i];

return s;

}

#include<iostream>

using namespace std;

int pick(int [3][5],int[15]);

main()

{

int r,c,size,odd[15]={0};

int arr[3][5]={{14,61,13,14,71},

{34,55,33,22,11},

{12,56,44,2,98}};

//int pick (int[3][5],int[15]);

size = pick(arr,odd); //size variable is initialize to pick the array and carry odd values like 13,71,33,11,98

cout<<"\n values of table\n"; //print statement

for(r=0;r<=2;r++) //for loop condition(row initialize=0;r<=2;r=r+1)

{

for(c=0;c<=4;c++)

cout<<arr[r][c]<<"\t"; //print the array and column

cout<<endl; // ?

}

cout<<"\n values of array odd\n";

for(r=0;r<=size-1;r++) //?

cout<<odd[r]<<endl;

cout<<"Total odd values are="<<size;

}

// definition of pick() function

int pick(int tab[3][5],int x[15])

{

int c=0;

for(int i=0;i<=2;i++)

for(int j=0;j<=4;j++)

if(tab[i][j]%2==1)

{

x[c]=tab[i][j];

c++;

}

return c;

}

/\*write a program that initializes values in two tables A and B. it adds tables A and B and stores result into talbe c.

it also displays the values of three tables in tabular form on the screen.\*/

#include<iostream>

using namespace std;

void addition(int[3][3],int [3][3],int [3][3]); //function prototype

void print(char [15],int [3][3]); //

main()

{

/\*Matrix A [2 3 8]

[5 8 3]

[2 6 3] and other so on \*/

int A[3][3]={{2,3,8},{5,8,3},{2,6,3}};

int B[3][3]={{2,3,1},{6,1,2},{6,2,1}};

int C[3][3]; //integer variable is initialize c [3 rows and 3 column]

//void addition(int[3][3],int [3][3],int [3][3]);

//void print(char [15],int [3][3]);

addition(A,B,C); // initialization of addition(A+B+C)

print("Matrix A:",A); // print matrix A on screen

print("Matrix B:",B); //print matrix B on screen

print("addition of A & B",C); //Addtion of A and B in C matrix

return 0;

}

// definition of addition()function

void addition(int x[3][3],int y[3][3],int z[3][3]) //function calling

{

int r,c; //integer variable is initialize row and column

for(r=0;r<=2;r++) //for loop condition(initialization row;condition;increment)

for(c=0;c<=2;c++) //similar column condition

z[r][c]=x[r][c]+y[r][c]; //Addition of Matrix c=A+B == z=x+y

}

// definition of print() function

void print(char str[],int R[3][3]) // ???????????????? //print function

{ //????????????

int r,c;

cout<<endl<<str<<endl;

for(r=0;r<=2;r++)

{

for(c=0;c<=2;c++)

cout<<R[r][c]<<"\t";

cout<<endl;

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Error on output \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/\*write a program to initialize values in two tables A and B.Multiply matrix A and B and Store result into table C.

Display the values of three matrices in tabular form on the screnn.\*/

/\*The multiplication of two matrices A and B is

Possible only when the number of clumns of matrix A is equal to number of rows of matrix B.\*/

#include<iostream>

using namespace std;

void multiply(int [3][3],int[3][3],int[3][3]);

void print(int [25] ,int [3][3]);

main()

{

int A[3][3]={{2,3,8},{4,5,3},{2,6,3}};

int B[3][3]={{3,9,1},{1,2,4},{6,2,1}};

int C[3][3];

multiply(A,B,C);

cout<<("Matrix A:",A);

cout<<("Matrix B:",B);

cout<<("Multiplication of A & B:",C);

}

//function for multiplying data of tables

void multiply(int x[3][3],int y[3][3],int z[3][3])

{

int r,c,k,sum;

for(r=0;r<=2;r++)

{

for(c=0;c<=2;c++)

{

for(k=0,sum=0;k<=2;k++)

sum=sum+x[r][k]\*y[k][c];

z[r][c]=sum;

}

}

}

// function for displyaing data of tables in tabular form

void print(char str[],int R[3][3])

{

int r,c;

cout<<endl<<str<<endl;

for(r=0;r<=2;r++)

{

for(c=0;c<=2;c++)

cout<<R[r][c]<<"\t";

cout<<endl;

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Error\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/\* Write a program that transposes a matrix A and displays result of the screen.

Note:A matrix which is obtained by the interchanging of rows and columns of a matrix is called the transpose

of given matrix and is denoted by A superscript of t.

\*/

#include<iostream>

using namespace std;

void tanspose (int [2][4],int[4][2]);

void print(char [25],int[4][2]);

main()

{

int A[2][4]={{24,31,18,51},{4,15,13,2}};

int T[4][2];

char str[25];

cout<<endl<<"Matrix A:"<<endl;

for(int r=0;r<=1;r++)

{

for(int c=0;c<=3;c++)

cout<<A[r][c]<<"\t"

cout<<endl;

}

transpos e(A,T);

print("Transpose of A:",T);

}

//definition of transpose()function

void tranpose (int x[2][4],int y[4][2])

{

int r,c;

for(r=0;r<=1;r++)

for(c=0;c<=3;c++)

y[c][r]=x[r][c];

}

void print(char str[],int R[4][2])

{

int r,c;

cout<<endl<<str<<endl;

for(r=0;r<=3;r++)

{

for (c=0;c<=1;c++)

cout<<R[r][c]<<"\t";

cout<<endl;

}

}

Pointer is a variable which is capable of storing the initial storage of object which its want to point to.

Pointer is a special variable that is capable of storing some address. Computer memory is a collection of consecutive memeory locations.the size of each memory location is one byte. Each memory location has a unique address.The memeory addresses start from 0 and rises to 1,2,3 and so on.

When a computer program is loaded into memory for execution,it occupies a certain range of memory locations.similarly ,each function and variable defined in the program also occupies certain memory locations.For example,a char type variable occupies one byte and double type variable occupies eight bytes.

/\*write a program that declares and assigns a value'245' to a variable 'x'.

It displays the value and address of this variable on the screen.\*/

#include<iostream>

using namespace std;

main()

{

int x=245;

cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Zeeshan Ali\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

cout<<"Values of variable=\t"<<x<<endl;

cout<<"Address of variable=\s\t"<<&x<<endl;

return 0;

}

When we write int x; computer allocates the **memory int memory allocated is 4 bytes ,x is a variable ,x represent the 4 bytes of location from memory.**&x represents address of x, pointer is looks like a name which allocate or point to the memory of address

Px is a pointer variable and, &x is a address of x of memory location.

/\*write a program tha assigns values to variables using pointer variables

it computes the sum of values and displays the result on the screen.\*/

#include<iostream>

using namespace std;

main()

{

int x,y,\*px,\*py; //in simple program we can only write a program in this scope.

px=&x; //px is a pointer variable which is initilize the the address of x ,where x=5.

py=&y; //px is a pointer variable which is initilize the the address of x ,where x=8.

\*px=5; //pointer of x is initialize =5

\*py=8; //pointer of y is initialize=8

cout<<"sum of x and y="<<x+y<<endl;

return 0;

}

/\*write a program that initializes a pointer with memory address of a variable.

it gets a value into variable through pointer and displays the value of the variable with reference to the name of variable\*/

#include<iostream>

using namespace std;

main()

{

float x; //float value is x variable

float \*px=&x; // &x,address of x is initialize the point of x.

cout<<"enter any value = "; //print the statement

cin>>\*px; //get the user value in floating number

cout<<"value of variable x="<<x; // print the value of variable x which the user entered.

return 0;

}

/\*write a program that declares and initializes variables x of 'int' type and y of 'float' type.

It displays the values and address of these variables on the screen using pointer to 'void'.\*/

#include<iostream>

using namespace std;

main()

{

int x=1122; //integer type variable x=1122

float y=25.5; // float type variable is initialize=25.5

//important

void \*pv; //A void type pointer can hold memory address of variable of any data type.

pv=&x; //address of x is initialize pointer variable of memory pv.

cout<<"value of x="<<x<<endl; //1122, enter value of x

cout<<"address of x="<<pv<<endl; // address of x point to the memory address of x.

pv=&y; //&y,is the address of y,pv is a variable to point the address of y.

cout<<"value of y="<<y<<endl; //print the value of y.

cout<<"address of y="<<pv<<endl; //memory address of y is printed.

return 0;

}

/\*.\*/

#include<iostream>

using namespace std;

main()

{

int a[5]={5,51,4,11,579},\*pa;

pa=a; //pointer variable pa is initialize arry a[5]

cout<<"value of array ="<<endl;

for(int i=0;i<=4;i=i+1)

{

cout<<\*(pa+i)<<endl; // cout<<\*(pa++)<<endl;

}

return 0;

}

/\*write a program that inputs values into an array and displays the odd values of array on the screen using pointer notation.\*/

#include<iostream>

using namespace std;

main()

{

int a[6],i,\*pa;

pa=a;

cout<<"enter six values ="<<endl;

for (i=0;i<=5;i++)

cin>>a[i]; //array is always declare in this scope, by get the user value

cout<<"\n odd values are ="<<endl;

for(i=0;i<=5;i++) //for loop condition

//decision condition ,if the pointer variables remainder is 1 print the odd value

if(\*(pa+i)%2==1)

cout<<pa[i]<<endl;

return 0;

}

/\*write a program that inputs values into an array and displays the odd values of array on the screen using pointer notation.\*/

#include<iostream>

using namespace std;

main()

{

int a[6],i,\*pa; //integer variable: is initialize a[6 arrays values],i and pointer of a.

pa=a; //pointer of a is initialize=a variable.

cout<<"enter six values"<<endl; //print the statement

for(i=0;i<=5;i++) //for loop condition (initialization;condition;increment operator)

cin>>a[i]; //a[i] is also written as \*pa++. //get the user value

pa=&a[5]; //address of 6 user value is initialize of pointer a.

cout<<"\n values in reverse order:"<<endl; // Print the statement.

for(i=0;i<=5;i++) //i is initialize 0;user value is <=5 ,loop continue,when i is equal to 5 terminated.

cout<<\*pa--<<endl; //get the reverse order by the user pointer values

return 0; //

}

/\*.\*/

#include<iostream>

using namespace std;

main()

{

int \*p[3],x,y,z,i;

x=51,y=42,z=33;

p[0]=&x;

p[1]=&y;

p[2]=&z;

cout<<"values of variables x,y,z are:\n";

for(i=0;i<=2;i++)

cout<<\*p[i]<<endl;

}

/\*.write a program that computes the cube of a number using pointer as argument to function\*/

#include<iostream>

using namespace std;

void cube (int\*); // function prototype//void declared is no return value in program.

main()

{

//void cube (int\*);

int n=3;

cout<<"value of n before function call:"<<n<<endl;

cube(&n); //cube &n,address of n=3

cout<<"value of n after function call:"<<n<<endl; //function declaration

}

//Definition of cube() function

void cube(int \*px)

{

\*px=\*px \* \*px \* \*px; //calling the function

}

/\*.write a program that computes the cube of a number using pointer as argument to function\*/

#include<iostream>

using namespace std;

void cube (int\*); // function prototype//void declared is no return value in program.

main()

{

void temp (int \*);

int x[]={4,5,6,7,8,9};

temp(x);

for(int i=0;i<=5;i++)

cout<<x[i]<<endl;

}

//Definition of cube() function

void temp(int \*p)

{

int i=0;

for(;i<=5;i++) //calling the function

{

\*p=\*p\*5;

p++;

}

}

/\*.\*/

#include<iostream>

using namespace std;

void swap (int\*,int\*); // function prototype//void declared is no return value in program.

main()

{

int x,y;

cout<<"enter 1st value =";

cin>>x;

cout<<"enter 2nd value =";

cin>>y;

swap(&x,&y);

cout<<"values after swapping"<<endl;

cout<<"enter 1st value ="<<x<<endl;

cout<<"enter 2nd value ="<<y<<endl;

}

// definitio of function

void swap(int \*a,int \*b)

{

int temp;

temp=\*a;

\*a=\*b;

\*b=temp;

}

/\* write a program that swaps(exchange) two values by passing pointers as arguments to the function.\*/

#include<iostream>

using namespace std;

void swap (int\*,int\*); // function prototype//void declared is no return value in program.

main()

{

int x,y; //integer variable is declared x and y.

cout<<">>>>>>>>>ZEESHAN ALI>>>>>>>>"<<endl;

cout<<">>>>>>>>>03088604026>>>>>>>>"<<endl;

cout<<"enter 1st value ="; //print the statement

cin>>x; // get the user value

cout<<"enter 2nd value ="; //print the statement

cin>>y; //get the user value

swap(&x,&y); //swap definition interchange the value(&x,address of x value,&y address of y value)

cout<<"values after swapping"<<endl;

cout<<"enter 1st value ="<<x<<endl;

cout<<"enter 2nd value ="<<y<<endl;

}

// definition of function

void swap(int \*a , int \*b)

{

int temp; //int variable is declared (temp) temporary

temp=\*a; //temp is initialize the pointer a.

