OBJECT ORIENTED PROGRAMMING USING C++ (CIC-211)

Faculty Name: Student Name:

(ASSISTANT PROFESSOR) Roll No:

Semester : 3

Group:

Maharaja Agrasen Institute of Technology, PSP Area, Sector – 22, Rohini, New Delhi -110085

SIGNATURE

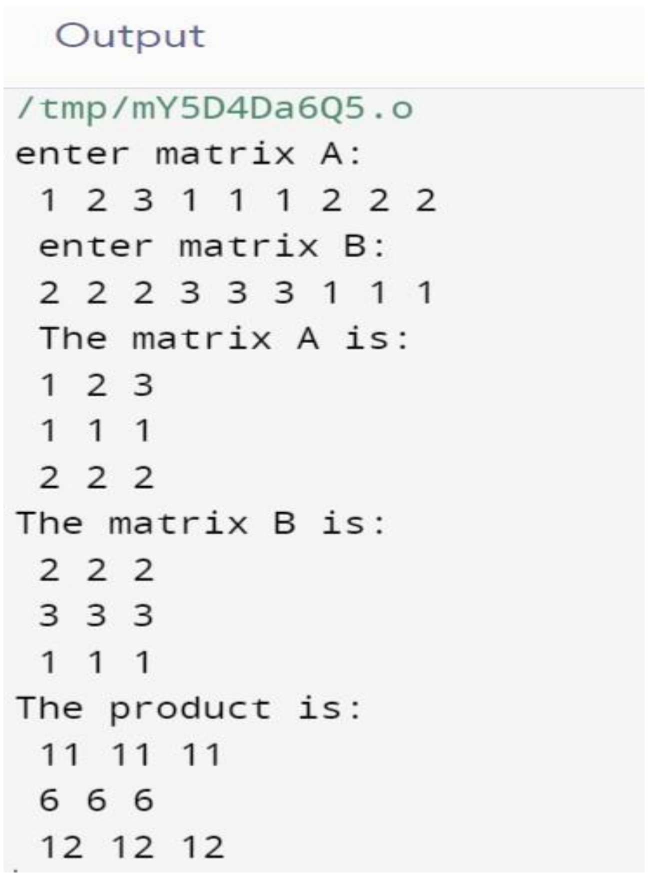
Date: 14/09/2023

Program : 1

Aim: write a program to multiply two matrices using classes.

Code:

```
#include<iostream>
using namespace std;
class matrix{
int a[3][3],b[3][3],c[3][3],i,j,k;
public:
void getdata(){
 cout<<" enter matrix A:\n ";
 for(i=0;i<3;i++){
 for(j=0;j<3;j++){
 cin>>a[i][j];}}
 cout<<"enter matrix B:\n ";
 for(i=0;i<3;i++){
 for(j=0;j<3;j++){
 cin>>b[i][j];}}}
void multiply(){
 for(i=0;i<3;i++){
 for(j=0;j<3;j++){
 c[i][j]=0;
 for(k=0;k<3;k++)
 c[i][j]=c[i][j]+(a[i][k]*b[k][j]);} }}
void display(){
cout<<"\nThe matrix A is:\n";
 for(i=0;i<3;i++){
 for(j=0;j<3;j++){
 cout<<" "<<a[i][j];}
 cout<<endl;}
 cout<<"The matrix B is:\n";
 for(i=0;i<3;i++){
 for(j=0;j<3;j++){
 cout<<" "<<b[i][j];}
 cout<<endl;}
 cout<<"The product is:\n";
 for(i=0;i<3;i++){
 for(j=0;j<3;j++){
 cout<<" "<<c[i][j];}
cout<<endl;}}};
int main(){
matrix x;
x.getdata();
x.multiply();
x.display();
return 0;}
```



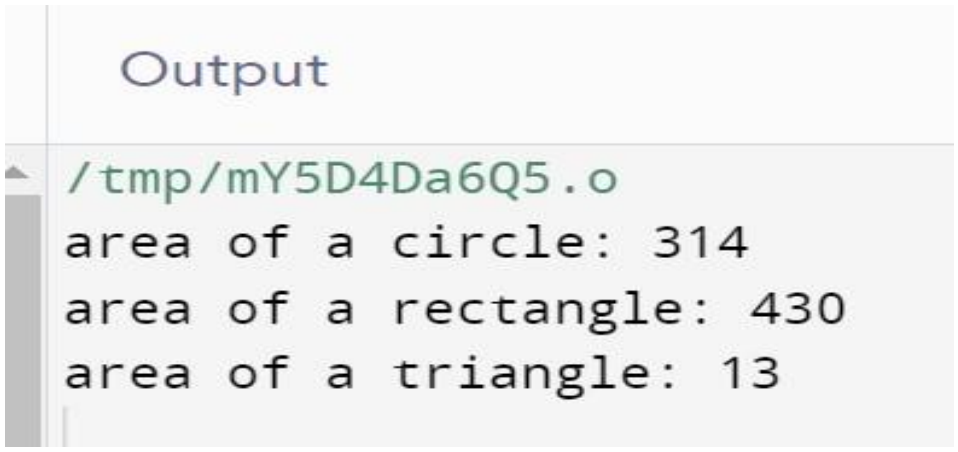
```
Output

/tmp/mY5D4Da6Q5.o
enter matrix A:
 1 2 3 1 1 1 2 2 2
 enter matrix B:
 2 2 2 3 3 3 1 1 1
 The matrix A is:
 1 2 3
 1 1 1
 2 2 2
The matrix B is:
 2 2 2
 3 3 3
 1 1 1
The product is:
 11 11 11
 6 6 6
 12 12 12
```

Program : 2

Aim: Write a program to calculate the area of different shapes using function overloading.

Code:

```
#include
<iostream>
using
namespace   std;
class shapes{
 public:
 int area(float ,int ); //triangle
 long long area(long long ,long long );
 //rectangledouble area(float ); // circle};
int
 main()
 {
 shapes
 a;
 cout<<"area of a circle: "<<a.area(10)<<endl;
 cout<<"area of a rectangle:
 "<<a.area(10.9,79)<<endl;cout<<"area of a
 triangle: "<<a.area(8.9,3)<<endl; return 0;}
double shapes::
 area(float r){return
 3.14*r*r;}
int shapes:: area(float
 b,int h){return 0.5*b*h;}
long long shapes:: area(long long a,long
 long b){return a*b;}
```


Output

```
/tmp/mY5D4Da6Q5.o
area of a circle: 314
area of a rectangle: 430
area of a triangle: 13
```

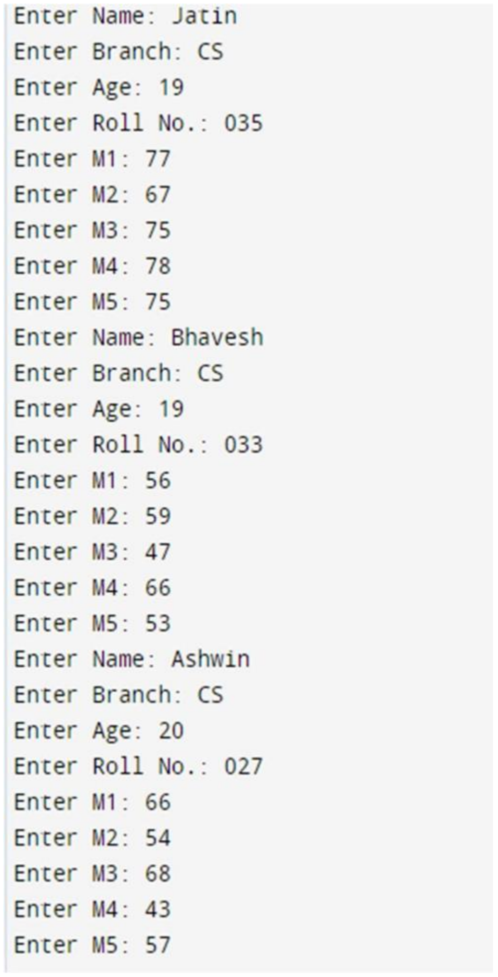
Program: 3

AIM: Write a program to create a class ‘student’ which has data members name, branch, roll no., marks of five subjects and display the name of the student & having percentage greater than 70%.

