Index

	initialization.		
25	Write a program to create Virtual Function.		
26	Write a program for using a (+) operator overloading.		
27	Write a program for using new and delete operator.		
28	Write a program for using function overloading.		
29	Write a program for using function overriding.		
30	Write a program for using pointer and string.		

1. **Write a program to create function with Return type and Argument.**

Answer:-

a. Algorithm

a. Program start with **#include<iostream>** and **<conio.h>** header file.

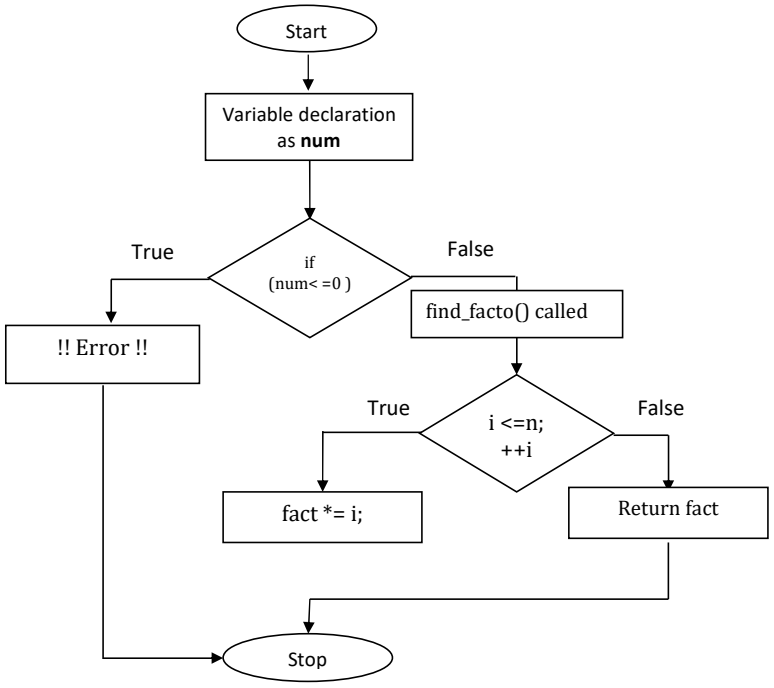
b. **int find_facto()** function defined, **return type with argument**.

c. In the **main( ) function**, declaration of variable with their **data type.**

d. **find_facto()** function called and pass them **num,** it return the factorial of the given number which is going to store in **fact** variable.

e. Print the value of **fact**, then it will terminate.

b. Flowchart



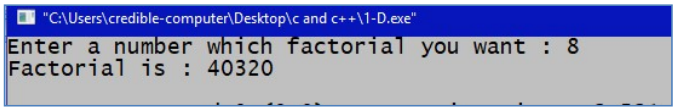
Formatted[Windows User]: Centered

Formatted[Windows User]: Centered

c. Coding for the program

```
#include<iostream>
#include<conio.h>
using namespace std;
int find_facto(int n)                                   // function definition
{
 int fact=1;
     for(int i = 1; i <=n; ++i)
     {      fact *= i;          }

return fact;
}

int main()
{
        int num;
        cout<<"Enter a number which factorial you want : ";
        cin>>num;

        if (num <= 0)
                cout << "Error! Factorial of a negative number doesn't exist.";

        else
                cout<<"Factorial is : "<<find_facto(num)<<endl;

        return 0;
}
```

d. Output screen:-



2. Write a program to create inline function.

Answer:-

a. Algorithm

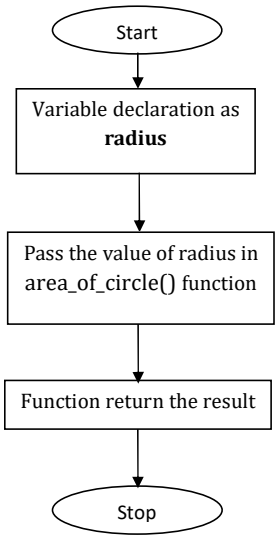
a. Program start with **#include<iostream>** and **<conio.h>** header file.

b. area_of_circle() defined as **Inline function** with argument.

c. Passing the **radius** as a function argument in area_of_circle() function.

d. It give us output then, program will terminate.

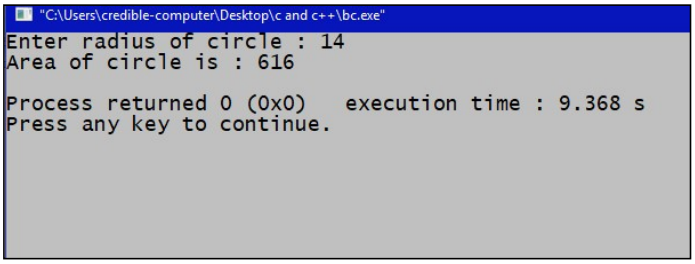
b. Flowchart



c. Coding for the program

```
#include<iostream>
#include<conio.h>
using namespace std;
inline float  area_of_circle(float radius)              //inline function definition
{
 cout<<"Area of circle is : "<<22/7.0 * radius * radius<<endl;              //area of circle
}

int main()
{     float radius;
      cout<<"Enter radius of circle : ";
     cin>>radius;
     area_of_circle(radius);                          //calculate the area of circle
     return 0;
}
```

d. Output screen:-



3. Write a program for recursive function.

Answer:-

Algorithm

a. Program start with **#include<iostream>** and **<conio.h>** header file.