\*a=\*b; //pointer(a) is initialize pointer b(\*b)

\*b=temp; //while pointer b is initialize temp.

}

The process of allocating and de-allocating memory is known as memory management.The operating system performs this activity during execution of application programs.In C++,the programs can be developed to allocate and de-allocate memory dynamically. The pointers play very important role for allocating memory dynamically.

The ‘new’ operator is used to allocate memory locations during program execution. This operator allocates memory for the specified object and returns its memory address to the pointer variable. The ‘new’ operator can be used to allocate memory for simple variables,arrays,or user-defined objects etc.

Once memory is allocated with ‘new’ operator, it cannot be de-allocated automatically. If the pointer goes out of scope, the allocated memory becomes unavailable.The ‘delete’ operator is used to de-allocate memory occupied by an object.The de-allocated memory can be used by operating system for other purposes.

/\* write a program that allocates memory locations for two integer values usin 'new' operaotr.It inputs values into the allocated memeory

locations and computes the sum of these values.It also displays the result on the computer screnn.

At the end,it de-allocates the allocated memory locations using 'delete'operator.\*/

#include<iostream>

using namespace std;

main()

{

int \*p1,\*p2; //int variable is pointer p1,pointer p2.

p1=new int; // p1 = new memory is allocated of integer type in pointer p1.

p2=new int; // p2 = new memory is allocated of integer type in pointer p2.

cout<<"Enter Ist value ="<<endl;

cin>>\*p1; //get the user value allocated by pointer p1

cout<<"Enter 2nd value ="<<endl;

cin>>\*p2;

int sum; //sum declare of two integer values

sum=\*p1+\*p2;

cout<<"sum of values ="<<sum<<endl; //value 1+value2

delete p1; // de-allocate the memory of pointer 1

delete p2; // de-allocate the memory of pointer 2

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\* write a program that allocates memory locations for an array of 'int' type usig 'new' operator

and inputs a value for the length of the array during the program execution.

it also inputs values into the array and finds out the largest value from the array.\*/

#include<iostream>

using namespace std;

main()

{

float \*p1,\*p2 ,max;

int n ,i;

cout<<"enter the size of array = ";

cin>>n;

p1=new float[n];

p2=p1; //assigns the starting address of p1 to p1.

cout<<"enter"<<n<<"values in array\n";

for (i=0;i<n;i++)

cin>>\*p1++;

/\*float max;

int i;\*/

i=0;

p1=p2;

max=\*p1;

while(i<n)

{

if(max<\*p1)

max=\*p1;

\*p1++;

i++;

}

cout<<"The Largest value is :"<<max;

delete[] p1;

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*write a program that initialize string variable and computes the total number of characters of string.\*/

#include<iostream>

#include<string>

using namespace std;

main()

{

char str[]="Pakistan Punjab "; //one space take one null character.

int i=0;

for( ; str[i]!='\0';i++) //find the length of string //loop started length of Rawalpindi is 0,1,2,3,4,......10,answer is 10

cout<<"length of "<<str<<"is"<< i<<endl;

return 0;}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*write a program that initializes strings to a table of char type and displays on the screen.\*/

#include<iostream>

using namespace std;

main()

{

/\* array of strings can be initialized like other arrays.In this case,the first index indicates athe total number of strings while the second index indicates the maximum size of each string,including null character.\*/

//char str[number of string][each string character maximum size]

char str[5][15]={"karachi","peshawar","quetta","lahore","islamabad"};

for(int i=0;i<=4;i++) //for loop int i is intialize=0;if i < and equal to 4 its true karachi print and so on ,i=i+1

cout<<str[i]<<endl; // print the output statement.

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Char str[]=”welcome”; in this statement,character type array’str’ is declared and string ”welcome ” is assigned to it. The array ‘str’ of type ‘char’ will be created having 8 elements (including null character).

We can initialize a pointer variable of type char. For example;

Char \*pstr=”Welcome”;

The string will be stored somewhere in memeory and the starting address of the string will be stored in poinbter ‘pstr’.A string defined as a poiner is considered more flexible than defined as an array. The main difference betweenvarialbe ‘str’ and pointer variable ‘pstr’ is that:

* **‘str’ is a pointer constant and it cannot be changed.**
* **‘pstr’ is a pointer variable and it can be changed.**

/\*write a porgram that inputs a sting in a string variable and searches a specific character from it using pointer notation\*/

#include<iostream>

using namespace std;

main()

{

char str[50],\*pstr,ch; //char type variable declare str[size of string 50],pointer str,character.

int loc=0; //integer location initialzie=0;

pstr=str; //pointer string initialize string.

cout<<"enter a string ?"; //print the statement

cin.getline(str,25); //cin.getline , it is used to get string input from user

cout<<"enter a character ?"; //print the statement

cin>>ch; // get the user character

while(\*pstr!='\0') //while loop(condition,pointer of string not initialize null character)

{

if(\*pstr==ch) //if pointer is equal to character

loc=1; //location initialize 1

\*pstr++; //pointer incrememt operator.

}

if(loc==1) //if location is equal to 1

cout<<"character found"; //print this statement

else //otherwise print another.

cout<<"character not found";

return 0;

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

/\*write a porgram that inputs data into a string variable and finds out the length of string using pointer notation\*/

#include<iostream>

using namespace std;

main()

{

char str[16],\*pstr;

int i=0;

pstr=str;

cout<<"Enter a string ?"<<endl;

cin>>str;

while(\*pstr!='\0') //find the length of string//it means pointer string is not initialize with null character size.

{

i++; //increment of i value

\*pstr++; //increment of pointer string variable while pstr=str.

}

cout<<"length of string is:"<<i;

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*write a porgram that inputs data into a string variable and finds out the length of string using pointer notation\*/

#include<iostream>

using namespace std;

main()

{

char str[50],\*pstr,ch; //string[size 50],pointer string,character

int l=0; //integer variable l,letter is initialize is 0

pstr=str; //string is initialize is pointer string

cout<<"Enter a string ?"; //print the statement

cin.getline(str,25); //cin.getline//get the user string value (string,size of character)

while(\*pstr!='\0') //while condition(pointer string variable not initialize a null character)//fing the length of string

{

if(\*pstr=='a'||\*pstr=='A')

l++;

\*pstr++;

}

// int l=0;

cout<<"Total letters (a or A)are :"<<l; // get the total letter in string character being used.

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*write a porgram that copies one string to another using pointer as argument to function\*/

#include<iostream>

using namespace std;

void copy(char\*,char\*); //void is used to no returen vale ..variable data type ,declaration (statement),prototype declaration

main()

{

char str1[15],str2[]="Zeeshan Ali"; // char str1[size of string],str2[size is not initialize]="always written in double cottes"

// void copy(char\*,char\*); //also written in this scope

copy(str1,str2); //definition of copy statement declaration(string1,string2)

cout<<"Data of str1:"<<str1<<endl; //print the string 1

cout<<"Data of str2:"<<str2<<endl; //print the string 2

}

//function definition

void copy(char \*s1, char \*s2) //function calling statement

{

int i; // integer variable declare i

for(i=0;s2[i]!='\0';i++) //for loop condition

s1[i]=s2[i]; //string 2 is initialize string 1

s1[i]='\0'; //null character is also initialize in string 1,while s1==s2.

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*write a program that converts a string to uppercase letters using pointers as arguments to function\*/

#include<iostream>

using namespace std;

void uppercase(char\*);

main()

{

char str[25];

cout<<"enter a string ?";

cin>>str;

uppercase(str);

cout<<"string in uppercase:"<<str<<endl;

}

//function definition

void uppercase(char \*s)

{

for(int i=0;s[i]!='\0';i=i+1)

if(s[i]>=97&&s[i]<=122)

s[i]=s[i]-32;

} enter the string value : Zeeshan

Output: ZEESHAN

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

A **Structure** is a collection of related dta items.T eh data items are called elements or members or fields of the structure.

A structure is unlike an array.All elements in an array are of same type.But in Structure,the elements cab be of different types.The structure is used to define user-defined data types or varialbesl known as structures.

A simple variable can store only one data value.

A structure variable can store multiple data values. These values may be of different data type or same type.

Structures are used for file processing. A file is a collection of related redcords.A record is a collection of related fields of same or different data types. The structure is used to define the template or structure of a record.

Structure is one of the important building block in understanding of objects and classes. A structure is a collection of related data members, while a class is a collection of data members and member functions .

The definition of structure is terminated by a semicolon**(;).**The use of semicolon is optional.

**Accessing Structure Members**

A member of structure is accessed by using a dot operator(.). The dot operator is also called the member access operator.

To access a member of a specific structure:

The name of **structure variable** is written on the **left side of dot operator**.

The name of **structure member** is written on the **right side** of dot operator.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*write a program that defines a structure "student" that has the following members:

roll\_no,name,city and phone ,it declare the structure variable"std",inputs data into it and displays on screen\*/

#include<iostream>

using namespace std;

struct student // student represent the name of structure

{

int roll\_no; //represent the int data type,it is also called structure tag.

// An array is a collection of data that holds fixed number of values of same type.

//while Pointers are used in C++ program to access the memory and manipulate the address.

/\*char data types name[number of values collection defined is 20]\*/

char name[20],city[15],phone[15]; //represent the char data type,it is also called structure tag.

}; // address of structure is terminated. ; is important for declare the structure

int main()

{

student std; //student tag name and variable is std;

cout<<"enter roll number ?"; // input values into data member of structure variable

/\*variable.member ,name of structure variable is written on LHS of dot operator & structure member is written on the right side of operator\*/

cin>>std.roll\_no; // A member of structure is accessed by using the dot operator,also called member access operator.

cout<<"enter the name of student ?";

cin>>std.name;

cout<<"enter name city of student ?";

cin>>std.city;

cout<<"enter phone number of student";

cin>>std.phone;

cout<<endl;

/\*following values of data member student of structure variable displays on screen\*/

cout<<"Roll no.:"<<std.roll\_no<<endl;

cout<<"Name. :"<<std.name<<endl;

cout<<"City. :"<<std.city<<endl;

cout<<"Phone no.:"<<std.phone<<endl;

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*write a program that defines a structure "time" that has the following members of data type int:h,m and s.

it declare the structure variable "pst",inputs data into data members and displays time on the screen in standard time format.\*/

#include<iostream>

using namespace std;

struct time

{

int h,m,s; //int type varibe is declared

}; //structure declaration

main()

{

time st; //name of structure variable ,time is name and st is structure variable.

cout<<"enter hours =";

cin>>st.h; // get the hour by user (st is variable is defined on above ,h is data member)

cout<<"enter minutes =";

cin>>st.m;

cout<<"enter seconds =";

cin>>st.s;

cout<<"Time:"<<st.h<<":"<<st.m<<":"<<st.s; //print the time on screen

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*write a program that defines a structure "employee"having the following numbers

emp\_no,basic\_pay,house\_rent,medical\_allow,conveyance\_allow,net\_pay

it declares structure variable "emp",and inputs values into its members

emp\_no and basic\_pay.it also computes the allowances based on basic pay and computes net pay.

The allowances are computed as under:

House rent is 45% of basic pay

Medical allowance is 5% of the basic pay

conveyance allowance is 10% of the basic pay\*/

#include<iostream>

using namespace std;

struct employee //structure declare employee

{

int emp\_no; //int variable emp\_no.

float basic\_pay,house\_rent,medical\_allow,

conveyance\_allow,net\_pay; //float variable declaration

};

main()

{

employee emp; // structure variable declare emp.

cout<<"enter employee number = "; //print the statement

cin>>emp.emp\_no; //get the user value (emp is variable and .emp\_no is data member structure)

cout<<"enter employee basic pay = "; //print the statement

cin>>emp.basic\_pay; //variable emp and .basic\_pay is structure data member.

emp.house\_rent=emp.basic\_pay\*45/100.0; //house rent is initialize of basic pay of 45%.

emp.medical\_allow=emp.basic\_pay\*5/100.0; //medical\_allowance is initialize of basic pay of 5%.

emp.conveyance\_allow=emp.basic\_pay\*10/100.0; //conveyance allowance is initialize of basic pay of 10%.

