Object Oriented Programming (OOP)

Dr. Mohamed Ezz
Al-Azhar University

Lecture 2

First OOP Class

Review

1. What is the class?
2. What is the Object?
3. Example of classes:
4. Example of Objects:
5. The state or behavior for class or object !

Lecture Objectives

- Create First Class
- Create Object from Class
- Access Object member variable
- Methods & class's behaviors
- How to declare instance variables & Method in a class
- How to access an object's member (instance variable) & Method.

Trace Circle Example

Circule.java

```
public class Circle
{
        double radius;
}
```

TestCircle.java

```
public class TestCircle
{
        public static void main (String arg[])
        {
                Circle myCircle = new Circle();
                Circle yourCircle = new Circle();
                yourCircle.radius = 100;
        }

}
```

Create 1st Object – step 1

```
Circle myCircle = new Circle();

Circle yourCircle = new Circle();

yourCircle.radius = 100;
```

Declare myCircle

myCircle | no value

Create 1st Object – step 2

```
Circle myCircle = new Circle();

Circle yourCircle = new Circle();

yourCircle.radius = 100;
```

myCircle | no value

: Circle

radius=0

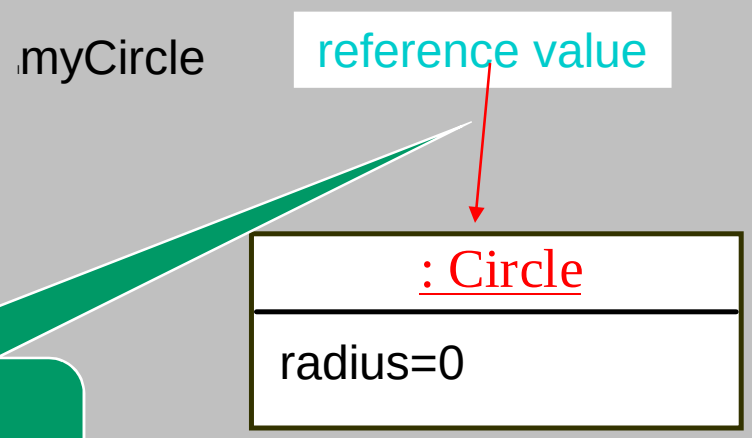
Create a circle

Create 1st Object – step 3

```
Circle myCircle = new Circle();

Circle yourCircle = new Circle();

yourCircle.radius = 100;
```



