**Implementing the Concept of Inheritance**

**LAB #** **05**

**Fall 2019**

**CSE208L Object Oriented Programming Lab**

Submitted by: **Shah Raza**

Registration No. : **18PWCSE1658**

Class Section: **B**

“On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work.”

Student Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Submitted to:

**Engr. Sumayyea Salahuddin**

November 20, 2019

Department of Computer Systems Engineering

University of Engineering and Technology, Peshawar

**Objectives of the Lab:**

Objectives of the lab are to:

# Understand the concept of inheritance and its different forms.

# Write derived class from base class using inheritance.

* Understand the constructor and destructor chaining in inheritance hierarchy.
* Know how to use base class constructors within derived class.

# Activity # 01

**Title:**

Make a Parent class for Point and its Child classes Circle and Cylinder.

**Problem analysis:**

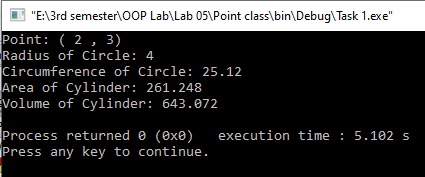
Create a parent class, **Point** that contains x and y coordinates, department. Define a constructor that takes no parameters and also a parameterized one. Define the methods **setx, sety, getx, gety** and **show**. Define child classes of Point, Circle and Cylinder. Demonstrate the use of Point, Circle and Cylinder class.

**Algorithm:**

* First make class Point
* Declare x and y as protected integer field
* Define a no argument constructor.
* Define a constructor with two parameters.
* Define Setx and Sety method to set the values of x and y respectively.
* Define Getx and Gety method to return the values of x and y respectively.
* Define show function to display the output
* Make subclass of Point called Circle.
* Make subclass of Point called Cylinder.
* In main function, make objects of Point, Circle and Cylinder to demonstrate theeuse of Point, Circle and Cylinder.
* Call each function one after the other and display the show function as shown in the flow chart.

**In C++**

**Source code: Output:**

#include <iostream>

using namespace std;

class Point{

protected:

float x,y;

public:

Point();

Point(int a, int b);

void Setx(int a);

void Sety(int a);

int Getx();

int Gety();

void show();

};

Point::Point()

{

x=0,y=0;

}

Point::Point(int a, int b)

{

x=a,y=b;

}

void Point::Setx(int a)

{

x=a;

}

void Point::Sety(int a)

{

y=a;

}

int Point::Getx()

{

return x;

}

int Point::Gety()

{

return y;

}

void Point::show()

{

cout<<"Point: "<<"( "<<x<<" , "<<y<<")";

}

class Circle: public Point

{

protected:

float radius;

public:

Circle();

Circle(float r, Point p);

Circle(float r, float a, float b);

void SetR(float R);

float GetR();

float Radius();

float Circumference();

};

Circle::Circle()

{

radius=0,x=0,y=0;

}

Circle::Circle(float r, Point p)

{

radius=r,x=p.Getx(),y=p.Gety();

}

Circle::Circle(float r, float a, float b)

{

radius=r, x=a, y=b;

}

void Circle::SetR(float R)

{

radius=R;

}

float Circle::GetR()

{

return radius;

}

float Circle::Circumference()

{

return 2\*3.14\*radius;

}

float Circle::Radius()

{

return Circumference()/(2\*3.14);

}

class Cylinder: public Point

{

protected:

float radius,height;

public:

Cylinder(float r,float h, Point p);

Cylinder(float r,float h, float xx,float yy);

Cylinder(float h, Circle C);

float Area();

float Vol();

};

Cylinder::Cylinder(float r,float h, Point p)

{

radius=r;

height=h;

x=p.Getx();

y=p.Gety();

}

Cylinder::Cylinder(float r,float h,float xx,float yy)

{

radius=r;

height=h;

x=xx;

y=yy;

}

Cylinder::Cylinder(float h, Circle C)

{

height=h;

radius=C.GetR();

x=C.Getx();

y=C.Gety();

}

float Cylinder::Area()

{

Circle c(radius,x,y);

return c.Circumference()\*(radius+height);

}

float Cylinder::Vol()

{

Circle c(radius,x,y);

return c.Circumference()\*radius\*height;

}

int main()

{

Point P(2,3);

Circle C(4,P);

C.show();

cout<<"\nRadius of Circle: "<<C.GetR()<<endl;

cout<<"Circumference of Circle: "<<C.Circumference()<<endl;

Cylinder Cy(6.4,C);

cout<<"Area of Cylinder: "<<Cy.Area()<<endl;

cout<<"Volume of Cylinder: "<<Cy.Vol()<<endl;

return 0;

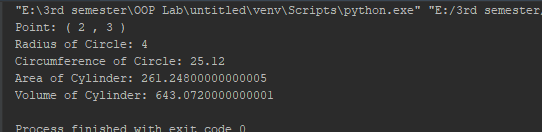
}

**In Python**

**Source code:**

class Point:  
 def \_\_init\_\_(self,a=0,b=0):  
 self.x=a  
 self.y=b  
 def Setx(self,a):  
 self.x=a  
 def Sety(self,b):  
 self.y=b  
 def Getx(self):  
 return self.x  
 def Gety(self):  
 return self.y  
 def Show(self):  
 print("Point: ( {} , {} )".format(self.x,self.y))  
  
class Circle(Point):  
 def \_\_init\_\_(self,r=0,a=0,b=0):  
 self.radius=r  
 self.x=a  
 self.y=b  
 def \_\_init\_\_(self,r,p):  
 self.radius=r  
 self.x=p.Getx()  
 self.y=p.Gety()  
 def SetR(self,r):  
 self.radius=r  
 def GetR(self):  
 return self.radius  
 def Circumference(self):  
 return 2\*3.14\*self.radius  
class Cylinder(Point):  
 def \_\_init\_\_(self,h,C):  
 self.height=h  
 self.radius=C.GetR()  
 self.x=C.Getx()  
 self.y=C.Gety()  
 def Area(self):  
 return 2\*3.14\*self.radius\*(self.radius+self.height)  
 def Vol(self):  
 return 2\*3.14\*self.radius \* (self.radius \* self.height)  
P = Point(2,3)  
C = Circle(4,P)  
C.Show()  
print("Radius of Circle: {}".format(C.GetR()))  
print("Circumference of Circle: {}".format(C.Circumference()))  
Cy = Cylinder(6.4,C)  
print("Area of Cylinder: {}".format(Cy.Area()))  
print("Volume of Cylinder: {}".format(Cy.Vol()))

**Output:**



**Conclusion:**

This program helps us in understanding the basic concepts of **Inheritance** in different languages. It acts as a base for us and helps us in preparing ourselves for the higher level of programming. We get to know about the constructor and method in OOP with the help of this program.