//only salary or pay is initialize of employee basic pay+employee house rent+employee medical allowance.

emp.net\_pay=emp.basic\_pay+emp.house\_rent

+emp.medical\_allow

+emp.conveyance\_allow;

cout<<endl; //all of the above statement should be declared on distinct newline

cout<<"Basic pay :"<<emp.basic\_pay<<endl; //basic pay statement is printed.

cout<<"House rent:"<<emp.house\_rent<<endl;//House rent statement is printed.

cout<<"Medical allowance:"<<emp.medical\_allow<<endl;//Medical allowance statement is printed.

cout<<"conveyance allowance:"<<emp.conveyance\_allow<<endl;//conveyance allowance statement is printed.

cout<<endl;

cout<<"Net pay:"<<emp.net\_pay<<endl; //total net pay or net salary with out allowances.

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*writ ea program tha defines a structure"employee" that has the following members

code of string type

name of string type

salary of float type

it declares the structure variable"emp",initialize values to it and displayes the values on the screen\*/

#include<iostream>

using namespace std;

struct employee

{

char code[10];

char name[50];

float salary;

};

main()

{

/\* name of defined structure ,name of structure variable,

{represent the list of values to be assigned to the sturcutre variable.separated by commas} \*/

employee emp={"AD001","Bashir",25782.0} ;

cout<<"Code of employee:"<<emp.code<<endl; //print the defined structure of employee code.

cout<<"Name of employee:"<<emp.name<<endl; //print the defined structure of above (#21 line) of employee code.

cout<<"Salary of employee:"<<emp.salary<<endl; //print the defined structure of employee salary ,mentioned #21 linde code.

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*\*/

#include<iostream>

using namespace std;

struct employee

{

char code[10];

char name[50];

float salary;

};

main()

{

/\* name of defined structure ,name of structure variable,

{represent the list of values to be assigned to the sturcutre variable.separated by commas} \*/

employee emp1={"AD001","Faizan",25782.0};

employee emp2={"AD058","Zeeshan",16712.0},temp;

cout<<"contents of variables before swapping\n";

cout<<"contents of emp1"<<endl;

cout<<"code:"<<emp1.code<<endl;

cout<<"Name:"<<emp1.name<<endl;

cout<<"Salary:"<<emp1.salary<<endl;

cout<<"contents of emp2"<<endl;

cout<<"code:"<<emp2.code<<endl;

cout<<"Name:"<<emp2.name<<endl;

cout<<"Salary:"<<emp2.salary<<endl;

//swapping the values

temp=emp1; //temp variable is intialize=employee1

emp1=emp2; //emp1 is initialize the content of employee 2

emp2=temp; //emp2 is initialize the temp variable while temp is initialize the content of emp1.

cout<<"\n contents of variable after swapping\n";

cout<<"contents of emp1"<<endl;

cout<<"code:"<<emp1.code<<endl;

cout<<"Name:"<<emp1.name<<endl;

cout<<"Salary:"<<emp1.salary<<endl;

cout<<"contents of emp2"<<endl;

cout<<"code:"<<emp2.code<<endl;

cout<<"Name:"<<emp2.name<<endl;

cout<<"Salary:"<<emp2.salary<<endl;

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*Write a program that defines a structure "student" tha thas the following members

roll\_no of 'int'data type

name of string data type

makrs of array of float data type

total of float type

it declares a structure of above-mentioned structure and inputs data into it.

if findsout the sum of marks and stores the result in the member 'total'.It also displays the record of student on the computer screen\*/

#include<iostream>

using namespace std;

struct student

{

int roll\_no;

char name[20];

float marks[3];

float total;

};

main()

{

student std;

int i;

float sum=0.0;

cout<<"enter the roll number ?";

cin>>std.roll\_no;

cout<<"enter the name of student ?";

cin>>std.name;

for(i=0;i<=2;i=i+1)

{

cout<<"enter marks of subject"<<i+1<<"?";

cin>>std.marks[i];

sum=sum+std.marks[i];

}

cout<<endl;

cout<<"Roll no :"<<std.roll\_no<<endl;

cout<<"Name :"<<std.name<<endl;

for(i=0;i<=2;i++)

{

cout<<"Marks of subject"<<i+1<<":"<<std.marks[i]<<endl;

cout<<"Total Marks:"<<sum<<endl;

}

// retrurn 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*Write a program that defines a structure "student" tha thas the following members

roll\_no of 'int'data type

name of string data type

makrs of array of float data type

total of float type

it declares a structure of above-mentioned structure and inputs data into it.

if findsout the sum of marks and stores the result in the member 'total'.It also displays the record of student on the computer screen\*/

#include<iostream>

using namespace std;

struct student

{

int roll\_no;

char name[200];

float marks[6];

float total;

};

main()

{

student std;

int i;

float sum=0.0;

cout<<"enter the roll number ?";

cin>>std.roll\_no;

cout<<"enter the name of student ?";

cin>>std.name;

for(i=0;i<=6;i=i+1)

{

cout<<"enter marks of subject"<<i+1<<"?";

cin>>std.marks[i];

sum=sum+std.marks[i];

}

cout<<endl;

cout<<"Roll no :"<<std.roll\_no<<endl;

cout<<"Name :"<<std.name<<endl;

for(i=0;i<=6;i++)

{

cout<<"Marks of subject"<<i+1<<":"<<std.marks[i]<<endl;

cout<<"Total Marks:"<<sum<<endl;

}

// retrurn 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*write a program that defines a structure"student" to store name and marks of 5 subjects. It declares a structure variable

of type 'student' and initialize values into it.

it calculates the total marks and average marks and then displays the complete record of the student\*/

#include<iostream>

using namespace std;

struct student

{

char name[12]; //character variable is used for alphabetic character, name type of variable 0 to 11+1=12 ,1 for null character.

float marks[5]; //float variable is used for real values,number of array is 5 because it is initialize 85,75....79

};

main() //body of pragram starting

{

//name of structure ,variable of structure = character name ,marks of structure builtin identified.

student std ={"Zeeshan Ali",{85,75,95,74,79}};

float total=0; //float type variable variable name is initialize 0.

float avg; //vaerage variable define

for(int i=0;i<=4;i=i+1) //for loop,i is initialize =0;0is<=4 value is true of 'i' increment the i value

total=total+std.marks[i]; // total is initialize 0 marks+ total of string marks from loop value

avg=total/5; // average formaula

cout<<"Name :"<<std.name<<endl; //print the name array of structure already defined

for(int i=0;i<=5;i++) // for loop condition ,step by step marks initialze\*\*\*\*\*\*

{

cout<<"Marks of Subject"<<i+1<< ":"<<std.marks[i]<<endl;

}

cout<<"Total Marks :"<<total<<endl; // sum of total marks print

cout<<"Average Marks:"<<avg<<endl; //average marks print

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*write a program that defines a structure"student" to store name and marks of 5 subjects. It declares a structure variable

of type 'student' and initialize values into it.

it calculates the total marks and average marks and then displays the complete record of the student\*/

#include<iostream>

using namespace std;

struct student

{

char name[12]; //character variable is used for alphabetic character, name type of variable 0 to 11+1=12 ,1 for null character.

float marks[5]; //float variable is used for real values,number of array is 5 because it is initialize 85,75....79

};

main() //body of pragram starting

{

//name of structure ,variable of structure = character name ,marks of structure builtin identified.

student std ={"Zeeshan Ali",{85,75,95,74,79}};

float total=0; //float type variable variable name is initialize 0.

float avg; //vaerage variable define

for(int i=0;i<=4;i=i+1) //for loop,i is initialize =0;0is<=4 value is true of 'i' increment the i value

total=total+std.marks[i]; //for total marks, is initialize 0 marks+ total of string marks from loop value

avg=total/5; // average formaula

cout<<"Name :"<<std.name<<endl; //print the name array of structure already defined

for(int i=0;i<=5;i++) // for loop condition ,step by step marks initialze\*\*\*\*\*\*

{

cout<<"Marks of Subject"<<i+1<<":"<<std.marks[i]<<endl;

}

cout<<"Total Marks :"<<total<<endl; // sum of total marks print

cout<<"Average Marks:"<<avg<<endl; //average marks print

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*write a program that defines a structure 'Student' that contains three fields

roll\_no,name,and city and then declares an array of structure to store

record of 3 students.It inputs records of 3 students into an array of structure and then displays on the screen.\*/

#include<iostream>

using namespace std;

struct student

{

int roll\_no;

char name[20],city[25];

};

main()

{

student std[3]; //record of 3 studetns initialization

int i;

for (i=0;i<=2;i++)

{

cout<<endl; //each statement is initialize for new line

cout<<"Enter record #"<<i+1<<endl; //print the statement<<0+1=1,0+2=2,0+3=3.

cout<<"Enter roll no. ?"; //print the statement of roll number

cin>>std[i].roll\_no; //get the user value of roll no in structure program,std is variable & data member structure

cout<<"Enter name ?"; // print the statement of student name

cin>>std[i].name; //get the name by user

cout<<"Enter city ?"; // print the statement of city of student

cin>>std[i].city; //get the city name from user

}

for (i=0;i<=2;i++) //for print the screen of record,roll #,name and city.

{

cout<<endl;

cout<<"Record #"<<i+1<<endl;

cout<<"Roll Number:"<<std[i].roll\_no<<endl;

cout<<"Name :"<<std[i].name<<endl;

cout<<"City :"<<std[i].city<<endl;

}

//return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*write a program that defines a structure to store roll number and marks of a student.

it declares an array of structure to store the record of five students.

it inputs the records of five students and displays the record of the studetn having highest marks.\*/

#include<iostream>

using namespace std;

struct student

{

int roll\_no;

float marks;

};

main()

{

student std[5]; //studetn variable is array structure is declared std[5].

int i,p; //integer variable i and p

float max; //float variable max

for(i=0;i<=4;i++) //0to5 time loop condition apply

{

cout<<endl;

cout<<"Enter Roll Number\t:";

cin>>std[i].roll\_no;

cout<<"Enter Marks\t:";

cin>>std[i].marks;

}

max=std[0].marks; //highest value display on screen,max is initialize marks

for(i=1;i<=4;i++) //for loop condition 0 to 5

if(std[i].marks>max) //if student marks is greater than max or highest marks

{

max=std[i].marks; //max=marks

p=i; //in this case p=i

}

cout<<"\nHighest marks of studetn\n";

cout<<"Roll Number:"<<std[p].roll\_no<<endl;

cout<<"Marks:"<<std[p].marks<<endl;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*write a program that initializes records of five students to an array of structure"std" during its declaration.

This program sorts the records in ascending order on the field 'name'and then displays the records on the screen.\*/

#include<iostream>

using namespace std;

struct student

{

int roll\_no;

char name[20],city[15];

};

main()

{

student std[5]={{1,"M.Khalil","Karachi"},

{2,"Faizan Ali","sialkot"},

{3,"Zeeshan Ali","Islamabad"},

{4,"Noor ","Canada"},

{5,"Mariam","Karachi"}

};

student temp;

int i,u;

//sorting records

for(u=4;u>=1;u--)

for(i=0;i<u;i++)

if(strcmp(std[i].name,std[i+1].name)>0)////////\*\*\*\*\*\*\*\*\*\*\*\*error\*\*\*\*\*\*

{

temp=std[i];

std[i]=std[i+1];

std[i+1]=temp;

}

cout<<"\nsorted records\n";

for(i=0;i<=4;i++)

cout<<std[i].roll\_no<<"\t"<<

std[i].name<<"\t"<<

std[i].city<<endl;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#include<iostream>

using namespace std;

struct addr

/\*The member of a structure can be of structure type.

when a structure is a member of another structure,it is called nested sturcture

The members of nested structure are accessed by using more than one dot operator(.),the first dot operator refers the member variable

of outer sturcture.The second dot operator refers the inner sturcture and so on.\*/

{

int roll\_no;

char name[12],city[15];

};

struct subject

{

addr x;

float marks;

};

main()

{

subject rec;

cout<<endl;

cout<<"Enter roll number=";

cin>>rec.x.roll\_no;

cout<<"Enter Name=";

cin>>rec.x.name;

cout<<"Enter City=";

cin>>rec.x.city;

cout<<"Enter Marks=";

cin>>rec.marks;

cout<<endl;

cout<<"Roll number:"<<rec.x.roll\_no<<endl;

cout<<"Name:"<<rec.x.name<<endl;

cout<<"City:"<<rec.x.city<<endl;

cout<<"Marks:"<<rec.marks<<endl;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*write a program that defines a structure'Book' to store Book\_id,Book\_Name,and its price

.it also defins another sturcture'order' that contains two members:order\_id and array of book of 5 elements.

