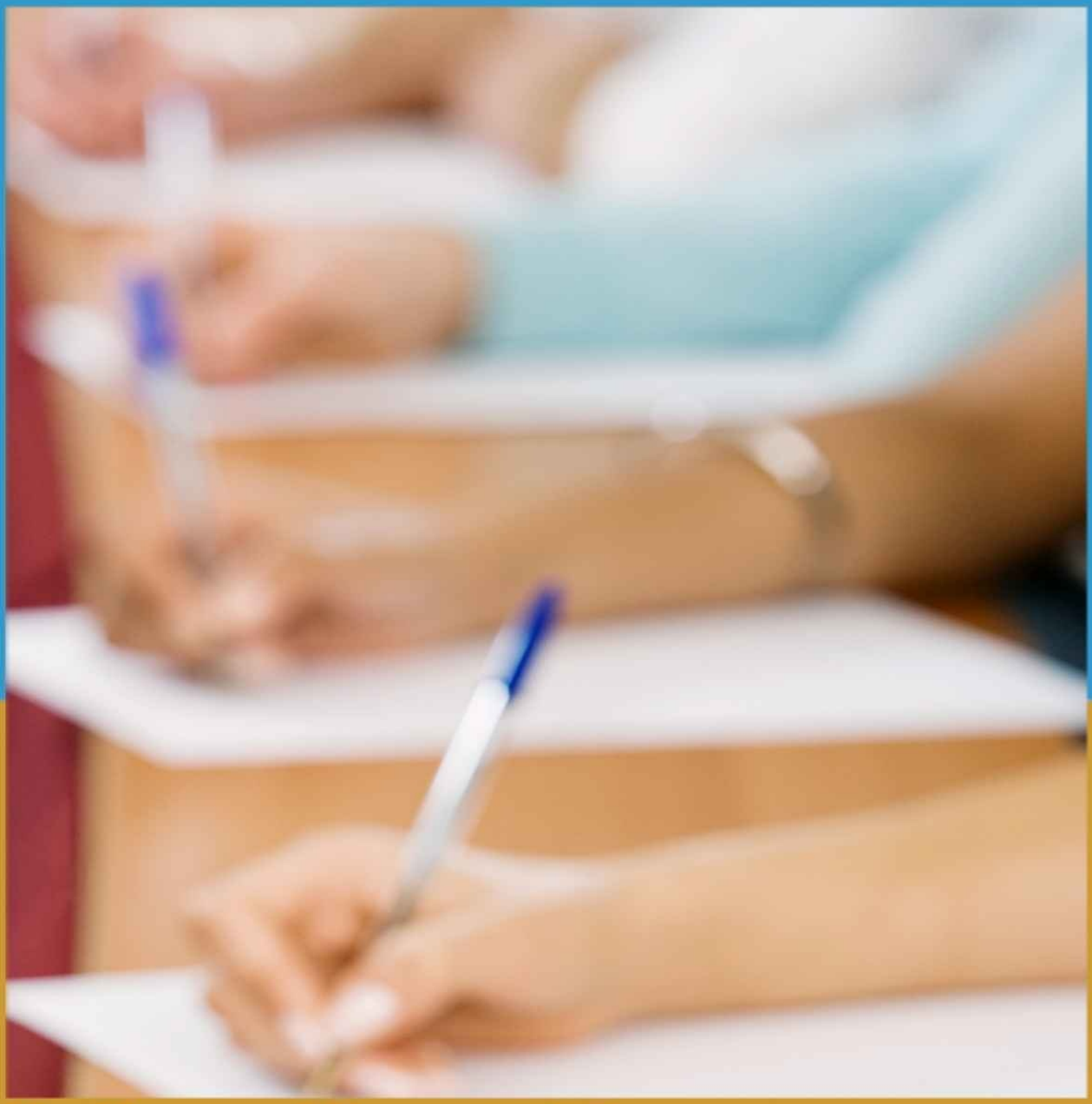
ANIKET PATASKAR



# 100+ C++ Programs with Output

For Students & Professionals

# C++ 100+ Programs with Output

By Aniket Pataskar

## Preface



**First Published :** Jan 2016

**Email :** pataskaraniket@gmail.com

**Contact :** +91 9423448273

Cover image by Aniket Pataskar

# About The Author

**Aniket O. Pataskar** is originally from Aurangabad, India. He is an IT Software engineer with working experience on Programming in various languages & Mac OSX Systems. Along with that he has interest in various social things. After the continuous efforts, he has completed this book.

Author also has skills in software development, iOS (iPhone, iPad & iPod) & AndroidApplication Development, Multimedia such as - Animation, Digital film making, Video editing, visual effects & Sound Editing. He also has his online learning course in Udemy Market.

**This book is dedicated to :**

My family for their continued love and support, and for always believing in me. I also want to dedicate this book to the God who always inspires me to do things.

This book would not have been possible without your love and understanding.

Thank you from the bottom of my heart.

# Index

## Program Name Page No.

# 1. Hello World C++ Example Program

```
#include<iostream>
#include<conio.h>

//Standard namespace declaration
using namespace std;

//Main Function
int main()
{

    //Standard Ouput Statement
    cout<<“My First C++ Program”;

    // Wait For Output Screen
    getch();

    //Main Function return Statement
    return 0;
}
```

**Output :**

```
My First C++ Program
```

# 2. if Statement Example Program in C++

```
#include<iostream>
#include<conio.h>

using namespace std;
int main()
{
    // Variable Declaration
    int a;

    //Get Input Value
    cout<<“Enter the Number :”;
    cin>>a;

    //If Condition Check
    if(a > 10)
    {
      // Block For Condition Success
      cout<<a<<” Is Greater than 10”;
    }

    // Wait For Output Screen
    getch();
    return 0;
 }
```

**Output :**

```
Enter the Number :15
```

15 Is Greater than 10

# 3. if..else Statement Example Program In C++

```
#include<iostream>
#include<conio.h>

using namespace std;

int main()
{

    // Variable Declaration
    int a;

    // Get Input Value
    cout<<“Enter the Number :”;
    cin>>a;

    //If Condition Check
    if(a > 10)
    {
      // Block For Condition Success
      cout<<a<<” Is Greater than 10”;
    }
    else
    {
     // Block For Condition Fail
     cout<<a<<” Is Less than/Equal 10”;
    }

    // Wait For Output Screen
    getch();
```

```
    return 0;
}
```

**Output 1:**

Enter the Number :20
20 Is Greater than 10

**Output 2:**

Enter the Number :5
5 Is Less than/Equal 10

# 4. For Loop Example Program In C++

```
#include<iostream>
#include<conio.h>

using namespace std;

int main()
{

    // Variable Declaration
    int a;

    // Get Input Value
    cout<<“Enter the Number :”;
    cin>>a;

    //for Loop Block
    for (int counter = 1; counter <= a; counter++)
    {
       cout<<“Execute “<<counter<<” time”<<endl;
    }

    // Wait For Output Screen
    getch();
    return 0;
 }
```

**Output:**

Enter the Number :5

Execute 1 time

Execute 2 time

Execute 3 time

Execute 4 time

Execute 5 time

# 5. While Loop Example Program In C++

```
#include<iostream>
#include<conio.h>

using namespace std;

int main()
{
    // Variable Declaration
    int a;

    // Get Input Value
    cout<<“Enter the Number :”;
    cin>>a;

    int counter = 1;
    //while Loop Block
    while (counter <= a)
    {
       cout<<“Execute While “<<counter<<” time”<<endl;
       counter++;
    }

    // Wait For Output Screen
    getch();
    return 0;
 }
```

**Output :**

Enter the Number :4

Execute While 1 time

Execute While 2 time

Execute While 3 time

Execute While 4 time

# 6. Do While Loop Example Program In C++

```
#include<iostream>
#include<conio.h>

using namespace std;

int main()
{
    // Variable Declaration
    int a;

    // Get Input Value
    cout<<“Enter the Number :”;
    cin>>a;

    int counter = 1;
    //Do while Loop Block
    do
    {
       cout<<“Execute Do While “<<counter<<” time”<<endl;
       counter++;
    }while (counter <= a);

    // Wait For Output Screen
    getch();
    return 0;
 }
```

**Output :**

```
Enter the Number :6
```

Execute Do While 1 time

Execute Do While 2 time

Execute Do While 3 time

Execute Do While 4 time

Execute Do While 5 time

Execute Do While 6 time

# 7. Simple Example Program For Constructor In C++

```
#include<iostream>
#include<conio.h>

using namespace std;

class Example        {
    // Variable Declaration
    int a,b;
    public:

     //Constructor
    Example()            {
    // Assign Values In Constructor
    a=10;
    b=20;
    cout<<“Im Constructor\n”;
    }

    void Display()    {
    cout<<“Values :”<<a<<”\t”<<b;
    }
};

int main()                {
        Example Object;
        // Constructor invoked.
        Object.Display();
```

```
    // Wait For Output Screen
    getch();
    return 0;
}
```

**Output :**

```
Im Constructor
Values :10      20
```

# 8. Simple Example Program For Parameterized Constructor In C++

```
#include<iostream>
#include<conio.h>

using namespace std;

class Example        {
    // Variable Declaration
    int a,b;
    public:

    //Constructor
    Example(int x,int y)            {
    // Assign Values In Constructor
    a=x;
    b=y;
    cout<<“Im Constructor\n”;
    }

    void Display()    {
    cout<<“Values :”<<a<<”\t”<<b;
    }
};

int main()                {
        Example Object(10,20);
        // Constructor invoked.
        Object.Display();
```

```
    // Wait For Output Screen
    getch();
    return 0;
}
```

**Output :**

```
Im Constructor
Values :10      20
```

# 9. Simple Example Program For Constructor Overloading In C++

```
#include<iostream>
#include<conio.h>

using namespace std;

class Example       {
   // Variable Declaration
   int a,b;
   public:

   //Constructor wuithout Argument
   Example()          {
   // Assign Values In Constructor
   a=50;
   b=100;
   cout<<”\nIm Constructor”;
   }

   //Constructor with Argument
   Example(int x,int y)          {
   // Assign Values In Constructor
   a=x;
   b=y;
   cout<<”\nIm Constructor”;
   }

   void Display()   {
```

```
    cout<<"\nValues :"<<a<<"\t"<<b;
    }
};

int main()              {
        Example Object(10,20);
        Example Object2;
        // Constructor invoked.
        Object.Display();
        Object2.Display();
        // Wait For Output Screen
        getch();
        return 0;
}
```

**Output :**

```
Im Constructor
Im Constructor
Values :10      20
Values :50      100
```

# 10. Simple Constructor Example Program For Find Prime Number In C++

**Algorithm:**

STEP 1: Start the program.

STEP 2: Declare the class as Prime with data members,
Member functions.

STEP 3: Consider the argument constructor Prime() with integer
Argument.

STEP 4: To cal the function calculate() and do the following steps.

STEP 5: For i=2 to a/2 do

STEP 6: Check if a%i==0 then set k=0 and break.

STEP 7: Else set k value as 1.

STEP 8: Increment the value i as 1.

STEP 9: Check whether the k value is 1 or 0.

STEP 10:If it is 1 then display the value is a prime number.

STEP 11:Else display the value is not prime.

STEP 12:Stop the program.

```
#include<iostream>
#include<conio.h>

using namespace std;

// Class Declaration
class prime
{
   //Member Varibale Declaration
 int a,k,i;
 public:
```

```
 prime(int x)
 {
 a=x;

 k=1;
 {
 for(i=2;i<=a/2;i++)
 if(a%i==0)
 {
 k=0;
 break;
 }
 else
 {
 k=1;
 }
 }
 }

 void show()
 {
 if(k==1)
 cout<<"\n"<<a<<" is Prime Number.";
 else
 cout<<"\n"<<a<<" is Not Prime Numbers.";
 }
};

//Main Function
int main()
{
```

```
 int a;
 cout<<“Enter the Number:”;
 cin>>a;

 // Object Creation For Class
 prime obj(a);

 // Call Member Functions
 obj.show();
 getch();
 return 0;
}
```

**Output :**

Enter the Number:7

7 is Prime Number.

# 11. Simple Example Program For Copy Constructor In C++

```
#include<iostream>
#include<conio.h>

using namespace std;

class Example        {
    // Variable Declaration
    int a,b;
    public:

    //Constructor with Argument
    Example(int x,int y)            {
    // Assign Values In Constructor
    a=x;
    b=y;
    cout<<"\nIm Constructor";
    }

    void Display()    {
    cout<<"\nValues :"<<a<<"\t"<<b;
    }
};

int main()                {
        Example Object(10,20);

        //Copy Constructor
```

```
    Example Object2=Object;

    // Constructor invoked.

    Object.Display();
    Object2.Display();

    // Wait For Output Screen
    getch();
    return 0;
}
```

**Output :**

```
Im Constructor
Values :10      20
Values :10      20
```

# 12. Simple Class Example Program In C++

```
#include <iostream>
#include<conio.h>
using namespace std;

// Class Declaration
class person
{
//Access - Specifier
public:

//Varibale Declaration
  string name;
  int number;
};

//Main Function
int main()
{
    // Object Creation For Class
    person obj;

    //Get Input Values For Object Varibales
    cout<<“Enter the Name :”;
    cin>>obj.name;

    cout<<“Enter the Number :”;
    cin>>obj.number;

    //Show the Output
```

```
    cout << obj.name << ": " << obj.number << endl;

    getch();
    return 0;
}
```

**Output:**

Enter the Name :Byron

Enter the Number :100

Byron: 100

# 13. Simple Class Example Program For Find Prime Number In C++

```
#include<iostream>
#include<conio.h>

using namespace std;

// Class Declaration
class prime
{
   //Member Varibale Declaration
 int a,k,i;
 public:

 prime(int x)
 {
 a=x;
 }

 // Object Creation For Class
 void calculate()
 {
 k=1;
 {
 for(i=2;i<=a/2;i++)
 if(a%i==0)
 {
 k=0;
 break;
```

```
 }
 else
 {
 k=1;
 }
 }
 }

 void show()
 {
 if(k==1)
 cout<<"\n"<<a<<" is Prime Number.";
 else
 cout<<"\n"<<a<<" is Not Prime Numbers.";
 }
};

//Main Function
int main()
{
 int a;
 cout<<"Enter the Number:";
 cin>>a;

 // Object Creation For Class
 prime obj(a);

 // Call Member Functions
 obj.calculate();
 obj.show();
 getch();
 return 0;
```

}

**Output:**

Enter the Number:10

10 is Not Prime Numbers.