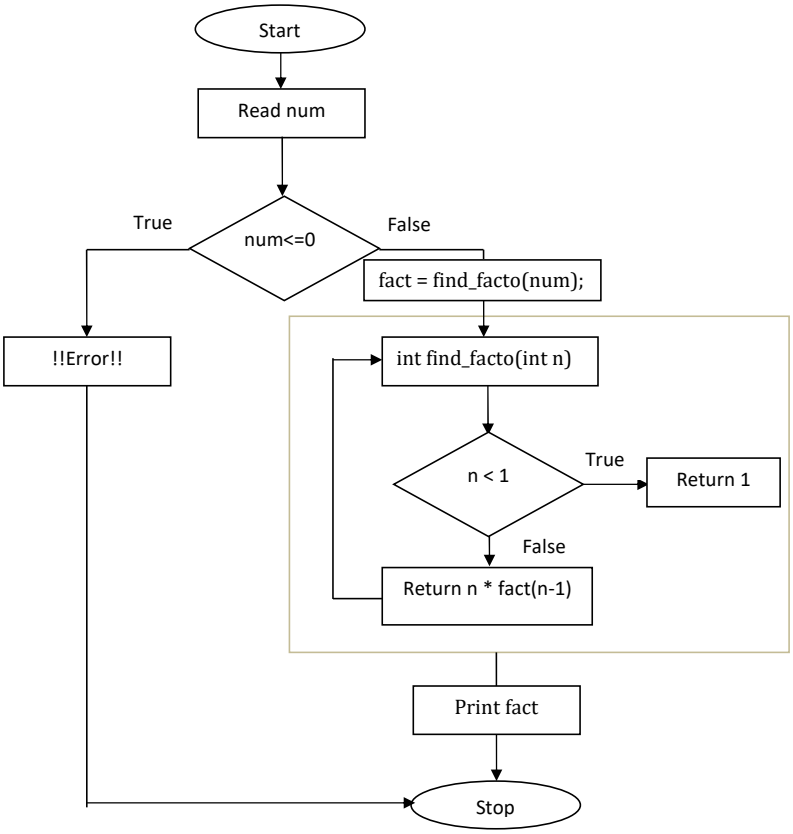
b. **find_facto()** function defined, **return type with argument** as **recursive** function.

c. In the **main( ) function**, declaration of variable with their **data type.**

d. **find_facto()** function called and pass them **num,** it return the factorial of the given number which is going to store in **fact** variable.

e. Print the value of **fact**, then it will terminate.

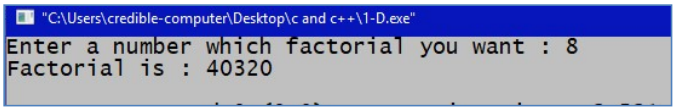
Flowchart



Coding for the program

```
#include<iostream>
#include<conio.h>
using namespace std;
find_facto(int n)              // function definition
{
    if( n < 1)
    return 1;
    else
    return n*find_facto(n-1);              //recursion
}

int main()
{
    int num, fact;
    cout<<"Enter a number which factorial you want : ";
    cin>>num;
    if (num <= 0)
            cout << "Error! Factorial of a negative number doesn't exist.";

    else
            cout<<"Factorial is : "<<find_facto(num)<<endl;

    return 0;
}
```

Output screen:-



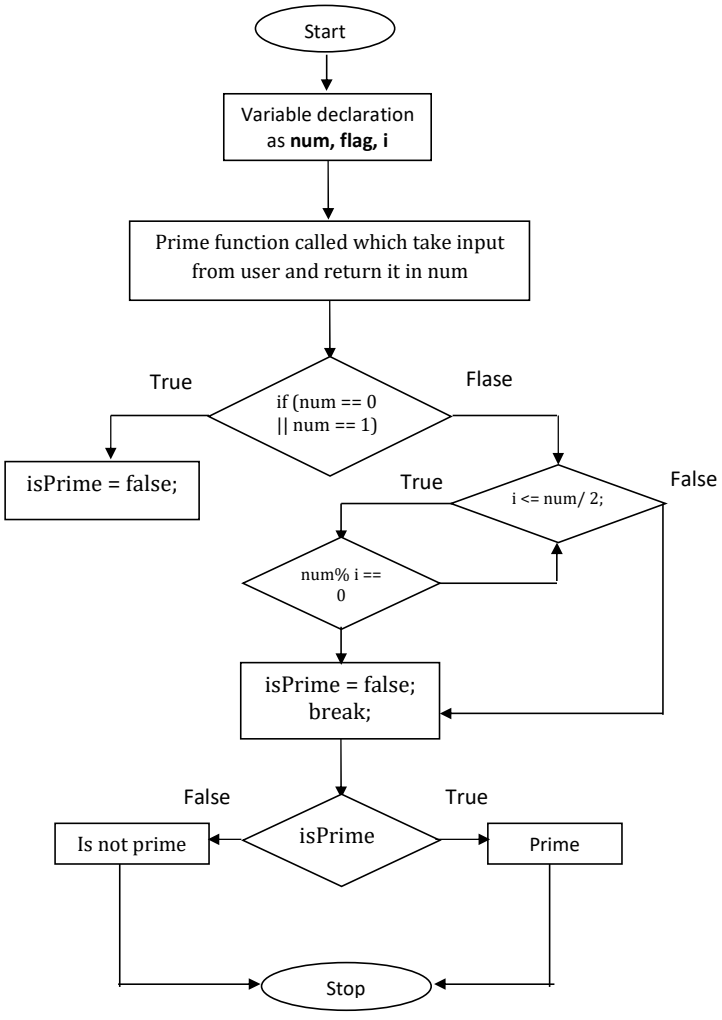
4. Write a program to create function with Return type and no Argument.

Answer:-

a. Algorithm

a. Program start with header files, **prime ()** function defined with no argument and return type. It take input from user and return it to main function
b. If number satisfied some condition then it be the prime otherwise is not prime.
c. At last program print the output and it will terminate.

