1] SINGLE INHERITANCE

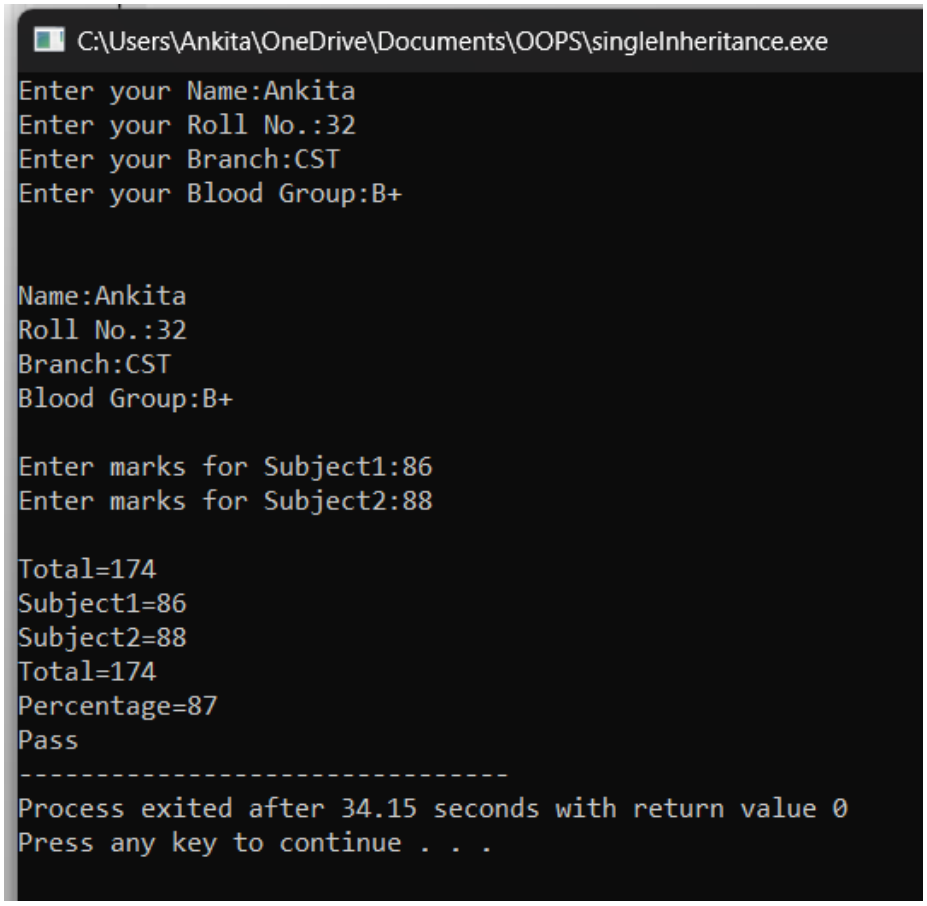
INPUT:

```
#include<iostream>
using namespace std;
class student_info
{
   char name[50];
   int roll_no;
   char branch[50];
   char blood[10];
   public:
   void getdetails()
   {
      cout << "Enter your Name:";
      cin >> name;
      cout << "Enter your Roll No.:";
      cin >> roll_no;
      cout << "Enter your Branch:";
      cin >> branch;
      cout << "Enter your Blood Group:";
      cin >> blood;
   }
   void showdetails()
   {
      cout << "\n\n";
      cout <<"Name:" << name << endl;
      cout <<"Roll No.:" << roll_no << endl;
      cout <<"Branch:" << branch << endl;
      cout <<"Blood Group:" << blood << endl;
```

```
    }
};
class stu_result : public student_info
{
    int sub1, sub2, total;
    float per;
    public:
    void get()
    {
        cout <<"\nEnter marks for Subject1:";
        cin >> sub1;
        cout << "Enter marks for Subject2:";
        cin >> sub2;
        total = (sub1+sub2);
        per = total/2;
        cout << "\nTotal=" << total <<endl;
    }
    void show()
    {
        cout << "Subject1=" << sub1 <<endl;
        cout << "Subject2=" << sub2 <<endl;
        cout << "Total=" << total <<endl;
        cout << "Percentage=" << per <<endl;
        if(per > 40)
        {
            cout <<"Pass";
        }
        else
        {
```

```
            cout << "Fail";
        }
    }
};
int main()
{
    stu_result s1;
    s1.getdetails();
    s1.showdetails();
    s1.get();
    s1.show();
    return 0;
}
```