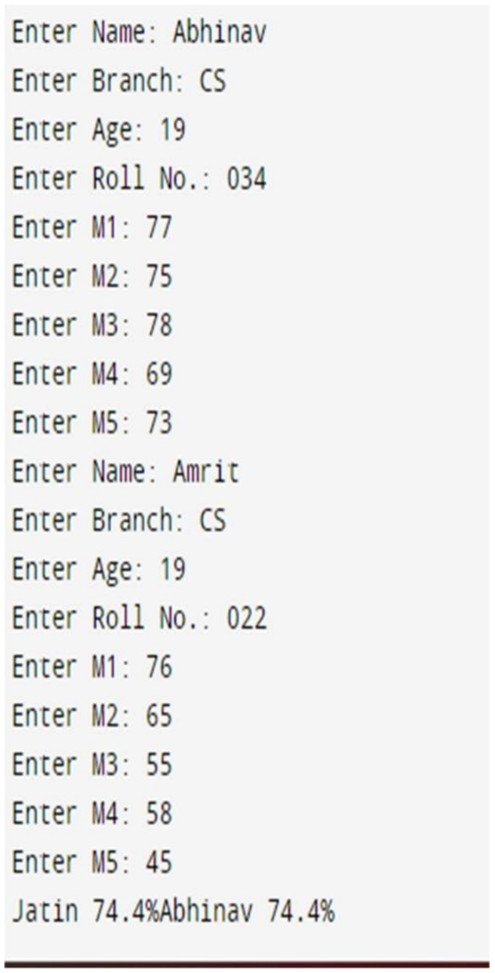
CODE:

```
#include<iostream>
using namespace std;
class student
{
        string name,branch;
        int age,rollno;
        float m1,m2,m3,m4,m5;
        public:
                void getdata(void)
                {
                        int i;
                        cout<<"Enter Name: ";
                        cin>>name;
                        cout<<"Enter Branch: ";
                        cin>>branch;
                        cout<<"Enter Age: ";
                        cin>>age;
                        cout<<"Enter Roll No.: ";
                        cin>>rollno;
                }
                void average(void)
                {
                        float av,avpr;
                        cout<<"Enter M1: ";
                        cin>>m1;
                        cout<<"Enter M2: ";
                        cin>>m2;
                        cout<<"Enter M3: ";
                        cin>>m3;
                        cout<<"Enter M4: ";
                        cin>>m4;
                        cout<<"Enter M5: ";
                        cin>>m5;
                        av=(m1+m2+m3+m4+m5)/5;
                        avpr=av*100;
                }
                void display(void)
                {
                        int i;
                        float av;
                        av=(m1+m2+m3+m4+m5)/5;
                        if(av>70)
                        {
                          cout<<name;
                          cout<<" "<<av<<"%";
                        }
                }
};
int main()
{

        student s[5];
        int i;
        for(i=0;i<5;i++)
        {
                s[i].getdata();
                s[i].average();
        }
        for(i=0;i<5;i++)
        {
                s[i].display();
        }

}
```

OUTPUT:



```
Enter Name: Jatin
Enter Branch: CS
Enter Age: 19
Enter Roll No.: 035
Enter M1: 77
Enter M2: 67
Enter M3: 75
Enter M4: 78
Enter M5: 75
Enter Name: Bhavesh
Enter Branch: CS
Enter Age: 19
Enter Roll No.: 033
Enter M1: 56
Enter M2: 59
Enter M3: 47
Enter M4: 66
Enter M5: 53
Enter Name: Ashwin
Enter Branch: CS
Enter Age: 20
Enter Roll No.: 027
Enter M1: 66
Enter M2: 54
Enter M3: 68
Enter M4: 43
Enter M5: 57
```



```
Enter Name: Abhinav
Enter Branch: CS
Enter Age: 19
Enter Roll No.: 034
Enter M1: 77
Enter M2: 75
Enter M3: 78
Enter M4: 69
Enter M5: 73
Enter Name: Amrit
Enter Branch: CS
Enter Age: 19
Enter Roll No.: 022
Enter M1: 76
Enter M2: 65
Enter M3: 55
Enter M4: 58
Enter M5: 45
Jatin 74.4%Abhinav 74.4%
```

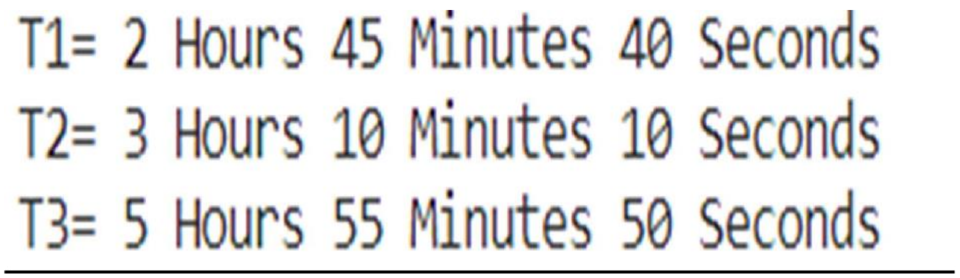
Program: 4

AIM: Write a program to create a class 'Time' with data members hours, minutes and seconds and add two time objects by passing the objects to a function and display the result.

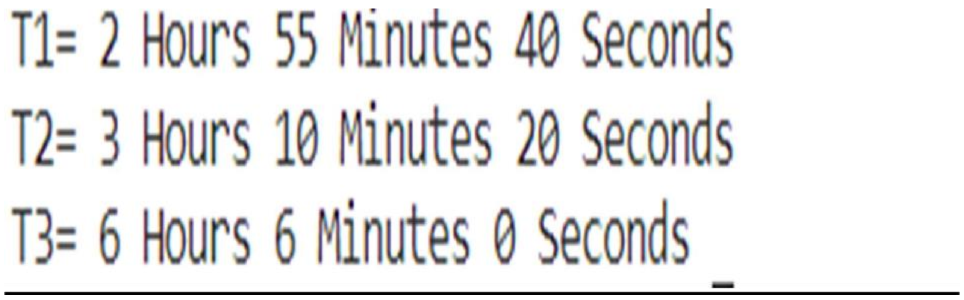
CODE:

```
#include<iostream>
using namespace std;
class Time
{
   int h,m,s;
   public:
   void gettime(int hour,int min,int sec)
   {
      h=hour;
      m=min;
      s=sec;
   }
   void puttime()
   {
      cout<<h<<" Hours ";
      cout<<m<<" Minutes ";
      cout<<s<<" Seconds ";
   }
   void sum(Time t1,Time t2)
   {

      s=t1.s+t2.s;
      if(s>59)
      {
         m=s/60;
         s=s%60;
         m=t1.m+t2.m+m;
      }
      else
      m=t1.m+t2.m;
      if(m>59)
      {
         h=m/60;
         m=m%60;
         h=t1.h+t2.h+h;
      }
      else
      h=t1.h+t2.h;
   }
};
int main()
{
   Time t1,t2,t3;
   t1.gettime(2,55,40);
   t2.gettime(3,10,20);
   t3.sum(t1,t2);
   cout<<"T1= ";t1.puttime();
   cout<<"\nT2= ";t2.puttime();
   cout<<"\nT3=  ";t3.puttime();
}
```

OUTPUT:



```
T1= 2 Hours 45 Minutes 40 Seconds
T2= 3 Hours 10 Minutes 10 Seconds
T3= 5 Hours 55 Minutes 50 Seconds
```



```
T1= 2 Hours 55 Minutes 40 Seconds
T2= 3 Hours 10 Minutes 20 Seconds
T3= 6 Hours 6 Minutes 0 Seconds _
```

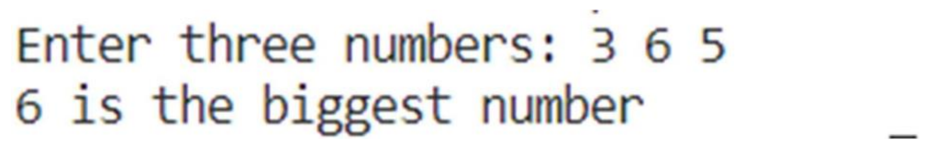
Program : 5

AIM: Write a program to find biggest of three numbers using a friend function.