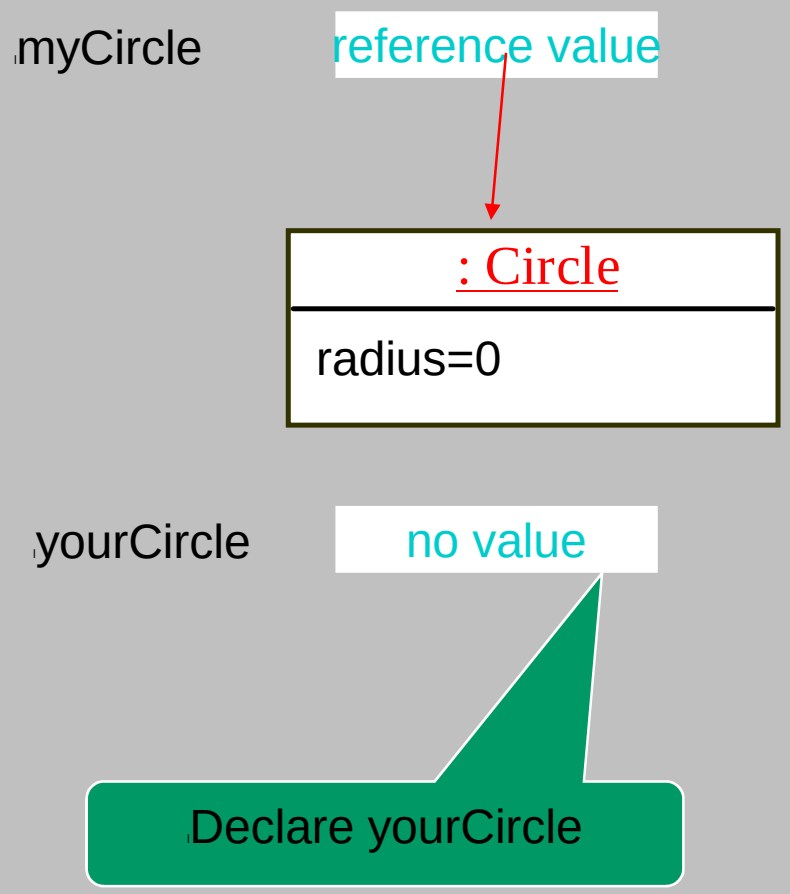
Assign object reference to myCircle

Create 2nd Object – step 1

```
Circle myCircle = new Circle();

Circle yourCircle = new Circle();

yourCircle.radius = 100;
```

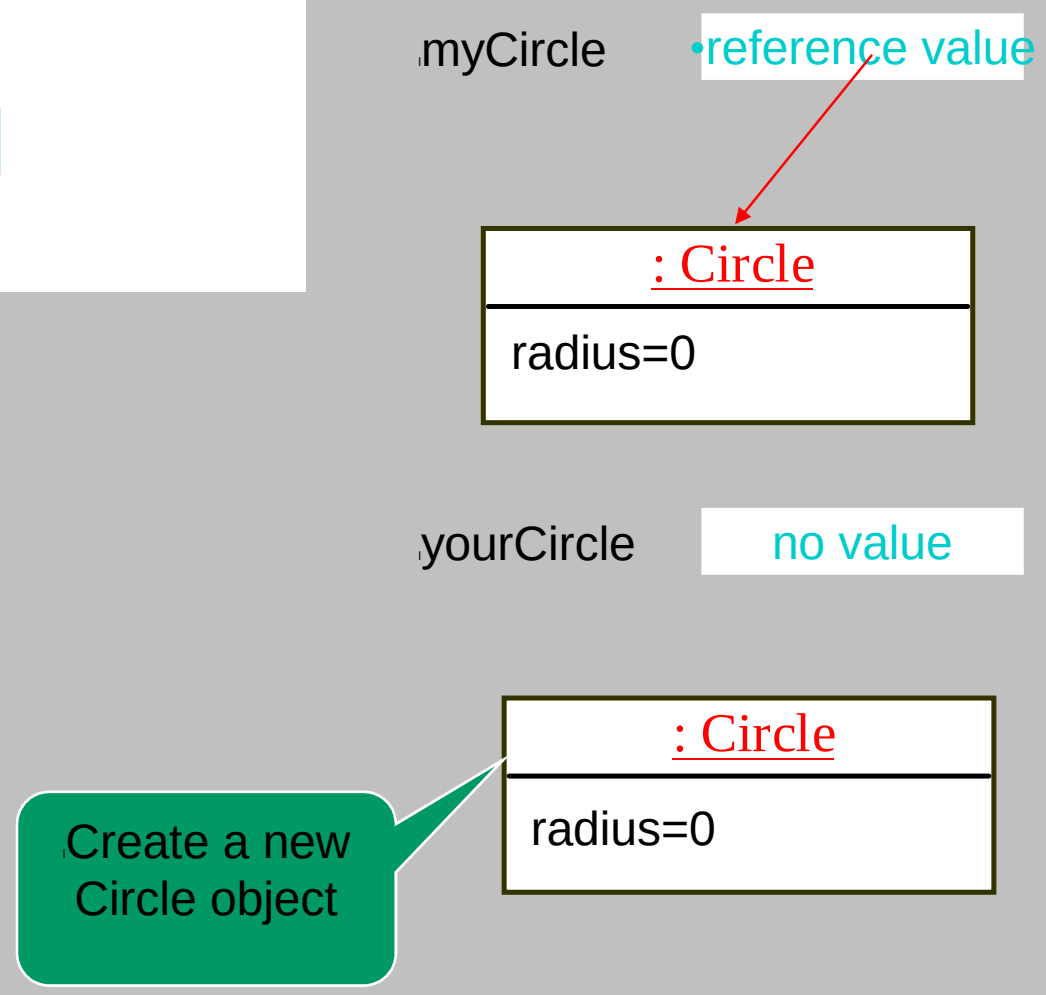


Create 2nd Object – step 2

```
Circle myCircle = new Circle();

Circle yourCircle = new Circle();

yourCircle.radius = 100;
```