OUTPUT:



2] MULTIPLE INHERITANCE

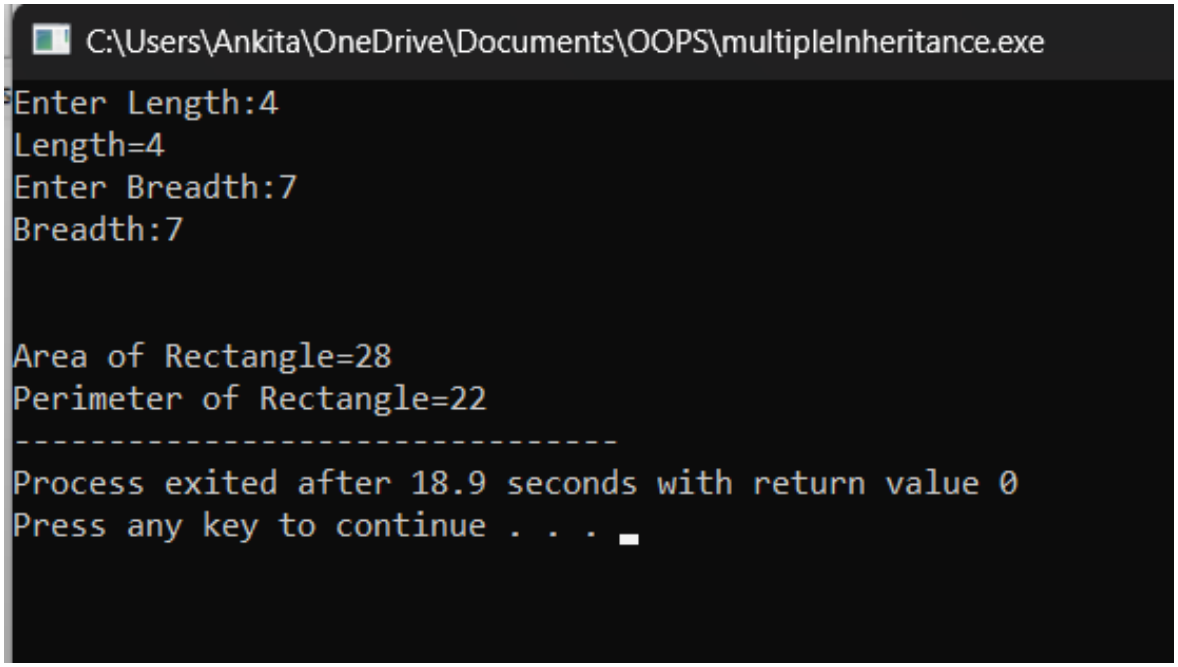
INPUT:

```
#include<iostream>
using namespace std;
class area
{
   protected:
   float l;
   public:
   void getarea()
   {
      cout << "Enter Length:";
      cin >> l;
   }
   void showarea()
   {
      cout << "Length=" << l;
   }
};
class perimeter
{
   protected:
   float b;
   public:
   void getp()
   {
      cout << "\nEnter Breadth:";
      cin >> b;
   }
```

```
    void showp()
    {
        cout << "Breadth:" << b;
    }
};
class rectangle : public area, public perimeter
{
    protected:
    float area, per;
    public:
    void get()
    {
        area = (l*b);
        per = 2*(l+b);
    }
    void show()
    {
        cout << "\n\n";
        cout << "\nArea of Rectangle=" << (l * b);
        cout << "\nPerimeter of Rectangle=" << 2*(l + b);
    }
};
int main()
{
    rectangle r;
    r.getarea();
    r.showarea();
    r.getp();
    r.showp();
```

```
    r.get();
    r.show();
    return 0;
}
```

