Q1. Create a base class called shape. Use this class to store two double type values that could be used to compute the area of figures. Derive two specific classes called triangle and rectangle from the base shape. Add to the base class, a member function get_data() to initialise base class data members and another member function display_area() to compute and display the area of figures. Make display_area() as a virtual function and redefine this function in the derived classes to suit their requirements. Using these three classes, design a program that will accept dimensions of a triangle or a rectangle interactively, and display the area. Remember the two values given as input will be treated as lengths of two sides in the case of rectangles, and as base and height in the case of the triangles, and used as follows: Area of rectangle = x * y Area of triangle = 1/2 * x * y

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

class Shape
{
private:
   double x, y;

public:
   void get_data()
   {
      cout<<"Enter data x >>> ";
      cin>>x;

      cout<<"Enter data y >>> ";
      cin>>y;
   }

   virtual void display_area()
   {
      cout<<"Area of rectangle >>> "<< x * y<<endl;
      cout<<"Area of triangle >>> "<< 1/2 * x * y<<endl;
   }

   double ret_x(){return x;}
   double ret_y(){return y;}
};

class Triangle: public Shape
{
public:
   void display_area()
   {
      cout<<"Area of triangle >>> "<< 1/2 * (ret_x() * ret_y())<<endl;
   }
};

class Rectangle: public Shape
{
public:
   void display_area()
   {
      cout<<"Area of rectangle >>> "<< ret_x() * ret_y()<<endl;
   }
};
```

```
int main()
{
   Shape *ptr;
   Triangle t1;
   Rectangle r1;

   ptr = &t1;
   t1.get_data();
   ptr->display_area();

   ptr = &r1;
   r1.get_data();
   ptr->display_area();
}
```

Q2. Extend the above program to display the area of circles. This requires the addition of a new derived class 'circle' that computes the area of a circle. Remember, for a circle we need only one value, its radius, but the get_data() function in the base class requires two values to be passed. (Hint: Make the second argument of get_data() function as a default one with zero value.)

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

class Shape
{
private:
   float pie = 22.7f;
   float r;

public:
   void get_data(float r, int dummy=0)
   {
      this->r = r;
   }

   float ret_r(){return r;}
   float ret_pie(){return pie;}
};

class Circle: public Shape
{
public:
   void disp()
   {
      cout<<"Area of circle >>> "<<ret_pie() * (ret_r() * ret_r())<<endl;
   }
};

int main()
{
   Circle c1;

   c1.get_data(5);
   c1.disp();
}
```

Q3.Using the concept of pointers, write a function that swaps the private data values of two objects of the same class type.

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

class Base
{
private:
   int a, b;

public:
   void input()
   {
      cout<<"a = ";
      cin>>a;

      cout<<"b = ";
      cin>>b;
   }

   void disp()
   {
      cout<<"a = "<<a<<endl;
      cout<<"b = "<<b<<endl;
   }

   void swapping(Base *b2)
   {
      int tmp;

      tmp = this->a;
      this->a = b2->a;
      b2->a = tmp;

      tmp = this->b;
      this->b = b2->b;
      b2->b = tmp;
   }
};
int main()
{
   Base b1, b2;
   cout<<"Before swapping"<<endl<<endl;
   cout<<"Enter b1 data"<<endl;
   b1.input();
   cout<<"Enter b2 data"<<endl;
   b2.input();