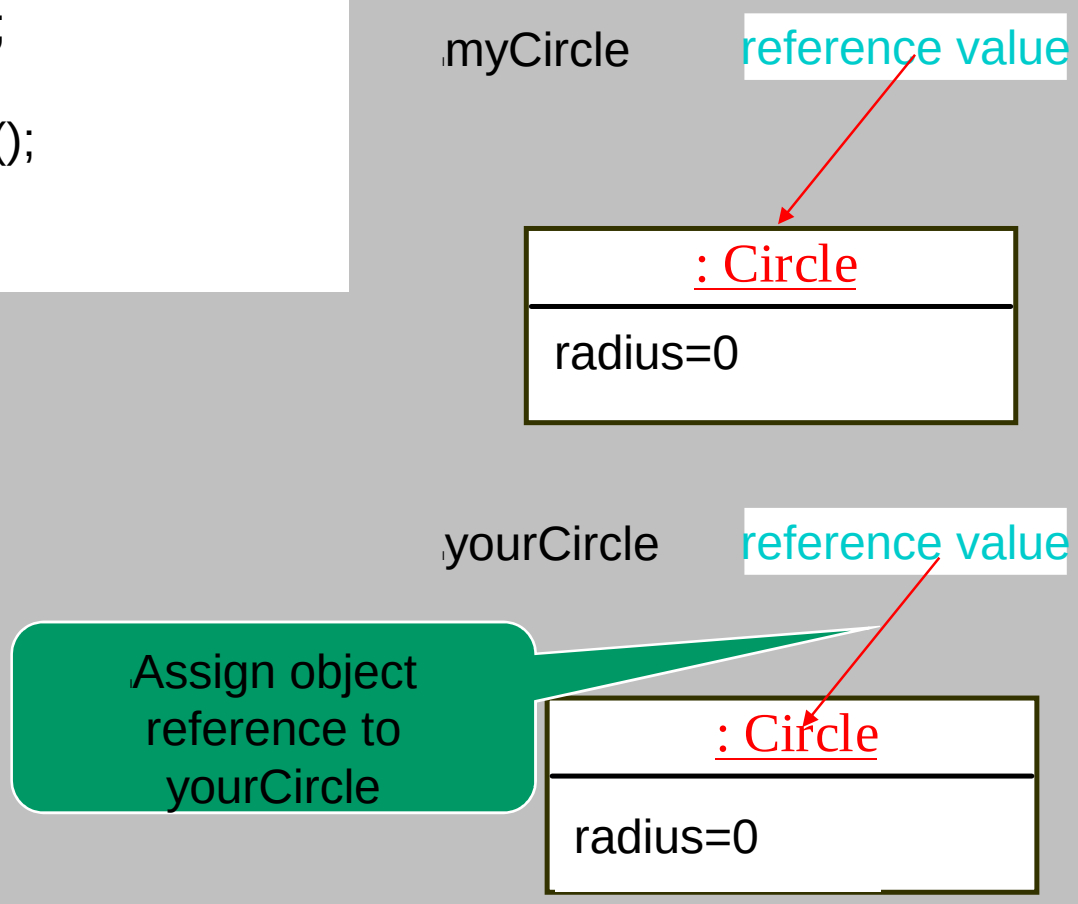


Create 2nd Object – step 3

```
Circle myCircle = new Circle();

Circle yourCircle = new Circle();

yourCircle.radius = 100;
```