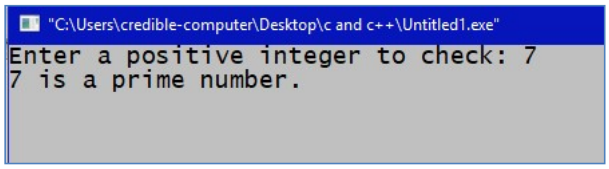
b. Flowchart



c. Coding for the program

```
#include <iostream>
using namespace std;
int prime();
int prime()                                  // Return type of function is int
{       int n;
       Cout<<"Enter a positive integer to check: ";
       cin >> n;
       return n;
}

int main()
{         int num, i;
          bool isPrime = true;
          num = prime();                     // No argument is passed to prime()

           if (num == 0 || num == 1)
                  {      isPrime = false;   }

          else {
                    for (i = 2; i <= num/ 2; ++i) {
                              if (num% i == 0) {
                                        isPrime = false;  break;
                              }
                    }
          }

  if (isPrime)
    cout << num << " is a prime number";

  else
    cout << num << " is not a prime number";
  return 0;
}
```

d. Output screen:-



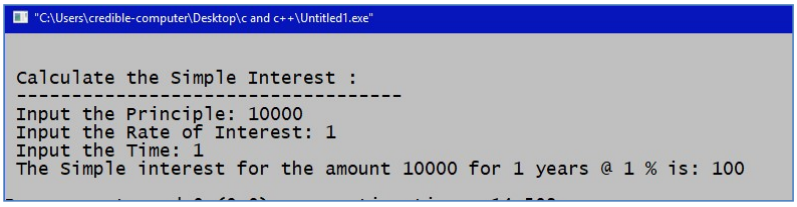
5. Write a program to Default Argument and calculates Simple Interest.

Answer:-

a. Coding for the program

```
#include<iostream>
using namespace std;
float sim_inter(int , int , int );

float sim_inter(int x, int y=1, int z=1)     // A function with default arguments, it can be called with
{                                            // 2 arguments or 3 arguments or 4 arguments
float sim_inter;
    sim_inter =(x*y*z)/100;
    return sim_inter;
}

int main()                                   /* Driver program to test above function*/
{
   int p,r,t;
   float i;
            cout << "\n\n Calculate the Simple Interest :\n";
            cout << " -----------------------------------\n";
    cout<<" Input the Principle: ";
    cin>>p;
    cout<<" Input the Rate of Interest: ";
    cin>>r;
    cout<<" Input the Time: ";
    cin>>t;
   i = sim_inter(p, r, t);
   cout<<" The Simple interest for the amount "<<p<<" for "<<t<<" years @ "<<r<<" %
is: "<<i;
      cout << endl;
   return 0;
}
```

Output screen:-



6. Write a program to demonstrate Friend function.

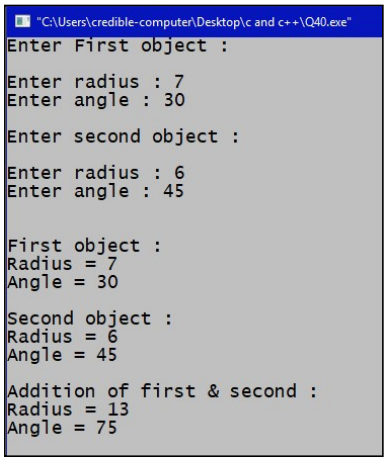
Answer:-

Coding for the program

```
#include<iostream>
#include<conio.h>
using namespace std;
class Polar
{
    float radius, angle;                                    //data members
public:
     void input()
    {       cout<<"Enter radius : ";
            cin>>radius;
            cout<<"Enter angle : ";
            cin>>angle;
    }

    void display()                                          //member fun. definition
    {
            cout<<"Radius = "<<radius<<endl;
            cout<<"Angle = "<<angle<<endl;
    }

friend Polar add(Polar,Polar);                  //friend function declaration
};
Polar add(Polar p1,Polar p2)                            //friend function definition
{
  Polar temp;
    temp.radius = p1.radius + p2.radius;
    temp.angle = p1.angle + p2.angle;           //Adding two Polar object
    return temp;                                        // then return a Polar object
}
```

```
int main()
{
    Polar p1, p2, addition;                              //objects created
            cout<<"Enter First object : "<<endl<<endl;
    p1.input();
            cout<<endl<<"Enter second object : "<<endl<<endl;
    p2.input();
            cout<<endl;
    addition = add(p1, p2);              //calling friend function add(Polar,Polar)
            cout<<endl<<"First object : "<<endl;
    p1.display();
            cout<<endl<<"Second object : "<<endl;
    p2.display();
            cout<<endl<<"Addition of first & second : "<<endl;
    addition.display();
    return 0;
}
```

Output screen:-



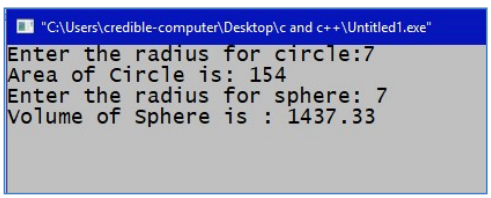
7. **Write a program for Single Inheritance.**

Answer:-

a. Coding for the program

```
#include<iostream>
#include<conio.h>
using namespace std;
const float pi=22/7.0;                                  //constant variable pi
class circle                                            //base class
{
    public:
      float radius;
      void area()
      {       cout<<"Enter the radius for circle:";
              cin>>radius;
              cout<<"Area of Circle is: "<<(pi)*radius*radius;
      }
};

class sphere: public circle                             //intermediate base class
{
public:
  void volume()
  {     cout<<"\nEnter the radius for sphere: ";
        cin>>radius;
        cout<<"Volume of Sphere is : "<<(4*(pi)*radius*radius*radius)/3;
  }
};
int main()
{       circle  c1;
        sphere sp;
        c1.area();
        sp.volume();
        return 0;
}
```

b. Output screen:-



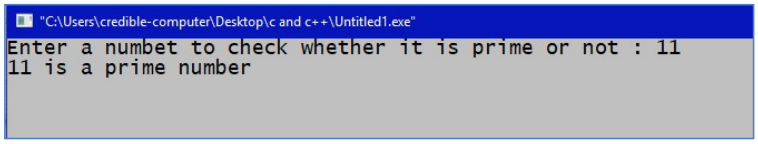
8. Write a program to create a class with Static Member function.