The program should declare/create a structure

variable of type order and input the data inoto. The Program should also display the data on the screen\*/

#include<iostream>

using namespace std;

struct book

{

int Book\_Id;

char Book\_Name[20];

float price;

};

struct order

{

int order\_Id;

book arr[5];

};

main()

{

order rec;

int i;

cout<<"\n Enter detail of 5 books:\n";

cout<<"Enter order ID =";

cin>>rec.order\_Id;

// cout<<"\n Enter detail of 5 books:\n"<<endl;

for(i=0;i<=4;i++)

{

cout<<"Book #"<<i+1<<endl;

cout<<"Enter Book ID ?"<<endl;

cin>>rec.arr[i].Book\_Id;

cout<<"Enter Book Name ?"<<endl;

cin>>rec.arr[i].Book\_Name;

cout<<"Enter Book Price ?"<<endl;

cin>>rec.arr[i].price;

}

cout<<endl<<"Order details of 5 Books is :"<<endl;

cout<<"Order ID:"<<rec.order\_Id<<endl;

cout<<"==========================================="<<endl;

cout<<"sr# \t ID \t Name \t Price:"<<endl;

for(i=0;i<=4;i++)

{

cout<<i+1<<'\t';

cout<<rec.arr[i].Book\_Id<<'\t';

cout<<rec.arr[i].Book\_Name<<'\t';

cout<<rec.arr[i].price<<endl;

}

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

INITIALIZING NESTED STRUCTURE

The nested structure is initialized in the similar way as array of structur is initialized.The values of the simple data members

are written in the normal form separated by commas.The values of the elements fo structure member are enclosed in braces.

The values enclosed in braces are also written separated by commas.

/\*write a program that initializes value to nested structure and then displays the values on the screen\*/

#include<iostream>

using namespace std;

struct addr

{

int roll\_no;

char name[20],city[15];

};

struct subject

{

addr x;

float marks;

};

main()

{

subject rec={{1,"Zeeshan Ali","Islamabad"},816}; //initialize nested structure in one line

cout<<"roll Number:\t"<<rec.x.roll\_no<<endl;

cout<<"Name:\t"<<rec.x.name<<endl;

cout<<"City:\t"<<rec.x.city<<endl;

cout<<"Marks:\t"<<rec.marks<<endl;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Passing Structures to function**

The structure can be passed as arguments to a function.If a structure is to be passed to a fauntion,then it must be defined before the declaration of the function that uses the structure type data as argument.

The type of arguments passing through function must match the type of parameter in the function header of function definition. The structure as argument can be passed to a function in two ways.

1. Passing structure by value ii) Passing structure by reference

#include<iostream>

using namespace std;

struct addr

{

int roll\_no;

char name[20];

float marks;

};

main()

{

addr rec={12,"Zeeshan",816.1}; //address variable (rec) record declaration in nested structure line

void t (addr); //void is used no return value ,t(address addr is declare assign the t.

t (rec); //t (rec)//for record line initialization

cout<<"OK"; //print the OK

}

//Definition of t=test function

void t(addr x)

{

cout<<"Roll No:"<<x.roll\_no<<endl;

cout<<"Name:"<<x.name<<endl;

cout<<"Mark:"<<x.marks<<endl; //marks will declare x.marks ,x is variable and mark is a member structure of x.

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Passing structure by Reference: A structure variable can also be passed to a function by reference .In this way,only the memeory address of the sturecture as argument is passed to the function.

#include<iostream>

using namespace std;

struct addr

{

int roll\_no;

char name[20];

float marks;

};

main()

{

addr rec;

void test(addr &);

test(rec);

cout<<"Roll No:"<<rec.roll\_no<<endl;

cout<<"Name:"<<rec.name<<endl;

cout<<"Mark:"<<rec.marks<<endl;

cout<<"OK";

}

//definition of test()function

void test(addr &x)

{

cout<<"Enter Roll No ?";

cin>>x.roll\_no;

cout<<"Enter Name? ";

cin>>x.name;

cout<<"Enter Mark ?";

cin>>x.marks;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*Write a program that inputs a date and displays on the screen by using class.\*/

#include<iostream>

using namespace std;

class ddd //class is declared

{

/\*The commands that determine wheter a member of a class can be accessed

from outside the class or not are called member access specifier \*/

/\*To restrict the access of a class members within the class

They can not be access from outside the class.private is used to acces as a private in the program\*/

private:

int y,m,d; //declare the object of the class

/\*Acces of class members outside the class as wellas with in class

The member of the class that follow the public specifier can be accessed from inside and outside the class

.public class is used to access anywhere of the program.\*/

public:

/\*void is variable which do not return value from user

,getdate(void) is used to initialize the getdate from user.\*/

void getdate(void){

cout<<"Enter year =";cin>>y;

cout<<"Enter Month=";cin>>m;

cout<<"Enter Day =";cin>>d;

}

void printdate(void) //for print the date in date formate

{

cout<<"Date is :"<<endl;

cout<<d<<"-"<<m<<"-"<<y;

}

}; //end of structure class program with closed curly braces and ; .

main() //for call the above program in main function

{

ddd dt; //variable | ddd means date,month and year ,dt is member function

/\* dt. is a dot operator, members of the class associated with a specific object are accessed by the dot operator for disply the date in Date format.\*/

dt.getdate(); /\* print the class of date .getdate and printdate is a member function of class\*/

dt.printdate();

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*Write a program by using a class that inputs and displays the data of a student on the computer screen.\*/

#include<iostream>

using namespace std;

class rec //class is declared

{

/\*The commands that determine wheter a member of a class can be accessed

from outside the class or not are called member access specifier \*/

/\*To restrict the access of a class members within the class

They can not be access from outside the class.

private is used to acces as a private in the program\*/

private:

char name[20]; //declare the object of the class

float age;

/\*Access of class members outside the class as well as with in class

The member of the class that follow the public specifier can be accessed from inside and outside the class

.public class is used to access anywhere of the program.\*/

public:

/\*void is variable which is used no return value of the program

,input(void) is used to initialize the user input.\*/

void input(void)

{

cout<<"Enter the name of Student =";cin>>name;

cout<<"Enter the age of Student =";cin>>age;

}

void print(void) //for print the student name and age.

{

cout<<endl;

cout<<"Name of Student :"<<name<<endl;

cout<<"Age of Student:"<<age<<endl;

}

}; //end of structure class program with closed curly braces and ; .

main() //for call the above program in main function

{

rec obj; //rec is a variable name of keyword to initialize the the object.

/\* obj. is a dot operator, members of the class associated with

a specific object are accessed by the dot operator for disply the date in Date format.\*/

obj.input();

/\* print the objecet of class to calling in main function\*/

obj.print();

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*write a program by using a class that inputs two values

using member function of the class and displays the sum of the two

values by using another member function of the class.\*/

#include<iostream>

using namespace std;

class sum

{

private:

int n1,n2;

public:

void input(int x,int y)

{

n1=x;

n2=y;

}

void print(void)

{

cout<<"Sum ="<<n1+n2;

}

};

main()

{

sum s;

int a,b;

cout<<"Enter the First Value:"; cin>>a;

cout<<"Enter the second Value:"; cin>>b;

s.input(a,b);

s.print();

return 0;}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*write a program using a class that gets the name of student and marks of three subjects. it calculates the total

marks and average marks.Each subject has a maximum of 100 marks.

It also displays the record of the studetn on the computer Screen.\*/

#include<iostream>

using namespace std;

class student //class of student

{

private:

char name[17]; // char is variable name and array value is declared number of 17 of the name.

float s1,s2,s3,total,avg; //float variable is used for student details

public:

void input\_data(void) //for get the value by user and input/print on the screen

{

cout<<"enter name of student= \t"; gets(name); //gets(name) is used for disply the complete name on screen \*important.

cout<<"enter the marks of 1st Subject= \t"; cin>>s1;

cout<<"enter the marks of 2nd Subject= \t"; cin>>s2;

cout<<"enter the marks of 3rd Subject= \t"; cin>>s3;

total=s1+s2+s3; //total=sum of three subjects

avg=total/3.0; //avergae=sum of total number/number of subject

}

void print\_data(void) //for print the students details

{

cout<<endl; //this line is used to make a one line space of display the output when user enter the value.

cout<<"Name of Students:"<<name<<endl;

cout<<"Marks of 1st Subject:"<<s1<<endl;

cout<<"Marks of 2nd Subject:"<<s2<<endl;

cout<<"Marks of 3rd Subject:"<<s3<<endl;

cout<<endl;

cout<<"Total Marks:"<<total<<endl; //print the total marks on screen and average marks

cout<<"Average Marks:"<<avg<<endl;

}

};

main() //call the function in main

{

student ostd;

/\*The public members of the class associated with a specificc object are accessed by using the dot operator.

The (. operator) is also called the class member of access operator.\*/

ostd.input\_data(); //for calling th input data

ostd.print\_data(); //for calling the print data

return 0;}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*.Write a program using class 'circle' that computes

and displays the area of a circle.The class 'circle' has the following members:\*/

#include<iostream>

using namespace std;

class circle //class is consist of circle class

{

private:

float r,area;

public:

void compute\_area(float n) // process of area

{

r=n; // r is a radius of circle while r is initialize n (number of value)

area=3.1417\*r\*r; //area is deifne is pi.(r)2

}

void print\_area(void) //for print the radius and area

{

cout<<endl;

cout<<"Radius of cricle ="<<r<<endl;

cout<<"Area of circle = "<<area<<endl;

}

};

main() //function of calling in main

{

circle c; //circle is define c

float radius; //float variable is define radius

cout<<"enter radius of circle ?";

cin>>radius;

/\* c. is a dot operator, members of the class associated with

a specific object are accessed by the dot operator for disply the area of radius format.\*/

c.compute\_area(radius);

c.print\_area();

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**M**ember function can also defined outside the class. Usually ,the member functions are defined outside the class,if the size of the functions are larger.

/\*write a program that uses two classes to explain the concept of member functions that are defined outside the classes.\*/

#include<iostream>

using namespace std;

class abc{

public:

void temp(void);

};

class xyz

{

public:

void temp(void);

};

main()

{

abc x;

xyz y;

x.temp();

y.temp();

}

/\*function are defined outside the main() ,outside the classes\*/

void abc::temp(void)

{

cout<<"Member function of class: abc"<<endl;

}

void xyz::temp(void)

{

cout<<"Member function of class: xyz"<<endl;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Error\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/\*write a program using class 'Time1' that displays the given time in standard format.

The class contain separate data members for hours,minutes & seconds.

\*/

#include<iostream>

using namespace std;

{

private:

int hours,minutes,seconds;

public:

void setdata(int,int,int);

void print\_time(void);

};

main()

{

time ttt;

int h,m,s;

cout<<"Enter hours =";cin>>h;

cout<<"Enter minutes =";cin>>m;

cout<<"Enter seconds =";cin>>s;

ttt.setdata(h,m,s);

ttt.print\_time();

}

void time1::setdata(int h1,int m1,int s1)

{

if((h1>=0 && h1<=12) &&

(m1>=0 &&,1<=59)

&&(s1>=0&&s1<=59))

{

hours=h1;

minutes=m1;

seconds=s1;

}

else

{

cout<<"Invalid time";

exit(0);

}

}

void time1::print\_time(void)

{

cout<<endl;

cout<<hours<<":"<<minutes<<":"<<seconds;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Error\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#include<iostream>

using namespace std;

class date1

{

private:

int day,month,year;

public:

void setdata(void);

void print\_date(void);

};

main()

{

date1 ddd;

ddd.setdata();

ddd.print\_date();

}

void date1::setdata(void)

{

date d;

getdate (&d);

day=d.da\_day;

month=d.da\_mon;

year=d.da\_year;

}

void date1::print\_date(void)

{

cout<<"your system date:";

cout<<day<<"/"<<month<<"/"<<year;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*write a program using class 'employee'that computes the allowances,

net pay and prints on the screen.The class hould have the following data members

emp\_no,basic\_pay,

house\_rent,

medical\_allow,

conveyance\_allow

and net\_pay.