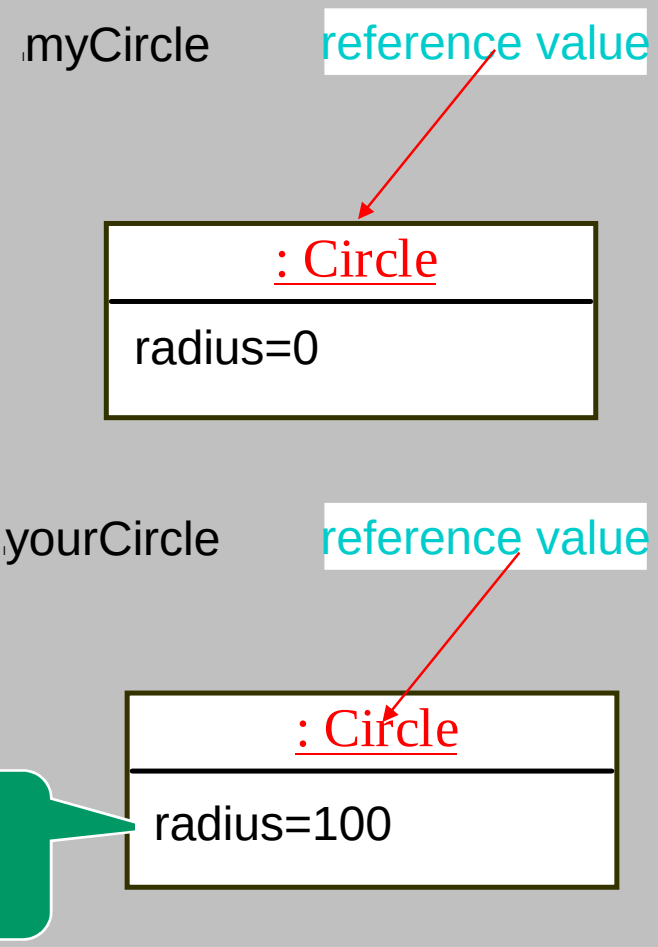


Access 2nd Object member

```
Circle myCircle = new Circle();

Circle yourCircle = new Circle();

yourCircle.radius = 100;
```



Declaring Object Reference

- To reference an **object**, assign the object to a **reference variable.**

- To declare a **reference variable**, use the syntax:

  - `ClassName objectRefVar;`

  - Example: `Circle myCircle;`

Declaring/Creating Objects

```
ClassName objectRefVar = new ClassName();
```

Example:

Assign object reference

Create an object

```
Circle myCircle = new Circle();
```

Accessing Objects

Referencing the object's data:

`objectRefVar.data`

e.g., `myCircle.radius`

Using Member Variables

Member variable declaration

- Declaration is the same as common variables
  - Any where in a class
  - Outside all methods

Used within the class

- Can be used/referenced anywhere in the class
  - Inside instance method (**directly**)

Used with prefixed with objects

- Can be used/referenced using the "." operator and prefixed with object name
  - outside the class (e.g. from inside test class)
  - Inside static method (**directly**)
  - e.g. : `myCircle.radius`

Using Member Method

Member method declaration

- Declaration is the same as common method/function
  - Any where in a class
  - Outside all methods

Used within the class

- Can be used/invoked anywhere in the class
  - Inside instance method (**directly**)

Used with objects

- Can be used/invoked using the "." operator and preceded with object name
  - outside the class
  - Inside static method (**directly**)
  - e.g. : myCircle.getArea()

Syntax OF Member Variable &Method

Member Variable

- Variable_type variable_name;