Enter the Number:7

7 is Prime Number.

# 14. Simple Example Program For Namespace In C++

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

//Namespace namespacefirst
namespace namespacefirst
{
  int value = 5;
}

//Namespace namespacesecond
namespace namespacesecond
{
  double value = 3.1416;
}

int main () {
  //Namespace namespacefirst Varibale Usage
  cout << “namespacefirst value : ” <<namespacefirst::value << endl;

  //Namespace namespacesecond Varibale Usage
  cout << “namespacesecond value : “<<namespacesecond::value << endl;
  return 0;
}
```

**Output:**

namespacefirst value : 5

namespacesecond value : 3.1416

# 15. Find Prime Number Example Program In C++

```
#include<iostream>
#include<conio.h>
#include<math.h>    // Math.h For sqrt function

using namespace std;

int main()
{

    // Variable Declaration
    int n;

    // Get Input Value
    cout<<“Enter the Number :”;
    cin>>n;

     cout<<“List Of Prime Numbers Below “<<n<<endl;

    //for Loop Block For Find Prime Number

    for (int i=2; i<n; i++)
      for (int j=2; j*j<=i; j++)
      {
         if (i % j == 0)
            break;
         else if (j+1 > sqrt(i)) {
            cout << i << endl;
```

```
        }
      }

    // Wait For Output Screen
    getch();
    return 0;
}
```

**Output:**

```
Enter the Number :50
List Of Prime Numbers Below 50
5
7
11
13
17
19
23
29
31
37
41
43
47
```

# 16. Find Prime Number by another method Example Program In C++

```
#include<iostream>
#include<conio.h>
#include<math.h>    // Math.h For sqrt function

using namespace std;

int main()
{

    // Variable Declaration
    int n;

    // Get Input Value
    cout<<“Enter the Number :”;
    cin>>n;

     cout<<“List Of Prime Numbers Below “<<n<<endl;

    //for Loop Block For Find Prime Number
    for (int i=2; i<n; i++)
    {
      bool prime=true;
      for (int j=2; j*j<=i; j++)
      {
          if (i % j == 0)
          {
```

```
            prime=false;
            break;
        }
      }
      if(prime) cout << i << endl;
    }

    // Wait For Output Screen
    getch();
    return 0;
}
```

**Output:**

```
Enter the Number :50
List Of Prime Numbers Below 50
5
7
11
13
17
19
23
29
31
37
41
43
47
```

# 17. Fibonacci series Example Program In C++

```
#include<iostream>
#include<conio.h>

using namespace std;

int main()
{

    // Variable Declaration
    int counter, n;
    long last=1,next=0,sum;
    // Get Input Value
    cout<<“Enter the Number :”;
    cin>>n;

    //Fibonacci Series Calculation
    while(next<n/2)
    {
     cout<<last <<”  ”;
     sum=next+last;
     next=last;
     last=sum;
    }

    // Wait For Output Screen
    getch();
    return 0;
```

}

**Output :**

Enter the Number :300

1

1

2

3

5

8

13

21

34

55

89

144

233

# 18. Factorial Using Recursion Example Program In C++

```
#include<iostream>
#include<conio.h>

using namespace std;

//Function
long factorial(int);

int main()
{

    // Variable Declaration
    int counter, n;

    // Get Input Value
    cout<<“Enter the Number :”;
    cin>>n;

    // Factorial Function Call
    cout<<n<<” Factorial Value Is “<<factorial(n);

    // Wait For Output Screen
    getch();
    return 0;
 }

// Factorial recursion Function
```

```
long factorial(int n)
{
  if (n == 0)
    return 1;
  else
    return(n * factorial(n-1));
}
```

**Output :**

```
Enter the Number :6
6 Factorial Value Is 720
```

# 19. Factorial Using Function Example Program In C++

```
#include<iostream>
#include<conio.h>

using namespace std;

//Function
long factorial(int);

int main()
{

    // Variable Declaration
    int counter, n;

    // Get Input Value
    cout<<“Enter the Number :”;
    cin>>n;

    // Factorial Function Call
    cout<<n<<” Factorial Value Is “<<factorial(n);

    // Wait For Output Screen
    getch();
    return 0;
 }

// Factorial Function
```

```
long factorial(int n)
{
    int counter;
    long fact = 1;

     //for Loop Block
     for (int counter = 1; counter <= n; counter++)
     {
         fact = fact * counter;
     }

  return fact;
}
```

**Output :**

```
Enter the Number :6
6 Factorial Value Is 720
```

# 20. Factorial Using Loop Example Program In C++

```
#include<iostream>
#include<conio.h>

using namespace std;

int main()
{

    // Variable Declaration
    int counter, n, fact = 1;

    // Get Input Value
    cout<<“Enter the Number :”;
    cin>>n;

    //for Loop Block
    for (int counter = 1; counter <= n; counter++)
    {
       fact = fact * counter;
    }

    cout<<n<<” Factorial Value Is “<<fact;
    // Wait For Output Screen
    getch();
    return 0;
 }
```

**Output :**

Enter the Number :6

6 Factorial Value Is 720

# 21. Simple Example Program for Function In C++

```
#include<iostream>
#include<conio.h>

using namespace std;

// Simple Function
void printmessage ()
{
  cout << “Im Function In C++”;
}

int main()              {
        printmessage ();
        getch();
        return 0;
}
```

**Output :**

Im Function In C++

# 22. Simple Example Program for Function Find Smallest Number In C++

```
#include<iostream>
#include<conio.h>

using namespace std;

// Simple Function
int compare( int a, int b )
{
   return (a+4 < b)? a : b;
}

int main()              {
   cout<<"\nSmallest Number :"<<compare(1,10);
   cout<<"\nSmallest Number :"<<compare(31,10);
   cout<<"\nSmallest Number :"<<compare(11,8);
   getch();
   return 0;
}
```

**Output :**

```
Smallest Number :1
Smallest Number :10
Smallest Number :8
```

# 23. Simple Example Program for Function to Find Factorial In C++

```
#include<iostream>
#include<conio.h>

using namespace std;

// Simple factorial Function
int factorial(int var)
{
   int fact=1;
   for(int i=1;i<=var;i++)
   fact = fact * i;
   return fact;
}

int main()              {
   cout<<“5 Factorial Number :”<<factorial(5);
   getch();
   return 0;
}
```

**Output :**

5 Factorial Number :120

# 24. Simple Example Program for Inline Function Using C++ Programming

**ALGORITHM:**

Step 1: Start the pogram.

Step 2: Declare the class.

Step 3: Declare and define the inline function for multiplication and cube.

Step 4: Declare the class object and variables.

Step 5: Read two values.

Step 6: Call the multiplication and cubic functions using class objects.

Step 7: Return the values.

Step 8: Display.

Step 9: Stop the program.

**Example Program Of Inline Function**

```
#include<iostream.h>
#include<conio.h>
class line
{
  public:
        inline float mul(float x,float y)
        {
                return(x*y);
        }
        inline float cube(float x)
        {
                return(x*x*x);
        }
};
void main()
```

```
{
        line obj;
        float val1,val2;
        clrscr();
        cout<<“Enter two values:”;
        cin>>val1>>val2;
        cout<<”\nMultiplication value is:”<<obj.mul(val1,val2);
        cout<<”\n\nCube value is          :”<<obj.cube(val1)<<”\t”<<obj.cube(val2);
        getch();
}
```

**Output:**

```
Enter two values: 5  7
Multiplication Value is: 35
Cube Value is: 25 and 343
```

# 25. Simple Addition in C++ Binary Operator Overloading Using C++ Programming

```
// Header Files
#include<iostream>
#include<conio.h>
//Standard namespace declaration
using namespace std;

class overloading
{
 int value;
 public:
 void setValue(int temp)
 {
      value = temp;
 }
 overloading operator+(overloading ob)
 {
  overloading t;
  t.value=value+ob.value;
  return(t);
  }
void display()
{
 cout<<value<<endl;
}
};

//Main Functions
```

```
int main()
{
  overloading obj1,obj2,result;
  int a,b;

  cout<<“Enter the value of Complex Numbers a,b:”;
  cin>>a>>b;
  obj1.setValue(a);
  obj2.setValue(b);

  result = obj1+obj2;

  cout<<“Input Values:\n”;
  obj1.display();
  obj2.display();
  cout<<“Result:”;
  result.display();
  getch();
  return 0;
}
```

**Output:**

```
Enter the value of Complex Numbers a,b:10
5
Input Values:
10
5
Result:15
```

# 26. Simple Program for Multiple Inheritance Using C++ Programming

To find out the student details using multiple inheritance.

**ALGORITHM:**

Step 1: Start the program.

Step 2: Declare the base class student.

Step 3: Declare and define the function get() to get the student details.

Step 4: Declare the other class sports.

Step 5: Declare and define the function getsm() to read the sports mark.

Step 6: Create the class statement derived from student and sports.

Step 7: Declare and define the function display() to find out the total and average.

Step 8: Declare the derived class object,call the functions get(),getsm() and display().

Step 9: Stop the program.

**PROGRAM:**

```
#include<iostream.h>
#include<conio.h>

class student
{
    protected:
      int rno,m1,m2;
    public:
            void get()
           {
                    cout<<“Enter the Roll no :”;
                    cin>>rno;
                    cout<<“Enter the two marks   :”;
                    cin>>m1>>m2;
```

```
        }
};
class sports
{
    protected:
      int sm;              // sm = Sports mark
    public:
          void getsm()
        {
          cout<<"\nEnter the sports mark :";
          cin>>sm;

        }
};
class statement:public student,public sports
{
    int tot,avg;
    public:
    void display()
        {
          tot=(m1+m2+sm);
          avg=tot/3;
          cout<<"\n\n\tRoll No    : "<<rno<<"\n\tTotal      : "<<tot;
         cout<<"\n\tAverage    : "<<avg;
        }
};
void main()
{
   clrscr();
   statement obj;
   obj.get();
   obj.getsm();
```

```
    obj.display();
    getch();
}
```

**Output:**

```
Enter the Roll no: 100

Enter two marks

90
80

Enter the Sports Mark: 90

Roll No: 100
Total    : 260
Average: 86.66
```

# 27. Simple Program for Inline Function Using C++ Programming

**ALGORITHM:**

Step 1: Start the pogram.

Step 2: Declare the class.

Step 3: Declare and define the inline function for multiplication and cube.

Step 4: Declare the class object and variables.

Step 5: Read two values.

Step 6: Call the multiplication and cubic functions using class objects.

Step 7: Return the values.

Step 8: Display.

Step 9: Stop the program.

**PROGRAM:**

```
#include<iostream.h>
#include<conio.h>

class line
{
    public:
            inline float mul(float x,float y)
            {
                    return(x*y);
            }
            inline float cube(float x)
            {
                    return(x*x*x);
            }
};
```

```
void main()
{
        line obj;
        float val1,val2;
        clrscr();
        cout<<“Enter two values:”;
        cin>>val1>>val2;
        cout<<”\nMultiplication value is:”<<obj.mul(val1,val2);
        cout<<”\n\nCube value is        :”<<obj.cube(val1)<<”\t”<<obj.cube(val2);
        getch();
}
```

**Output:**

```
Enter two values: 5  7
Multiplication Value is: 35
Cube Value is: 25 and 343
```

# 28. Simple Program for Function Template Using C++ Programming

To swap the numbers using the concept of function template.

**ALGORITHM:**

STEP 1: Start the program.

STEP 2: Declare the template class.

STEP 3: Declare and define the functions to swap the values.

STEP 4: Declare and define the functions to get the values.

STEP 5: Read the values and call the corresponding functions.

STEP6: Display the results.

STEP 7: Stop the program.

**PROGRAM:**

```
#include<iostream.h>
#include<conio.h>

template<class t>

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

void fun(int a,int b,float c,float d)
{
    cout<<"\na and b before swaping :"<<a<<"\t"<<b;
    swap(a,b);
```

```
    cout<<"\na and b after swaping  :"<<a<<"\t"<<b;
    cout<<"\n\nc and d before swaping :"<<c<<"\t"<<d;
    swap(c,d);
    cout<<"\nc and d after swaping  :"<<c<<"\t"<<d;
}

void main()
{
     int a,b;
     float c,d;
     clrscr();
     cout<<"Enter A,B values(integer):";
     cin>>a>>b;
     cout<<"Enter C,D values(float):";
     cin>>c>>d;
     fun(a,b,c,d);
     getch();
}
```

**Output:**

```
Enter A, B values (integer): 10  20
Enter C, D values (float):    2.50  10.80