CODE:

```
#include<iostream>
using namespace std;
class compare
{
   int a,b,c;
   public:
      void getdata()
      {
         cout<<"Enter three numbers: ";
         cin>>a>>b>>c;
      }
      friend void big(compare );
};
void big(compare x)
{
   int y;
   y=(x.a>x.b)?(x.a>x.c?x.a:x.c):(x.b>x.c?x.b:x.c);
   cout<<y<<" is the biggest number";
}
int main()
{
   compare x;
   x.getdata();
   big(x);
}
```

OUTPUT:



```
Enter three numbers: 3 6 5
6 is the biggest number
```

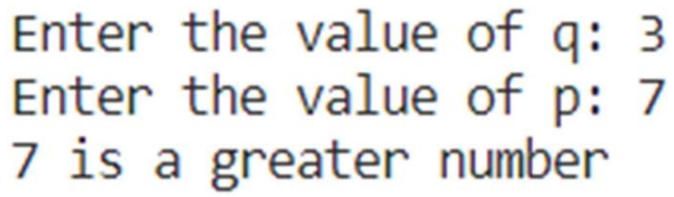
Program : 6

AIM: Write a program to find bigger of two numbers in two different classes using friend function.

CODE:

```
#include<iostream>
using namespace std;
class abc;
class xyz
{
   int q;
   public:
      void indata()
      {
         cout<<"Enter the value of p: ";
         cin>>q;
      }
      friend void big(abc ,xyz );
};
class abc
{
   int p;
   public:
      void getdata()
      {
         cout<<"Enter the value of q: ";
         cin>>p;
      }
      friend void big(abc ,xyz );
};
void big(abc a,xyz x)
{
   int y;
   y=a.p>x.q?a.p:x.q;
   cout<<y<<" is a greater number";
}
int main()
{
   abc a;
   xyz x;
   a.getdata();
   x.indata();
   big(a,x);
}
```

OUTPUT:



```
Enter the value of q: 3
Enter the value of p: 7
7 is a greater number
```

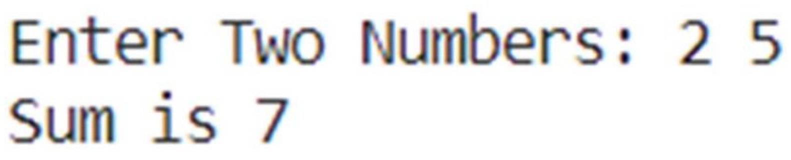
Program : 7

AIM: Write a program to demonstrate the use of a friend function using inline assignment.

CODE:

```
#include<iostream>
using namespace std;
class num
{
   int a,b;
   public:
   inline void getdata(void)
   {
      cout<<"Enter Two Numbers: ";
      cin>>a>>b;
   }
   friend void display(num );
};
void display(num x)
{
   cout<<"Sum is "<<(x.a+x.b);
}
int main()
{
   num n;
   n.getdata();
   display(n);
}
```

OUTPUT:



```
Enter Two Numbers: 2 5
Sum is 7
```

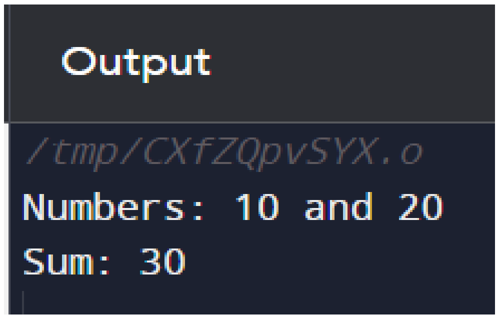
Program : 8

AIM: Write a program to display two numbers declared in a class and find their sum using a friend function.

CODE:

```
#include<iostream>
using namespace std;
class num
{
    int a,b;
    public:
        void getdata()
        {
            cout<<"Enter two numbers: ";
            cin>>a>>b;
        }
        friend void sum(num );
};
void sum(num x)
{
    int s;
    s=x.a+x.b;
    cout<<s<<" is the sum";
}
int main()
{
    num x;
    x.getdata();
    sum(x);
}
```

OUTPUT:



```
/tmp/CXfZQpvSYX.o
Numbers: 10 and 20
Sum: 30
```

Program: 9

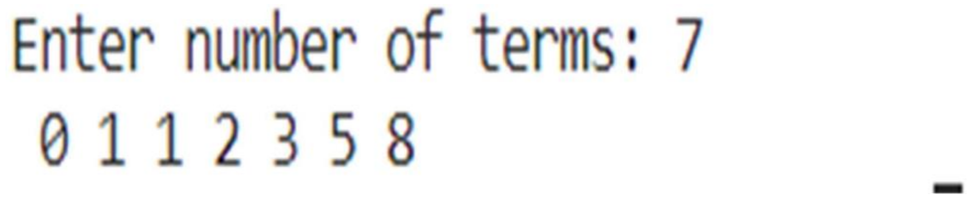
AIM: Write a program to generate Fibonacci series using Copy Constructor.

CODE:

```
#include<iostream>
using namespace std;
class fib
{
   int a,b,n;
   public:
   fib(int x)
   {
      a=0;
      b=1;
      n=x;
   }
   fib(fib &f)
   {
      a=f.a;
      b=f.b;
      n=f.n;
   }
   void fibno()
   {
      for(int i=0,s=0;i<n;i++)
      {
         if(i<=1)
         s=i;
         else
         {
            s=a+b;
            a=b;
            b=s;
         }
         cout<<" "<<s;
      }
   }

};
int main()
{
   int n;
   cout<<"Enter number of terms: ";
   cin>>n;
```

```
    fib a(n);
    fib b(a);
    b.fibno();
}
```

OUTPUT:



```
Enter number of terms: 7
 0 1 1 2 3 5 8
```

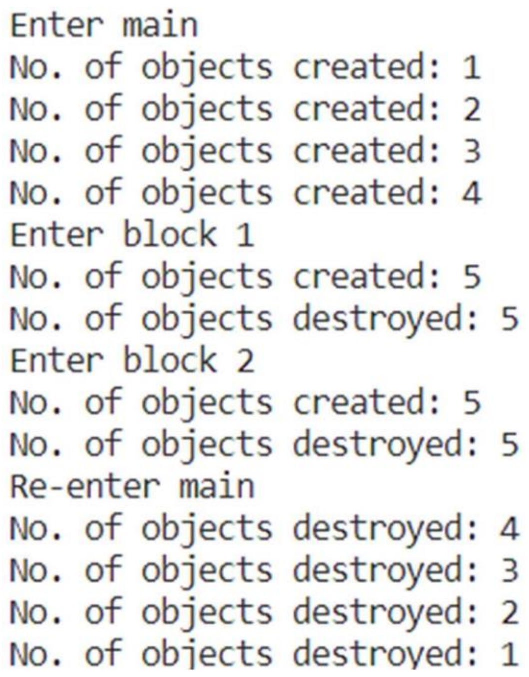
Program: 10

AIM: Create a class which keep track of number of it's instances. Use static data member, constructors, and destructors to maintain updated information about active objects.