   cout<<endl<<"After swapping"<<endl<<endl;
   b1.swapping(&b2);
   cout<<"b1 data"<<endl;
   b1.disp();
   cout<<"b2 data"<<endl;
   b2.disp();
}
```

Q4. Create a base class called shape. Use this class to store two double type values that could be used to compute the area of figures. Derive two specific classes called triangle and rectangle from the base shape. Add to the base class, a member function get_data() to initialise base class data members and another member function display_area() to compute and display the area of figures. Make display_area() as a virtual function and redefine this function in the derived classes to suit their requirements. Using these three classes, design a program that will accept dimensions of a triangle or a rectangle interactively, and display the area. Remember the two values given as input will be treated as lengths of two sides in the case of rectangles, and as base and height in the case of the triangles, and used as follows: Area of rectangle = x * y Area of triangle = 1/2 * x * y

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

class Shape
{
private:
   double x, y;

public:
   void get_data()
   {
      cout<<"Enter data x >>> ";
      cin>>x;

      cout<<"Enter data y >>> ";
      cin>>y;
   }

   virtual void display_area()
   {
      cout<<"Area of rectangle >>> "<< x * y<<endl;
      cout<<"Area of triangle >>> "<< 1/2 * x * y<<endl;
   }

   double ret_x(){return x;}
   double ret_y(){return y;}
};

class Triangle: public Shape
{
public:
   void display_area()
   {
      cout<<"Area of triangle >>> "<< 1/2 * (ret_x() * ret_y())<<endl;
   }
};

class Rectangle: public Shape
{
public:
   void display_area()
   {
      cout<<"Area of rectangle >>> "<< ret_x() * ret_y()<<endl;
   }
};
```

```
int main()
{
   Shape *ptr;
   Triangle t1;
   Rectangle r1;

   ptr = &t1;
   t1.get_data();
   ptr->display_area();

   ptr = &r1;
   r1.get_data();
   ptr->display_area();
}
```

Q5. Create a base class called Photon. Use this class to store double type value of wavelength that could be used to calculate photon energy. Create class calculate_photonEnergy which will photon energy. Using these classes, calculate photon energy.

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

class Photon
{
private:
   double lambda;
public:

   double input()
   {
      cout<<"Enter wavelength(nm) = ";
      cin>>lambda;
      return lambda;
   }
};

class calculate_photonEnergy: public Photon
{
private:
   double E, c, h;
public:
   calculate_photonEnergy()
   {
      h = 6.626 * pow(10, -34);
      c = 3 * pow(10, 8);
      E = 0;
   }
   void photon_energy()
   {
      E = (h * c)/input();
      cout<<"photon energy "<<E<<" Joule/photon"<<endl<<endl;
   }
};

int main()
{
   calculate_photonEnergy obj;

   obj.photon_energy();
 }
```

Q6. Extend above to display the area of circles. For a circle,only one value is needed i.e. radius but in get_data() function 2 values are passed.

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

class Shape
{
private:
   float pie = 22.7f;
   float r;

public:
   void get_data(float r, int dummy=0)
   {
      this->r = r;
   }

   float ret_r(){return r;}
   float ret_pie(){return pie;}
};

class Circle: public Shape
{
public:
   void disp()
   {
      cout<<"Area of circle >>> "<<ret_pie() * (ret_r() * ret_r())<<endl;
   }
};

int main()
{
   Circle c1;

   c1.get_data(5);
   c1.disp();
}
```

Q7. Create a base class called Matrix. Use this class to store 4 int type values that could be used to calculate determinants and create matrices. Create class calculate_determinant which will calculate the determinant of a matrix. Using these classes, calculate the determinant of the matrix.

```
#include <iostream>

using namespace std;

class Matrix
{
private:
   int *ptr;

public:

   Matrix()
   {
      cout<<"matrix";
      ptr = (int *)malloc(sizeof(int) * 4);
   }
   int* setvalue()
   {
      cout<<"Enter a, b, c, d = ";

      for(int i = 0; i < 4; i++)
      {
         cin>>ptr[i];
      }

      return ptr;
   }
};

class calculate_determinant: public Matrix
{
public:
   int calculate()
   {
      int*ptr= setvalue();

      int ans = (ptr[0] * ptr[3]) - (ptr[1] * ptr[2]);

      return ans;
   }
};

int main()
{
   calculate_determinant obj;

   cout<<obj.calculate();
}
```