A and B before swapping: 10 20
A and B after    swapping:  20 10

C and D before swapping: 2.50  10.80
C and D after              swapping: 10.80  2.50
```

# 29. Simple Program for Function Overloading Using C++ Programming

**ALGORITHM:**

STEP 1: Start the program.

STEP 2: Declare the class name as fn with data members and member functions.

STEP 3: Read the choice from the user.

STEP 4: Choice=1 then go to the step 5.

STEP 5: The function area() to find area of circle with one integer argument.

STEP 6: Choice=2 then go to the step 7.

STEP 7: The function area() to find area of rectangle with two integer argument.

STEP 8: Choice=3 then go to the step 9.

STEP 9: The function area() to find area of triangle with three arguments, two as Integer and one as float.

STEP 10: Choice=4 then stop the program.

**PROGRAM:**

```
#include<iostream.h>
#include<stdlib.h>
#include<conio.h>
#define pi 3.14
class fn
{
      public:
        void area(int);  //circle
        void area(int,int);  //rectangle
        void area(float ,int,int);  //triangle
};

void fn::area(int a)
{
```

```
        cout<<“Area of Circle:”<<pi*a*a;
}
void fn::area(int a,int b)
{
        cout<<“Area of rectangle:”<<a*b;
}
void fn::area(float t,int a,int b)
{
        cout<<“Area of triangle:”<<t*a*b;
}

void main()
{
      int ch;
      int a,b,r;
      clrscr();
      fn obj;
      cout<<”\n\t\tFunction Overloading”;
      cout<<”\n1.Area of Circle\n2.Area of Rectangle\n3.Area of Triangle\n4.Exit\n:”;
      cout<<”Enter your Choice:”;
      cin>>ch;

      switch(ch)
      {
            case 1:
              cout<<“Enter Radious of the Circle:”;
              cin>>r;
              obj.area(r);
              break;
            case 2:
              cout<<“Enter Sides of the Rectangle:”;
```

```
            cin>>a>>b;
            obj.area(a,b);
            break;
          case 3:
            cout<<“Enter Sides of the Triangle:”;
            cin>>a>>b;
            obj.area(0.5,a,b);
            break;
          case 4:
            exit(0);
    }
getch();
}
```

**Output:**

```
Function Overloading
1. Area of Circle
2. Area of Rectangle
3. Area of Triangle
4. Exit
Enter Your Choice: 2

Enter the Sides of the Rectangle: 5 5

Area of Rectangle is: 25

1. Area of Circle
2. Area of Rectangle
3. Area of Triangle
4. Exit
Enter Your Choice: 4
```

# 30. Simple Program for Friend Function Using C++ Programming

**ALGORITHM:**

STEP 1: Start the program.

STEP 2: Declare the class name as Base with data members and member functions.

STEP 3: The function get() is used to read the 2 inputs from the user.

STEP 4: Declare the friend function mean(base ob) inside the class.

STEP 5: Outside the class to define the friend function and do the following.

STEP 6: Return the mean value (ob.val1+ob.val2)/2 as a float.

STEP 7: Stop the program.

**PROGRAM:**

```
#include<iostream.h>
#include<conio.h>
class  base
{
     int val1,val2;
    public:
     void get()
     {
       cout<<“Enter two values:”;
       cin>>val1>>val2;
     }
     friend float mean(base ob);
};
float mean(base ob)
{
    return float(ob.val1+ob.val2)/2;
}
```

```
void main()
{
    clrscr();
    base obj;
    obj.get();
    cout<<"\n Mean value is : "<<mean(obj);
    getch();
}
```

**Output:**

Enter two values: 10, 20

Mean Value is: 15

# 31. Simple Program for Exception Handling with Multiple Catch Using C++ Programming

**ALGORITHM:**

Step 1: Start the program.

Step 2: Declare and define the function test().

Step 3: Within the try block check whether the value is greater than zero or not.

a. if the value greater than zero throw the value and catch the corresponding exception.

b. Otherwise throw the character and catch the corresponding exception.

Step 4: Read the integer and character values for the function test().

Step 5: Stop the program.

**PROGRAM:**

```
#include<iostream.h>
#include<conio.h>
void test(int x)
{
    try
    {
            if(x>0)
              throw x;
        else
              throw ‘x’;
    }

    catch(int x)
    {
            cout<<“Catch a integer and that integer is:”<<x;
    }
```

```
    catch(char x)
    {
            cout<<“Catch a character and that character is:”<<x;
    }
}

void main()
{
    clrscr();
    cout<<“Testing multiple catches\n:”;
    test(10);
    test(0);
    getch();
}
```

**Output:**

Testing multiple catches

Catch a integer and that integer is: 10

Catch a character and that character is: x

# 32. Simple Program for Exception Handling Divide by zero Using C++ Programming

**ALGORITHM:**

Step 1: Start the program.

Step 2: Declare the variables a,b,c.

Step 3: Read the values a,b,c,.

Step 4: Inside the try block check the condition.

a. if(a-b!=0) then calculate the value of d and display.

b. otherwise throw the exception.

Step 5: Catch the exception and display the appropriate message.

Step 6: Stop the program.

**PROGRAM:**

```
#include<iostream.h>
#include<conio.h>
void main()
{
    int a,b,c;
    float  d;
    clrscr();
    cout<<“Enter the value of a:”;
    cin>>a;
    cout<<“Enter the value of b:”;
    cin>>b;
    cout<<“Enter the value of c:”;
    cin>>c;
    try
    {
          if((a-b)!=0)
```

```
        {
          d=c/(a-b);
          cout<<“Result is:”<<d;
        }
        else
        {
          throw(a-b);
        }
    }

    catch(int i)
    {
        cout<<“Answer is infinite because a-b is:”<<i;
    }
    getch();
}
```

**Output:**

```
        Enter the value for a: 20
        Enter the value for b: 20
        Enter the value for c: 40

        Answer is infinite because a-b is: 0
```

# 33. Simple Program for Constructor Using C++ Programming

**ALGORITHM:**

STEP 1: Start the program.

STEP 2: Declare the class as Prime with data members,
Member functions.

STEP 3: Consider the argument constructor Prime() with integer
Argument.

STEP 4: To cal the function calculate() and do the following steps.

STEP 5: For i=2 to a/2 do

STEP 6: Check if a%i==0 then set k=0 and break.

STEP 7: Else set k value as 1.

STEP 8: Increment the value i as 1.

STEP 9: Check whether the k value is 1 or 0.

STEP 10:If it is 1 then display the value is a prime number.

STEP 11:Else display the value is not prime.

STEP 12:Stop the program.

```
#include<iostream.h>
#include<conio.h>
class prime
{
        int a,k,i;
      public:
      prime(int x)
      {
            a=x;
      }
```

```
        void calculate()
        {
          k=1;
         {
            for(i=2;i<=a/2;i++)

    if(a%i==0)
            {
                k=0;
                break;
            }
            else
           {
                k=1;
          }
         }
        }

void show()
        {
         if(k==1)
           cout<< "\n\tA is prime Number. ";
         else
           cout<<"\n\tA is Not prime.";
        }
};

void main()
{
    clrscr();
    int a;
    cout<<"\n\tEnter the Number:";
```

```
    cin>>a;
    prime obj(a);
    obj.calculate();
    obj.show();
    getch();
}
```

**Output:**

Enter the number: 7

Given number is Prime Number

# 34. Simple Program for Binary Operator Overloading Using C++ Programming

**ALGORITHM:**

Step 1: Start the program.

Step 2: Declare the class.

Step 3: Declare the variables and its member function.

Step 4: Using the function getvalue() to get the two numbers.

Step 5: Define the function operator +() to add two complex numbers.

Step 6: Define the function operator –()to subtract two complex numbers.

Step 7: Define the display function.

Step 8: Declare the class objects obj1,obj2 and result.

Step 9: Call the function getvalue using obj1 and obj2

Step 10: Calculate the value for the object result by calling the function operator + and operator -.

Step 11: Call the display function using obj1 and obj2 and result.

Step 12: Return the values.

Step 13: Stop the program.

**PROGRAM:**

```
#include<iostream.h>
#include<conio.h>

class complex
{
        int a,b;
    public:
        void getvalue()
        {
          cout<<“Enter the value of Complex Numbers a,b:”;
          cin>>a>>b;
```

```
        }
        complex operator+(complex ob)
        {
                complex t;
                t.a=a+ob.a;
                t.b=b+ob.b;
                return(t);
        }
        complex operator-(complex ob)
        {
                complex t;
                t.a=a-ob.a;
                t.b=b-ob.b;
                return(t);
        }
        void display()
        {
                cout<<a<<"+"<<b<<"i"<<"\n";
        }
};



void main()
{
    clrscr();
    complex obj1,obj2,result,result1;

    obj1.getvalue();
    obj2.getvalue();
```

```
    result = obj1+obj2;
    result1=obj1-obj2;

    cout<<“Input Values:\n”;
    obj1.display();
    obj2.display();
    cout<<“Result:”;
    result.display();
   result1.display();
    getch();
}
```

**Output:**

```
Enter the value of Complex Numbers a, b
4               5
Enter the value of Complex Numbers a, b
2               2
Input Values
4 + 5i
2 + 2i
Result
6 +   7i
2 +   3i
```

# 35. Simple Program Book Entry Using structure Variable in C++ Programming

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

struct books
{
   char name[20],author[20];
}a[50];
int main()
{

   int i,n;
   cout<<“No Of Books[less than 50]:”;
   cin>>n;
 cout<<“Enter the book details\n”;
 cout<<”————-\n”;

   for(i=0;i<n;i++)
   {
  cout<<“Details of Book No “<<i+1<<”\n”;
  cout<<“Book Name :”;
  cin>>a[i].name;
  cout<<“Book Author :”;
  cin>>a[i].author;
  cout<<”————-\n”;
   }
   cout<<”==========================================\n”;
   cout<<” S.No\t| Book Name\t|author\n”;
```

```
    cout<<"=================================================";
    for(i=0;i<n;i++)
    {
  cout<<"\n  "<<i+1<<"\t|"<<a[i].name<<"\t| "<<a[i].author;
    }
    cout<<"\n=================================================";

    return 0;
}
```

**Output :**

```
No Of Books[less than 50]:2
Enter the book details
————-
Details of Book No 1
Book Name :Programming
Book Author :Dromy
Details of Book No 2
Book Name :C
Book Author :Byron
=================================================
 S.No | Book Name |author
=================================================
  1 |Programming | Dromy
  2 |C | Byron
=================================================
```

# 36. Simple Program for Write File Operation Using C++ Programming

**ALGORITHM:**

STEP 1: Start the program.

STEP 2: Declare the variables.

STEP 3: Read the file name.

STEP 4: open the file to write the contents.

STEP 5: writing the file contents up to reach a particular condition.

STEP 6: Stop the program.

**PROGRAM:**

```
#include<iostream.h>
#include<stdio.h>
#include<conio.h>
#include<fstream.h>
void main()
{
        char c,fname[10];
        ofstream out;
        cout<<“Enter File name:”;
        cin>>fname;
        out.open(fname);
        cout<<“Enter contents to store in file (Enter # at end):\n”;
        while((c=getchar())!=’#’)
        {
                out<<c;
        }
        out.close();
        getch();
```

}

**Output:**

Enter File name: one.txt

Enter contents to store in file (enter # at end)

Master of Computer Applications#

# 37. Simple Program for Virtual Functions Using C++ Programming

**ALGORITHM:**

Step 1: Start the program.

Step 2: Declare the base class base.

Step 3: Declare and define the virtual function show().

Step 4: Declare and define the function display().

Step 5: Create the derived class from the base class.

Step 6: Declare and define the functions display() and show().

Step 7: Create the base class object and pointer variable.

Step 8: Call the functions display() and show() using the base class object and pointer.

Step 9: Create the derived class object and call the functions display() and show() using the derived class object and pointer.

Step 10: Stop the program.

**PROGRAM:**

```
#include<iostream.h>
#include<conio.h>
class base
{
    public:
      virtual void show()
      {
            cout<<"\n  Base class show:";
      }
      void display()
      {
          cout<<"\n  Base class display:" ;
      }
};
```

```
class drive:public base
{
    public:
      void display()
      {
            cout<<"\n  Drive class display:";
      }
      void show()
      {
            cout<<"\n  Drive class show:";
      }
};

void main()
{
    clrscr();
    base obj1;
    base *p;
    cout<<"\n\t P points to base:\n"  ;

    p=&obj1;
    p->display();
    p->show();

    cout<<"\n\n\t P points to drive:\n";
    drive obj2;
    p=&obj2;
    p->display();
    p->show();
    getch();
}
```

**Output:**

P points to Base

Base class display
Base class show

P points to Drive

Base class Display
Drive class Show

# 38. Simple Program for Virtual Base Class Using C++ Programming

**ALGORITHM:**

Step 1: Start the program.

Step 2: Declare the base class student.

Step 3: Declare and define the functions getnumber() and putnumber().

Step 4: Create the derived class test virtually derived from the base class student.

Step 5: Declare and define the function getmarks() and putmarks().

Step 6: Create the derived class sports virtually derived from the base class student.

Step 7: Declare and define the function getscore() and putscore().

Step 8: Create the derived class result derived from the class test and sports.

Step 9: Declare and define the function display() to calculate the total.

Step 10: Create the derived class object obj.

Step 11: Call the function get number(),getmarks(),getscore() and display().

Step 12: Stop the program.