CODE:

```
#include<iostream>
using namespace std;
static int count=0;
class alpha
{
   public:
   alpha()
   {
      count++;
      cout<<"\nNo. of objects created: "<<count;
   }
   ~alpha()
   {
      cout<<"\nNo. of objects destroyed: "<<count;
      count--;
   }
};
int main()
{
   cout<<"Enter main";
   alpha a,b,c,d;
   {
      cout<<"\nEnter block 1";
      alpha e;
   }
   {
      cout<<"\nEnter block 2";
      alpha f;
   }
   cout<<"\nRe-enter main";
}
```

OUTPUT:



```
Enter main
No. of objects created: 1
No. of objects created: 2
No. of objects created: 3
No. of objects created: 4
Enter block 1
No. of objects created: 5
No. of objects destroyed: 5
Enter block 2
No. of objects created: 5
No. of objects destroyed: 5
Re-enter main
No. of objects destroyed: 4
No. of objects destroyed: 3
No. of objects destroyed: 2
No. of objects destroyed: 1
```

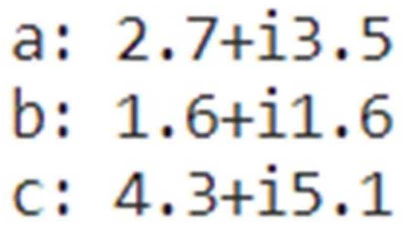
Program: 11

AIM: Write a program to perform addition of two complex numbers using constructor overloading, The first constructor which takes no argument is used to create objects which are not initialized, second which takes one argument is used to initialize real and imag parts to equal values and third which takes two arguments is used to initialize real and imag parts to two different values.

CODE:

```
#include<iostream>
using namespace std;
class complex
{
   float x,y;
   public:
   complex(void)
   {
      x=0;
      y=0;
   }
   complex(float a)
   {
      x=y=a;
   }
   complex(float r,float i)
   {
      x=r;
      y=i;
   }
   friend complex sum(complex ,complex);
   friend void show(complex);
};
complex sum(complex c1,complex c2)
{
   complex c3;
   c3.x=c1.x+c2.x;
   c3.y=c1.y+c2.y;
   return c3;
}
void show(complex c)
{
   cout<<c.x<<"+i"<<c.y;
}
int main()
{
    complex a(2.7,3.5);
    complex b(1.6);
    complex c;
    c=sum(a,b);
    cout<<"a: ";
    show(a);
    cout<<"\nb: ";
    show(b);
    cout<<"\nc: ";
    show(c);
}
```

OUTPUT:



```
a: 2.7+i3.5
b: 1.6+i1.6
c: 4.3+i5.1
```

Program 12 :

AIM: Implement a class string containing the following functions:

Overload + operator to carry out concatenation of strings.

Overload = operator to carry out string copy.

Overload <= operator to carry out the comparison of strings.

Function to display the length of a string.

Function tolower() to convert upper case letters to lower case.

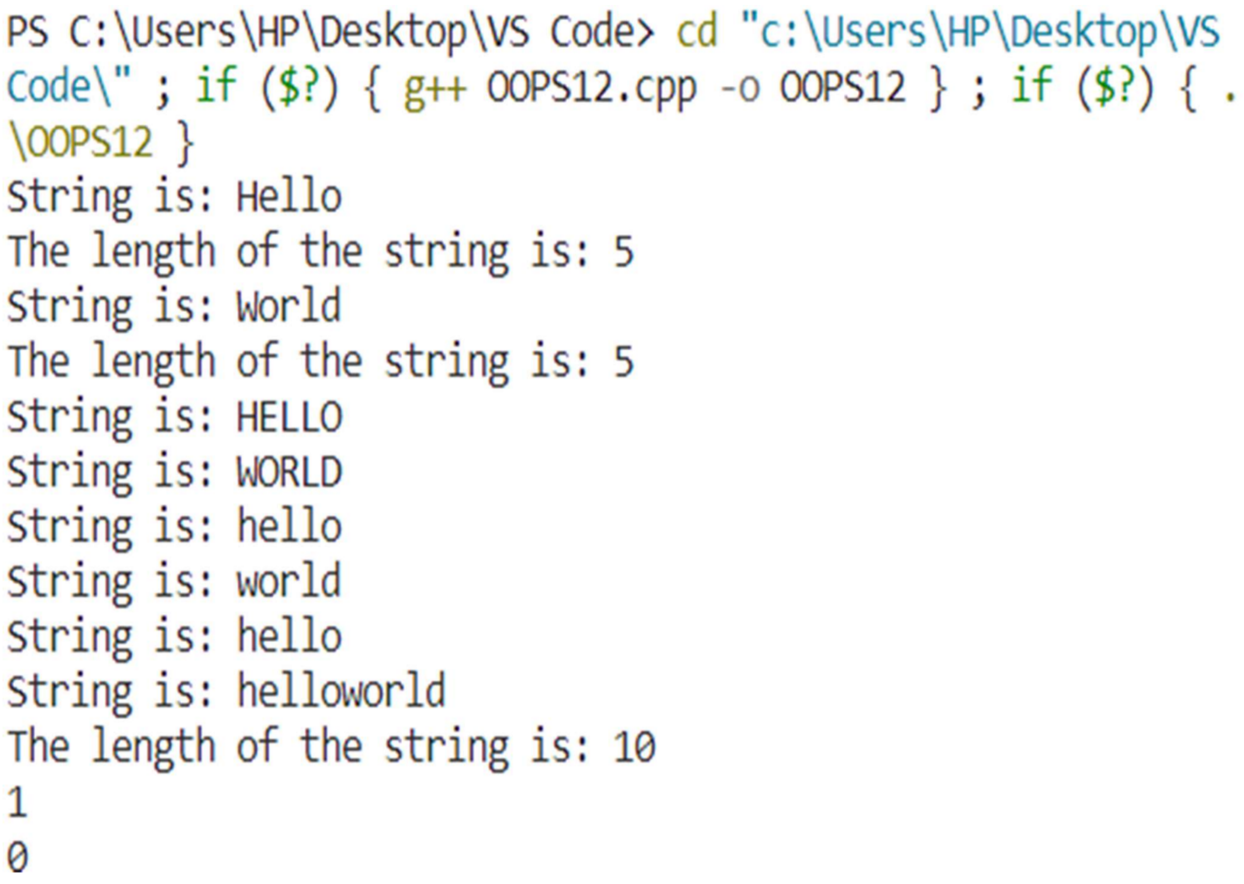
Function toupper() to convert lower case letters to upper case.

CODE:

```
  #include<iostream>
  #include<string>
  using namespace std;
  class str
  {
string s;public:
str()
{
        s="";
}
str(string x)
{
        s=x;
}
str operator+(str obj)
{
        s=s+obj.s;
        return s;
}
str operator=(str obj)
{
        s=obj.s;
        return s;
}
int operator<=(str obj)
{
        if(s.length()<obj.s.length())
        return 1;
        else
        if(s.length()>obj.s.length())
        return 0;
        else
        for(int i=0;i<s.length();i++)
        {
           if(s[i]>obj.s[i])
           {
              return 1;
              break;
           }
           else
           return 0;
        }
        return -1;
}
void displaylen()
{
        cout<<"The length of the string is: "<<s.length()<<endl;;
}
void tolower()
{
        for(int i=0;i<=s.length();i++)
        {
           if(s[i]>=65 && s[i]<=90)
           s[i]+=32;
        }
}
void toupper()
{
        for(int i=0;i<=s.length();i++)
        {
           if(s[i]>=97 && s[i]<=122)
           s[i]-=32;
        }
}
void print()
{
        cout<<"String is: "<<s<<endl;
}
  };
  int main()
  {
str a("Hello");
str  b("World"); a.print();
a.displaylen();
b.print(); b.displaylen();a.toupper();
b.toupper();
```

```
a.print();
b.print();
a.tolower();
b.tolower();
a.print();
b.print();
str c=a;
c.print();
str d=a+b;
d.print(); d.displaylen();
int res=b<=a;
cout<<res<<endl;
res=d<=a;
cout<<res<<endl;
  }
```