OUTPUT:



3] MULTILEVEL INHERITANCE

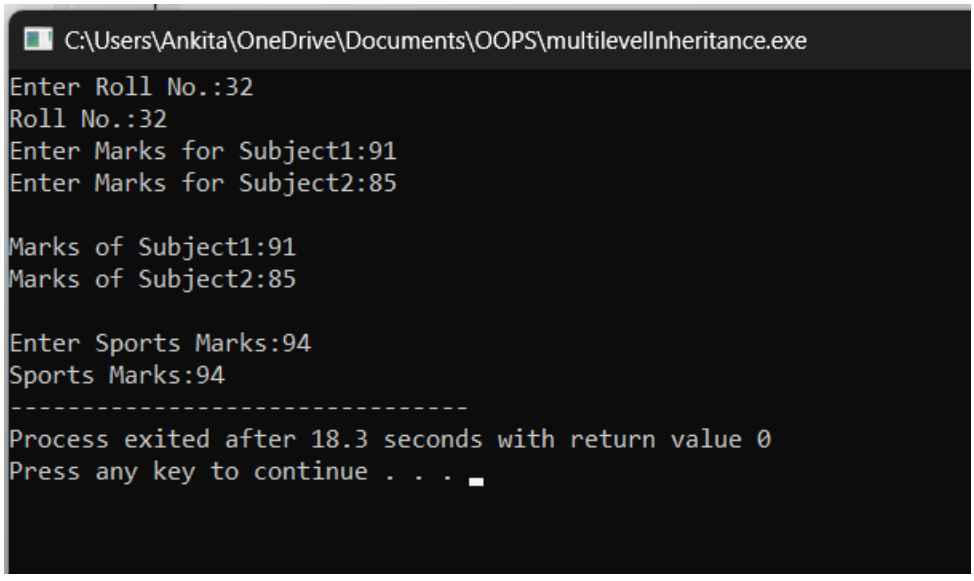
INPUT:

```
#include<iostream>
using namespace std;
class A
{
   public:
   int roll_no;
   void getdata()
   {
      cout << "Enter Roll No.:";
      cin >> roll_no;
```

```
    }
    void showdata()
    {
        cout << "Roll No.:" << roll_no <<endl;
    }
};
class B:public A
{
    public:
    int sub1, sub2;
    void getmarks()
    {
        cout << "Enter Marks for Subject1:";
        cin >> sub1;
        cout << "Enter Marks for Subject2:";
        cin >> sub2;
    }
    void showmarks()
    {
        cout << "\nMarks of Subject1:" << sub1 << endl;
        cout << "Marks of Subject2:" << sub2 << endl;
    }
};
class C:public B
{
    public:
    int spm;
    void getspm()
    {
```

```
        cout << "\nEnter Sports Marks:";
        cin >> spm;
    }
    showspm()
    {
        cout << "Sports Marks:" << spm;
    }
};
int main()
{
    C obj;
    obj.getdata();
    obj.showdata();
    obj.getmarks();
    obj.showmarks();
    obj.getspm();
    obj.showspm();
    return 0;
}
```

OUTPUT:



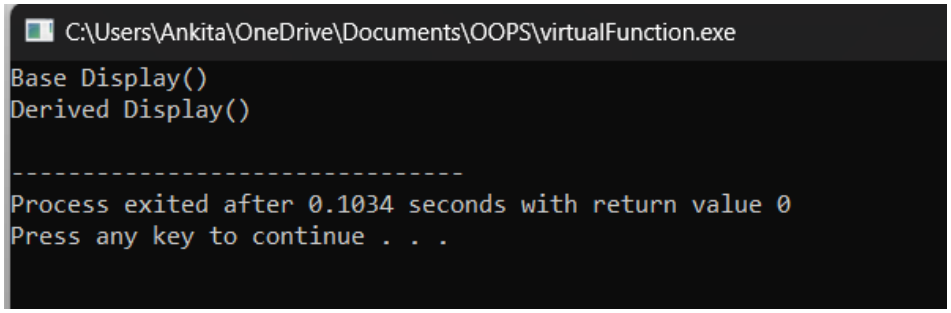
4] VIRTUAL FUNCTION

INPUT:

```
#include<iostream>
using namespace std;
class base
{
   public:
   virtual void display()
   {
      cout << "Base Display()" << endl;
   }
};
class derived : public base
{
   public:
   void display()
   {
      cout << "Derived Display()" << endl;
   }
};
int main()
{
   base *ptr;
   derived d1;
   base b1;
   ptr = &b1;
   ptr -> display();
   ptr = &d1;
```

```
    ptr -> display();
    return 0;
}
```

OUTPUT:



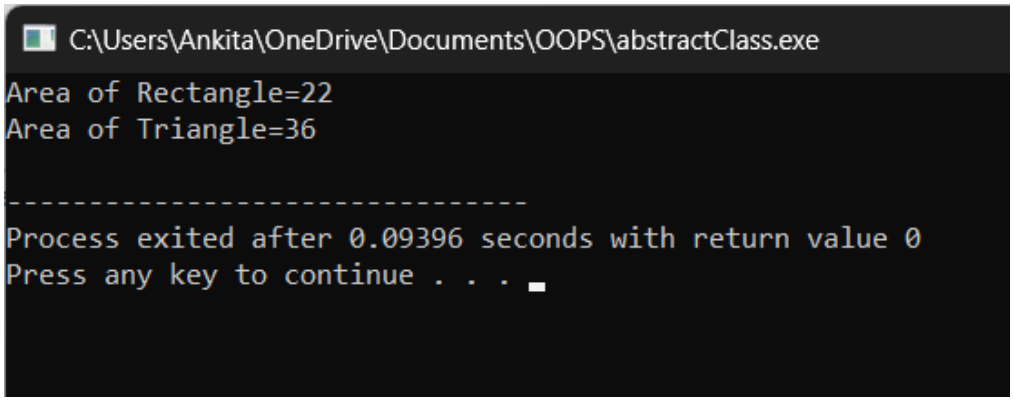
5] ABSTRACT CLASS

INPUT:

```
#include<iostream>
using namespace std;
class shape
{
   protected:
   int width;
   int height;
   public:
   virtual int area() =0;
   void setwidth(int w)
   {
      width = w;
   }
   void setheight(int h)
   {
      height = h;
   }
```

```
};
class rectangle:public shape
{
   public:
   int area()
   {
      return (width*height);
   }
};
class triangle:public shape
{
   public:
   int area()
   {
      return ((width*height)/2);
   }
};
int main()
{
   rectangle r;
   triangle t;
   r.setwidth(2);
   r.setheight(11);
   t.setwidth(12);
   t.setheight(6);
   cout << "Area of Rectangle=" << r.area() << endl;
   cout << "Area of Triangle=" << t.area() << endl;
}
```

OUTPUT:



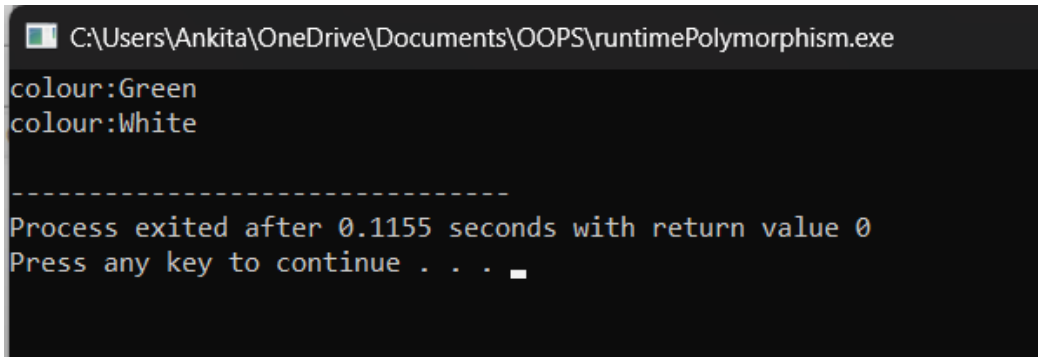
6] RUNTIME POLYMORPHISM

INPUT:

```
#include<iostream>
using namespace std;
class A
{
   public:
   string colour = "Green";
};
class B:public A
{
   public:
   string colour = "White";
};
int main()
{
   A a = B();
   cout << "colour:" << a.colour << endl;
   B b = B();
   cout << "colour:" << b.colour << endl;
   return 0;
```