**PROGRAM:**

```
#include<iostream.h>
#include<conio.h>

class student
{
     int rno;
    public:
     void getnumber()
     {
              cout<<“Enter Roll No:”;
              cin>>rno;
     }
```

```
    void putnumber()
    {
            cout<<"\n\n\tRoll No:"<<rno<<"\n";
    }
};

class test:virtual public student
{
   public:
    int part1,part2;
    void getmarks()
    {
            cout<<"Enter Marks\n";
            cout<<"Part1:";
            cin>>part1;
            cout<<"Part2:";
            cin>>part2;
    }
    void putmarks()
    {
            cout<<"\tMarks Obtained\n";
            cout<<"\n\tPart1:"<<part1;
            cout<<"\n\tPart2:"<<part2;
    }
};

class sports:public virtual student
{

   public:
     int score;
     void getscore()
```

```
    {
        cout<<“Enter Sports Score:”;
        cin>>score;
    }
    void putscore()
    {
        cout<<”\n\tSports Score is:”<<score;
    }
};

class result:public test,public sports
{
    int total;
  public:
    void display()
    {
      total=part1+part2+score;
      putnumber();
      putmarks();
      putscore();
      cout<<”\n\tTotal Score:”<<total;
    }
};

void main()
{
    result obj;
    clrscr();
    obj.getnumber();
    obj.getmarks();
    obj.getscore();
    obj.display();
```

```
    getch();
}
```

**Output:**

```
Enter Roll No: 200

Enter Marks

Part1: 90
Part2: 80
Enter Sports Score: 80


Roll No: 200
Marks Obtained
Part1: 90
Part2: 80
Sports Score is: 80
Total Score is: 250
```

# 39. Simple Program for Unary Operator Overloading Using C++ Programming

To write a program to find the complex numbers using unary operator overloading.

**ALGORITHM:**

Step 1: Start the program.

Step 2: Declare the class.

Step 3: Declare the variables and its member function.

Step 4: Using the function getvalue() to get the two numbers.

Step 5: Define the function operator ++ to increment the values

Step 6: Define the function operator - -to decrement the values.

Step 7: Define the display function.

Step 8: Declare the class object.

Step 9: Call the function getvalue

Step 10: Call the function operator ++() by incrementing the class object and call the function display.

Step 11: Call the function operator - -() by decrementing the class object and call the function display.

Step 12: Stop the program.

**PROGRAM:**

```
#include<iostream.h>
#include<conio.h>

class complex
{
      int a,b,c;
     public:
        complex(){}
        void getvalue()
```

```
        {
                cout<<“Enter the Two Numbers:”;
                cin>>a>>b;
        }



    void operator++()
        {
                a=++a;
                b=++b;
        }

        void operator—()
        {
                a=—a;
                b=—b;
         }

         void display()
         {
                cout<<a<<”+\t”<<b<<“i”<<endl;
          }
};

void main()
{
       clrscr();
       complex obj;
       obj.getvalue();
       obj++;
       cout<<“Increment Complex Number\n”;
```

```
    obj.display();
    obj—;
    cout<<“Decrement Complex Number\n”;
    obj.display();
    getch();
}
```

**Output**:

Enter the two numbers: 3 6

Increment Complex Number

4 + 7i

Decrement Complex Number

3 + 6i

# 40. Simple Program for Static Data and Member Function Using C++ Programming

**ALGORITHM:**

STEP 1: Start the program.

STEP 2: Declare the class name as Stat with data member s and member functions.

STEP 3: The constructor Stat() which is used to increment the value of count as 1 to to assign the variable code.

STEP 4: The function showcode() to display the code value.

STEP 5: The function showcount() to display the count value.

STEP 6: Stop the program.

**PROGRAM:**

```
#include<iostream.h>
#include<conio.h>

class stat
{
     int code;
     static int count;

    public:
     stat()
     {
       code=++count;
     }
     void showcode()
     {
       cout<<"\n\tObject number is :"<<code;
     }
```

```
    static void showcount()
    {
        cout<<"\n\tCount Objects :"<<count;
    }
};

int stat::count;

void main()
{
    clrscr();
    stat obj1,obj2;

    obj1.showcount();
    obj1.showcode();
    obj2.showcount();
    obj2.showcode();
    getch();
}
```

**Output:**

Count Objects: 2

Object Number is: 1

Count Objects: 2

Object Number is: 2

# 41. Simple Program for Single Inheritance Using C++ Programming

**ALGORITHM**:

Step 1: Start the program.

Step 2: Declare the base class emp.

Step 3: Define and declare the function get() to get the employee details.

Step 4: Declare the derived class salary.

Step 5: Declare and define the function get1() to get the salary details.

Step 6: Define the function calculate() to find the net pay.

Step 7: Define the function display().

Step 8: Create the derived class object.

Step 9: Read the number of employees.

Step 10: Call the function get(),get1() and calculate() to each employees.

Step 11: Call the display().

Step 12: Stop the program.

**PROGRAM:PAYROLL SYSTEM USING SINGLE INHERITANCE**

```
#include<iostream.h>
#include<conio.h>

class emp
{
    public:
      int eno;
      char name[20],des[20];
      void get()
      {
            cout<<“Enter the employee number:”;
            cin>>eno;
```

```
        cout<<“Enter the employee name:”;
        cin>>name;
        cout<<“Enter the designation:”;
        cin>>des;
    }
};


class salary:public emp
{
    float bp,hra,da,pf,np;
  public:
    void get1()
    {
        cout<<“Enter the basic pay:”;
        cin>>bp;
        cout<<“Enter the Humen Resource Allowance:”;
        cin>>hra;
        cout<<“Enter the Dearness Allowance :”;
        cin>>da;
        cout<<“Enter the Profitablity Fund:”;
        cin>>pf;
    }
    void calculate()
    {
        np=bp+hra+da-pf;
    }
    void display()
    {

 cout<<eno<<”\t”<<name<<”\t”<<des<<”\t”<<bp<<”\t”<<hra<<”\t”<<da<<”\t”<<pf<<”\t
    }
};
```

```
void main()
{
    int i,n;
    char ch;
    salary s[10];
    clrscr();
    cout<<“Enter the number of employee:”;
    cin>>n;
    for(i=0;i<n;i++)
    {
        s[i].get();
        s[i].get1();
        s[i].calculate();
    }
    cout<<”\ne_no \t e_name\t des \t bp \t hra \t da \t pf \t np \n”;
    for(i=0;i<n;i++)
    {
        s[i].display();
    }
    getch();
}
```

**Output:**

Enter the Number of employee:1

Enter the employee No: 150

Enter the employee Name: ram

Enter the designation: Manager

Enter the basic pay: 5000

Enter the HR allowance: 1000

Enter the Dearness allowance: 500

Enter the profitability Fund: 300

| E.No | E.name | des | BP | HRA | DA | PF | NP |
|---|---|---|---|---|---|---|---|
| 150 | ram | Manager | 5000 | 1000 | 500 | 300 | 6200 |

# 42. Simple Program for Copy Constructor Using C++ Programming

**ALGORITHM:**

STEP 1: Start the program.

STEP 2: Declare the class name as Copy with data members and member functions.

STEP 3: The constructor Copy() with argument to assign the value.

STEP 4: To cal the function calculate() do the following steps.

STEP 5: For i=1 to var do

STEP 6: Calculate fact*i to assign to fact.

STEP 7: Increment the value as 1.

STEP 8: Return the value fact.

STEP 9: Print the result.

STEP 10: Stop the program.

**PROGRAM:**

```
#include<iostream.h>
#include<conio.h>
class copy
{
        int var,fact;
        public:

          copy(int temp)
          {
           var = temp;
          }

          double calculate()
          {
                fact=1;
```

```
                    for(int i=1;i<=var;i++)
                    {
                    fact = fact * i;
                    }
                    return fact;
            }
};
void main()
{
    clrscr();
    int n;
    cout<<"\n\tEnter the Number : ";
    cin>>n;
    copy obj(n);
    copy cpy=obj;
    cout<<"\n\t"<<n<<" Factorial is:"<<obj.calculate();
    cout<<"\n\t"<<n<<" Factorial is:"<<cpy.calculate();
    getch();
}
```

**Output:**

Enter the Number: 5

Factorial is: 120

Factorial is: 120

# 43. Simple Program for Read File Operation Using C++ Programming

Program for read the content of  a file.

**ALGORITHM:**

STEP 1:  Start the program.

STEP 2:  Declare the variables.

STEP 3:  Get the file name to read.

STEP 4:  Using ifstreamin(filename) check whether the file exist.

STEP 5:  If the file exist then check for the end of file condition.

STEP 6:  Read the contents of the file.

STEP 7:  Print the contents of the file.

STEP 8:  Stop the program.

**PROGRAM:**

```
#include<iostream.h>
#include<conio.h>
#include<fstream.h>
void main()
{
        char c,fname[10];
        clrscr();
        cout<<“Enter file name:”;
        cin>>fname;
        ifstream in(fname);
        if(!in)
        {
                cout<<“File Does not Exist”;
                getch();
```

```
            return;
        }
        cout<<"\n\n";
        while(in.eof()==0)
        {
            in.get(c);
            cout<<c;
        }
        getch();
}
```

**Output:**

```
        Enter File name: one.txt
        Master of Computer Applications
```

# 44. Simple Program for Read & Write File Operation (Convert lowercase to uppercase) Using C++ Programming

**ALGORITHM:**

STEP 1: Start the program.

STEP 2: Declare the variables.

STEP 3: Read the file name.

STEP 4: open the file to write the contents.

STEP 5: writing the file contents up to reach a particular condition.

STEP6: write the file contents as uppercase.

STEP7: open the file to read the contents.

STEP 8: Stop the program.

**PROGRAM:**

```
#include<fstream.h>
#include<stdio.h>
#include<ctype.h>
#include<string.h>
#include<iostream.h>
#include<conio.h>
void main()
{
        char c,u;
        char fname[10];
        clrscr();
        ofstream out;
        cout<<“Enter File Name:”;
        cin>>fname;
        out.open(fname);
```

```
        cout<<"Enter the text(Enter # at end)\n";   //write contents to file
        while((c=getchar())!='#')
        {
                u=c-32;
                out<<u;
        }
        out.close();
        ifstream in(fname);        //read the contents of file
        cout<<"\n\n\t\tThe File contains\n\n";
        while(in.eof()==0)
        {
                in.get(c);
                cout<<c;
        }
        getch();

}
```

**Output:**

Enter File Name: two.txt
Enter contents to store in file (enter # at end)
oops programming
The File Contains
OOPS PROGRAMMING

# 45. C++ program which adds two integers

```
#include <iostream.h>

main() {

  // A simple C++ program

  int x, y, sum;
  cout << “A program which adds two integers\n”;
  cout << “Enter 1st integer: “;
  cin >> x;
  cout << “Enter 2nd integer: “;
  cin >> y;
  sum = x + y;
  cout << “Sum is ” << sum << endl;
  return 0;
}
```

**Output :**

```
A program which adds two integers
Enter 1st integer: 3
Enter 2nd integer: 4
Sum is 7
```

# 46. C++ Program to output an integer, a floating point number and a character

```
#include <iostream.h>
#include <conio.h>

void main()
{
  clrscr();
  int x = 10;
  float y = 10.1;
  char z = ‘a’;
  cout << “x = ” << x << endl;
  cout << “y = ” << y << endl;
  cout << “z = ” << z << endl;
  getch();
}
```

This program has pre-defined values for an integer x, floating point number y, and a character z.

These three values are outputted using the ‘cout’ command.

**INPUT :**

No inputs from the user for this program.

**OUTPUT :**

```
x = 10
y = 10.1
```

z = a

# 47. C++ Program to find the sum, difference, product and quotient of two integers

```
#include <iostream.h>
#include <conio.h>

void main()
{
  clrscr();
  int x = 10;
  int y = 2;
  int sum, difference, product, quotient;
  sum = x + y;
  difference = x - y;
  product = x * y;
  quotient = x / y;
  cout << “The sum of ” << x << ” & ” << y << ” is ” << sum << “.” << endl;
  cout << “The difference of ” << x << ” & ” << “y <<  is ” << difference << “.” << endl;
  cout << “The product of ” << x << ” & ” << y << ” is ” << product << “.” << endl;
  cout << “The quotient of ” << x << ” & ” << y << ” is ” << quotient << “.” << endl;
  getch();
}
```

This program has pre-defined values for two integers x and y.

The sum, difference, product and quotient of these two values are calculated and then outputted using the ‘cout’ command.

**OUTPUT :**

The sum of 10 & 2 is 12.

The difference of 10 & 2 is 8.

The product of 10 & 2 is 20.

The quotient of 10 & 2 is 5.

# 48. Program to enter two integers and find their sum and average

```
#include <iostream.h>
#include <iostream.h>
#include <conio.h>

void main()
{
clrscr();
int x,y,sum;
float average;
cout << “Enter 2 integers : ” << endl;
cin>>x>>y;
sum=x+y;
average=sum/2;
cout << “The sum of ” << x << ” and ” << y << ” is ” << sum << “.” << endl;
cout << “The average of ” << x <<  ” and ” << y << ” is ” << average << “.” << endl;
getch();
}
```

This program takes in two integers x and y as a screen input from the user.

The sum and average of these two integers are calculated and outputted using the ‘cout’ command.

**INPUT :**

8 6

**OUTPUT :**

The sum of 8 and 6 is 14.

The average of 8 and 6 is 7.