OUTPUT:



```
PS C:\Users\HP\Desktop\VS Code> cd "c:\Users\HP\Desktop\VS Code\" ; if ($?) { g++ OOPS12.cpp -o OOPS12 } ; if ($?) { .\OOPS12 }
String is: Hello
The length of the string is: 5
String is: World
The length of the string is: 5
String is: HELLO
String is: WORLD
String is: hello
String is: world
String is: hello
String is: helloworld
The length of the string is: 10
1
0
```

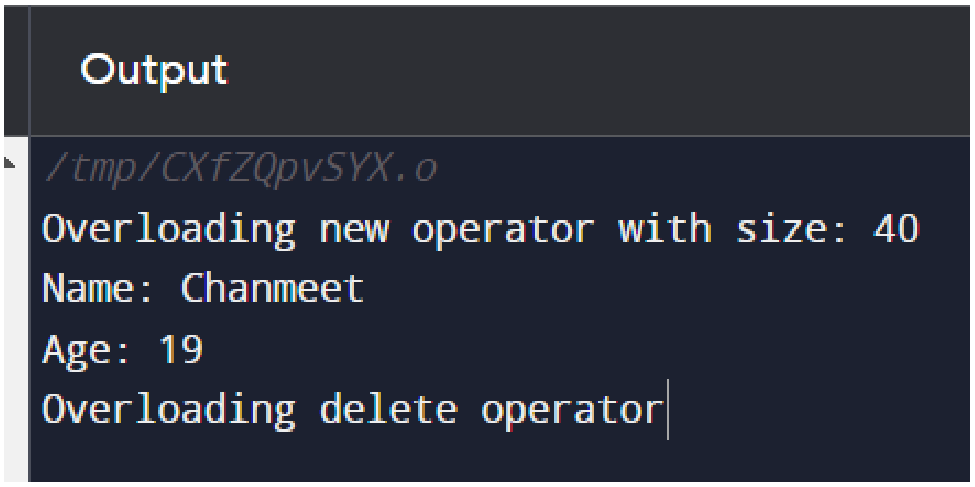
Program:13

AIM: Write a program to overload new and delete operator.

CODE:

```
  #include<iostream>
  using namespace std;
  class student
  {
string name;int
age; public:
student()
{}
student(string x,int n)
{
        name=x;
        age=n;
}
void display()
{
        cout<<"Name: "<<name<<endl;
        cout<<"Age: "<<age<<endl;
}
void *operator new(size_t size)
{
        cout<<"Overloading new operator with size: "<<size<<endl;
        void *p=::operator new(size);
        return p;
}
void operator delete(void *p)
{
        cout<<"Overloading delete operator";
        free(p);
}
  };
  int main()
  {
student *p=new
student("Chanmeet",19);p->display();
delete p;
  }
```

OUTPUT:



Output

```
/tmp/CXfZQpvSYX.o
Overloading new operator with size: 40
Name: Chanmeet
Age: 19
Overloading delete operator
```

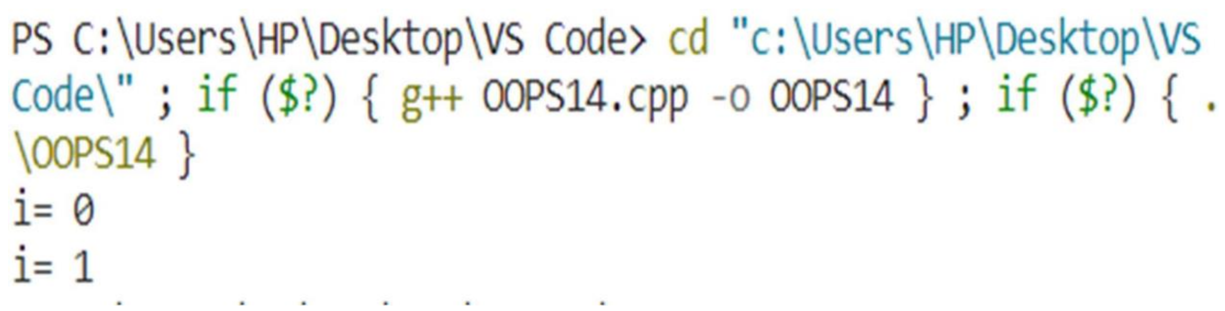
Program: 14

AIM: Write a program to overload unary increment (++) operator.

CODE:

```
  #include<iostream
  > using namespace
  std;class inc
  {
int i;
public:
inc()
{
        i=0;
}
int operator ++()
{
        ++i;
        return i;
}
void display()
{
        cout<<"i= "<<i<<endl;
}
  };
  int main()
  {
inc a;
a.display();
++a;
a.display();
  }
```

OUTPUT:



```
PS C:\Users\HP\Desktop\VS Code> cd "c:\Users\HP\Desktop\VS
Code\" ; if ($?) { g++ OOPS14.cpp -o OOPS14 } ; if ($?) { .
\OOPS14 }
i= 0
i= 1
```

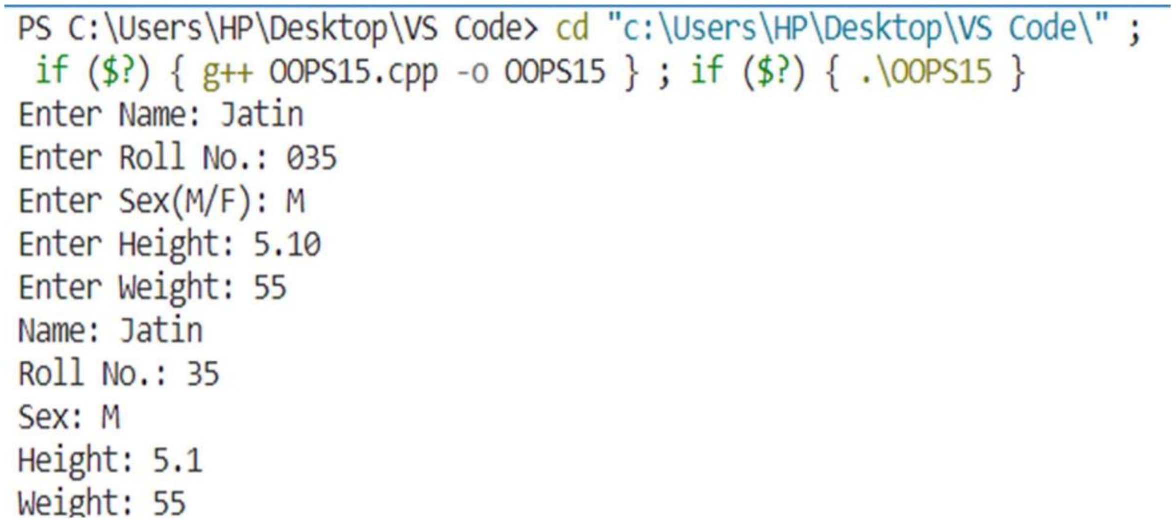
Program: 15

AIM: Write a program to create a base class basic_info with data members name, roll no, sex and two member functions getdata and display. Derive a class physical_fit from basic_info which has data members height and weight and member functions getdata and display. Display all the information using object of derived class.