The class should have the following member functions:

\* one member function 'inputdata'to get values into emp\_no and basic\_pay

\* one member function'compute\_pay' to compute allowances and net pay.

The allowances and net pay is calculated as:

1:house\_rent is 45% of basic\_pay.

2:medical\_allow is 15% of basic \_pay.

3:conveyance\_allow is 10% of basic\_pay.

4:net\_pay is calculated as= net\_pay=basic\_pay+house\_rent+medical\_allow+convey\_allow.

\*: one member function print\_pay to display the pay.\*/

#include<iostream>

using namespace std;

class employee //class of employee is declared.

{

private:

int emp\_no; //data member of employee

float basic\_pay,

house\_rent,

medical\_allow,

conveyance\_allow,net\_pay;

public:

void inputdata(void);

void compute\_pay(void);

void print\_pay(void);

};

main() //outside the data member calling in main

{

employee emp; //to get value into emp\_no and basic\_pay.

emp.inputdata(); //to get allowances and net pay.

emp.compute\_pay();

emp.print\_pay(); //to get the print on the screen allowance and net pay.

}

/\*The member functions are defined outside the class,if the size of the functions are larger.\*/

void employee::inputdata(void)

{

cout<<"Enter employee's number =";

cin>>emp\_no;

cout<<"Enter employee's basic pay =";

cin>>basic\_pay;

}

void employee::compute\_pay(void)

{

house\_rent=basic\_pay\*45/100; /\*1:house\_rent is 45% of basic\_pay\*/

medical\_allow=basic\_pay\*15/100; /\*2:medical\_allow is 15% of basic \_pay.\*/

conveyance\_allow=basic\_pay\*10/100;/\*3:conveyance\_allow is 10% of basic\_pay.\*/

net\_pay=basic\_pay+ /\*4:net\_pay is calculated as= net\_pay=basic\_pay+house\_rent+medical\_allow+convey\_allow.\*/

house\_rent+

medical\_allow+

conveyance\_allow;

}

void employee::print\_pay(void) //only for print the employee data on screen

{

cout<<endl;

cout<<"employee number:"<<emp\_no<<endl;

cout<<"Basic pay:"<<basic\_pay<<endl;

cout<<"House Rent:"<<house\_rent<<endl;

cout<<"Medical allowance:"<<medical\_allow<<endl;

cout<<"Conveyance allowance:"<<conveyance\_allow<<endl;

cout<<endl;

cout<<"Calculated net pay:"<<net\_pay<<endl;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*A constructor is a member function of a class with the same name as name of class itself.

A constructor is executed automatically each time when a new object of that class is created.

A constructor may have arguments but it has no return type.

The compiler will display an error if we specify a return type for a constructor,

we must not even write it as void.\*/

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#include<iostream>

using namespace std;

class abc

{

public:

/\*in this line no void return type variable is used ,it is called constructor\*/

abc (void) //a constructor may have arguments but it has no return type

{

cout<<"PM meaning Pakman"<<endl;

}

void temp(void) //compiler will disp;ay an error if we spedify a return type

{

cout<<"Welcome"<<endl;

}

};

main() //calling the program in main

{

abc x,y;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*write a program that explains the concept of initializing class object.\*/

#include<iostream>

using namespace std;

class temp //object of class is temp

{

private:

int n;

public:

temp(int x) //in this line no return value is assign in initial stage //constructor

{

n=x;

}

int fact(void) //integer variable is fact(void)

{

int c,f=1; //int variable is declared c and f is initialize 1.

c=n; //while c is initialize n.

for(;c>=1;c--) //for loop condition

f=f\*c;

cout<<"Factorial of"<<n<<"is:"<<f<<endl;

}

};

main() //call the main function

{

temp a(4),b(6); //temp factorial of 4 and 6 assign

cout<<a.fact();

cout<<b.fact();

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*write a program using class 'math'that passes three values as arguments to the constructor of class ans initialize

these values to data members x,y & z of the class.

1:one member function 'max' to find and return the maximum of three numbers.

2: second member function 'sum'to compute and return the sum of three numbers.

3:Third member functionn 'factorial'to find minimum number

of three numbers and computes its factorial and return to callin function.\*/

#include<iostream>

using namespace std;

class math

{

private:

int x,y,z; //initialize these values to data members x,y & z

public: //class 'math'that passes three values as arguments to the constructor.

math(int,int,int);

int max(void);

int sum(void);

int factorial(void);

};

main() //calling the main function.

{

math zz(40,4,13); //assign the values of max,sum and factorial.

cout<<"Maximum number is :"<<zz.max()<<endl; //maximum number is print <<zz.max(value is assigned 13).

cout<<"Sum of three number is :"<<zz.sum()<<endl;

cout<<"Factorial number is :"<<zz.max()<<endl;

}

/\*Member function can also defined outside the class.

Usually ,the member functions are defined outside the class

,if the size of the functions are larger..\*/

math::math(int a,int b,int c)

{

x=a;

y=b;

z=c;

}

int math::sum(void)

{

return x+y+z;

}

/\*max:member functions are defined outside the class

,if the size of the functions are larger.\*/

int math::max(void)

{

int mx; //integer variable is declare mx

mx=x; //mx is initialize x,while x is initialize a and a is initialize 13.

if(mx<y) //mx=x,13<4.

mx=y; //mx=y,13=4.

if(mx<z) //mx<z,mx=13,z=33

mx=z; //13=33.

return mx; // while return mx,mx=x,x=13

}

int math::factorial(void)

{

int mn; //integer variable initialize mn.

int f=1; // factorial is initialize 1 of integer variable.

mn=x; //mn initialize x,where x==13

if(mn>y)

mn=y;

if(mn>z)

mn=x;

for(;mn>1;mn--)

f=f\*mn;

return f;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

:Using an ordaniary function,a constructor may contain default arguments,when default values are provided to the constructor in its prototype then these values are automatically passed to the function even no values are provided in a constructor call.

A constructor is called automatically when object of class is created.please note that there may be only one default constructor per class.

#include<iostream>

using namespace std;

class pak

{

public:

pak(char[] ="Pakistan");

};

main()

/\*two objects p1 and p2 are declared of class 'pak' in the main() function.

Both of the objects belong to the same class but object 'p2' uses value

while 'p1' does not use any value .

when 'p1' is created, the constructor will use default value,

i.e Pakistan.

similaraly,when 'p2' is created the value "Islamabad"is passed to the constructor and default values is ignored. \*/

{

pak p1,p2("Islamabad");

}

pak::pak(char st[])

{

cout<<st<<endl;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Constructor Overloading: when a program that uses the construcotrs overloading is compiled ,C++ compiler marks them differently by examining the number of parameters,their order and data type.

When an object of class is created,the corresponding constructor of the class that matches the number of parameters is executed.

/\*write a program that explain the constructors overloading\*/

#include<iostream>

using namespace std;

class sum //class is declared sum

{

public:

sum(float x,float y)

{

cout<<"sum of "<<x<<"and"<<y<<"="<<x+y<<endl;

}

sum(float x,float y,float z)

{

cout<<"sum of "<<x<<","<<y<<"and"<<z<<"="<<x+y+z<<endl;

}

};

main() // calling the program

{

sum a(6.3,2.5,6.7);

cout<<endl;

sum b(7.6,3.5);

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*write a program by using class 'area' that uses the constructors overloading.

it initializes data to objects and finds out the area of Circle,Square,Rect-angle and Tri-angle.\*/

/\*

#include<iostream>

#include<math.h>

using namespace std;

class area

{

private:

int radius, a,b,c,length,height;

public:

area(int r)

{

radius=r;

}

area(int l,int w)

{

length=1;height=w;

}

area (int x,int y,int z)

{

a=x;

b=y;

c=z;

}

void rectangle(void);

void triangle(void);

void circle(void);

};

main() //calling the main function outside the member

{

area cir(5),

tri(144,167,150),

rect(12,6);

cir.circle();

rect.rectangle();

tri.triangle();

}

void area::rectangle(void)

{

cout<<endl;

cout<<"Height of Rectangle is ="<<height<<endl;

cout<<"Length of rectangle is ="<<length<<endl;

cout<<"Area of rectangle ="<<length\*height<<endl;

}

void area::circle(void)

{

cout<<endl;

cout<<"Radius of circle ="<<radius<<endl;

cout<<"Area of circle ="<<3.1417\*radius\*radius<<endl;

}

void area::triangle(void)

{

float s,res;

cout<<endl;

cout<<"1st side of triangle ="<<a<<endl;

cout<<"second side of triangle ="<<b<<endl;

cout<<"3rd side of triangle ="<<c<<endl;

s=(a+b+c)/2.0;

res=sqrt (s\*(s-a)\*(s-b)\*(s-c));

cout<<"Area of triangle ="<<res<<endl;}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*Write a program by defining two constructors of a class to find out the maximum values.\*/

#include<iostream>

using namespace std;

class findmax

{

private:

int mx;

public:

findmax(int x,int y,int z)

{

if(x>y)

if(x>z)

mx=x; //mx initialize x.

else

mx=z; //mx initialize z.

else if(y>z)

mx=y; //mx initialize y

else

mx=z; //while mx=z

cout<<"Maximum value among three numbers is :"<<mx<<endl;

}

findmax(int x,int y)

{

if(x>y)

mx=x; //mx=x.

else

mx=y; //mx initialize y.

cout<<"Maximum value between two numbers is"<<mx<<endl;

}

};

main() //calling to the main funcion

{

int a,b,c; //integer is assigned variable a,b,c.

a=25; //a is initialize 25.

b=14; //b is initialize 14.

c=67; //c is initialize 67.

findmax two(a,b); //findmax is keyword for compareing two values of a and b.

findmax three(a,b,c); //findmax is keyword for compareing three values of a,b & c.

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

A destructor is also a special member function of a class with the same name as the name of the class ,but complement operator ~ (tilde sign) is used before its name.

A destructor works opposite to a constructor.A destructor overloading is not allowed.Like a constructor ,destructor does not return any value.The destructor is alos does not take any arguments.

A destructor is used to destroy the reserve memory location which was made in constructor . destructor is always used in public not in a private member. We cannot take a argument in destructor while we can take a argument **test (int a int b) in** constructor.

/\*Write a program that explains the concept of destructor.\*/

#include<iostream>

using namespace std;

class test{

public: //public member is used in destructor.

test (int a,int b)

{

cout<<"Sum of two numbers:"<<a+b<<endl;

}

test(int a,int b,int c)

{

cout<<"sum of three numbers"<<a+b+c<<endl;

}

~test() //we did not use arguments in destructor.

{

cout<<"Welcome to destructor\n";

}

};

/\*Two objects are declared.The object x uses two values as arguments,so construcotr that uses two arguments will be called for execution when x is created.

similarly ,object y uses three values as arguments,so constructor that uses three arguments will be called for execution

when y is created.

After executing the last statement of the main()function,the object x and y will be destroyed one by one.

So the destructor function will be called automatically fo each object.\*/

main()

{

test x(6,3),y(5,5,10);

cout<<"Ok"<<endl;

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*Write a program that explains the concept of passing objects(by values) as arguments to function.\*/

#include<iostream>

using namespace std;

class temp{

private:

int m,n;

public:

void input(void)

{

cout<<"Enter 1st value =";

cin>>m;

cout<<"Enter the 2nd value=";

cin>>n;

}

void print(void)

{

cout<<"1st value"<<m<<endl;

cout<<"2nd value"<<n<<endl;

}

~temp()

{

cout<<"object is destroyed"<<endl;

}

};

// definition of test() function

void test(temp ob)

{

cout<<"Enter values through function object \n\n";

ob.input();

cout<<"values through function object are: \n\n";

ob.print();

}

main()

{

temp x;

void test(temp);

test(x);

cout<<"Enter value into object x \n\n";

x.input();

cout<<"Values of object x are \n\n";

x.print();

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

An object can be passed to a function by reference.In this case,only the memory address of the object as argument is passed to the function.

