ESc101 : Fundamental of Computing

I Semester 2008-09

Lecture 21

Object Oriented programming

- class and object (recap from previous lecture)
- Object : reference, creation
- Object : accessing attributes and executing methods
- Example : class of Point
- Example : class of Triangle (discussed partially)

Crucial Observation which forms the basis of OOP

For a given real life problem :

whenever we think of the (attributes of the) data, ...

Crucial Observation which forms the basis of OOP

For a given real life problem :

whenever we think of the (attributes of the) data, the thoughts of the methods on them comes automatically in our mind.

Crucial Observation which forms the basis of OOP

For a given real life problem :

whenever we think of the (attributes of the) data, the thoughts of the methods on them comes automatically in our mind.

So it is useful to think of (the attributes of) data in conjunction of the methods which work on them.

Objects

Objects are self contained entity which has its own collection of

- attributes
- methods to access them, manipulate them and compute some functions on them.

Encapsulation : one of the fundamental principle of OOP

The ability of an object to be a container (or capsule) for its attributes (i.e. data variables) and its related methods (i.e. functions).

Object-oriented programming may be seen as *a collection of cooperating objects*.

What is a class ?

Definition : A class specifies the attributes (data) and methods (actions) that objects can work with.

Class for Point

```
public class Point
{ double x;
  double y;

  public void setX(double x_value)  {x = x_value;}
  public void setY(double y_value)  { y = y_value;}
  public double distance_from_origin()
  {
    return Math.sqrt(x*x+y*y);
  }
}
```

How to create an object, say a point

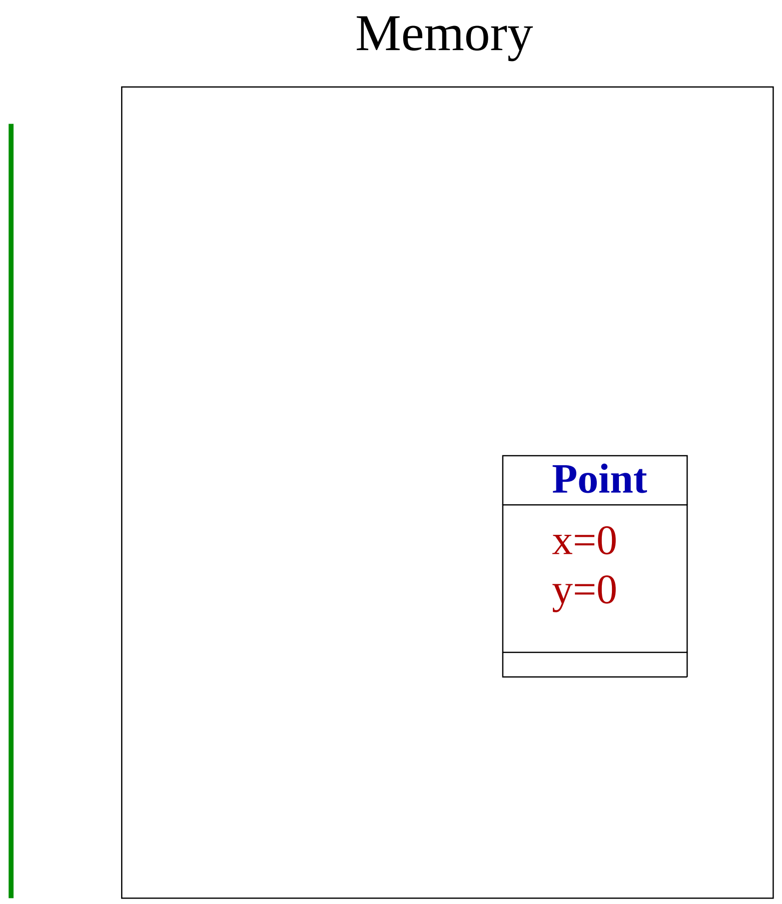
new Point()

Here `Point()` is a special method called **constructor** since it *constructs* one object of class Point.