CODE:

```
  #include<iostream>
  using namespace std;
  class basic_info
  {
string name; int
rollno; char sex;
public:
void getdata(void)
     {
        cout<<"Enter Name: ";
        cin>>name;
        cout<<"Enter Roll No.: ";
        cin>>rollno;
        cout<<"Enter Sex(M/F): ";
        cin>>sex;
     }
void display(void)
     {
        cout<<"Name: "<<name<<endl;
        cout<<"Roll No.: "<<rollno<<endl;
        cout<<"Sex: "<<sex<<endl;
     }
  };
  class physical_fit : public basic_info
  {
float height,weight;
public:
void getdata(void)
     {
        cout<<"Enter Height: ";
        cin>>height;
        cout<<"Enter Weight: ";
        cin>>weight;
     }
void display(void)
     {
          cout<<"Height: "<<height<<endl;
          cout<<"Weight:  "<<weight<<endl;
      }
  };
  int main()
  {
physical_fit a;
a.basic_info::getdata();
a.getdata();
a.basic_info::display();
a.display();
  }
```

OUTPUT:



```
PS C:\Users\HP\Desktop\VS Code> cd "c:\Users\HP\Desktop\VS Code\" ;
 if ($?) { g++ OOPS15.cpp -o OOPS15 } ; if ($?) { .\OOPS15 }
Enter Name: Jatin
Enter Roll No.: 035
Enter Sex(M/F): M
Enter Height: 5.10
Enter Weight: 55
Name: Jatin
Roll No.: 35
Sex: M
Height: 5.1
Weight: 55
```

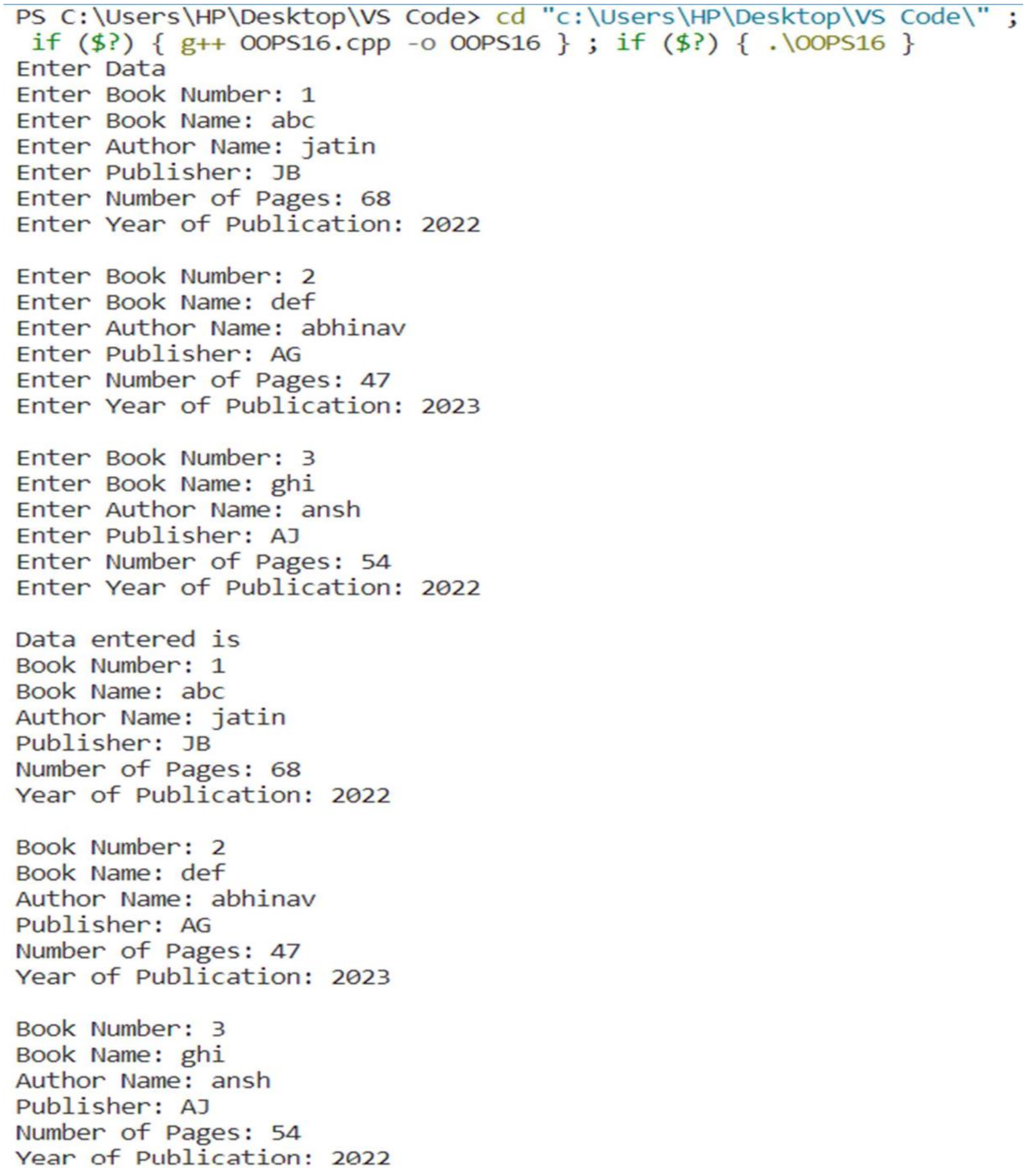
Program: 16

AIM: write a program to create a class first with data members book no, book name and member function getdata and putdata. Create a class second with data members author name, publisher and member functions getdata and showdata. Derive a class third from first and second with data member no of pages and year of publication. Display all this information using array of objects.

CODE:

```
  #include<iostream>
  using namespace std;
  class first
  {
int bookno;   string
bookname; public:
void getdata(void)
      {
         cout<<"Enter Book Number: ";
         cin>>bookno;
         cout<<"Enter Book Name: ";
         cin>>bookname;
      }
void putdata(void)
      {
         cout<<"Book Number: "<<bookno<<endl;
         cout<<"Book Name: "<<bookname<<endl;
      }
  };
  class second
  {
string authorname,publisher; public:
void getdata(void)
      {
         cout<<"Enter Author Name: ";
         cin>>authorname;
         cout<<"Enter Publisher: ";
         cin>>publisher;
      }
void showdata(void)
      {
         cout<<"Author Name: "<<authorname<<endl;
         cout<<"Publisher: "<<publisher<<endl;
      }
  };
  class third : public first,public second
  {
int noofpages,yearofpublication; public:
void data(void)
     {
        cout<<"Enter Number of Pages: ";
        cin>>noofpages;
        cout<<"Enter Year of Publication: ";
        cin>>yearofpublication;
     }
void display()
     {
        cout<<"Number of Pages: "<<noofpages<<endl;
        cout<<"Year of Publication: "<<yearofpublication<<endl;
     }
  };
  int main()
  {
third a[3];
cout<<"Enter Data"<<endl;
for(int i=0;i<3;i++)
     {
        a[i].first::getdata();
        a[i].second::getdata();
        a[i].data();
        cout<<endl;
     }
cout<<"Data entered is"<<endl;
for(int i=0;i<3;i++)
     {
        a[i].putdata();
        a[i].showdata();
        a[i].display();
        cout<<endl;
     }
  }
```

OUTPUT:



```
PS C:\Users\HP\Desktop\VS Code> cd "c:\Users\HP\Desktop\VS Code\" ;
 if ($?) { g++ OOPS16.cpp -o OOPS16 } ; if ($?) { .\OOPS16 }
Enter Data
Enter Book Number: 1
Enter Book Name: abc
Enter Author Name: jatin
Enter Publisher: JB
Enter Number of Pages: 68
Enter Year of Publication: 2022

Enter Book Number: 2
Enter Book Name: def
Enter Author Name: abhinav
Enter Publisher: AG
Enter Number of Pages: 47
Enter Year of Publication: 2023

Enter Book Number: 3
Enter Book Name: ghi
Enter Author Name: ansh
Enter Publisher: AJ
Enter Number of Pages: 54
Enter Year of Publication: 2022

Data entered is
Book Number: 1
Book Name: abc
Author Name: jatin
Publisher: JB
Number of Pages: 68
Year of Publication: 2022

Book Number: 2
Book Name: def
Author Name: abhinav
Publisher: AG
Number of Pages: 47
Year of Publication: 2023

Book Number: 3
Book Name: ghi
Author Name: ansh
Publisher: AJ
Number of Pages: 54
Year of Publication: 2022
```