/\*Write a program that explains the use of an object as argument by reference to a function.\*/

#include<iostream>

using namespace std;

class sss

{

private:

int m,n,s;

public:

input(int x,int y)

{

m=x; //m should be initialize x

n=y; //n should be initialize y

s=m+n; //sum of m and n.

}

print(void) //for print the sum

{

cout<<"sum of"<<m<<"and"<<n<<"="<<s;

}

};

main() // calling out the main function

{

sss obj; //sss means 3 value sss is object.

void sum(sss &); // address of sum is declared

sum(obj); //sum of object is sum(obj)

obj.print(); //member function is print

}

//definition of function 'Sum'

void sum(sss &obx)

{

int a,b;

cout<<"Enter First Value :";cin>>a;

cout<<"Enter Second Value :";cin>>b;

obx.input(a,b);

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Returning Object from Function.

A function or member function can also return an object.The return statement is used inside the function to return an object to calling function.Usually , the object returned by a function is assigned to an object same type.

/\*.write a program that explains the concept of returning object from function.\*/

#include<iostream>

using namespace std;

class sss{

private:

int m,n,s;

public:

input(int x,int y)

{

m=x;

n=y;

s=m+n;

}

print (void)

{

cout<<"Sum of "<<m<<"and"<<n<<"="<<s;

}

};

main()

{

sss obj;

sss sum(void);

obj=sum();

obj.print();

}

//definition of function 'sum'

sss sum()

{

sss x;

int a,b;

cout<<"Enter the Ist value =";cin>>a;

cout<<"Enter the 2nd value =";cin>>b;

x.input(a,b);

return x;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*The private and protected member of any class cannot be access from outside the class.

sometimes,it may require to access private and protected members.

A friend function is a non-member function of the class.It allows for accessing the private and protected members of the

class from outside the class.

A function is declared to be a friend of a class by using the keyword "friend" before the name of the function

progotype.When a friend function is defined outside the class,no (::) resulation operator or

name of class is mentioned along with the function header.\*/

/\*Write a program that explains the concept of friend function.\*/

#include<iostream>

using namespace std;

class abc //class name is declared is abc

{

friend void xy(abc &,int,int); //friend function is defined.

private: //private member is defined ,declared two variable a,b.

int a,b;

public:

void print(void) //in public ,print private variable

{

cout<<"Sum ="<<a+b<<endl;

}

};

main() //calling the main function of sum of two variables

{

abc obj; //abc is class of declared an abj (an object)

xy(obj,3,5); //friend function is used in this line to assign the value of a=3 and b=5 while obj is abc.

obj.print(); //for print of abc class

cout<<"OK";

}

//definition of friend function 'xy'

void xy(abc &c,int x,int y)

{

c.a=x;

c.b=y;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*write a program by using classes A and B that initializes the

data to private data member 'a' of class A and private data member 'b'

of class B using their respective constructors.

it also uses a friend function to access and

compute the sum of the values of data members 'a' & 'b' of class A and Class B respectively.

The calculated value should return to the calling function.\*/

#include<iostream>

using namespace std;

class B;

class A //class A

{

private:

int a; //private data member 'a' of class A

public:

friend int sum(A,B);

A(int x)

{

a=x;

}

};

class B

{

private: //private data member 'b' of class b

int b;

public:

friend int sum(A,B); //friend function

B(int y)

{

b=y;

}

};

main() //sum function calling the main

{

A aa(5); //A is declared aa(5)

B bb(7); //B is declared bb(7)

cout<<"Sum ="<<sum(aa,bb); //sum statement is printed

}

//definition of friend function 'sum'

int sum(A x , B y) //sum of friend function is (A is declare x,and B function is declare y)

{

return(x.a+y.b); //return statement (sum of 5 and 7 is 12 )

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*write a program that explains the concept of friend class.\*/

#include<iostream>

using namespace std;

class A{

private:

float m;

protected:

float n;

public:

void assign(float x,float y)

{

m=x;

n=y;

}

friend class B;

};

class B

{

public:

float sum(A obj)

{

return obj.m+obj.n;

}

};

main()

{

A aa;

B bb;

float a,b;

cout<<"Enter first value ="<<endl;

cin>>a;

cout<<"Enter second Value ="<<endl;

cin>>b;

aa.assign(a,b);

cout<<"Sum of two values ="<<bb.sum(aa);

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*write a program that explains the concept of static data member of a class.\*/

#include<iostream>

using namespace std;

class A //class A is define

{

private:

int M,N; //M and N are two member in private class

static int ss; //one is static member ss

public:

A()

{

M=0;

N=0;

}

void add(int x) //for adding the M and N

{

M=M+x; //

N=N+x; //\*\*

ss=ss+x; //\*\*

}

void print(void) //print the value of M,N and SS

{

cout<<endl;

cout<<"value of M="<<M<<endl;

cout<<endl;

cout<<"value of N="<<N<<endl;

cout<<endl;

cout<<"value of ss="<<ss<<endl;

}

};

int A::ss=0; //This statement must be written to declare and initialize the static data member of the specified class.

main() // for calling the main function

{

A aa,bb,cc;

aa.add(2);

aa.print();

bb.add(5);

bb.print();

cc.add(3);

cc.print();

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

OOP stands for Object-Oriented Programming.

Procedural programming is about writing procedures or functions that perform operations on the data, while object-oriented programming is about creating objects that contain both data and functions.

Object-oriented programming has several advantages over procedural programming:

* OOP is faster and easier to execute
* OOP provides a clear structure for the programs
* OOP helps to keep the C++ code DRY "Don't Repeat Yourself", and makes the code easier to maintain, modify and debug
* OOP makes it possible to create full reusable applications with less code and shorter development time

**Tip:** The "Don't Repeat Yourself" (DRY) principle is about reducing the repetition of code. You should extract out the codes that are common for the application, and place them at a single place and reuse them instead of repeating it.

## C++ What are Classes and Objects?

Classes and objects are the two main aspects of object-oriented programming.

Look at the following illustration to see the difference between class and objects:

## class

Fruit

## objects

Apple

Banana

Mango

Another example:

## class

Car

## objects

Volvo

Audi

Toyota

So, a class is a template for objects, and an object is an instance of a class.

When the individual objects are created, they inherit all the variables and functions from the class.

## C++ Classes/Objects

C++ is an object-oriented programming language.

Everything in C++ is associated with classes and objects, along with its attributes and methods. For example: in real life, a car is an **object**. The car has **attributes**, such as weight and color, and **methods**, such as drive and brake.

Attributes and methods are basically **variables** and **functions** that belongs to the class. These are often referred to as "class members".

A class is a user-defined data type that we can use in our program, and it works as an object constructor, or a "blueprint" for creating objects.

Create a class called "MyClass":

class MyClass {       // The class  
  public:             // Access specifier  
    int myNum;        // Attribute (int variable)  
    string myString;  // Attribute (string variable)  
};

* The class keyword is used to create a class called MyClass.
* The public keyword is an **access specifier**, which specifies that members (attributes and methods) of the class are accessible from outside the class. You will learn more about [access specifiers](https://www.w3schools.com/cpp/cpp_access_specifiers.asp) later.
* Inside the class, there is an integer variable myNum and a string variable myString. When variables are declared within a class, they are called **attributes**.
* At last, end the class definition with a semicolon ;.
* The class keyword is used to create a class called MyClass.
* The public keyword is an **access specifier**, which specifies that members (attributes and methods) of the class are accessible from outside the class. You will learn more about [access specifiers](https://www.w3schools.com/cpp/cpp_access_specifiers.asp) later.
* Inside the class, there is an integer variable myNum and a string variable myString. When variables are declared within a class, they are called **attributes**.
* At last, end the class definition with a semicolon ;.

## Multiple Objects

You can create multiple objects of one class:

### Example

// Create a Car class with some attributes  
class Car {  
  public:  
    string brand;     
    string model;  
    int year;  
};  
  
int main() {  
  // Create an object of Car  
  Car carObj1;  
  carObj1.brand = "BMW";  
  carObj1.model = "X5";  
  carObj1.year = 1999;  
  
  // Create another object of Car  
  Car carObj2;  
  carObj2.brand = "Ford";  
  carObj2.model = "Mustang";  
  carObj2.year = 1969;  
  
  // Print attribute values  
  cout << carObj1.brand << " " << carObj1.model << " " << carObj1.year << "\n";  
  cout << carObj2.brand << " " << carObj2.model << " " << carObj2.year << "\n";  
  return 0;  
}

## Multiple Objects

You can create multiple objects of one class:

### Example

// Create a Car class with some attributes  
class Car {  
  public:  
    string brand;     
    string model;  
    int year;  
};  
  
int main() {  
  // Create an object of Car  
  Car carObj1;  
  carObj1.brand = "BMW";  
  carObj1.model = "X5";  
  carObj1.year = 1999;  
  
  // Create another object of Car  
  Car carObj2;  
  carObj2.brand = "Ford";  
  carObj2.model = "Mustang";  
  carObj2.year = 1969;  
  
  // Print attribute values  
  cout << carObj1.brand << " " << carObj1.model << " " << carObj1.year << "\n";  
  cout << carObj2.brand << " " << carObj2.model << " " << carObj2.year << "\n";  
  return 0;  
}

### Inside Example

class MyClass {        // The class  
  public:              // Access specifier  
    void myMethod() {  // ***Method/function defined inside the class***  
      cout << "Hello World!";  
    }  
};  
  
int main() {  
  MyClass myObj;     // Create an object of MyClass  
  myObj.myMethod();  // Call the method  
  return 0;  
}

To define a function outside the class definition, you have to declare it inside the class and then define it outside of the class. This is done by specifiying the name of the class, followed the scope resolution :: operator, followed by the name of the function:

### Outside Example

class MyClass {        // The class  
  public:              // Access specifier  
    void myMethod();   // Method/function declaration  
};  
  
// Method/function definition outside the class  
void **MyClass::myMethod()** {  
  cout << "Hello World!";  
}  
  
int main() {  
  MyClass myObj;     // Create an object of MyClass  
  myObj.myMethod();  // Call the method  
  return 0;  
}

## Parameters

You can also add parameters:

### Example

#include <iostream>  
using namespace std;  
  
class Car {  
  public:  
    int speed(int maxSpeed);  
};  
  
int Car::speed(int maxSpeed) {  
  return maxSpeed;  
}  
  
int main() {  
  Car myObj; // Create an object of Car  
  cout << myObj.speed(200); // Call the method with an argument  
  return 0;  
}

## Constructors

A constructor in C++ is a **special method** that is automatically called when an object of a class is created.

To create a constructor, use the same name as the class, followed by parentheses ():

### Example

class MyClass {     // The class  
  public:           // Access specifier  
    MyClass() {     // Constructor  
      cout << "Hello World!";  
    }  
};  
  
int main() {  
  MyClass myObj;    // Create an object of MyClass (this will call the constructor)  
  return 0;  
}

The constructor has the same name as the class, it is always public, and it does not have any return value.

## Constructor Parameters

Constructors can also take parameters (just like regular functions), which can be useful for setting initial values for attributes.