Q8. Create a base class called proof. Use this class to store two int type values that could be used to prove that triangle is a right angled triangle. Create a class compute which will determine whether a triangle is a right angled triangle. Using these classes, design a program that will accept dimensions of a triangle, and display the result. (Summary: Prove that triangle is a right angled triangle using pythagoras theorem).

```
#include <iostream>

using namespace std;

class Proof
{
private:
   int a, b, c;

public:
   int input()
   {
      cout<<"Enter a side, b side and c side = ";
      cin>>a>>b>>c;

      return ((a*a)+(b*b));
   }

   int retc(){return c*c;}
};

class Compute: public Proof
{
public:
   void triangle()
   {
      if(input() == retc())
      {
         cout<<"This is right angle triangle";
      }
      else
      {
         cout<<"This is not right angle triangle";
      }
   }

};

int main()
{
   Compute obj;

   obj.triangle();
}
```

Q9.Create a base class called volume. Use this class to store two double type values that could be used to compute the volume of figures. Derive two specific classes called cube and sphere from the base shape. Add to the base class, a member function get_data() to initialise base class data members and another member function display_volume() to compute and display the volume of figures. Make display_volume() as a virtual function and redefine this function in the derived classes to suit their requirements. Using these three classes, design a program that will accept dimensions of a cube or a sphere interactively, and display the volume.

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

class Shape
{
private:
   double x, y;

public:

   void get_data()
   {
      cout<<"Enter side of cube : ";
      cin>>x;

      cout<<"Enter radius of sphere : ";
      cin>>y;
   }

   virtual void display_volume()
   {
      cout<<"Volume of cube : "<<cube()<<endl;
      cout<<"Volume of sphere : "<<sphere()<<endl;
   }

   int cube()
   {
      return (x*x*x);
   }

   float sphere()
   {
      return ((4/3)*3.14*y*y*y);
   }
};

class Cube: public Shape
{
public:
   void display_volume()
   {
      cout<<"Volume of cube : "<<cube()<<endl;
   }
};
```

```
class Sphere: public Shape
{
public:
   void display_volume()
   {
      cout<<"Volume of sphere : "<<sphere()<<endl;
   }
};

int main()
{
   Shape *s1;
   Cube c1;
   Sphere p1;

   s1 = &c1;
   s1->get_data();
   s1->display_volume();

   cout<<endl;

   s1 = &p1;
   s1->get_data();
   s1->display_volume();
}
```

Q10. Create a base class called shape. Use this class to store two double type values that could be used to compute the area of figures. Derive two specific classes called square and parallelogram from the base shape. Add to the base class, a member function get_data() to initialise base class data members and another member function display_area() to compute and display the area of figures. Make display_area() as a virtual function and redefine this function in the derived classes to suit their requirements.
Using these three classes, design a program that will accept dimensions of a square or a parallelogram interactively, and display the area.

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

class Shape
{
private:
   double x, y, z;

public:

   void get_data()
   {
      cout<<"Enter side of square : ";
      cin>>x;

      cout<<"Enter base and height : ";
      cin>>y>>z;
   }

   virtual void display_area()
   {
      cout<<"Area of square : "<<square()<<endl;
      cout<<"Area of parallelogram : "<<para()<<endl;
   }

   int square()
   {
      return (x*x);
   }

   float para()
   {
      return (y*z);
   }
};

class Square: public Shape
{
public:
   void display_area()
   {
      cout<<"Area of square : "<<square()<<endl;
   }
};
```

```
class Parallelogram: public Shape
{
public:
   void display_area()
   {
      cout<<"Area of parallelogram : "<<para()<<endl;
   }
};

int main()
{
   Shape *s1;
   Square c1;
   Parallelogram p1;

   s1 = &c1;
   s1->get_data();
   s1->display_area();
   cout<<endl;
   s1 = &p1;
   s1->get_data();
   s1->display_area();
}
```