(We shall discuss in details about *constructor* in next class)

What happens when new Point() is executed

new Point( );



We shall have to use an identifier for an object

Point **P**;

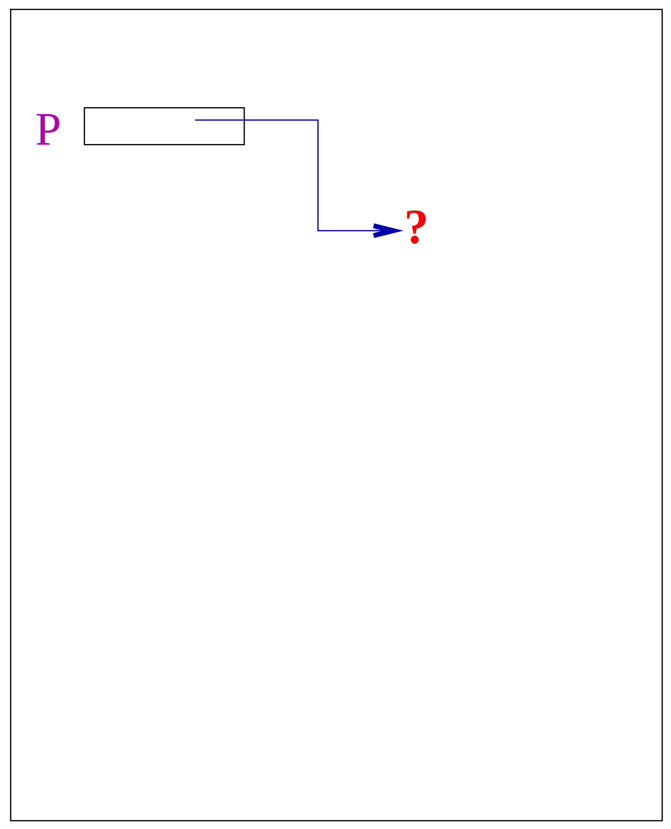
Here **P** is a variable which is a reference to a **Point** object.

P stores *the location* of a **Point** object.

What happens when Point P; is executed

```
Point     P ;
```

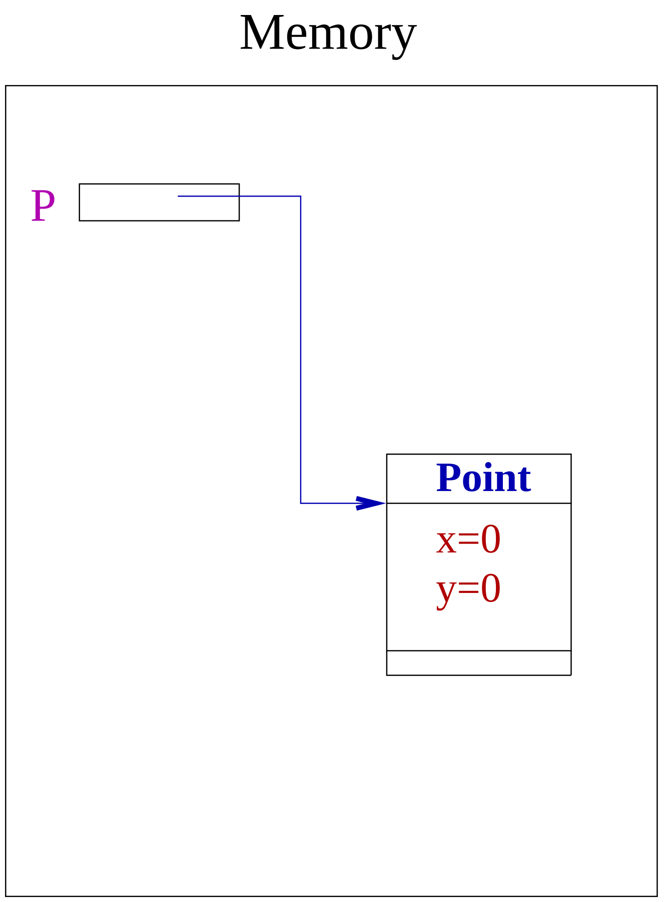
Memory



The complete description of creation of an object

```
Point P;
P=new Point();
```

```
Point    P ;
P= new   Point( );
```