# 49. Program to enter velocity, acceleration and time and print final velocity using the formula : v = u + a * t

```
#include <iostream.h>
#include <conio.h>

void main()
{
clrscr();
int v,u,a,t;
cout << “Enter the velocity, acceleration, time as integers : ” << endl;
cin>>u>>a>>t;
v=u+a*t;
cout << “The final velocity is ” << v << “.” << endl;
getch();
}
```

This program takes in the velocity, acceleration and the time as a screen input from the user.

The final velocity is calculated using the formula v = u + a * t, and then outputted using the ‘cout’ command.

**INPUT :**

20 10 5

**OUTPUT :**

The final velocity is 70.

# 50. Program to enter your age and print if you should be in grade 10

```
#include <iostream.h>
#include <conio.h>

void main()
{
clrscr();
int age;
cout << “Enter your present age : ” << endl;
cin>>age;
if(age==16)
{
cout << “Your present age is ” << age << ” years.” << endl;
cout << “You are of the right age for joining grade 10 !” << endl;
}
else
{
cout << “Your present age is ” << age << ” years.” << endl;
cout << “You are not of the right age for joining grade 10 !” << endl;
}
getch();
}
```

This program takes in the age as a screen input from the user.

The program tells the user whether he/she should be in grade 10 or not by using the ‘IF-ELSE’ command.

It then prints out the appropriate message using the ‘cout’ command.

**INPUT :**

15

**OUTPUT :**

Your present age is 15 years.
You are not of the right age for joining grade 10 !

# 51. Program to enter an integer and print if it is greater or less than 100

```
#include <iostream.h>
#include <conio.h>

void main(){
clrscr();
int x;
cout << “Enter an integer : ” << endl;
cin>>x;
if(x>100)
{
cout << x << ” is greater than 100.” << endl;
}
else
{
cout << x << ” is less than 100.” << endl;
}
getch();
}
```

This program takes in an integer x as a screen input from the user.
The program tells the user whether that integer is greater than 100 or less than 100.
It then prints out the appropriate message using the ‘cout’ command.

**INPUT :**

**OUTPUT :**

74 is less than 100.

# 52. Program to enter an integer and output its 15 multiples

```
#include <iostream.h>
#include <conio.h>
int main()
{
clrscr();
int x;
cout << “Enter an integer less than 2185 : “;
cin>>x;
cout << “The first 15 multiples of ” << x << ” are : “;
for(int y=1;y<16;y++)
cout << “\n” << x << “x” << y << “=” << x*y;
getch();
return 0;
}
```

This program takes in an integer x as a screen input from the user.

It then calculates the first fifteen multiples of that integer and outputs it using the ‘cout’ command.

**INPUT :**

12

**OUTPUT :**

The first 15 multiples of 12 are :

12x1=12

12x2=24

12x3=36

12x4=48

12x5=60

12x6=72

12x7=84

12x8=96

12x9=108

12x10=120

12x11=132

12x12=144

12x13=156

12x14=168

12x15=180

# 53. Program to enter two integers and print the quotient and remainder

```
#include <iostream.h>
#include <conio.h>
int main()
{
clrscr();
int x,y,quotient,remainder;
cout << “Enter 2 integers greater than 0 : “;
cin>>x>>y;
quotient=x/y;
remainder=x-(quotient*y);
cout << “Quotient of ” << x << ” & ” << y << ” = ” << quotient << “\n”;
cout << “Remainder” << ” = ” << remainder << “\n”;
getch();
return 0;
}
```

This program takes in two integers x and y as a screen input from the user.

It then calculates their quotient and remainder and outputs them using the ‘cout’ command.

**INPUT :**

23 4

**OUTPUT :**

Quotient of 23 & 4 = 5

Remainder = 3

# 54. Program to enter an integer and find out if it is even or odd

```
#include <iostream.h>
#include <conio.h>

void main()
{
clrscr();
int x;
cout << “Enter an integer : “;
cin>>x;
if(x%2==0)
cout << “The number ” << x << ” is even.”;
else
cout << “The number ” << x << ” is odd.”;
getch();
}
```

This program takes in an integer x as a screen input from the user.

It then determines whether the integer is odd or even and outputs the appropriate message using the ‘cout’ command.

**INPUT :**

86

**OUTPUT :**

The number 86 is even.

# 55.Program to change the foreground colors and draw circles on the screen

```
#include <conio.h>
#include <graphics.h>
#include <stdlib.h>
#include <stdio.h>

void main (int)
{
int gdriver=DETECT,gmode,errorcode; //Requesting auto-detection.
int midx,midy,x;
//Initializing graphics and local variables.
initgraph(&gdriver,&gmode,“d:\bc3\bgi”);
//Reading result of initialization.
errorcode=graphresult();
if(errorcode!=grOk)
//An error occurred.
{
printf(“Graphics error occurred : %s \n”,grapherrormsg(errorcode));
printf(“Press any key to stop : “);
getch();
exit(1); //Terminate the program due to error.
}
/*Changing the foreground color.
Note : Press enter to exit the last screen as it is black and
it may appear as if the program has stopped running.*/
for(x=15;x>=0;x—)
{
setcolor(x);
```

```
circle(20+(x*40),200,15);/*Changing x-coordinate by 50 each time so that
the circles do not overlap.*/
getch();
}
cleardevice(); //Clearing the screen in the graphics mode.
circle(200,200,50);
getch();
closegraph();
}
```

This graphics program changes the foreground colors on the screen gradually from white to black, in-turn drawing circles of that foreground color, using the ‘setcolor’ command.

# 56. Program to draw 2 rectangles and fill 1 of them

```
#include <iostream.h>
#include <conio.h>
#include <graphics.h>
#include <ctype.h>
#include <stdlib.h>
#include <stdio.h>

void main()
{
  clrscr();
  int gd = DETECT,gm,errorcode; //Requesting auto-detection.

  //Initializing graphics and local variables.
  initgraph (&gd, &gm, "d:\bc3\bgi"); //Path where graphics drivers are installed

  //Read result of initialization.
  errorcode = graphresult();

  //An error occured.
  if (errorcode != grOk)
    {
      cout << "Graphics error occured : \n" << grapherrormsg(errorcode) << endl;
      cout << "Press any key to stop : ";
      getch();
      exit(1);
    }
```

```
  /*Drawing a rectangle having top LHS vertex at (300, 300)
  and bottom RHS vertex at (600, 400)*/
  rectangle(300, 300, 600, 400);
  rectangle(100, 100, 200, 200);
  getch();
  floodfill(120, 120, WHITE);
  getch();
  closegraph();
}
```

This graphics program draws two rectangles, but fills in only one of them with a white color using the ‘floodfill’ command.

# 57. Program to enter three integers and output the biggest integer

```
#include <iostream.h>
#include <conio.h>
int main()
{
clrscr();
int x,y,z,biggest;
cout << “Enter 3 integers : “;
cin>>x>>y>>z;
biggest=x>y?(x>z?x:z):(y>z?y:z);
cout << “The biggest integer out of the 3 integers you typed “;
cout << x << “, ” << y << ” & ” << z << ” is : ” << “\n” << biggest << “\n”;
getch();
return 0;
}
```

This program takes in three integers x, y and z as a screen input from the user.
It then determines the biggest integer of the three and outputs it using the ‘cout’ command.

**INPUT :**

63 73 79

**OUTPUT :**

The biggest integer out of the 3 integers you typed 64, 73 & 79 is :

# 58. Program to draw circles

```
#include <iostream.h>
#include <conio.h>
#include <graphics.h>
#include <ctype.h>
#include <stdlib.h>
#include <stdio.h>

void main()
{
clrscr();
int gd=DETECT,gm,errorcode; //Requesting auto-detection.
//Initializing graphics and local variables.
initgraph(&gd,&gm,“d:\bc3\bgi”); //Path where graphics drivers are installed
//Reading result of initialization.
errorcode=graphresult();
//An error occured.
if (errorcode!=grOk)
{
cout << “Graphics error occured : \n” << grapherrormsg(errorcode) << endl;
cout << “Press any key to stop : “;
getch();
exit(1);
}
circle(200,200,50); //Drawing a circle having center(200,200) and radius(50).
getch();
circle(300,203,40); //Drawing a circle having center(300,203) and radius(40).
getch();
circle(500,303,80); //Drawing a circle having center(500,303) and radius(80).
```

```
getch();
closegraph();
}
```

**Output :**

This graphics program draws three circles on the screen.

# 59. Program to change the background colors on the screen

```
#include <conio.h>
#include <graphics.h>
#include <stdlib.h>
#include <stdio.h>

void main (int)
{
int gdriver=DETECT,gmode,errorcode; //Requesting auto-detection.
int midx,midy,x;
//Initializing graphics and local variables.
initgraph(&gdriver,&gmode,“d:\bc3\bgi”);
//Reading result of initialization.
errorcode=graphresult();
if(errorcode!=grOk)
//An error occurred.
{
printf(“Graphics error occurred : %s \n”,grapherrormsg(errorcode));
printf(“Press any key to stop : “);
getch();
exit(1); //Terminate the program due to error.
}
/*Changing the background color.
Note : Press enter to see the first screen as it is black and
it may appear as if the program has stopped running.*/
for(x=0;x<=15;x++)
{
```

```
setbkcolor(x);
getch();
}
closegraph();
}
```

**Output :**

This graphics program changes the background colors on the screen gradually from black to white using the ‘setbkcolor’ command.

# 60. Program to enter three integers and output the biggest integer using IF

```
#include <iostream.h>
#include <conio.h>
int main()
{
clrscr();
int x,y,z,biggest;
cout << “Enter 3 integers : “;
cin>>x>>y>>z;
biggest=x;
if(y>biggest)
biggest=y;
if(z>biggest)
biggest=z;
cout << “The biggest integer out of the 3 integers you typed “;
cout << x << “, ” << y << ” & ” << z << ” is : ” << “\n” << biggest << “\n”;
getch();
return 0;
}
```

This program takes in three integers x, y and z as a screen input from the user.
It then determines the biggest integer of the three using the ‘IF’ statement.
It then outputs the biggest integer using the ‘cout’ command.

**INPUT :**

32 47 44

**OUTPUT :**

The biggest integer out of the 3 integers you typed 32, 47 & 44 is :
47

# 61. Program to enter an integer and print out its successor

```
#include <iostream.h>
#include <conio.h>
void value(int);

void main()
{
clrscr();
int x;
cout << “Enter an integer : “;
cin>>x;
cout << “The successor of ” << x << ” is “;
value(x);
getch();
}
void value(int x)
{
x++;
cout << x << “.” << endl;
}
```

This program takes in an integer x as a screen input from the user.
It then determines the successor of the integer and outputs it using the ‘cout’ command.

**INPUT :**

**OUTPUT :**

The successor of 49 is 50.

# 62. Program to enter an integer and output the cube of that integer

```
#include <iostream.h>
#include <conio.h>
int cube(int x); //The prototype.

void main()
{
clrscr();
int a;
cout << “Enter an integer : “;
cin>>a;
cout << “The cube of ” << a << ” is : ” << cube(a) << endl; //Call the function cube(a).
getch();
}
//Defining the function.
int cube(int x)
{
int y;
y=x*x*x;
return(y);
}
```

This program takes in an integer a as a screen input from the user.

It then determines the integer’s cube and outputs it using the ‘cout’ command.

**INPUT :**

**OUTPUT :**

The cube of 8 is : 512

# 63. Program to enter a string and find its length

```
#include <iostream.h>
#include <conio.h>
#include <string.h>

void main()
{
clrscr();
int slength;
char x[81]; //Allowing the user to input a maximum of 80 characters.
cout << “Enter the string : ” << endl;
cin>>x;
slength=strlen(x);
cout << “The length of the string ” << x << ” is ” << slength << “.” << endl;
getch();
}
```

This program takes in a string x as a screen input from the user.
It then determines the length of the string using the ‘strlen’ function.
This length is finally outputted using the ‘cout’ command.

**INPUT :**

goldfish

**OUTPUT :**

The length of the string goldfish is 8.

# 64. Program to switch between different cases

```
#include <iostream.h>
#include <conio.h>
int main()
{
clrscr();
int choice;
cout << “1. Talk” << endl;
cout << “2. Eat” << endl;
cout << “3. Play” << endl;
cout << “4. Sleep” << endl;
cout << “Enter your choice : ” << endl;
cin>>choice;
switch(choice)
{
case 1 : cout << “You chose to talk…talking too much is a bad habit.” << endl;
break;
case 2 : cout << “You chose to eat…eating healthy foodstuff is good.” << endl;
break;
case 3 : cout << “You chose to play…playing too much everyday is bad.” << endl;
break;
case 4 : cout << “You chose to sleep…sleeping enough is a good habit.” << endl;
break;
default : cout << “You did not choose anything…so exit this program.” << endl;
}
getch();
}
```

This program takes in the user’s choice as a screen input.

Depending on the user’s choice, it switches between the different cases.

The appropriate message is then outputted using the ‘cout’ command.

**INPUT :**

3

**OUTPUT :**

You chose to play…playing too much everyday is bad.

# 65. Program to enter the principal, rate & time and print the simple interest

```
#include <iostream.h>
#include <conio.h>

void main()
{
clrscr();
int x;
float sinterest,principal,rate,time;
for(x=4;x>=0;x—)
{
cout << “Enter the principal, rate & time : ” << endl;
cin>>principal>>rate>>time;
sinterest=(principal*rate*time)/100;
cout << “Principal = $” << principal << endl;
cout << “Rate = ” << rate << “%” << endl;
cout << “Time = ” << time << ” years” << endl;
cout << “Simple Interest = $” << sinterest << endl;
}
getch();
}
```

This program takes in the prinicipal, rate and time as a screen input from the user.