Answer:-

a. Coding for the program

```
#include<iostream>
#include<conio.h>
using namespace std;
class Static_demo
{   public:
        static void check_prime(int);          //sta_mem. function declaration
};

void Static_demo::check_prime(int a)           //sta_member function definition
{
    int i, j;
    int c=0;
    for(i=1; i<=a; i++)
    {
        if(a%i == 0)
        c++;
    }
    if(c<=2)
    cout<<a<<" is a prime number"<<endl;
    else
    cout<<a<<" is not a prime number"<<endl;
}
int main()
{
    int n;
    cout<<"Enter a numbet to check whether it is prime or not : ";
    cin>>n;
    Static_demo::check_prime(n);
                                    //calling static member function
   getch();
    return 0;
}
```

b. Output screen:-



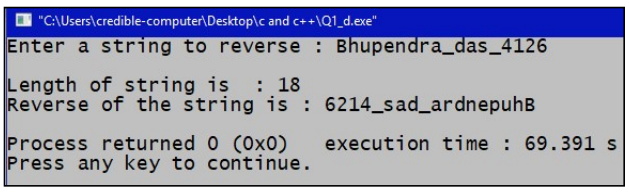
9. Write a program for String Manipulation with function.

Answer:-

a. Coding for the program

```
#include<iostream>
using namespace std;
int string_length(char);
void reverse_string(char);                          //Function declearation
void reverse_string(char rev[20],int length)        //function definition
{       int i;
        cout<<"Reverse of the string is : ";
        for(i =length-1; i>=0; --i)
        {
                cout<<rev[i];
        }
        cout<<endl;
}

int string_length(char p[20])
{
        int count;
        for(count = 0; p[count]!= '\0'; ++count);   //loop till getting null character
        return count;
}
int main()
{       char str[20];
        cout<<"Enter a string to reverse: ";
        cin>>str;
        cout<<endl;
        int len = string_length(str);               //storing length of string
        cout<<"Length of string is : "<<len<<endl;
        reverse_string(str,len);                    //passing length and string to the function
        return 0;
        }
```

b. Output screen:-



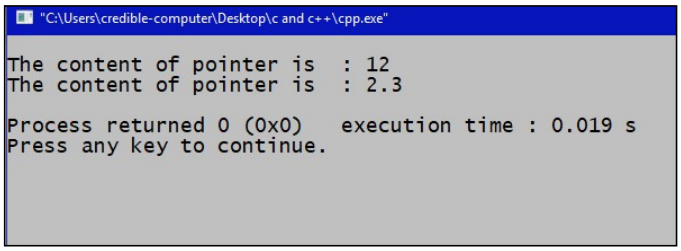
10. Write a program for Void Pointer.

Answer:-

a. Coding for the program

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

int main()
 {
            void* ptr;
            float f = 2.3;
            int a=12;

                                    // assign integer address to void pointer
          ptr=&a;
          cout << "\nThe content of pointer is  : ";
                                    // use type casting to print pointer content
          cout << *(static_cast<int*>(ptr));
                                    // assign float address to void pointer
          ptr = &f;

           cout << "\nThe content of pointer is  : ";
                                    // use type casting to print pointer content
           cout << *(static_cast<float*>(ptr));
          cout<<endl;
           return 0;
}
```

b. Output screen:-



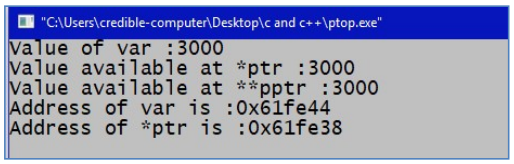
11. Write a program to create Pointer to Pointer variable.

Answer:-

a. Coding for the program

```
#include <iostream>
using namespace std;
int main ()
{
        int  var;
        int  *ptr;
        int  **pptr;

        var = 3000;

                        // take the address of var
        ptr = &var;

                        // take the address of ptr using address of operator &
        pptr = &ptr;

                        // take the value using pptr
        cout << "Value of var :" << var << endl;
        cout << "Value available at *ptr :" << *ptr << endl;
        cout << "Value available at **pptr :" << **pptr << endl;
        cout << "Address of var is :" << ptr << endl;
         cout << "Address of *ptr is :" << pptr << endl;

        return 0;
}
```

Output screen:-



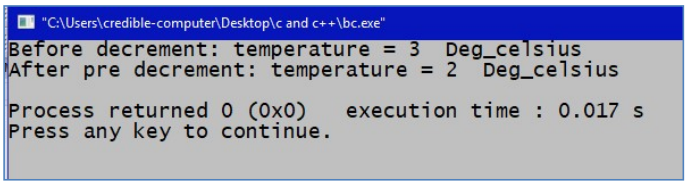
12. Write a program for Unary Operator Overloading.

Answer:-

a. Coding for the program

```
#include <iostream>
using namespace std;
class Degree_Celsius {
private:
        int temperature;
public:
    Degree_Celsius(int i = 0)          // Parameterised constructor
        {
        this->temperature= i;
        }
    Degree_Celsius operator--()        // Overloading the prefix operator
        {
        Degree_Celsius temp;
        temp.temperature= --temperature;
        return temp;
        }
    void display()                     // Function to display the value of i
        {
        cout << "temperature = " << temperature<< "  Deg_celsius"<<endl;
        }
        };
int main()
{
    Degree_Celsius i1(3);
    cout << "Before decrement: ";
    i1.display();
    Degree_Celsius i2 = --i1;      // Using the pre-decrement operator
    cout << "After pre decrement: ";
    i2.display();
return 0;
}
```

b. Output screen:-



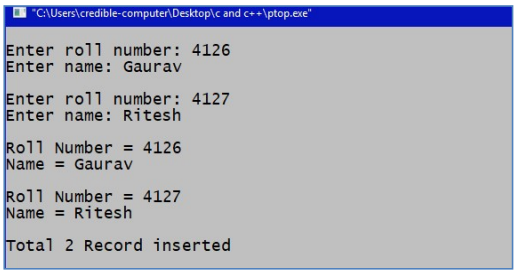
13. Write a program to create classes which have Static Member (Data member).

Answer:-

a. Coding for the program