Operations on the object

accessing an attribute : **object_name.attribute_name**

executing a method : **object_name.method_name(parameters)**

Accessing attributes

```
class program00
{
  public static void main(String args[])
  {
    Point P;
    P = new Point();
    System.out.println("The point P has coordinates :");
    double x_coord = P.x;
    double y_coord = P.y;
    System.out.println(x_coord+’’,’’+y_coord);
  }
}
```

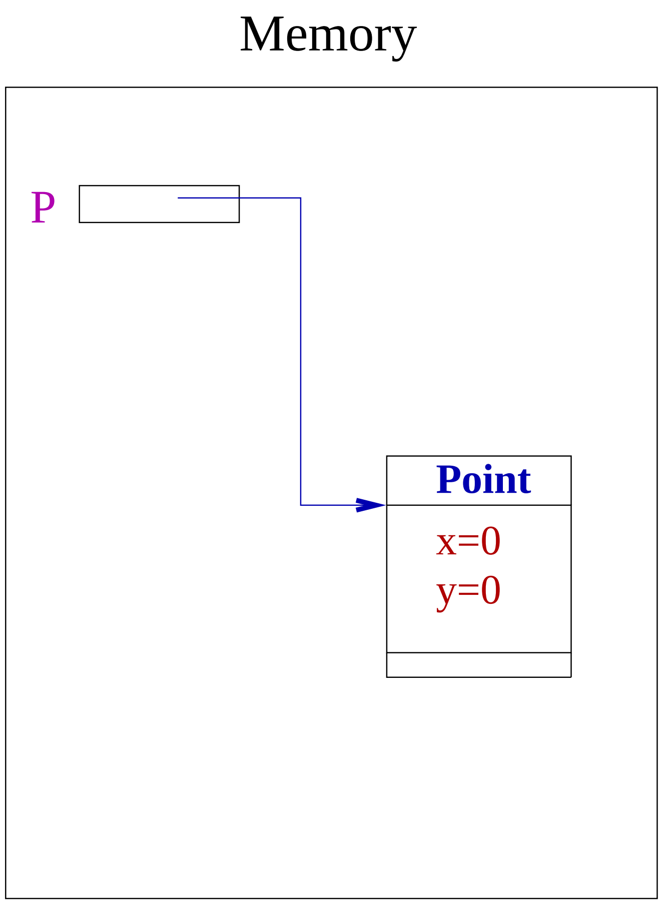
Ouput will be   0.0 , 0.0

Program0 : methods applied on the object

```
class program0
{
  public static void main(String args[])
  {
    Point P;
    P = new Point();
    P.setX(1);
    P.setY(2.5);
    System.out.println("P is ("+P.x+","+P.y+")");
    }
}
```

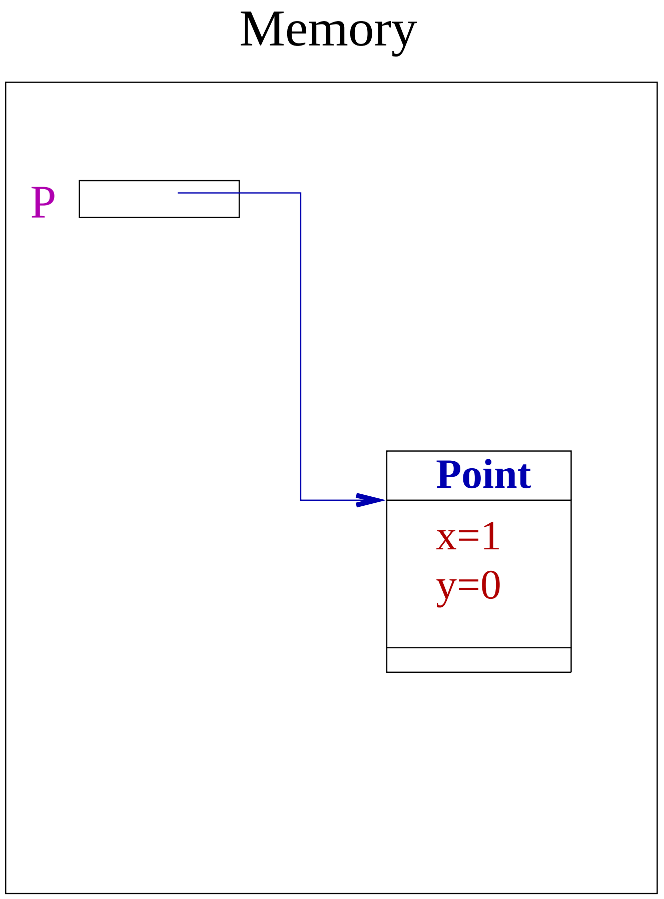
After first two steps of the Execution of Program0

```
Point      P ;
P= new   Point( );
P.setX(1);
P.setY(2.5);
```