The program is executed (run) 5 times using the ‘FOR’ loop.

It calculates the simple interest using the formula I = PTR/100.

The principal, rate, time and the simple interest are then outputted using the ‘cout’ command.

**INPUT :**

1000 5 3

**OUTPUT :**

Principal = $1000
Rate = 5%
Time = 3 years
Simple Interest = $150

# 66. Program to enter an integer and print if it is prime or composite

```
#include <iostream.h>
#include <conio.h>
#include <process.h>

void main()
{
clrscr();
int num1,x;
cout << “Enter an integer : ” << endl;
cin>>num1;
for(x=2;x<num1;x++)
{
if(num1%x==0)
{
cout << num1 << ” is a composite number.” << endl;
getch();
exit(0);
}
else
{
cout << num1 << ” is a prime number.” << endl;
getch();
exit(0);
}
}
}
```

This program takes in an integer num1 as a screen input from the user.
It then determines whether the integer is prime or composite.
It finally outputs the approriate message by writing to the ‘cout’ stream.

**INPUT :**

23

**OUTPUT :**

23 is a prime number.

# 67. Program to enter the sale value and print the agent’s commission

```
#include <iostream.h>
#include <conio.h>

void main()
{
clrscr();
long int svalue;
float commission;
cout << “Enter the total sale value : ” << endl;
cin>>svalue;
if(svalue<=10000)
{
commission=svalue*5/100;
cout << “For a total sale value of $” << svalue << “, “;
cout << “the agent’s commission is $” << commission;
}
else if(svalue<=25000)
{
commission=svalue*10/100;
cout << “For a total sale value of $” << svalue << “, “;
cout << “the agent’s commission is $” << commission;
}
else if(svalue>25000)
{
commission=svalue*20/100;
cout << “For a total sale value of $” << svalue << “, “;
```

```
cout << “the agent’s commission is $” << commission;
}
getch();
}
```

This program takes in the total sale value as a screen input from the user.

The program then calculates the agent’s commission with the help of the ‘IF-ELSE’ command as follows :

5% if the total sale value is less than or equal to $10000.

10% if the total sale value is less than or equal to $25000.

20% if the total sale value is greater than $25000. It then outputs the agent’s commission using the ‘cout’ command.

**INPUT :**

26000

**OUTPUT :**

For a total sale value of $26000, the agent’s commission is $5200

# 68. Program to enter the unit reading and output the customer’s telephone bill

```
#include <iostream.h>
#include <conio.h>
int main()
{
clrscr();
long int units,charge=0;
float total;
const int rent=25;
cout << “Enter the number of units used : “;
cin>>units;
if(units>200)
charge=(units-200)*20+150*40+50*60;
else if(units>50)
charge=(units-50)*40+50*60;
else
charge=units*60;
total=0.01*charge+rent;
cout << “You have used ” << units << ” units.” << endl;
cout << “Your total telephone bill is $” << total;
getch();
return 0;
}
```

This program takes in the number of units used (‘units’) as a screen input from the user. It then calculates the total telephone bill for the customer on the following basis :

A compulsory fee of $25, plus
60 cents per unit for the first 50 units,
40 cents per unit for the next 150 units,
20 cents per unit for anything above 200 units.
It then outputs the bill using the ‘cout’ command.

**INPUT :**

250

**OUTPUT :**

You have used 250 units.
Your total bill is $125

# 69. Program to find the total days in the year till date

```
#include <iostream.h>
#include <conio.h>

void main()
{
clrscr();
int day,month,total;
int days_per_month[12]={31,28,31,30,31,30,31,31,30,31,30,31};
cout << “Enter the month : ” << endl;
cin>>month;
cout << “Enter the day : ” << endl;
cin>>day;
total=day;
for(int x=0;x<month-1;x++)
total+=days_per_month[x];
cout << “The number of days in this year till date = ” << total << endl;
getch();
}
```

This program takes in the current day and month as a screen input from the user.

It then calculates the total number of days in the current year till date and outputs it using the ‘cout’ command.

**INPUT :**

```
6
12
```

**OUTPUT :**

The number of days in this year till date = 163

# 70. Program to compute the fibonacci series

```
#include <iostream.h>
#include <conio.h>

void main()
{
clrscr();
int a,b,x,y,num1,ct;
a=0;
b=1;
cout << “Enter the number of terms (less than 25) : ” << endl;
cin>>num1;
cout << a << endl;
cout << b << endl;
for(ct=1;ct<=num1-2;ct++)
{
x=a+b;
cout << x << endl;
y=a;
a=b;
b=x;
}
getch();
}
```

This program takes in the number of terms num1 in the fibonacci series (less than 25) as a screen input from the user.

It then computes the fibonacci series and prints it out using the ‘cout’ command.

**INPUT :**

12

**OUTPUT :**

0
1
1
2
3
5
8
13
21
34
55
89

# 71. Program to enter an integer and print its total value based on the formula

# ‘x - 1/3!x^3 + 1/5!x^5 - 1/7!x^7 + 1/9!x^9’

```
#include <iostream.h>
#include <conio.h>
#include <math.h>
int main()
{
clrscr();
float factorial=1;
float num,tot,term,total;
int i,n=20,index,j=1;
cout << “Enter a single-digit integer : \n”;
cin>>num;
tot=num;
total=num;
for(i=2,index=3;i<=n;i++,index+=2)
{
for(j=1,factorial=1;j<=index;j++)
factorial*=j;
tot=tot*pow((double)(-1),(double)(2*i-1))*num*num;
term=tot/factorial;
total+=term;
}
cout << “Total = ” << total << endl;
getch();
return 0;
}
```

This program takes in an integer num as a screen input from the user.
It then calculates the total value of the integer based on the formula x - 1/3!x^3 + 1/5!x^5 -

1/7!x^7 + 1/9!x^9.

It then outputs the final answer using the ‘cout’ command.

**INPUT :**

3

**OUTPUT :**

Total = 0.14112

# 72. Program to convert days into years and weeks

```
#include <iostream.h>
#include <conio.h>

void main()
{
clrscr();
int days,years,weeks,num1;
cout << “Enter the number of days : ” << endl;
cin>>days;
years=days/365;
num1=days-(years*365);
weeks=days/7;
num1=days-(weeks*7);
cout << days << ” days = ” << endl;
cout << weeks << ” weeks OR ” << endl;
cout << years << ” years.” << endl;
getch();
}
```

This program takes in the number of days days as a screen input from the user.

It then converts the days into years as well as weeks and outputs it using the ‘cout’ command.

**INPUT :**

789

**OUTPUT :**

789 days =

112 weeks OR

2 years.

# 73. Program to find the roots of a quadratic equation

```
#include <iostream.h>
#include <conio.h>
#include <math.h>
int main()
{
clrscr();
float a,b,c,d,root1,root2;
cout << “Enter the 3 coefficients a, b, c : ” << endl;
cin>>a>>b>>c;
if(!a){
if(!b)
cout << “Both a and b cannot be 0 in ax^2 + bx + c = 0” << “\n”;
else
{
d=-c/b;
cout << “The solution of the linear equation is : ” << d << endl;
}
}
else
{
d=b*b-4*a*c;
if(d>0)
root1=(-b+sqrt(d))/(2*a);
root2=(-b-sqrt(d))/(2*a);
cout << “The first root = ” << root1 << endl;
cout << “The second root = ” << root2 << endl;
}
```

```
getch();
return 0;
}
```

This program takes in the values of the three coefficients a, b, and c as a screen input from the user.

It then determines the roots of the quadratic equation using the formula ax^2 + bx + c = 0.

The two roots are then outputted using the ‘cout’ command.

**INPUT :**

4 4 -3

**OUTPUT :**

The first root = 0.5

The second root = -1.5

# 74. Program to enter an integer and output it in the reversed form

```
#include <iostream.h>
#include <conio.h>

void main()
{
clrscr();
long int num1,num2,rnum=0;
cout << “Enter an integer : ” << endl;
cin>>num1;
num2=num1;
do
{
rnum=rnum*10;
int digit=num1%10;
rnum+=digit;
num1/=10;
}
while(num1);
cout << “The integer you typed is ” << num2 << “.” << endl;
cout << “The reversed integer is ” << rnum << “.” << endl;
getch();
}
```

This program takes in an integer num1 as a screen input from the user.
It then outputs the integer in its reversed form using the ‘cout’ command.

**INPUT :**

987

**OUTPUT :**

The integer you typed is 987.
The reversed integer is 789.

# 75. Program to count the number of words and characters in a sentence

```
#include <iostream.h>
#include <conio.h>

void main()
{
clrscr();
int countch=0;
int countwd=1;
cout << “Enter your sentence in lowercase: ” << endl;
char ch=‘a’;
while(ch!=’\r’)
{
ch=getche();
if(ch==’ ‘)
countwd++;
else
countch++;
}
cout << “\n Words = ” << countwd << endl;
cout << “Characters = ” << countch-1 << endl;
getch();
}
```

This program takes in a sentence as a screen input from the user.

It then determines the number of words and characters in the sentence using the ‘WHILE’ loop and outputs them using the ‘cout’ command.

**INPUT :**

this is a nice program

**OUTPUT :**

Words = 5
Characters = 18

# 76. write program to take the salary of the employee as a screen input from the user.

# It then deducts the income tax from the salary on the following basis :

# 30% income tax if the salary is above $15000.

# 20% income tax if the salary is between $7000 and $15000.

# 10% income tax if the salary is below $7000.

```
#include <iostream.h>
#include <conio.h>

void main()
{
clrscr();
int itrate;
float salary,itax,nsalary=0;
cout << “Enter the salary : “;
cin>>salary;
if(salary>15000)
{
itax=salary*30/100;
itrate=30;
}
else if(salary>=7000)
{
itax=salary*20/100;
itrate=20;
}
else
{
itax=salary*10/100;
itrate=10;
}
nsalary=salary-itax;
cout << “Salary = $” << salary << endl;
cout << “Your income tax @ ” << itrate << “% = $” << itax << endl;
```

```
cout << “Your net salary = $” << nsalary << endl;
getch();
}
```

This program takes in the salary of the employee as a screen input from the user.
It then deducts the income tax from the salary on the following basis :
30% income tax if the salary is above $15000.
20% income tax if the salary is between $7000 and $15000.
10% income tax if the salary is below $7000.
The salary, income tax and the net salary is then outputted using the ‘cout’ command.

**INPUT :**

12000

**OUTPUT :**

Salary = $12000
Your income tax @ 20% = $2400
Your net salary = $9600

# 77. Program to find the sum of each row & column of a matrix of size n x m and

# if matrix is square, find the sum of the diagonals also.

```
#include <iostream.h>
#include <conio.h>
int main()
{
clrscr();
int A[10][10],m,n,x,y,sum=0;
//Create a Matrix A
cout << “Enter number of rows and columns in Matrix A : \n”;
cin>>n>>m;
cout << “Enter elements of Matrix A : \n”;
for(x=1;x<n+1;++x)
for(y=1;y<m+1;++y)
cin>>A[x][y];
//Find sum of each row
for(x=1;x<n+1;++x)
{
A[x][m+1]=0;
for(y=1;y<m+1;++y)
A[x][m+1]=A[x][m+1]+A[x][y];
}
//Find sum of each column
for(y=1;y<m+1;++y)
{
A[n+1][y]=0;
for(x=1;x<n+1;++x)
A[n+1][y]+=A[x][y];
}
```

```
cout << “\nMatrix A, Row Sum (Last Column)” << ” and Column Sum (Last Row) : \n”;
for(x=1;x<n+1;++x)
{
for(y=1;y<m+2;++y)
cout << A[x][y] << ”     “;
cout << “\n”;
}
//Print sum of each column
x=n+1;
for(y=1;y<m+1;++y)
cout << A[x][y] << ”     “;
cout << “\n”;
if(m==n)
{
for(x=1;x<m+1;x++)
for(y=1;y<n+1;y++)
if(x==y)
sum+=A[x][y];
else
if(y==m-(x+1))
sum+=A[x][y];
}
cout << “Sum of diagonal elements is : ” << sum << endl;
getch();
return 0;
}
```

This program takes in the number of rows (n) and columns (m) as well as the elements as a screen input in a matrix n x m.

It then calculates the sum of each row and each column and outputs it using the ‘cout’ command.

Also, if it is a square matrix, it calculates the sum of diagonal elements and prints it out.

**INPUT :**

3 3
9 8 7 6 5 4 3 2 1

**OUTPUT :**

Matrix A, Row Sum(Last Column) and Column Sum(Last Row) :
9 8 7 24
6 5 4 15
3 2 1 6
18 15 12
Sum of diagonal elements is : 15

# 78. Program to convert 2-digit octal number into binary number and print it

```
 #include <iostream.h>
#include <conio.h>
void octobin(int);

void main()
{
clrscr();
int a;
cout << “Enter a 2-digit octal number : “;
cin>>a;
octobin(a);
getch();
}
void octobin(int oct)
{
long bnum=0;
int A[6];
//Each octal digit is converted into 3 bits, 2 octal digits = 6 bits.
int a1,a2,quo,rem;
a2=oct/10;
a1=oct-a2*10;
for(int x=0;x<6;x++)
{
A[x]=0;
}
//Storing the remainders of the one’s octal digit in the array.
for (x=0;x<3;x++)
```

```
{
quo=a1/2;
rem=a1%2;
A[x]=rem;
a1=quo;
}
//Storing the remainders of the ten's octal digit in the array.
for(x=3;x<6;x++)
{
quo=a2/2;
rem=a2%2;
A[x]=rem;
a2=quo;
}
//Obtaining the binary number from the remainders.
for(x=x-1;x>=0;x—)
{
bnum*=10;
bnum+=A[x];
}
cout << "The binary number for the octal number " << oct << " is " << bnum << "." << endl;
}
```

This program takes in a two-digit octal number a as a screen input from the user.