The following class have brand, model and year attributes, and a constructor with different parameters. Inside the constructor we set the attributes equal to the constructor parameters (brand=x, etc). When we call the constructor (by creating an object of the class), we pass parameters to the constructor, which will set the value of the corresponding attributes to the same:

### Example

class Car {        // The class  
  public:          // Access specifier  
    string brand;  // Attribute  
    string model;  // Attribute  
    int year;      // Attribute  
    Car(string x, string y, int z) { // Constructor with parameters  
      brand = x;  
      model = y;  
      year = z;  
    }  
};  
  
int main() {  
  // Create Car objects and call the constructor with different values  
  Car carObj1("BMW", "X5", 1999);  
  Car carObj2("Ford", "Mustang", 1969);  
  
  // Print values  
  cout << carObj1.brand << " " << carObj1.model << " " << carObj1.year << "\n";  
  cout << carObj2.brand << " " << carObj2.model << " " << carObj2.year << "\n";  
  return 0;  
}

Just like functions, constructors can also be defined outside the class. First, declare the constructor inside the class, and then define it outside of the class by specifying the name of the class, followed by the scope resolution :: operator, followed by the name of the constructor (which is the same as the class):

### Example

class Car {        // The class  
  public:          // Access specifier  
    string brand;  // Attribute  
    string model;  // Attribute  
    int year;      // Attribute  
    Car(string x, string y, int z); // Constructor declaration  
};  
  
// Constructor definition outside the class  
Car::Car(string x, string y, int z) {  
  brand = x;  
  model = y;  
  year = z;  
}  
  
int main() {  
  // Create Car objects and call the constructor with different values  
  Car carObj1("BMW", "X5", 1999);  
  Car carObj2("Ford", "Mustang", 1969);  
  
  // Print values  
  cout << carObj1.brand << " " << carObj1.model << " " << carObj1.year << "\n";  
  cout << carObj2.brand << " " << carObj2.model << " " << carObj2.year << "\n";  
  return 0;  
}

## Access Specifiers

By now, you are quite familiar with the public keyword that appears in all of our class examples:

### Example

class MyClass {  // The class  
  **public:**        // Access specifier  
    // class members goes here  
};

## Access Specifiers

By now, you are quite familiar with the public keyword that appears in all of our class examples:

### Example

class MyClass {  // The class  
  **public:**        // Access specifier  
    // class members goes here  
};

The public keyword is an **access specifier.** Access specifiers define how the members (attributes and methods) of a class can be accessed. In the example above, the members are public - which means that they can be accessed and modified from outside the code.

However, what if we want members to be private and hidden from the outside world?

In C++, there are three access specifiers:

* public - members are accessible from outside the class
* private - members cannot be accessed (or viewed) from outside the class
* protected - members cannot be accessed from outside the class, however, they can be accessed in inherited classes.
* In the following example, we demonstrate the differences between public and private members:

### Example

* class MyClass {  
    **public:**    // Public access specifier  
      int x;   // Public attribute  
    **private:**   // Private access specifier  
      int y;   // Private attribute  
  };  
    
  int main() {  
    MyClass myObj;  
    myObj.x = 25;  // Allowed (public)  
    myObj.y = 50;  // Not allowed (private)  
    return 0;  
  }
* If you try to access a private member, an error occurs:
* error: y is private // ((((((((output of the program)))))))

**Note:** It is possible to access private members of a class using a public method inside the same class.

**Tip:** It is considered good practice to declare your class attributes as private (as often as you can). This will reduce the possibility of yourself (or others) to mess up the code. This is also the main ingredient of the [Encapsulation](https://www.w3schools.com/cpp/cpp_encapsulation.asp) concept.

**Note:** By default, all members of a class are private if you don't specify an access specifier:

### Example

class MyClass {  
  int x;   // Private attribute  
  int y;   // Private attribute  
};

## Encapsulation

The meaning of **Encapsulation**, is to make sure that "sensitive" data is hidden from users. To achieve this, you must declare class variables/attributes as private (cannot be accessed from outside the class). If you want others to read or modify the value of a private member, you can provide public **get** and **set** methods.

## Access Private Members

To access a **private attribute**, use public "get" and "set" methods:

### Example

#include <iostream>  
using namespace std;  
  
class Employee {  
  private:  
    // Private attribute  
    int salary;  
  
  public:  
    // Setter  
    void setSalary(int s) {  
      salary = s;  
    }  
    // Getter  
    int getSalary() {  
      return salary;  
    }  
};  
  
int main() {  
  Employee myObj;  
  myObj.setSalary(50000);  
  cout << myObj.getSalary();  
  return 0;  
}

#### Example explained

The salary attribute is private, which have restricted access.

The public setSalary() method takes a parameter (s) and assigns it to the salary attribute (salary = s).

The public getSalary() method returns the value of the private salary attribute.

Inside main(), we create an object of the Employee class. Now we can use the setSalary() method to set the value of the private attribute to 50000. Then we call the getSalary() method on the object to return the value.

Why Encapsulation?

* It is considered good practice to declare your class attributes as private (as often as you can). Encapsulation ensures better control of your data, because you (or others) can change one part of the code without affecting other parts
* Increased security of data

## Inheritance (receive)

In C++, it is possible to inherit attributes and methods from one class to another. We group the "inheritance concept" into two categories:

* **derived class** (child) - the class that inherits from another class
* **base class** (parent) - the class being inherited from

To inherit from a class, use the : symbol.

In the example below, the Car class (child) inherits the attributes and methods from the Vehicle class (parent):

### Example

// Base class  
class Vehicle {  
  public:  
    string brand = "Ford";  
    void honk() {  
      cout << "Tuut, tuut! \n" ;  
    }  
};  
  
// Derived class  
**class Car: public Vehicle** {  
  public:  
    string model = "Mustang";  
};  
  
int main() {  
  Car myCar;  
  myCar.honk();  
  cout << myCar.brand + " " + myCar.model;  
  return 0;  
}

[Run example »](https://www.w3schools.com/cpp/showcpp.asp?filename=demo_inheritance)

#### Why And When To Use "Inheritance"?

- It is useful for code reusability: reuse attributes and methods of an existing class when you create a new class.

You learned from the [Access Specifiers](https://www.w3schools.com/cpp/cpp_access_specifiers.asp) chapter that there are three specifiers available in C++. Until now, we have only used public (members of a class are accessible from outside the class) and private (members can only be accessed within the class). The third specifier, protected, is similar to private, but it can also be accessed in the **inherited** class:

/\*write a program the result of a student by using the single inheritance.In this program

,the base class is "std\_address" and the derived class is "std\_result".\*/

#include<iostream>

using namespace std;

class std\_address{

private: //private class

int code; //attributes of private class

char name[15],phone[15]; //attributes of class

public: //access specifer public is access outside the the program

void input\_addr(void); //void is used for no return value

void print\_addr(void);

};

/\*class std\_result is define as a **derived class** (child class).

The keyword 'public'and the name of the base class is 'std\_address'.

objects of the derived class are able to access public members of the base class.It is called public inheritance.

The derived class'std\_result'can access the 'input\_addr' and 'print\_addr' member functions of the base class.

It can not access other private members of the base class.

\*/

class std\_result:public std\_address //derived class access of existing class (base class)

{

private:

float sub1,sub2,sub3,average,total;

public:

void input\_marks(void);

void result\_card(void);

};

main()

{

/\*An object 'res' of class "std\_result" is created.The member functions "input\_addr"and "print\_addr"of the class

"std\_address" are called through "res" object of class "std\_result".The object "res"of class "std\_result" also uses

its own member functions.\*/

std\_result res;

res.input\_addr();

res.input\_marks();

res.print\_addr();

res.result\_card();

}

void std\_address::input\_addr()

{

cout<<"Enter code of student =";

cin>>code;

cout<<"Enter name of student =";

cin>>name;

cout<<"Enter phone of student =";

cin>>phone;

}

void std\_address::print\_addr()

{

cout<<endl;

cout<<"Code :"<<code<<endl;

cout<<"Name :"<<name<<endl;

cout<<"Phone:"<<phone<<endl;

}

void std\_result::input\_marks()

{

cout<<"Enter marks of 1st subject =";

cin>>sub1;

cout<<"Enter marks of 2nd subject =";

cin>>sub2;

cout<<"Enter marks of 3rd subject =";

cin>>sub3;

total=sub1+sub2+sub3;

average=total/3;

}

void std\_result::result\_card()

{

cout<<"Marks of 1st subject :"<<sub1<<endl;

cout<<"Marks of 2nd subject :"<<sub2<<endl;

cout<<"Marks of 3rd subject :"<<sub3<<endl;

cout<<"Total Marks :"<<total<<endl;

cout<<"Average marks :"<<average<<endl;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*write a program the result of a student by using the single inheritance.In this program

,the base class is "std\_address" and the derived class is "std\_result".\*/

#include<iostream>

using namespace std;

class std\_address //base class is define

{

private: // private class is consist of student code # ,name of student and phone

int code;

char name[15],phone[15];

public: //publice is used to call the program anywhere

void input\_addr(void); // address of the imput

void print\_addr(void); //print the address

};

/\*class std\_result is define as a derived class (child class) .

The keyword 'public' is and the name of the base class is 'std\_address'.

objects of the derived class are able to access public members of the base class.It is called public inheritance.

The derived class'std\_result'can access the 'input\_addr' and 'print\_addr' member functions of the base class.

It can not access other private members of the base class.

\*/

class std\_result:public std\_address //derived class access of existing class (base class,parent class)

{

private:

float sub1,sub2,sub3,average,total;

public:

void input\_marks(void);

void result\_card(void);

};

main() //we can calling the program in main outside the program

{

/\*An object 'res' of class "std\_result" is created.The member functions "input\_addr"and "print\_addr"of the class

"std\_address" are called through "res" object of class "std\_result".The object "res"of class "std\_result" also uses

its own member functions.\*/

std\_result res;

res.input\_addr();

res.input\_marks();

res.print\_addr();

res.result\_card();

}

void std\_address::input\_addr() // for getting student information by user

{

cout<<"Enter code of student =";

cin>>code;

cout<<"Enter name of student =";

cin>>name;

cout<<"Enter phone of student =";

cin>>phone;

}

void std\_address::print\_addr() // print on screen the student code,name and phone

{

cout<<endl;

cout<<"Code :"<<code<<endl;

cout<<"Name :"<<name<<endl;

cout<<"Phone:"<<phone<<endl;

}

void std\_result::input\_marks()

{

cout<<"Enter marks of 1st subject =";

cin>>sub1;

cout<<"Enter marks of 2nd subject =";

cin>>sub2;

cout<<"Enter marks of 3rd subject =";

cin>>sub3;

total=sub1+sub2+sub3; //total marks of the subject

average=total/3; //average marks=total no of subject/no of subject quantity

}

void std\_result::result\_card() // for the statement of the marks subjects

{

cout<<"Marks of 1st subject :"<<sub1<<endl;

cout<<"Marks of 2nd subject :"<<sub2<<endl;

cout<<"Marks of 3rd subject :"<<sub3<<endl;

cout<<"Total Marks:"<<total<<endl; //print the total makrs

cout<<"Average marks :"<<average<<endl; //and average makrs

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*write a program the result of a student by using the single inheritance.In this program

,the base class is "std\_address" and the derived class is "std\_result".\*/

#include<iostream>

using namespace std;

class std\_address //parent class,base class is define (std\_address) name of class

{

private: // private class is consist of student code # ,name of student and phone

int code; //int attributes

char name[15],phone[15]; //char attributes

public: //public is access specifier for used outside the program

void input\_addr(void); //void is used for no return value address of the imput(void argument)

void print\_addr(void); //print the address

};

/\*class std\_result is define as a derived class.

The keyword 'public'and the name of the base class is 'std\_address'.

objects of the derived class are able to access public members of the base class.It is called public inheritance.

The derived class'std\_result'can access the 'input\_addr' and 'print\_addr' member functions of the base class.

It can not access other private members of the base class.

\*/

class std\_result:public std\_address //derived class access of existing class (base class)

{

private:

float sub1,sub2,sub3,average,total;

public:

void input\_marks(void);

void result\_card(void);

};

void std\_address::input\_addr() // for getting student information by user

{

cout<<"Enter code of student =";

cin>>code;

cout<<"Enter name of student =";

cin>>name;

cout<<"Enter phone of student =";

cin>>phone;

}

void std\_address::print\_addr() // print on screen the student code,name and phone

{

cout<<endl;

cout<<"Code :"<<code<<endl;

cout<<"Name :"<<name<<endl;

cout<<"Phone:"<<phone<<endl;

}

void std\_result::input\_marks()

{

cout<<"Enter marks of 1st subject =";

cin>>sub1;

cout<<"Enter marks of 2nd subject =";

cin>>sub2;

cout<<"Enter marks of 3rd subject =";

cin>>sub3;

total=sub1+sub2+sub3; //total marks of the subject

average=total/3; //average marks=total no of subject/no of subject quantity

}

void std\_result::result\_card() // for the statement of the marks subjects

{

cout<<"Marks of 1st subject :"<<sub1<<endl;

cout<<"Marks of 2nd subject :"<<sub2<<endl;

cout<<"Marks of 3rd subject :"<<sub3<<endl;

cout<<"Total Marks :"<<total<<endl; //print the total makrs

cout<<"Average marks :"<<average<<endl; //and average makrs

}

main() //we can calling the program in main outside the program

{

std\_result res; //std\_result is created by object of the class (res is object name)

res.input\_addr(); //get the student address

res.input\_marks(); //get the marks of student

res.print\_addr(); //print addr of student

res.result\_card(); //print in the form of result card shape.