After first three steps of the Execution of Program0

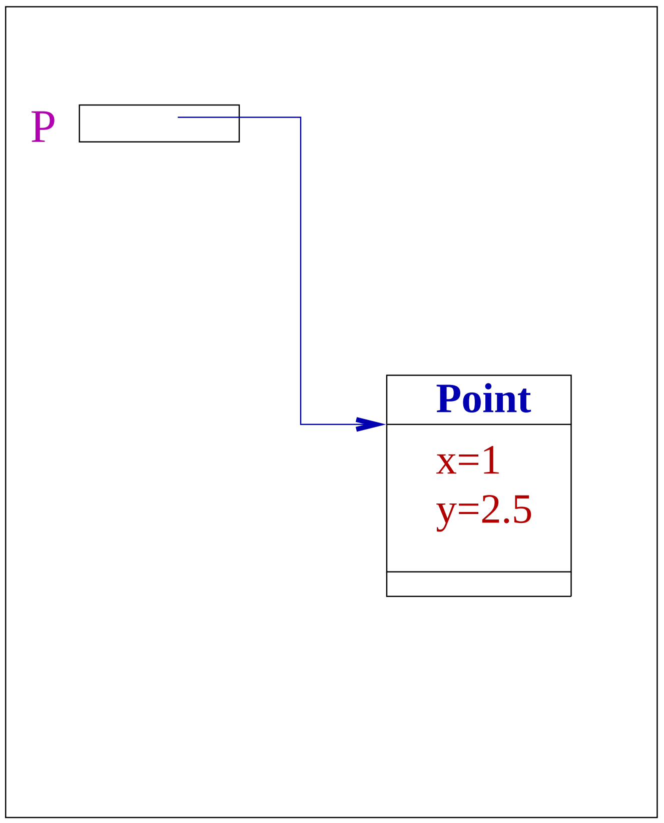
```
Point     P ;
P= new   Point( );
P.setX(1);
P.setY(2.5);
```



After first four steps of the Execution of Program0

```
Point    P ;
P= new   Point( );
P.setX(1);
P.setY(2.5);
```

Memory



More about a variable of object reference

Point **P**;

Note that **P** is a variable which stores reference of an object of class Point.