```
#include <iostream>
using namespace std;
class Student {      private:
                              int rollNo;
                              char name[10];
                       public:
                              static int objectCount;
                              Student() {
                                 objectCount++;
                               }
                               void getdata() {
                                cout << endl<<"Enter roll number: ";
                                cin >> rollNo;
                                cout <<"Enter name: ";
                                cin >> name;
                                 }
                               void putdata() {
                                cout<<endl<<"Roll Number = "<< rollNo <<endl;
                                cout<<"Name = "<< name <<endl;
                                  }
            };
int Student::objectCount = 0;
int main(void) {
                Student s1, s2;
                s1.getdata();
                s2.getdata();
                s1.putdata();
                s2.putdata();
                cout <<endl<< "Total "<<Student::objectCount <<" Record inserted  " ;
                return 0;
      }
```

Output screen:-



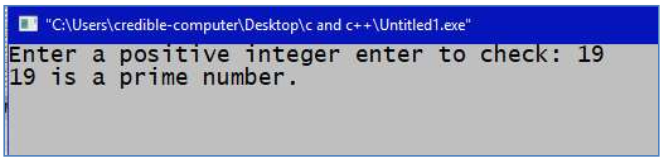
14. Write a program to create function with no Return type and no Argument.

Answer:-

a. Coding for the program

```
# include <iostream>
    using namespace std;
    void prime();
    int main()
    {
      prime();                              // No argument is passed to prime()
      return 0;
    }

    void prime()              // Return type of function is void because value is not returned
    {
      int num, i, flag = 0;
      cout << "Enter a positive integer enter to check: ";
      cin >> num;
      for(i = 2; i <= num/2; ++i)
      {
                if(num % i == 0)
                {  flag = 1;
                   break;
                }
      }
      if (flag == 1)
            {       cout << num << " is not a prime number.";
              }
      else
           {        cout << num << " is a prime number.";
              }
    }
```

Output:-



15. Write a program to create an Array of Objects.

Answer:-

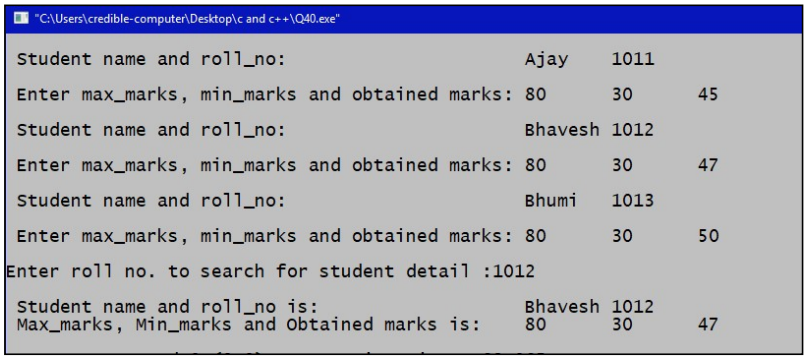
a. Coding for the program

```
#include<iostream>
using namespace std;
class student                                          //class defined
{   public:
     char  name[30];
     int   rollNo, max_marks, min_marks, obt_marks;       //data members
     void input()                                      //member function definition
 {
       cout<<"\n Student name and roll_no:\t\t\t";
       cin>>name>>rollNo;
       cout<<"\n Enter max_marks, min_marks and obtained marks:\t";
       cin>>max_marks>>min_marks>>obt_marks;
 }

 void display()                                        //member function definition
 {
       cout<<"\n Student name and roll_no is:\t\t\t"<<name<<"\t"<<rollNo;
       cout<<"\n Max_marks, Min_marks and Obtained marks is:\t";
       cout<<max_marks<<"\t"<<min_marks<<"\t"<<obt_marks;
       cout<<endl;
 }
};
int main()
{       int roll;
        student stu[2];
        for(int i=0;i<=2;i++)
        {       stu[i].input();
        }
        cout<<" \nEnter roll no. to search for student detail :";
        cin>>roll;
```

```
        for(int i=0;i<=2;i++)
        {
            if(roll==stu[i].rollNo)
            {   stu[i].display();
                }
        }
        return 0;
    }
```

---

c. Output screen:-



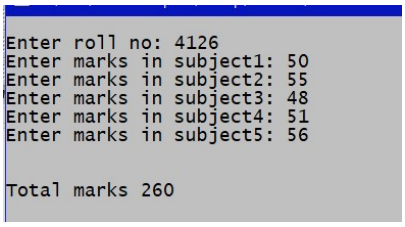
```
"C:\Users\credible-computer\Desktop\c and c++\Q40.exe"

Student name and roll_no:                    Ajay    1011
Enter max_marks, min_marks and obtained marks: 80      30      45
Student name and roll_no:                    Bhavesh 1012
Enter max_marks, min_marks and obtained marks: 80      30      47
Student name and roll_no:                    Bhumi   1013
Enter max_marks, min_marks and obtained marks: 80      30      50
Enter roll no. to search for student detail :1012

Student name and roll_no is:                 Bhavesh 1012
Max_marks, Min_marks and Obtained marks is:  80      30      47
```

16. Write a program to create a class which has Array as a data member.

Answer:-