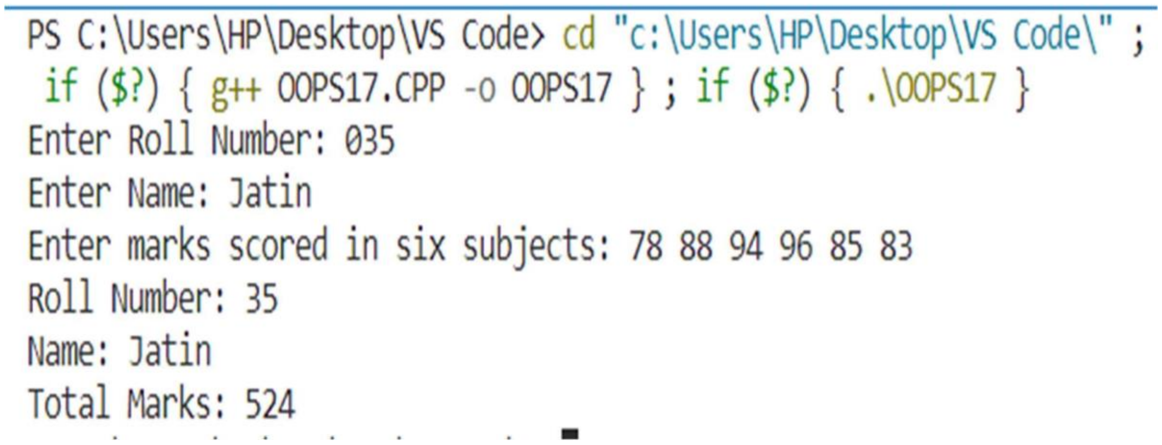
Program: 17

AIM: Write a program to design three classes STUDENT, EXAM and RESULT. The STUDENT class has data members such as roll no, name. Create a class EXAM by inheriting the STUDENT class. The EXAM class adds data members representing the marks scored in six subjects. Derive the RESULT class from the EXAM class and has its own data members such as total marks. Write a program to model this relationship.

CODE:

```
  #include<iostream>
  using namespace std;
  class STUDENT
  {
int rollno; string
name; public:
void getdata(void)
     {
        cout<<"Enter Roll Number: ";
        cin>>rollno;
        cout<<"Enter Name: ";
        cin>>name;
     }
void display(void)
     {
        cout<<"Roll Number: "<<rollno<<endl;
        cout<<"Name: "<<name<<endl;
     }
  };
  class EXAM : public STUDENT
  {
int m1,m2,m3,m4,m5,m6;
public:
void getmarks(void)
     {
        cout<<"Enter marks scored in six subjects: ";
        cin>>m1>>m2>>m3>>m4>>m5>>m6;
     }
int sum()
     {
        int sum=m1+m2+m3+m4+m5+m6;
        return sum;
     }
  };
  class RESULT : public EXAM
  {
int totalmarks;
public:
void total(void)
      {
          totalmarks=sum();
          display();
          cout<<"Total Marks: "<<totalmarks;
      }
  };
  int main()
  {
RESULT s;
s.getdata();
s.getmarks();
s.total();
  }
```

OUTPUT:



```
PS C:\Users\HP\Desktop\VS Code> cd "c:\Users\HP\Desktop\VS Code\" ;
 if ($?) { g++ OOPS17.CPP -o OOPS17 } ; if ($?) { .\OOPS17 }
Enter Roll Number: 035
Enter Name: Jatin
Enter marks scored in six subjects: 78 88 94 96 85 83
Roll Number: 35
Name: Jatin
Total Marks: 524
```

Program: 18

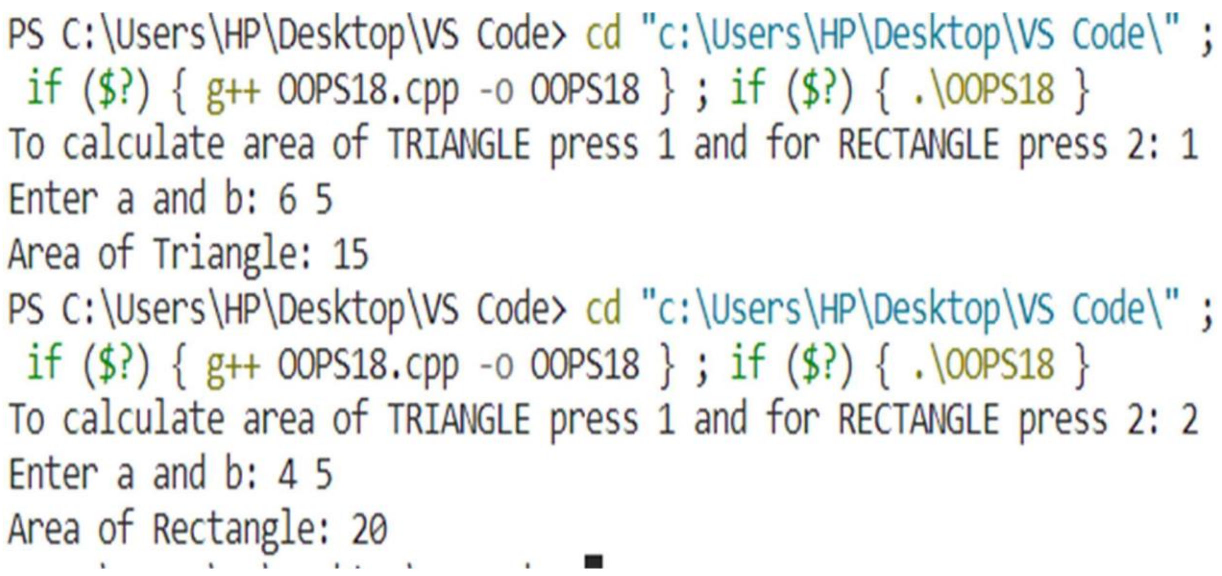
AIM: Write a program to create a base class called SHAPE. Use this class to store two double type values. Derive two specific classes called TRIANGLE and RECTANGLE from the base class. Add to the base class, a member function getdata to initialize base class data members and another member function display to compute and display the area of figures. Make display a virtual function and redefine this function in the derived class to suit their requirements. Using these three classes design a program that will accept driven of a TRIANGLE or RECTANGLE interactively and display the area.