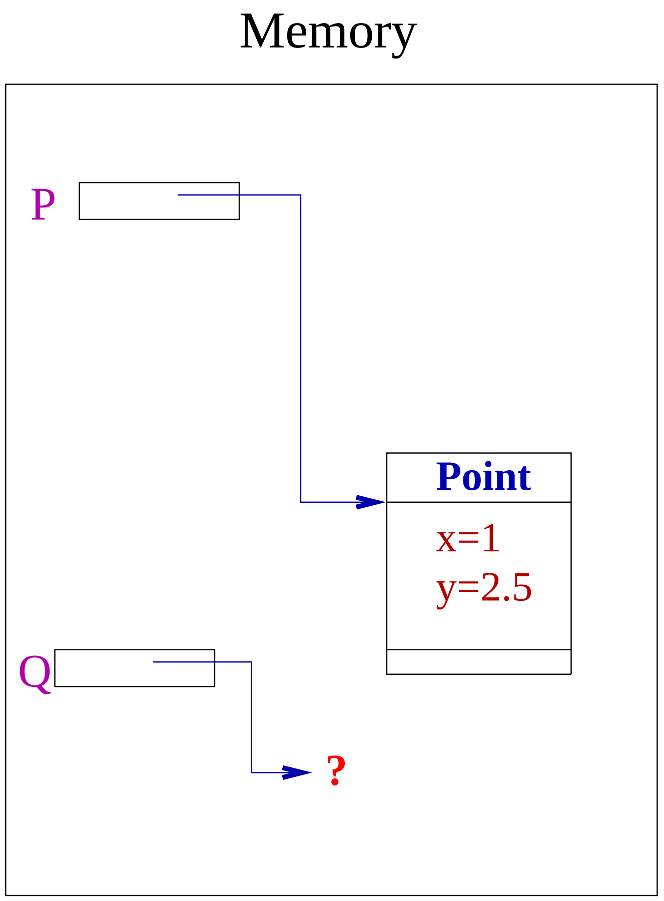
It is not an object

Program1

```
class program1
{ public static void main(String args[])
  {
    Point P;
    P = new Point();
    P.setX(1);    P.setY(2.5);
    System.out.println(P.x+’’,’’+P.y);
    Point Q;
    Q = new Point();
    System.out.println(Q.x+’’,’’+Q.y);
    P=Q;
    P.setX(4.37);
   System.out.println(Q.x+’’,’’+Q.y);
    }
}
```

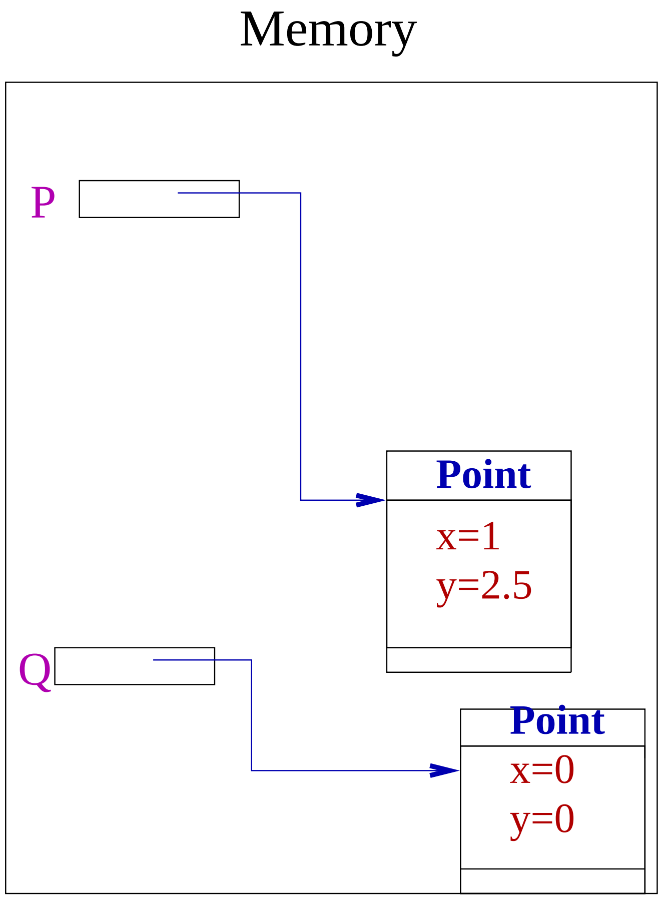
Execution of program1

```
Point   P ;
P= new  Point( );
P.setX(1);
P.setY(2.5);
Point   Q ;
```