```
#include<iostream>
using namespace std;
const int size=5;
class student
{
        int roll_no;
        int marks[size];                    //Array as data member
 public:
          void getdata ();
          void tot_marks ();
} ;

  void student :: getdata ()
 {   cout<<"\nEnter roll no: ";
     cin>>roll_no;
     for(int i=0; i<size; i++)
       {
          cout<<"Enter marks in subject"<<(i+1)<<": ";
          cin>>marks[i] ;
       }
  }

  void student :: tot_marks()                //calculating total marks
  {
     int total=0;
     for(int i=0; i<size; i++)
        total +=marks[i];
        cout<<"\n\nTotal marks "<<total;
}
int main() {
   student stu;
   stu.getdata() ;
   stu.tot_marks() ;

   cout<<endl<<endl;
   return 0;
}
```

Output screen:-

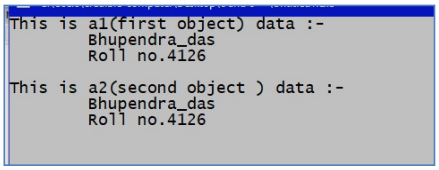


17. Write a program to demonstrate Copy Constructor.

Answer:-

```
#include <iostream>
#include <cstdio>
using namespace std;
class A
{
  public:
        int x;
        char name[20];

        A(char name1[], int a)                  // parameterized constructor.
        {       x=a;
                for(int j=0; j<=19;j++)
                name[j]=name1[j];
         }

        A(A &i)                                 // copy constructor
        {       x = i.x;
                for(int j=0; j<=19;j++)
                name[j]=i.name[j];

        }
};
int main()
{
        A a1("Bhupendra_das" , 100);          // Calling the parameterized
constructor.
        A a2(a1);                               //  Calling the copy constructor.

        cout<<"This is a1(first object) data :-"<<endl;
        puts(a1.name);
        cout<<"\t"<<a1.x<<"\t"<<endl;

        cout<<endl<<"This is a2(second object ) data :-"<<endl;
        puts(a2.name);
        cout<<"\t"<<a2.x<<"\t";

 return 0;
}
```

Output screen:-



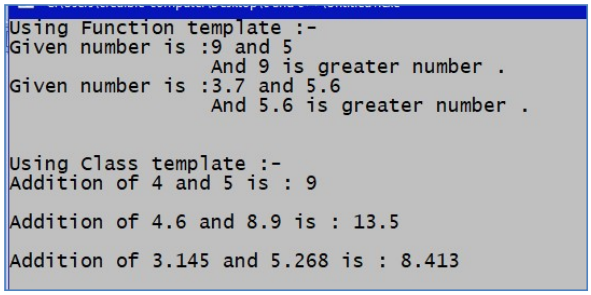
18. Write a program for Template Function.

Answer:-

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

template <class X>       //can replace 'class' keyword by 'typename' keyword
        X func(X a, X b)
        {
                cout<<"Given number is :"<<a<<" and " <<b<<endl;
                 if (a > b)
                        cout<<"\t\t And "<<a<<" is greater number ."<<endl;
                 else
                         cout<<"\t\t And "<<b<<" is greater number ."<<endl;
        }

template <class Obj>                    //class template
        class A
        {
                 private:
                        Obj a, b;
                  public:
                         A(Obj x, Obj y){
                                        a = x;
                                        b = y;
                                        }
                        void show(){
                                cout<<"Addition of "<<a<<" and ";
                                cout<<b<<" is : "<<add()<<endl;
                                }
                  Obj add(){
                         Obj c = a + b;
                         return c;
                }
        };

int main(){
        cout<<"Using Function template :-"<<endl;
        func(9, 5);                             // func(int, int);
        func(3.7, 5.6);                         //func(double, double);

        cout<<endl<<endl<<"Using Function template :-"<<endl;
        A <int>add_int(4, 5);
```

```
        A <float>add_float(4.6, 8.9);
        A  <double>add_double(3.145, 5.268);

        add_int.show();
        cout<<endl;
        add_float.show();
        cout<<endl;

        add_double.show();
        return 0;
    }
```

Output screen:-

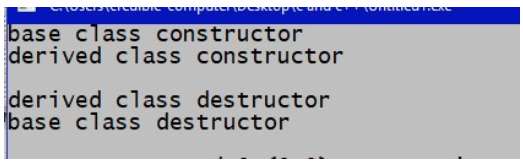


19. Write a program to illustrate the Constructor and Destructor.

Answer:-

```
//eg of constructor and destructor in single inheritance
#include<iostream>
using namespace std;
class base
{
public:
   base()
        {    cout<<"base class constructor"<<endl;
        }
 ~base()
        {    cout<<"base class destructor"<<endl;
        }
};

class derived:public base
{
public:
   derived()
        {     cout<<"derived class constructor"<<endl;
         }
~derived()
        {     cout<<"derived class destructor"<<endl;
        }
};

int main()
{
   derived d;
   cout<<endl;
   return 0;
}
```

Output screen:-



20. Write a program for Hybrid Inheritance.

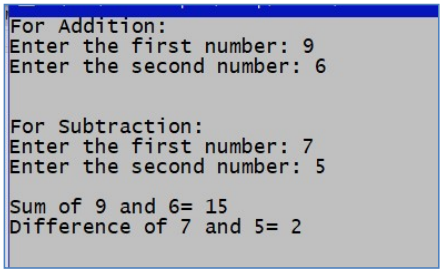
Answer:-

```
#include<iostream>
    using namespace std;
    class arithmetic
    {
       protected:
             int num1, num2;
       public:
            void getdata()
            {
                  cout<<"For Addition:";
                  cout<<"\nEnter the first number: ";
                  cin>>num1;
                  cout<<"Enter the second number: ";
                  cin>>num2;
                  cout<<endl;
            }
    };

    class plus1:public arithmetic
    {
       protected:
                int sum;
       public:
                void add()
                {
                   sum=num1+num2;
                }
    };

    class minus1
    {
       protected:
             int n1,n2,diff;
       public:
            void sub()
            {
                  cout<<"\nFor Subtraction:";
                  cout<<"\nEnter the first number: ";
                  cin>>n1;
                  cout<<"Enter the second number: ";
```

```
            cin>>n2;
            diff=n1-n2;
        }
};
```

```
class result : public plus1, public minus1
{
        public:
                void display()
                {
                cout<<"\nSum of "<<num1<<" and "<<num2<<"= "<<sum;
                cout<<"\nDifference of "<<n1<<" and "<<n2<<"= "<<diff;
                }
};

int  main()
{
   result z;
   z.getdata();
   z.add();
   z.sub();
   z.display();
   return 0;
}
```