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//write a program of **class \_car** which define it **model**,**brand** and **year of manufacturer** .

/\*

* #include tells the preprocessor to include these header file in the program.
* iostream is the header file which contains all the functions of program like cout, cin etc.
* when we run a program to print something, "using namespace std" says if

you find something that is not declared in the current scope go and check std.\*/

#include<iostream>

using namespace std;

class car //the class of car.

{

public: //access specifier

string brand; //string variable is used for word or sentence statement

string model;

int year; //int variable is used for year or integer

}; //endl of construct of class

/\*if you simply write int main() , which says the function does not take any argument

int main(void) means that the function takes NO arguments.\*/

int main() //calling the main of the program

{

//create an object of car

car carObj; //car is an object of the class\_car ,and carObj is member variable of the car.

carObj.model="X6";

carObj.brand="Ferrari";

carObj.year=1991; //year is only write **without** ("") inverted commas,quotation marks

//print attribute values

cout<<carObj.brand<<"\n "<<carObj.model<<"\n "<< carObj.year<<"\n";

return 0; // optional,the status of your program was successful and it could do what it was supposed to do.

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*

#include tells the preprocessor to **include these header file** in the program.

iostream is the **header file** which contains all the functions of program like cout, cin etc.

when we run a program to print something, "using namespace std" says if you find something that is not declared in the current scope go and check std.\*/

#include <iostream>

using namespace std;

class Employee //class of employee (employee is object)

{

private:

// Private attribute

int salary; //int attribute

//for getting the private acces we use setter and getter methods using by public

public:

// Setter

void setSalary(int s) //void is used for no return value ,setsalary (passing the argument)

{

salary = s; //salary is initialize s.

}

// Getter

int getSalary() //integer variable is used getsalary() without passing the argument

{

return salary; //return to the private attribute,must write (return).

}

};

int main() //calling the program in main

{

Employee myObj; //create an object of the Employee class,(Employee is object and object is the name of muObj).

myObj.setSalary(50000); //set the value of the private attribute to 50000.

cout << myObj.getSalary(); //then we call the object of getSalary.

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*write a program that defines two classes and creates public inheritance.\*/

#include<iostream>

using namespace std;

class x

{

private: //access spefifier

int a,b; //int attribute

protected: //access specifier

int m,n;

public: //public access specifier

void print (void) //using the public class for print the statement

{

cout<<"value of m of class x:"<<m<<endl;

cout<<"value of n of class x:"<<n<<endl;

}

};

//derived class y,: is used the relationship b/w derived and base class (base1 ,or base class public is used)

class y:public x

{

public:

void input(void) //void provide no return value of something ,input for print/get the the statements & value

{

cout<<"Enter value of m ?"; cin>>m;

//cin>>m;

cout<<"Enter value of n ?"; cin>>n;

//cin>>n;

}

};

int main() //call the program in main()

{

y temp; // y is created by class y,temp is object name

temp.input(); //calling and print the input values by user,for getting the 63 to 69 line.

temp.print(); //for print the 53 to 57 line//get the printed value on screen.

return 0; //optional used,exit the program

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Error Issue**

#include<iostream>

using namespace std;

class x{

private:

int a,b;

protected:

int m,n;

public:

void print(void) //error ,problem

{

cout<<"value of m of class x"<<m<<endl;

cout<<"value of n of class x"<<n<<endl;

}

};

class y : private x

{

public:

void input (void)

{

cout<<"Enter the value of m"; cin>>m;

cout<<"Enter the value of n"; cin>>n;

print();

}

};

main()

{

y temp;

temp.input();

temp.print();

return void;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /\*write a program that defines two classes and creates protected inheritance\*/

#include<iostream>

using namespace std;

class x{

private:

int a,b;

protected:

int m,n;

public:

void print (void)

{

cout<<"value of m of class x:"<<m<<endl;

cout<<"value of n of class x :"<<n<<endl;

}

};

class y:private x

{

public:

void input (void)

{

cout<<"Enter value of m"<<endl;

cin>>m;

cout<<"Enter value of n"<<endl;

cin>>n;

print();

}

};

int main()

{

y temp;

temp.input();

//temp.print();

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Destructor are called in the reverse order of constructor calls,so a derived class destructor is called before its base class destructor.

/\*concept of constructor and destructor with single inheritance.\*/

#include<iostream>

using namespace std;

class A

{

public:

A(void)

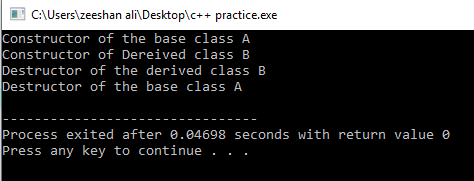
{

cout<<"Constructor of the base class A \n";

}

~A()

{

 cout<<"Destructor of the base class A \n";

}

};

/\*in this condition derived class B ,access the base class A .

output is 1st statement printed of base class of constructor and 2nd derived class printed.

But in destructor 3rd statement will be print of derived class of destructor and

then print destructor of base class. \*/

class B : public A //derived class B is acces the base class1 of (public A) with single inheritance

{

public:

B(void)

{

cout<<"Constructor of Dereived class B \n";

}

~B(void)

{

cout<<"Destructor of the derived class B \n";

}

};

main() //calling the main

{

B obj; // class B is print while class B is derived class,B is the object.

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Concept of constructor with **arguments** in single inheritance:

#include<iostream>

using namespace std;

class A{

private: // private access specifier

int x,y; //int attributes

public: //public access specifier

A (int m,int n) //class A (attributes with agguments)

{

x=m;

y=n;

}

void print(void) //for print the statement below,void is used no return value from program and declare the statements

{

cout<<"value of x ="<<x<<endl;

cout<<"value of y ="<<y<<endl;

}

}; //end of construct of class

class B:public A //derived class B is acces the Class A(base class)

{

private:

float i,j;

public:

B(int m,int n,float a,float b): A(m,n) // attributes(arguments) of class B of derived class: public, class A arguments

{

i=a; // i is initialize b

j=b; // j is initialize b

}

void show(void)

{

cout<<"value of i ="<<i<<endl;

cout<<"value of j ="<<j<<endl;

}

};

main() //calling the main function

{

B obj(6,2,19.2,6.5); //in derived class B, name of class is or object is B, and object of B class is declared obj(assign the values)

obj.print(); //print the statement of base class x and y which are declared 6 and 2.

obj.show(); //derived class print statement which are assigned (6,2,19.2,6,5)

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Error find**

/\*write a program that displays the record of an employee using base class and derived class.

The name of the base class should be "emp\_address"and name of the derived class should be "emp\_allowance".

Use the constructors with arguments in both classess and assign data to the object fo the derived class.\*/

#include<iostream>

using namespace std;

class emp\_address{

private:

int emp\_no;

char emp\_name[15];

public:

emp\_address(int ,char[]);

void print\_addr (void);

};

class emp\_allowance:public emp\_address

{

private:

float basic,hr,ma,ca,net\_pay;

public:

emp\_allowance(int,char[],float);

void print(void);

};

main()

{

emp\_allowance pay(142,"Zeeshan Ali",15743.58);

pay.print\_addr();

pay.print();

}

//definition of constructor of emp\_address class

emp\_address::emp\_address(int code,char mn[15])

{

emp\_no=code;

strcpy(emp\_name,nm); //error find,,,,,shit fuck you

}

//definition of print () member function of emp\_address class

void emp\_address::print\_addr()

{

cout<<endl;

cout<<"Employee number :"<<emp\_no<<endl;

cout<<"Employee name :"<<emp\_name<<endl;

}

//definition of constructor of emp\_allowance class

emp\_allowance::emp\_allowance(int code,char nm [15],float b):emp\_address(code,nm)

{

basic=b;

hr=basic\*45/100;

ma=basic\*15/100;

ca=basic\*10/100;

net\_pay=basic+hr+ma+ca;

}

//definition of print() member function of emp\_allowances class

void emp\_allowance::print()

{

cout<<"Basic pay :"<<basic<<endl;

cout<<"House rent :"<<hr<<endl;

cout<<"Medical allowance :"<<ma<<endl;

cout<<"Conveyance allowance :"<<ca<<endl;

cout<<"\n Net Pay :"<<net\_pay<<endl;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//write a program that inputs the record of a student and display on the screen by using the multiple inheritance.

#include<iostream>

using namespace std;

class std\_address //class name of class student address

{

private: // private access specifier

char name[15],city[15]; //attributes of class

public:

void input\_addr(void); // for input the address

void print\_addr(void); //for print the address

};

class std\_marks //class of student marks

{

private: //private access specifier

float sub1,sub2,sub3,average,total; //attributes of marks

public: // public access specifier is used to access the attributes anywhere of the program

void input\_marks(void); //for input the marks

void show\_marks(void); //for show the marks

};

//Derived class : base class (access specifier two classes-std\_addresss and std\_marks.

class std\_result: public std\_address,public std\_marks

{

public:

void show\_result(void) //for printed the statements of given below in this public class

{

cout<<"\n The complete record is:\n";

cout<<"============================\n";

print\_addr(); //for print the address like name of student and city name.

show\_marks(); //for showing the subject marks

}

};

/\*

main() //calling the program in main function outside of the program.

{

std\_result res; // derived class is created by the object of res.

res.input\_addr(); //for input the address

res.input\_marks(); //input the marks by user

res.show\_result(); // and show the result

}

\*/

void std\_address::input\_addr()

{

cout<<"Enter name of student =";

cin>>name;

cout<<"Enter name of city =";

cin>>city;

}

void std\_address::print\_addr()

{

cout<<endl;

cout<<"Name :"<<name<<endl;

cout<<"City :"<<city<<endl;

}

void std\_marks::input\_marks()

{

cout<<"Enter marks of 1st Subject = ";

cin>>sub1;

cout<<"Enter marks of 2nd Subject = ";

cin>>sub2;

cout<<"Enter marks of 3rd Subject = ";

cin>>sub3;

total=sub1+sub2+sub3;

average=total/3;

}

void std\_marks::show\_marks()

{

cout<<"Marks of first subject :"<<sub1<<endl;

cout<<"Marks of 2nd subject :"<<sub2<<endl;

cout<<"Marks of 3rd subject :"<<sub3<<endl;

cout<<"Total Marks:\t"<<total<<endl;

cout<<"Average Marks:\t"<<average<<endl;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*in this multiple inheritance constructor and destructor are declared but in this program

constructor is first declare of base1

and constructor is declared of derivea class.

and then

destructor are declared of

derive1 class

base1 class

base2 class

without arguments\*/

#include<iostream>

using namespace std;

class base1 //class ,name of class base1

{

public: //public access specifier

base1(void)

{

cout<<" Constructor of class base1 \n"; //constructor

}

~base1() //destructor

{

cout<<"Destructor of class base1 \n";

}

};

class base2 //class,name of classe base2

{

public:

base(void)

{

cout<<"Construcotor of class base2 \n";

}

~base2()

{

cout<<"Destructor of class base2 \n";

}

};

// derive class name is derive1:public(access specifier ) class name1 and class name2.

class derive1:public base1,public base2

{

public:

derive1(void)

{

cout<<"Constructor of class derive1 \n";

}

~derive1()

{

cout<<"Destructor of class deriv1 \n";

}

};

main()

{

derive1 x;

cout<<"Destructor are :"<<endl;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*Constructor in Multiple Inheritance with arguments \*/

#include<iostream>

using namespace std;

class base1

{

public:

base(int x,int y)

{

cout<<"sum of values passed to base1 :"<<x+y<<endl;

}

};

class base2{

public:

base2(char str[])

{

cout<<"String passed to base 2 :"<<str<<endl;

}

};

class derive1:public base1,public base2

{

public:

derive1(int x,int y,char str[],double d):

base1(x,y),base2 (str) //error

{

cout<<"Value passed to derive1 :"<<d;

}

};

main()

{

derive1 obj(6,2,"Zeeshan",2.5);

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#include<iostream>

using namespace std;

class B1{

public:

print()

{

cout<<"First Base Class :"<<endl;

}

};

class B2{

public:

print()

{

cout<<"Second Base Class :"<<endl;

}

};

class D1: public B1,public B2{

};

//if we write x.print(); ,error will be shown not executed the program

main()

{

D1 x;

x.B1::print();

x.B2::print();

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_