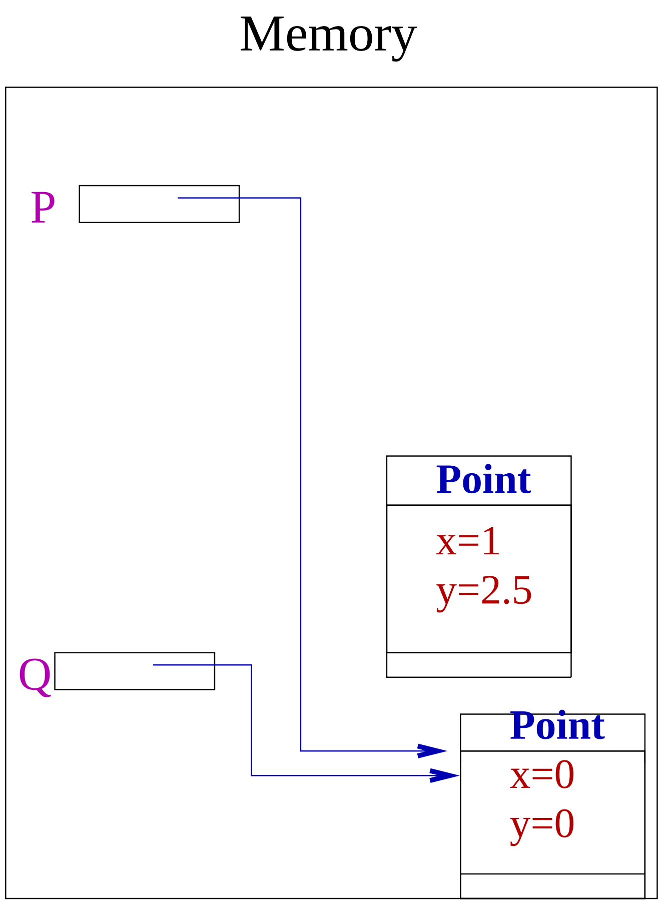
Execution of program1

```
Point    P ;
P= new  Point( );
P.setX(1);
P.setY(2.5);
Point    Q ;
Q= new  Point( );
```



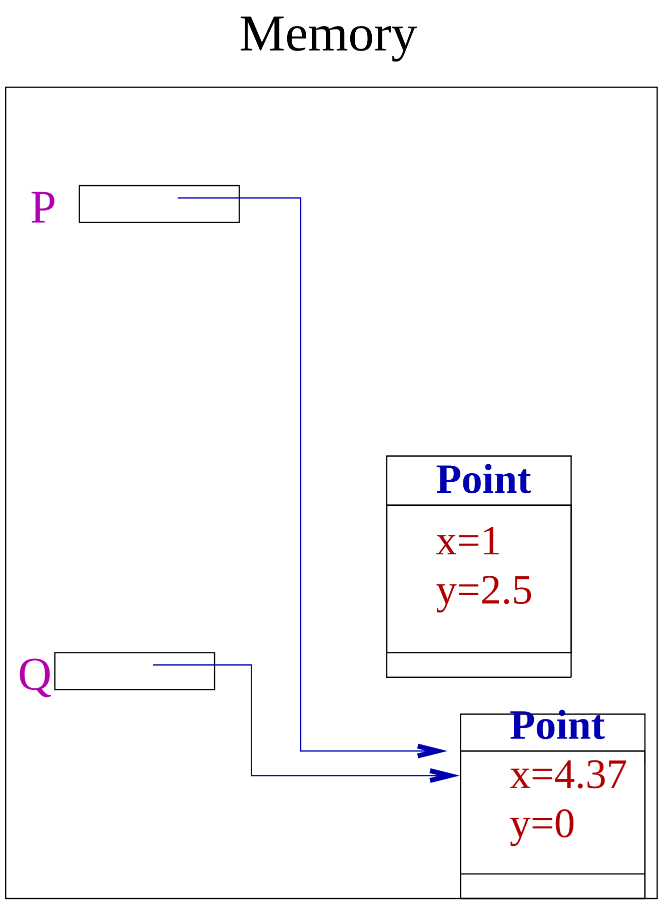
Execution of program1

```
Point   P ;
P= new  Point( );
P.setX(1);
P.setY(2.5);
Point   Q ;
Q= new  Point( );
P = Q ;
```



Execution of program1

```
Point   P ;
P= new  Point( );
P.setX(1);
P.setY(2.5);
Point   Q ;
Q= new  Point( );
P = Q ;
P.setX(4.37);
```



Using Objects, we may form complex data types

We can use Points to define classes for

1. triangle
2. segment
3. square
4. :
5. :

Class of triangle

```
class Triangle
{ Point P;
  Point Q;
  Point R;

  double perimeter()
  {
    :
    :
  }
}
```

To be continued from here on Monday ...