Output screen:-



21. Write a program to use this Pointer.

Answer:-

```
#include <iostream>
using namespace std;
class Box {
        public:
                                                    // Constructor definition
                Box(double l = 2.0, double b = 2.0, double h = 2.0, int a=0)
                {       cout <<"Constructor called. for Obj : " << a<<endl;
                        length = l;
                        breadth = b;
                        height = h;
                }
                double Volume()
                {       return length * breadth * height;
                }
                int compare(Box box)
                {       return this->Volume() > box.Volume();
                }
        private:
                double length;                  // Length of a box
                double breadth;                 // Breadth of a box
                double height;                  // Height of a box
        };

int main(void) {
                Box Box1(3.3, 1.2, 1.5, 1);             // Declare box1
                 Box Box2(8.5, 6.0, 2.0, 2);             // Declare box2

                if(Box1.compare(Box2))
                                cout << "Box2 is smaller than Box1" <<endl;
                else
                                cout << "Box2 is equal to or larger than Box1" <<endl;
                cout<<endl;
                cout<<endl;
                return 0;
}
```

Output screen:-

```
Constructor called. for Obj : 1
Constructor called. for Obj : 2
Box2 is equal to or larger than Box1
```

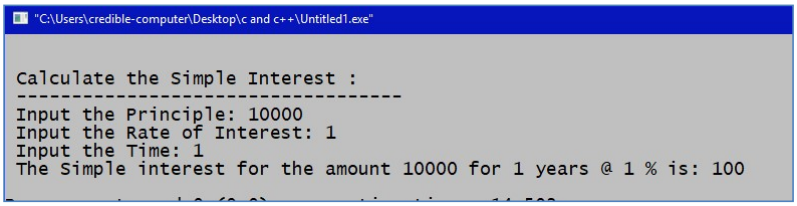
22. Write a program to create a function using Default Argument.

Answer:-

Coding for the program

```
#include<iostream>
using namespace std;
float sim_inter(int , int , int );
float sim_inter(int x, int y=1, int z=1)     // A function with default arguments, it can be called with
{                                            // 2 arguments or 3 arguments or 4 arguments
float sim_inter;
    sim_inter =(x*y*z)/100;
    return sim_inter;
}

int main()                                   /* Driver program to test above function*/
{
   int p,r,t;
   float i;
            cout << "\n\n Calculate the Simple Interest :\n";
            cout << " ----------------------------------\n";
    cout<<" Input the Principle: ";
    cin>>p;
    cout<<" Input the Rate of Interest: ";
    cin>>r;
    cout<<" Input the Time: ";
    cin>>t;
   i = sim_inter(p, r, t);
   cout<<" The Simple interest for the amount "<<p<<" for "<<t<<" years @ "<<r<<" %
is: "<<i;
      cout << endl;
   return 0;
}
```

Output screen:-



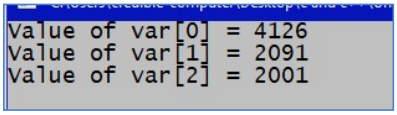
23. Write a program to create a Pointer and Array.

Answer:-

Coding for the program

```
#include <iostream>

using namespace std;
const int MAX = 3;

int main () {
  int  var[MAX] = {4126, 2091, 2001};
  int *ptr[MAX];

  for (int i = 0; i < MAX; i++) {
    ptr[i] = &var[i];                    // assign the address of integer.
  }

  for (int i = 0; i < MAX; i++) {
    cout << "Value of var[" << i << "] = ";
    cout << *ptr[i] << endl;
  }

  return 0;
}
```

Output screen:-



24. Write a program for Parameterized constructor with Dynamic initialization.

Answer:-

```
#include<iostream>
using namespace std;
class Obj
{       int *p, a;
        char *q;
public:
        Obj(int x)                      // Parameterized constructor
        {       a=x;
                p= new int[x];
                q=new char[x];
                cout<<"\nMemory allocated Enter records :- ";
        }
        void input(){
                for(int i=0; i<a;i++)
                {        cin>>*(p+i);
                        cin>>*(q+i);
                  }
           }
        void output(){
                cout<<"Enter records are: "<<endl;
                for(int i=0; i<a;i++)
                {       cout<<*(p+i)<<"\t";
                        cout<<*(q+i)<<endl;
                  }
                delete p;
           }
};
int main()
{       int n;
        cout<<"Enter number of records : ";
        cin>>n;
        Obj obj1(n);                    //dynamic initialization
        obj1.input();
        obj1.output();
        cout<<endl;
   return 0;
}
```

Output screen:-

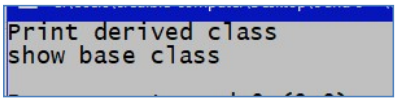
```
Enter number of records : 3

Memory allocated Enter records :-
101     D
102     A
103     S
Enter records are:
101     D
102     A
103     S
```