It then converts the octal number into a binary number and outputs it using the 'cout' command.

**INPUT :**

13

**OUTPUT :**

The binary number for the octal number 13 is 1011.

# 79. Program to identify if an input is a symbol, digit or character

```
#include <iostream.h>
#include <conio.h>

void main()
{
clrscr();
char charac;
cout << “Enter your input : ” << endl;
cin>>charac;
if(((charac>=‘A’)&&(charac<=‘Z’))||((charac>=‘a’)&&(charac<=‘z’)))
cout << “Your input ” << charac << ” is a character.” << endl;
else if((charac>=‘0’)&&(charac<=‘9’))
cout << “Your input ” << charac << ” is a digit.” << endl;
else
cout << “Your input ” << charac << ” is a symbol.” << endl;
getch();
}
```

This program takes in a character, a digit or a symbol charac as a screen input from the user.

It then identifies whether the input is a symbol, a digit or a character and outputs the appropriate message using the ‘cout’ command.

**INPUT :**

#

**OUTPUT :**

Your input # is a symbol.

# 80. Program to enter three integers and output the smallest integer using IF

```
#include <iostream.h>
#include <conio.h>
int main()
{
clrscr();
int x,y,z,smallest;
cout << “Enter 3 integers : “;
cin>>x>>y>>z;
smallest=x;
if(y<smallest)
smallest=y;
if(z<smallest)
smallest=z;
cout << “The smallest integer out of the 3 integers you typed “;
cout << x << “, ” << y << ” & ” << z << ” is : ” << “\n” << smallest << “\n”;
getch();
return 0;
}
```

This program takes in three integers x, y and z as a screen input from the user.

It then determines the smallest integer of the three and prints it out using the ‘cout’ command.

**INPUT :**

82 78 86

**OUTPUT :**

The smallest integer out of the 3 integers you typed 82, 78 & 86 is :
78

# 81. Program to enter a sentence and output the number of

# uppercase & lowercase consonants, uppercase & lowercase vowels in sentence.

```
#include <iostream.h>
#include <conio.h>

void main()
{
clrscr();
char line[80];
int number_of_vowels,uc,lc,uv,lv;
uc=lc=uv=lv=0;
cout << “Enter your sentence : ” << endl;
cin.getline(line,80);
for(int x=0; line[x]!=’\0’;x++)
{
if(line[x]==‘A’||line[x]==‘E’||line[x]==‘I’||line[x]==‘O’||line[x]==‘U’)
uv++;
else if(line[x]==‘a’||line[x]==‘e’||line[x]==‘i’||line[x]==‘o’||line[x]==‘u’)
lv++;
else if(line[x]>+65&&line[x]<=90)
uc++;
else if (line[x]>=97&&line[x]<=122)
lc++;
}
//Printing the output.
cout << “Uppercase Consonants = ” << uc << “.” << endl;
cout << “Lowercase Consonants = ” << lc << “.” << endl;
cout << “Uppercase Vowels = ” << uv << “.” << endl;
cout << “Lowercase Vowels = ” << lv << “.” << endl;
```

```
number_of_vowels=uv+lv;
cout << “Number of vowels = ” << number_of_vowels << endl;
getch();
}
```

This program takes in a sentence as a screen input from the user.

It then computes the number of uppercase and lowercase consonants, uppercase and lowercase vowels and the total number of vowels.

It then outputs it using the ‘cout’ command.

**INPUT :**

These programs are truly AMAZING

**OUTPUT :**

Uppercase Consonants = 5.

Lowercase Consonants = 13.

Uppercase Vowels = 3.

Lowercase Vowels = 7.

Number of vowels = 10

# 82. Program to print the first 10 lines of pascal’s triangle

```
#include <iostream.h>
#include <conio.h>
#include <iomanip.h>
long triangle(int x,int y);
int main()
{
clrscr();
const lines=10;
for (int i=0;i<lines;i++)
for (int j=1;j<lines-i;j++)
cout << setw(2) << ” “;
for (int j=0;j<=i;j++)
cout << setw(4) << triangle(i,j);
cout << endl;
getch();
}
long triangle(int x,int y)
{
if(x<0||y<0||y>x)
return 0;
long c=1;
for (int i=1;i<=y;i++,x—)
c=c*x/i;
return c;
}
```

This program does not take in any screen inputs from the user.

It just prints out the first ten lines of the pascal's triangle using the 'cout' command.

**INPUT :**

No input from the user for this program.

**OUTPUT :**

1 10 45 120 210 252 210 120 45 10 1

# 83. Program to enter 10 integers in a single-dimension array and then print out the array in ascending order

```
#include <iostream.h>
#include <conio.h>

void main()
{
clrscr();
int array[10],t;
for(int x=0;x<10;x++)
{
cout << “Enter Integer No. ” << x+1 << ” : ” << endl;
cin>>array[x];
}
for (x=0;x<10;x++)
{
for(int y=0;y<9;y++)
{
if(array[y]>array[y+1])
{
t=array[y];
array[y]=array[y+1];
array[y+1]=t;
}
}
}
cout << “Array in ascending order is : “;
```

```
for (x=0;x<10;x++)
cout << endl << array[x];
getch();
}
```

This program takes in the ten integers array[x] to be stored in the single-dimensional array as a screen input from the user.

It then sorts out these ten integers into ascending order and prints them out using the ‘cout’ command.

**INPUT :**

```
43
67
53
21
6
78
92
48
95
8
```

**OUTPUT :**

```
Array in ascending order is :
6
8
21
43
48
53
67
78
```

92
95

# 84. Program to find the sum of either of the diagonals of a 4 x 4 matrix

```
#include <iostream.h>
#include <conio.h>

void main()
{
clrscr();
int x;
int A[4][4],sum=0; //Reading the matrix.
cout << “Enter the elements of the matrix : ” << endl;
for(int y=0;y<4;y++)
for (int x=0;x<4;x++)
{
cout << “Element ” << x+1 << “, ” << y+1 << ” : “;
cin>>A[x][y];
}
//Sum of either of the diagonal elements.
for(x=0;x<4;x++)
for(y=0;y<4;y++)
if(x==y)
sum+=A[x][y];
else if(y==4-(1+1));
sum+=A[x][y];
cout << “Sum of either of the diagonal elements is : ” << sum;
getch();
}
```

This program takes in the elements A[x][y] of the 4 x 4 matrix as a screen input from the user.

It then calculates the sum of either of its diagonals and outputs it using the ‘cout’ command.

**INPUT :**

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

**OUTPUT :**

Sum of either of the diagonal elements is : 34

# 85. Program to convert temperatures from Celsius to Fahrenheit and vice versa

```
#include <iostream.h>
#include <conio.h>

void main()
{
clrscr();
int choice;
float ctemp,ftemp;
cout << “1.Celsius to Fahrenheit” << endl;
cout << “2.Fahrenheit to Celsius” << endl;
cout << “Choose between 1 & 2 : ” << endl;
cin>>choice;
if (choice==1)
{
cout << “Enter the temperature in Celsius : ” << endl;
cin>>ctemp;
ftemp=(1.8*ctemp)+32;
cout << “Temperature in Fahrenheit = ” << ftemp << endl;
}
else
{
cout << “Enter the temperature in Fahrenheit : ” << endl;
cin>>ftemp;
ctemp=(ftemp-32)/1.8;
cout << “Temperature in Celsius = ” << ctemp << endl;
}
```

```
getch();
}
```

This program takes in the user's choice choice as a screen input from the user.

It then asks the user for a temperature in Celsius or Fahrenheit depending on the choice.

It then converts the Celsius temperature to Fahrenheit or vice versa and prints it out using the 'cout' command.

**INPUT :**

2

98.6

**OUTPUT :**

Temperature in Celcius = 37

# 86. Program to enter a character and output its ASCII code

```
#include <iostream.h>
#include <conio.h>

void main()
{
clrscr();
char charac;
cout << “Enter the character : ” << endl;
cin>>charac;
int num1=charac;
cout << “The ASCII code for ” << charac << ” is ” << num1 << “.” << endl;
getch();
}
```

This program takes in any character charac as a screen input from the user.
It then finds out its ASCII code and outputs it using the ‘cout’ command.

**INPUT :**

a

**OUTPUT :**

The ASCII code for a is 97.

# 87. Program to enter a letter and output the next 2 letters

```
#include <iostream.h>
#include <conio.h>

void main()
{
clrscr();
char charac;
cout << “Enter your letter : ” << endl;
cin>>charac;
cout << “The 2 letters are : ” << endl;
cout << (char)(charac-1) << endl;
cout << (char)(charac+1) << endl;
getch();
}
```

This program takes in a letter charac of the English alphabet as a screen input from the user.

It then determines its previous letter and next letter and prints it out using the ‘cout’ command.

**INPUT :**

x

**OUTPUT :**

The next 2 letters after x are :

y

z

# 88. Print 1 to 100 in C++, without loop and recursion

.

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

template<int N>
class PrintOneToN
{
public:
    static void print()
    {
    PrintOneToN<N-1>::print();  // Note that this is not recursion
    cout << N << endl;
    }
};

template<>
class PrintOneToN<1>
{
public:
    static void print()
    {
    cout << 1 << endl;
    }
};
int main()
{
    const int N = 100;
    PrintOneToN<N>::print();
```

```
    return 0;
}
```

**Output:**

```
1
2
3
..
..
98
99
100
```

# 89. simple C++ program to show working of getline.

```
#include <iostream>
#include <cstring>
using namespace std;
int main()
{
    string str;
    int t = 4;
    while (t—)
    {
    // Read a line from standard input in str
    getline(cin, str);
    cout << str << ” : newline” << endl;
    }
    return 0;
}
```

**Input :**

```
This
is
Geeks
for
```

**Output :**

```
This : newline
is  : newline
```

Geeks : newline

for : newline

The above input and output look good, there may be problems when input has blank lines in between.

**Input :**

This

is

Geeks

for

**Output:**

This : newline

: newline

is : newline

: newline

It doesn’t print the last 2 lines. The reason is that getline() reads till enter is encountered even if no characters are read. So even if there is nothing in the third line, getline() considers it as a single line. Further observe the problem in the second line.

The code can be modified to exclude such blank lines.

**Modified code:**

```
// A simple C++ program that uses getline to read
// input with blank lines
```

```
#include <iostream>
#include <cstring>
using namespace std;
int main()
{
    string str;
    int t = 4;
    while (t—)
    {
    getline(cin, str);

    // Keep reading a new line while there is
    // a blank line
    while (str.length()==0 )
    getline(cin, str);

    cout << str << ” : newline” << endl;
    }
    return 0;
}
```

**Input :**

This

is

Geeks

for

**Output :**

This : newline
is  : newline
Geeks : newline
for : newline

# 90. Print 2D matrix in different lines and without curly braces in C/C++?

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

int main()
{
    int m = 2, n = 3;
    int mat[][3] = { {1, 2, 3},
    {4, 5, 6},
    };
    for (int i = 0; i < m; i++)
    for (int j = 0; j < n; j++)

    // Prints ‘ ‘ if j != n-1 else prints ‘\n’
    cout << mat[i][j] << ” \n”[j == n-1];

    return 0;
}
```

**Output:**

```
1 2 3
4 5 6
```

# 91. A simple and complete C++ program to demonstrate friend Class

```
#include <iostream>
class A {
private:
    int a;
public:
    A() { a=0; }
    friend class B;     // Friend Class
};

class B {
private:
    int b;
public:
    void showA(A& x) {
    // Since B is friend of A, it can access
    // private members of A
    std::cout << “A::a=” << x.a;
    }
};

int main() {
    A a;
    B b;
    b.showA(a);
    return 0;
}
```

**Output:**

A::a=0

# 92. A simple and complete C++ program to demonstrate friend function of another class.

```
#include <iostream>

class B;

class A
{
public:
    void showB(B& );
};

class B
{
private:
    int b;
public:
    B()  {  b = 0; }
    friend void A::showB(B& x); // Friend function
};
 void A::showB(B &x)
{ // Since show() is friend of B, it can
    // access private members of B
    std::cout << “B::b = ” << x.b;
}

int main()
{  A a;
    B x;
    a.showB(x);
```

```
    return 0;
}
```

**Output:**

```
B::b = 0
```

# 93. A simple and complete C++ program to demonstrate global friend

```
#include <iostream>

class A
{
    int a;
public:
    A() {a = 0;}
    friend void showA(A&); // global friend function
};

void showA(A& x) {
    // Since showA() is a friend, it can access
    // private members of A
    std::cout << “A::a=” << x.a;
}

int main()
{
    A a;
    showA(a);
    return 0;
}
```

**Output:**

```
A::a = 0
```

# 94. Write a programs to show Constructors in C++

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

class Point
{
private:
    int x, y;
public:
    /***Constructor****/
    Point(int x1, int y1) { x = x1; y = y1; }

    int getX()          {  return x; }
    int getY()          {  return y; }
};

int main()
{
    Point p1(10, 15); // constructor is called here

    // Let us access values assigned by constructor
    cout << “p1.x = ” << p1.getX() << “, p1.y = ” << p1.getY();

    return 0;
}
```

**Output:**

p1.x = 10, p1.y = 15

# 95. Write a program on sorting.