CODE:

```
  #include<iostream>
  using namespace std;
  class SHAPE
  {
protected: double
a,b; public:
void getdata(void)
     {
        cout<<"Enter a and b: ";
        cin>>a>>b;
     }
virtual void display(void)
     {
        cout<<"Area of Triangle: "<<(0.5*a*b);
        cout<<"Area of Rectangle: "<<(a*b);
     }
  };
  class TRIANGLE : public SHAPE
  {
public:
void display(void)
     {
        cout<<"Area of Triangle: "<<(0.5*a*b);
     }
  };
  class RECTANGLE : public SHAPE
  {
public:
void display(void)
     {
        cout<<"Area of Rectangle: "<<(a*b);
     }
  };
  int main()
  {
```

```
int n;
cout<<"To calculate area of TRIANGLE press 1 and for RECTANGLE press 2: "; cin>>n;
if(n==1)
     {
        TRIANGLE t;
        t.getdata();
        t.display();
     }
if(n==2)
     {
        RECTANGLE r;
        r.getdata();
        r.display();
     }
  }
```

OUTPUT:



```
PS C:\Users\HP\Desktop\VS Code> cd "c:\Users\HP\Desktop\VS Code\" ;
 if ($?) { g++ OOPS18.cpp -o OOPS18 } ; if ($?) { .\OOPS18 }
To calculate area of TRIANGLE press 1 and for RECTANGLE press 2: 1
Enter a and b: 6 5
Area of Triangle: 15
PS C:\Users\HP\Desktop\VS Code> cd "c:\Users\HP\Desktop\VS Code\" ;
 if ($?) { g++ OOPS18.cpp -o OOPS18 } ; if ($?) { .\OOPS18 }
To calculate area of TRIANGLE press 1 and for RECTANGLE press 2: 2
Enter a and b: 4 5
Area of Rectangle: 20
```

Program: 19

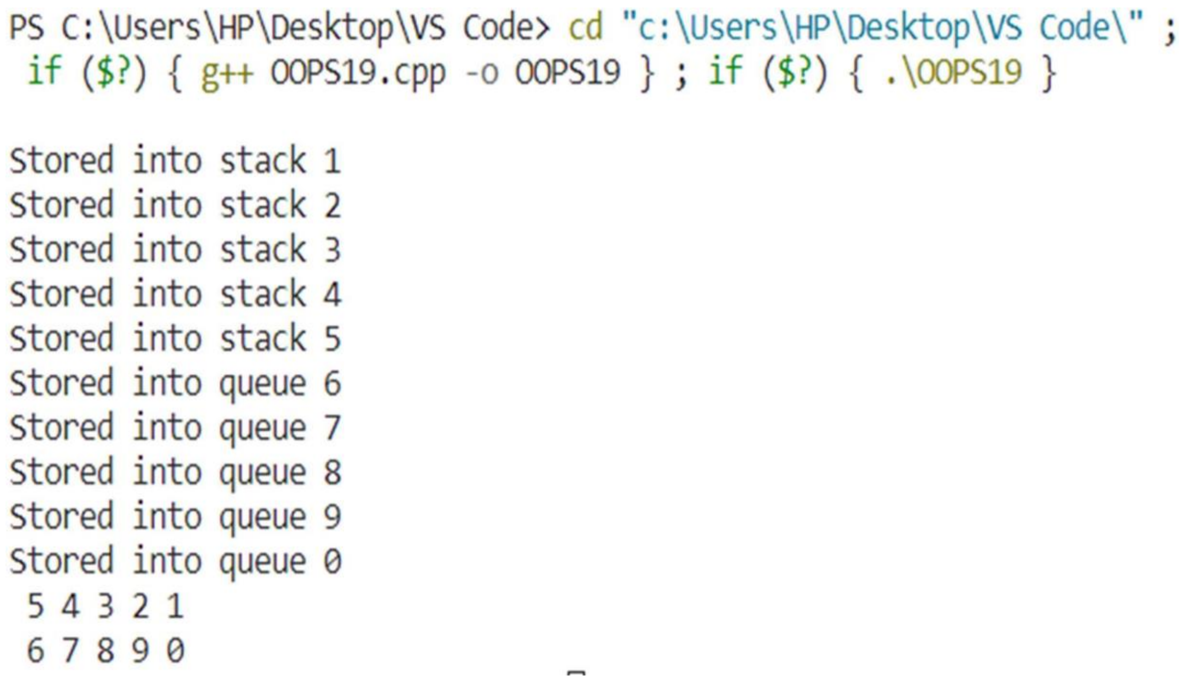
AIM: Write a program to create a class called LIST with two pure virtual function store() and retrieve(). To store a value call store() and to retrieve call retrieve() function. Derive two classes stack and queue from it and override store and retrieve.

CODE:

```
  #include<iostream>
  using namespace std;
  class LIST
  {
public:
virtual void store(int x=0)
     {
     }
virtual void retrieve(void)
     {
     }
  };
  class stack : public LIST
  {
int a[5],top=-1; public:
void store(int x)
     {
        if(top==5)
        cout<<"overflow";
        else
        {
           top++;
           a[top]=x;
           cout<<"\nStored into stack "<<x;
        }
     }
void retrieve(void)
     {
        while(top!=-1)
        {
           cout<<" "<<a[top];
           top--;
        }
     }
  };
  class queue : public LIST
  {
int front=0,rear=0,a[5]; public:
```

```
void store(int x)
      {
         if(rear==5)
         cout<<"Queue is full";
         else
         {
            a[rear]=x;
            rear++;
            cout<<"\nStored into queue "<<x;
         }
      }
void retrieve(void)
      {
         if(front==rear)
         cout<<"queue is empty";
         else
         {
            for(int i=0;i<5;i++)
            {
               cout<<" "<<a[i];
               a[i]=a[i+1];
               rear--;
            }
         }
      }
  };
  int main()
  {
stack s; queue q;
s.store(1);
s.store(2);
s.store(3);
s.store(4);
s.store(5);
q.store(6);
q.store(7);
q.store(8);
q.store(9);
q.store(0);
cout<<""<<endl;
s.retrieve();
cout<<""<<endl;
q.retrieve();
  }
```

OUTPUT:



```
PS C:\Users\HP\Desktop\VS Code> cd "c:\Users\HP\Desktop\VS Code\" ;
 if ($?) { g++ OOPS19.cpp -o OOPS19 } ; if ($?) { .\OOPS19 }

Stored into stack 1
Stored into stack 2
Stored into stack 3
Stored into stack 4
Stored into stack 5
Stored into queue 6
Stored into queue 7
Stored into queue 8
Stored into queue 9
Stored into queue 0
 5 4 3 2 1
 6 7 8 9 0
```

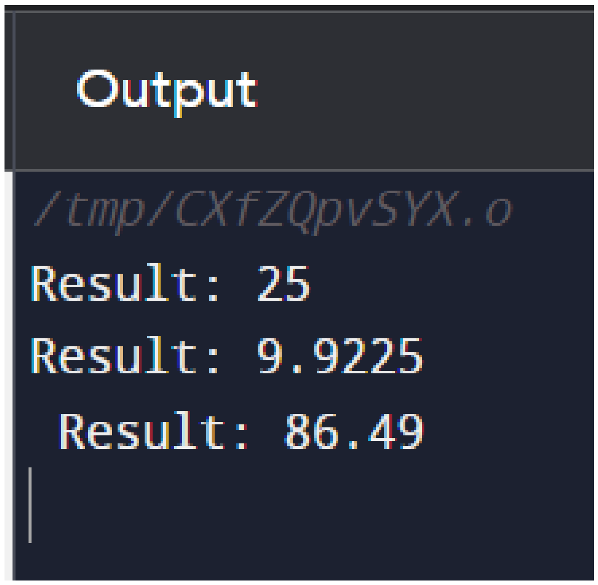
PROGRAM – 20

Aim : write a program to define the function template for calculating the square of givennumbers with different data types

CODE :

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

template
<typename Temp >
class
SquareCalculator {
private:
    Temp result;

public:
    void
        calculateSquare(Temp
        num) {result = num *
        num;
    }

    void displayResult() {
        cout << " Result: " << result << endl;
    }
};

int main() {
    SquareCalculator<int>
    intSquareCalc;
    intSquareCalc.calculateSquar
    e(5);
    intSquareCalc.displayResult(
    );

    SquareCalculator<double>
    doubleSquareCalc;
    doubleSquareCalc.calculateSquare(3.1
    5); doubleSquareCalc.displayResult();


    SquareCalculator<float>
    floatSquareCalc;
    floatSquareCalc.calculateSquare(
    9.3f);
    floatSquareCalc.displayResult();


    return 0;
}
```

OUTPUT :



```
/tmp/CXfZQpvSYX.o
Result: 25
Result: 9.9225
 Result: 86.49
```