25. Write a program to create Virtual Function.

Answer:-

```
#include <iostream>
using namespace std;
class base {
public:
        virtual void print()
        {
                cout << "print base class" << endl;
        }
        void show()
        {
                cout << "show base class" << endl;
        }
};

class derived : public base {
public:
        void print()
        {
                cout << "Print derived class" << endl;
        }

        void show()
        {
                cout << "Show derived class" << endl;
        }
};

int main()
{
        base* bptr;
        derived d;
        bptr = &d;
        bptr->print();          // virtual function, binded at runtime
        bptr->show();           // Non-virtual function, binded at compile time
return 0;
}
```

Output screen:-

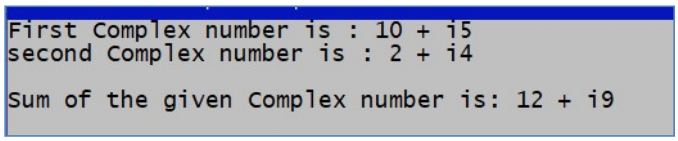


26. Write a program for using a (+) operator overloading.

Answer:-

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

class Complex {
private:
        int real, imag;
public:
        Complex(int r = 0, int i =0) {real = r; imag = i;}

        // This is automatically called when '+' is used with
        // between two Complex objects

        Complex operator + (Complex const &obj) {
                Complex res;
                res.real = real + obj.real;
                res.imag = imag + obj.imag;
                return res;
        }
        void print() { cout << real << " + i" << imag << endl; }
};

int main()
{
        Complex c1(10, 5), c2(2, 4);
        cout<<"First Complex number is : ";
        c1.print();
        cout<<"second Complex number is : ";
        c2.print();
        Complex c3 = c1 + c2;                   // An example call to "operator+"
        cout<< endl<<"Sum of the given Complex number is: ";
        c3.print();
}
```

Output screen:-

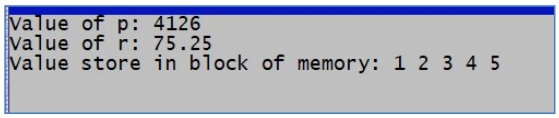


27. Write a program for using new and delete operator.

Answer:-

```
// dynamic allocation and de-allocation of memory using new and delete
#include <iostream>
using namespace std;
int main ()
{
        int* p = NULL;                  // Pointer initialization to null
                                        // using new operator
        p = new(nothrow) int;           // Request memory for the variable
        if (!p)
                cout << "allocation of memory failed\n";
        else
        {       *p = 4126;              // Store value at allocated address
                cout << "Value of p: " << *p << endl;
        }
                                        // using new operator
        float *r = new float(75.25);    // Request block of memory
        cout << "Value of r: " << *r << endl;

        int n = 5;                      // Request block of memory of size n
        int *q = new(nothrow) int[n];
        if (!q)
                cout << "allocation of memory failed\n";
        else
        {       for (int i = 0; i < n; i++)
                        q[i] = i+1;
                cout << "Value store in block of memory: ";
                for (int i = 0; i < n; i++)
                        cout << q[i] << " ";
        }
        delete p;                       // freed the allocated memory
        delete r;
        delete[] q;                     // freed the block of allocated memory
        cout<<endl;
        cout<<endl;
        return 0;
}
```

Output screen:-



28. Write a program for using function overloading.

Answer:-

Coding for the program

```
#include <iostream>
    using namespace std;
                                        // function with 2 parameters
    void display(int var1, double var2)
    {
      cout << "Integer number: " << var1;
      cout << " and double number: " << var2 << endl;
    }
                                        // function with double type single
                                    parameter
    void display(double var)
     {
      cout << "Double number: " << var << endl;
    }

                                        // function with int type single parameter
    void display(int var) {
      cout << "Integer number: " << var << endl;
    }

    int main()
    {

      int a = 5;
      double b = 5.5;
                                        // call function with int type parameter
      display(a);
                                         // call function with double type parameter
      display(b);
                                         // call function with 2 parameters
      display(a, b);
      return 0;
}
```

Output screen:-

```
Integer number: 5
Double number: 5.5
Integer number: 5 and double number: 5.5
```

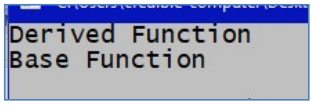
29. Write a program for using function overriding.

Answer:-

Coding for the program

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

    class Base {
            public:
                    void print() {
                            cout << "Base Function" << endl;
                    }
    };

    class Derived : public Base {
            public:
                    void print() {
                             cout << "Derived Function" << endl;
                            // Base::print();    // call overridden function
                    }
    };

    int main() {
       Derived derived1, derived2;
       derived1.print();
       derived2.Base::print();
       return 0;
}
```

Output screen:-



30. Write a program for using pointer and string.

Answer:-

Coding for the program

```
#include<string.h>
#include<iostream>
using namespace std;
int main ()
{
   char str1[]= "Test";              //Array of characters
   char *p1;
   p1= &str1[0];
   cout<<"Display str1 using the Array of characters:-"<<endl;
   int len = strlen(str1);
          for(int i =0; i<len;i++)
              {
               cout<<str1[i] ;
               }
cout<<endl<<endl<<"Display the str1 value by using pointer : "<<endl;
for(int i =0; i<len;i++)
cout<<*(p1+i);
cout<<endl;

string str2 = "Pizza";              // A str2 variable of type string
string* ptr = &str2;                 // A pointer variable ptr, that stores the address of str2

cout<<endl<<"Display the str2 value by using string variable : "<<endl;
                                    // Output the value of str2 (Pizza)
cout << str2 << "\n";

cout<<endl<<"Display the memory address of str2 : "<<endl;
cout << &str2 << "\n";

cout << ptr << "\n";              // Output the memory address of str2 with the pointer
return 0;

}
```

Output screen: -

```
Display str1 using the Array of characters:-
Test

Display the str1 value by using pointer :
Test

Display the str2 value by using string variable :
Pizza

Display the memory address of str2 :
0x61fdf0
0x61fdf0
```