```
#include <iostream>
#include <algorithm>

using namespace std;

void show(int a[])
{
    for(int i = 0; i < 10; ++i)
    cout << '\t' << a[i];
}

int main()
{
    int a[10]= {1, 5, 8, 9, 6, 7, 3, 4, 2, 0};
    cout << "\n The array before sorting is : ";
    show(a);

    sort(a, a+10);

    cout << "\n\n The array after sorting is : ";
    show(a);

    return 0;

}
```

**Output :**

```
The array before sorting is :    1      5      8
```

9 6 7 3 4 2 0

The array after sorting is : 0 1 2 3 4 5 6 7 8 9

# 96. Write a program to demonstrate Binary Search Algorithms in C++

```
#include <iostream>
#include <algorithm>

using namespace std;

void show(int a[], int arraysize)
{
    for(int i = 0; i < arraysize; ++i)
    cout << '\t' << a[i];
}



int main()
{
    int a[]= {1, 5, 8, 9, 6, 7, 3, 4, 2, 0};
    int asize = sizeof(a) / sizeof(a[0]);
    cout << "\n The array is : ";
    show(a, asize);

    cout << "\n\nLet's say we want to search for 2 in the array";
    cout << "\n So, we first sort the array";
    sort(a, a + 10);
    cout << "\n\n The array after sorting is : ";
    show(a, asize);
    cout << "\n\nNow, we do the binary search";
    if (binary_search(a, a + 10, 2))
    cout << "\nElement found in the array";
```

```
    else
    cout << "\nElement not found in the array";

    cout << "\n\nNow, say we want to search for 10";
    if (binary_search(a, a + 10, 10))
    cout << "\nElement found in the array";
    else
    cout << "\nElement not found in the array";

    return 0;

}
```

**Output :**

```
The array is :  1    5    8    9
6    7    3    4    2    0

Let's say we want to search for 2 in the array
 So, we first sort the array

 The array after sorting is :    0    1    2
3    4    5    6    7    8    9

Now, we do the binary search
Element found in the array

Now, say we want to search for 10
Element not found in the array
```

# 97. Can namespaces be nested in C++?Write the program.

```
#include <iostream>

int x = 20;
namespace outer {
    int x = 10;
    namespace inner {
    int z = x; // this x refers to outer::x
    }
}

int main()
{
    std::cout<<outer::inner::z; //prints 10
    getchar();
    return 0;
}
```

**Output :**

10.

# 98. Write a program to find the prime numbers from 2 to 100 by using nested for loop.

```
#include <iostream>

using namespace std;
int main () {
int i, j;
   for(i=2; i<100; i++) {
      for(j=2; j <= (i/j); j++)
if(!(i%j)) break; // if factor found, not prime
if(j > (i/j)) cout << i << ” is prime\n”; }
return 0;
}
```

**Output :**

```
2 is prime
3 is prime
5 is prime
7 is prime
11 is prime
13 is prime
17 is prime
19 is prime
23 is prime
29 is prime
31 is prime
37 is prime
41 is prime
43 is prime
```

47 is prime
53 is prime
59 is prime
61 is prime
67 is prime
71 is prime
73 is prime
79 is prime
83 is prime
89 is prime
97 is prime

# 99. Write a program using nested if statement.

```
#include <iostream>


using namespace std;

int main () {

// local variable declaration: int a = 100;
int b = 200;

// check the boolean condition if( a == 100 )
{

// if condition is true then check the following if( b == 200 )
{

// if condition is true then print the following

cout << “Value of a is 100 and b is 200” << endl; }

}
cout << “Exact value of a is : ” << a << endl; cout << “Exact value of b is : ” << b << endl;

return 0;

}
```

**Output :**

Value of a is 100 and b is 200

Exact value of a is : 100

Exact value of b is : 200

# 100. Write a program to demonstrate nested switch statement.

```
#include <iostream>


using namespace std;
int main ()
{
// local variable declaration: int a = 100;
int b = 200;
  switch(a) {
      case 100:
cout << “This is part of outer switch” << endl; switch(b)
    {
case 200:
cout << “This is part of inner switch” << endl;
                }
    }
cout << “Exact value of a is : ” << a << endl;
cout << “Exact value of b is : ” << b << endl;
return 0;
}
```

**Output :**

This is part of outer switch

This is part of inner switch

Exact value of a is : 100

Exact value of b is : 200

# 101. Predict the output of following C++ program.

**Program 1 :**

Assume that integers take 4 bytes.

```
#include<iostream>

using namespace std;

class Test
{
    static int i;
    int j;
};

int Test::i;

int main()
{
    cout << sizeof(Test);
    return 0;
}
```

Output:

4 (size of integer)

static data members do not contribute in size of an object. So ‘i’ is not considered in size of Test. Also, all functions (static and non-static both) do not contribute in size.

**Program 2 :**

```
#include<iostream>

using namespace std;
class Base1 {
 public:
    Base1()
    { cout << " Base1's constructor called" << endl;  }
};

class Base2 {
 public:
    Base2()
    { cout << "Base2's constructor called" << endl;  }
};

class Derived: public Base1, public Base2 {
    public:
    Derived()
    {  cout << "Derived's constructor called" << endl;  }
};

int main()
{
    Derived d;
```

```
    return 0;
}
```

**Ouput:**

Base1’s constructor called

Base2’s constructor called

Derived’s constructor called

In case of Multiple Inheritance, constructors of base classes are always called in derivation order from left to right and Destructors are called in reverse order.

**Program 3 :**

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

class A {
 public:
    A(int ii = 0) : i(ii) {}
    void show() { cout << “i = ” << i << endl;}
 private:
    int i;
};

class B {
 public:
    B(int xx) : x(xx) {}
    operator A() const { return A(x); }
 private:
    int x;
};
```

```
void g(A a)
{  a.show(); }

int main() {
    B b(10);
    g(b);
    g(20);
    getchar();
    return 0;
}
```

**Output:**

```
i = 10
i = 20
```

Since there is a Conversion constructor in class A, integer value can be assigned to objects of class A and function call g(20) works. Also, there is a conversion operator overloaded in class B, so we can call g() with objects of class B.

**Program 4 :**

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

class base {
    int arr[10];
};

class b1: public base { };

class b2: public base { };

class derived: public b1, public b2 {};
```

```
int main(void)
{
    cout<<sizeof(derived);
    getchar();
    return 0;
}
```

**Output:**

If integer takes 4 bytes, then 80.

**Program 5 :**

```
#include<iostream>

using namespace std;
class P {
public:
    void print()
    { cout <<" Inside P::"; }
};

class Q : public P {
public:
    void print()
    { cout <<" Inside Q"; }
};

class R: public Q {
};
```

```
int main(void)
{
    R r;

    r.print();
    return 0;
}
```

**Output:**

*Inside Q*

The print function is not defined in class R. So it is looked up in the inheritance hierarchy. *print()* is present in both classes *P* and *Q*, which of them should be called? The idea is, if there is multilevel inheritance, then function is linearly searched up in the inheritance heirarchy until a matching function is found.

**Program 6 :**

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

using namespace std;

class Base
{
public:
    Base()
    {
    fun(); //note: fun() is virtual
    }
    virtual void fun()
```

```
    {
    cout<<"\nBase Function";
    }
};

class Derived: public Base
{
public:
    Derived(){}
    virtual void fun()
    {
    cout<<"\nDerived Function";
    }
};

int main()
{
    Base* pBase = new Derived();
    delete pBase;
    return 0;
}
```

**Output:**

*Base Function*

See following excerpt from C++ standard for explanation.

*When a virtual function is called directly or indirectly from a constructor (including from the mem-initializer for a data member) or from a destructor, and the object to which the call applies is the object under construction or destruction, the function called is the one defined in the constructor or destructor's own class or in one of its bases, but not a function overriding it in a class derived from the constructor or destructor's class, or overriding it in one of the other base classes of the most derived object.*

Because of this difference in behavior, it is recommended that object's virtual function is

not invoked while it is being constructed or destroyed

**Program 7 :**

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

int x = 10;
void fun()
{
    int x = 2;
    {
    int x = 1;
    cout << ::x << endl;
    }
}

int main()
{
    fun();
    return 0;
}
```

***Output:***

10

If Scope Resolution Operator is placed before a variable name then the global variable is referenced. So if we remove the following line from the above program then it will fail in compilation.

int x = 10;

**Program 8 :**

```
#include<iostream>
using namespace std;
class Point {
private:
    int x;
    int y;
public:
    Point(int i, int j);  // Constructor
};

Point::Point(int i = 0, int j = 0)  {
    x = i;
    y = j;
    cout << “Constructor called”;
}

int main()
{
    Point t1, *t2;
    return 0;
}
```

***Output:***

Constructor called.

If we take a closer look at the statement “Point t1, *t2;:” then we can see that only one object is constructed here. t2 is just a pointer variable, not an object.

**Program 9 :**

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

class Point {
private:
    int x;
    int y;
public:
    Point(int i = 0, int j = 0);    // Normal Constructor
    Point(const Point &t); // Copy Constructor
};

Point::Point(int i, int j)  {
    x = i;
    y = j;
    cout << “Normal Constructor called\n”;
}

Point::Point(const Point &t) {
    y = t.y;
    cout << “Copy Constructor called\n”;
}

int main()
{
    Point *t1, *t2;
    t1 = new Point(10, 15);
    t2 = new Point(*t1);
    Point t3 = *t1;
    Point t4;
    t4 = t3;
    return 0;
```

```
}
```

***Output:***

```
Normal Constructor called
Copy Constructor called
Copy Constructor called
Normal Constructor called
```

See following comments for explanation:

```
Point *t1, *t2;   // No constructor call
t1 = new Point(10, 15);  // Normal constructor call
t2 = new Point(*t1);   // Copy constructor call
Point t3 = *t1;  // Copy Constructor call
Point t4;   // Normal Constructor call
t4 = t3;   // Assignment operator call
```

**Program 10 :**

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

class Test {
    int value;
public:
    Test(int v);
};

Test::Test(int v) {
    value = v;
}
```

```
int main() {
    Test t[100];
    return 0;
}
```

**Output**:

Compiler error

The class Test has one user defined constructor “Test(int v)” that expects one argument. It doesn’t have a constructor without any argument as the compiler doesn’t create the default constructor if user defines a constructor (See this).

Following modified program works without any error.

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

class Test {
    int value;
public:
    Test(int v = 0);
};

Test::Test(int v) {
    value = v;
}

int main() {
    Test t[100];
    return 0;
}
```

**Program 11 :**

```
#include<iostream>
using namespace std;
int &fun() {
    static int a = 10;
    return a;
}

int main() {
    int &y = fun();
    y = y +30;
    cout<<fun();
    return 0;
}
```

**Output:**

40

The program works fine because ‘a’ is static. Since ‘a’ is static, memory location of it remains valid even after fun() returns. So a reference to static variable can be returned.

**Program 12 :**

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

class Test
{
public:
    Test();
```

```
};

Test::Test()  {
    cout<<“Constructor Called \n”;
}

int main()
{
    cout<<“Start \n”;
    Test t1();
    cout<<“End \n”;
    return 0;
}
```

**Output:**

```
Start
End
```

Note that the line “Test t1();” is not a constructor call. Compiler considers this line as declaration of function t1 that doesn’t recieve any parameter and returns object of type Test.

**Program 13 :**

```
#include<iostream>

using namespace std;

class Test {
    int value;
public:
```

```
    Test (int v = 0) {value = v;}
    int getValue() { return value; }
};

int main() {
    const Test t;
    cout << t.getValue();
    return 0;
}
```

**Output:**

Compiler Error.

A const object cannot call a non-const function. The above code can be fixed by either making getValue() const or making t non-const. Following is modified program with getValue() as const, it works fine and prints 0.

```
#include<iostream>

using namespace std;

class Test {
    int value;
public:
    Test (int v = 0) { value = v; }
    int getValue() const { return value; }
};

int main() {
    const Test t;
    cout << t.getValue();
```

```
    return 0;
}
```

**Program 14 :**

```
#include<iostream>

using namespace std;

class Test {
    int &t;
public:
    Test (int &x) { t = x; }
    int getT() { return t; }
};

int main()
{
    int x = 20;
    Test t1(x);
    cout << t1.getT() << " ";
    x = 30;
    cout << t1.getT() << endl;
    return 0;
}
```

**Output:**

Compiler Error

Since t is a reference in Test, it must be initialized using Initializer List. Following is the modified program. It works and prints “20 30″.

```
#include<iostream>

using namespace std;

class Test {
    int &t;
public:
    Test (int &x):t(x) {  }
    int getT() { return t; }
};

int main() {
    int x = 20;
    Test t1(x);
    cout << t1.getT() << " ";
    x = 30;
    cout << t1.getT() << endl;
    return 0;
}
```

**Program 15:**

```
#include <iostream>

using namespace std;
```

```
class Complex
{
private:
    double real;
    double imag;

public:
    // Default constructor
    Complex(double r = 0.0, double i = 0.0) : real(r), imag(i) {}

    // A method to compare two Complex numbers
    bool operator == (Complex rhs) {
    return (real == rhs.real && imag == rhs.imag)? true : false;
    }
};

int main()
{
    // a Complex object
    Complex com1(3.0, 0.0);

    if (com1 == 3.0)
    cout << “Same”;
    else
    cout << “Not Same”;
    return 0;
}
```

**Output:**

Same

# Notes