}

OUTPUT:



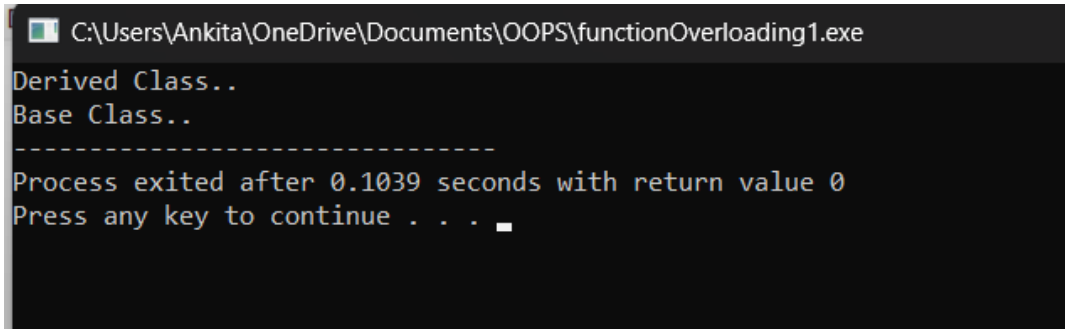
7] FUNCTION OVERLOADING

INPUT:

```
#include<iostream>
using namespace std;
class A
{
   public:
   void display()
   {
      cout << "\nBase Class..";
   }
};
class B:public A
{
   public:
   void display()
   {
      cout << "Derived Class..";
      //A::display();
   }
```

```
};
int main()
{
   B obj;
   obj.display();
   obj.A::display();
   return 0;
}
```

OUTPUT:



8] HIERARCHICAL INHERITANCE

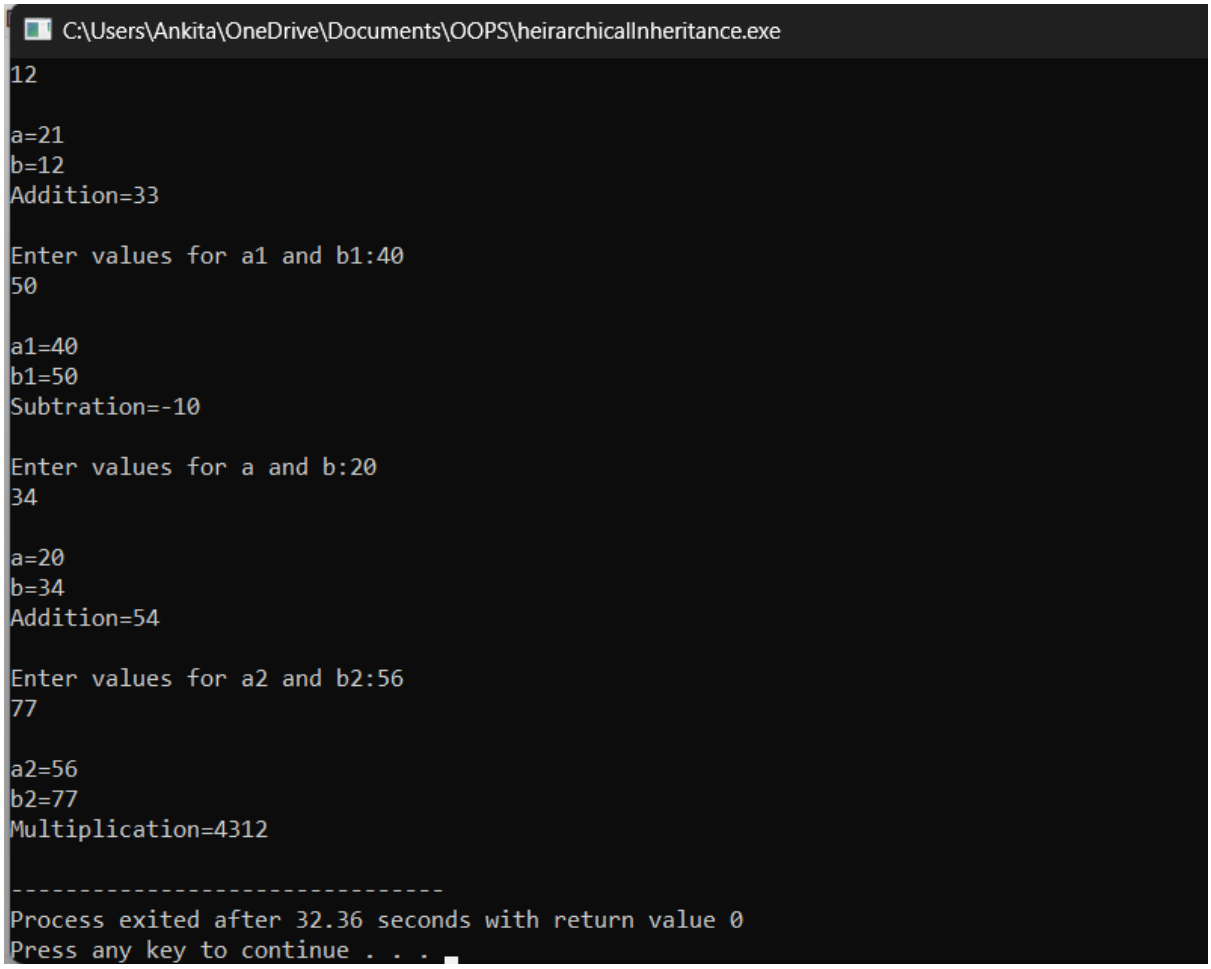
INPUT:

```
#include<iostream>
using namespace std;
class demo
{
   public:
      int a, b, sum;
      void getdata()
      {
         cout << "Enter values for a and b:";
         cin >> a >> b;
```

```
    }
    void showdata()
    {
      cout << "\na=" << a << endl << "b=" << b << endl;
      sum = a + b;
      cout << "Addition=" << a+b << endl;
    }
};
  class sub:public demo
  {
    public:
      int a1, b1, sub;
    void getdetails()
    {
      cout << "\nEnter values for a1 and b1:";
      cin >> a1 >> b1;
    }
    void showdetails()
    {
      cout << "\na1=" << a1 << endl << "b1=" << b1 << endl;
      sub = a1 - b1;
      cout << "Subtration=" << a1-b1 << endl;
      cout << "\n";
    }
};
class derived:public demo
{
  public:
  int a2, b2, mul;
```

```
    void getinput()
    {
        cout << "\nEnter values for a2 and b2:";
        cin >> a2 >> b2;
    }
    void showinput()
    {
        cout << "\na2=" << a2 << endl << "b2=" << b2 << endl;
        mul = a2*b2;
        cout << "Multiplication=" << a2*b2 << endl;
    }
};
int main()
{
    sub obj1;
    obj1.getdata();
    obj1.showdata();
    obj1.getdetails();
    obj1.showdetails();
    derived d1;
    d1.getdata();
    d1.showdata();
    d1.getinput();
    d1.showinput();
    return 0;
}
```

OUTPUT:


C:\Users\Ankita\OneDrive\Documents\OOPS\heirarchicalInheritance.exe
12
a=21
b=12
Addition=33
Enter values for a1 and b1:40
50
a1=40
b1=50
Subtration=-10
Enter values for a and b:20
34
a=20
b=34
Addition=54
Enter values for a2 and b2:56
77
a2=56
b2=77
Multiplication=4312

Process exited after 32.36 seconds with return value 0
Press any key to continue . . .