Example : int `radius;`

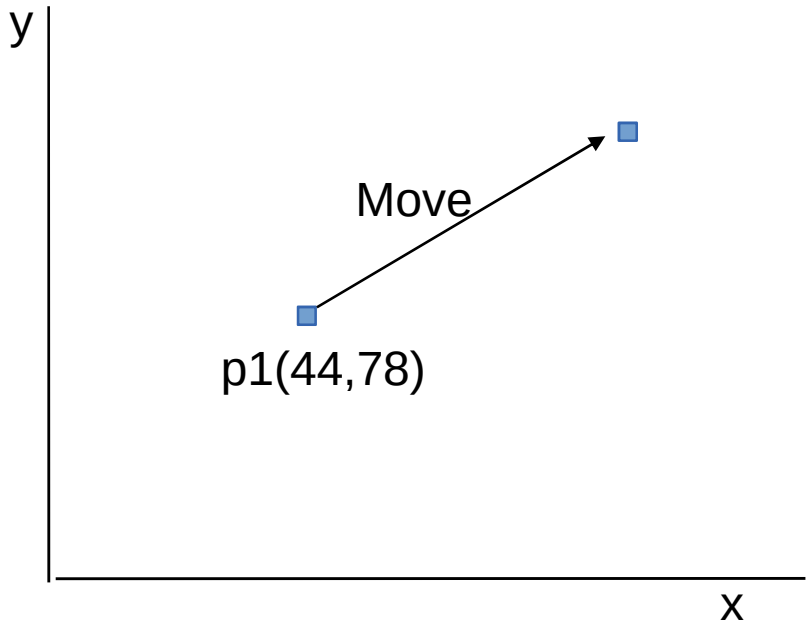
Member Method

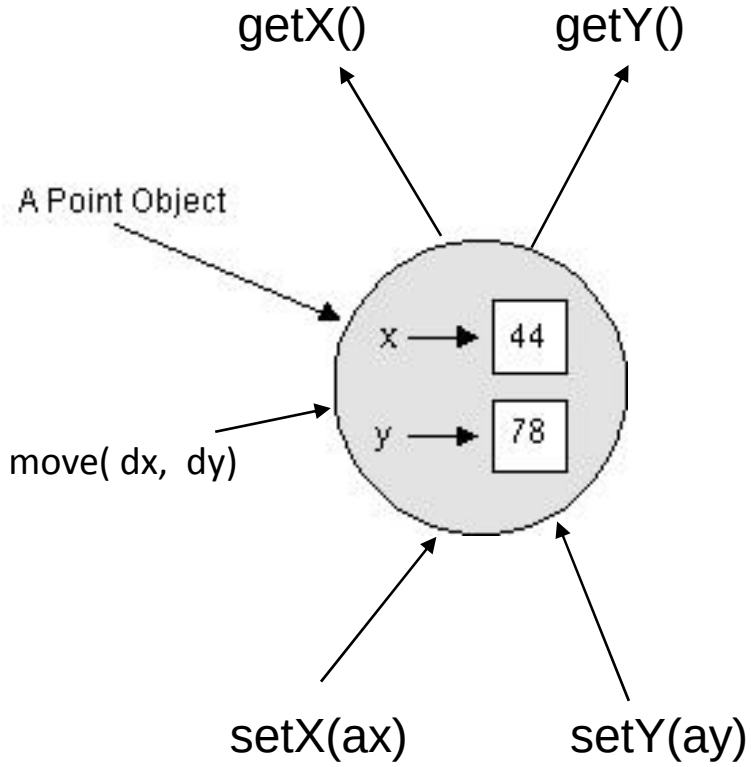
- Same as function : may have parameter or return type

Example :

```
public int getRadius ()
{
        return radius;
}
}
```

Point Class Example





Point Class

Point.java

```
public class Point
{
        //member variables
        private int x, int y;

        //Member Methods
        public void  setX(int ax) {
                x= ax;
        }

        public void  setY(int ay) {
                y= ay;
        }

        public int  getX() {
                return x;
        }

        public int  getY() {
                return y;
        }
```

Point.java

```
        public void move(int dx, int dy) {
                setX(x+dx);
                setY(y+dy);
        }
}
```

Test Point Class

TestPoint.java

```
public class TestPoint
{

        public static void main(String arg[]){
            System.out.println("Test Point Class");
            Point p1 =  new Point();
            Point p2 = new Point();
            p1.setX(1);
            p1.setY(3);
            p2.setX(4);
            p2.setY(5);
            System.out.println(p1);
            p1.move(2,2);
            System.out.println(p1);

      }// end of main
}// end of class
```

Defining methods

With a return value (type)

```
public int getX()
{
        return x;
}
```

Access member variable

Without a return value (type)

```
public void setX(int xx)
{
        x= xx;
}
```

Update member variable

Calling Methods

Within the class

- Called by method name directly

Example: inside **move** function

```
setX(x+dx);
setY(y+dy);
```

Used with objects or outside the class

- Using the "." operator and preceded with object name

Examples:

```
p1.setY(3);
System.out.println("Point at" +p1.getX() + "," p1.getY() );
p1.move(2,2);
```

Instance Variables Setter &Getter

Ensure data encapsulation instance variable accessed through interface (set & get methods)

Setter method

- Set the value of the instance variable

Example: setting radius of circle

```
public setRadius(int r) {
        radius = r;        // what extra benefits from set method?
}
```

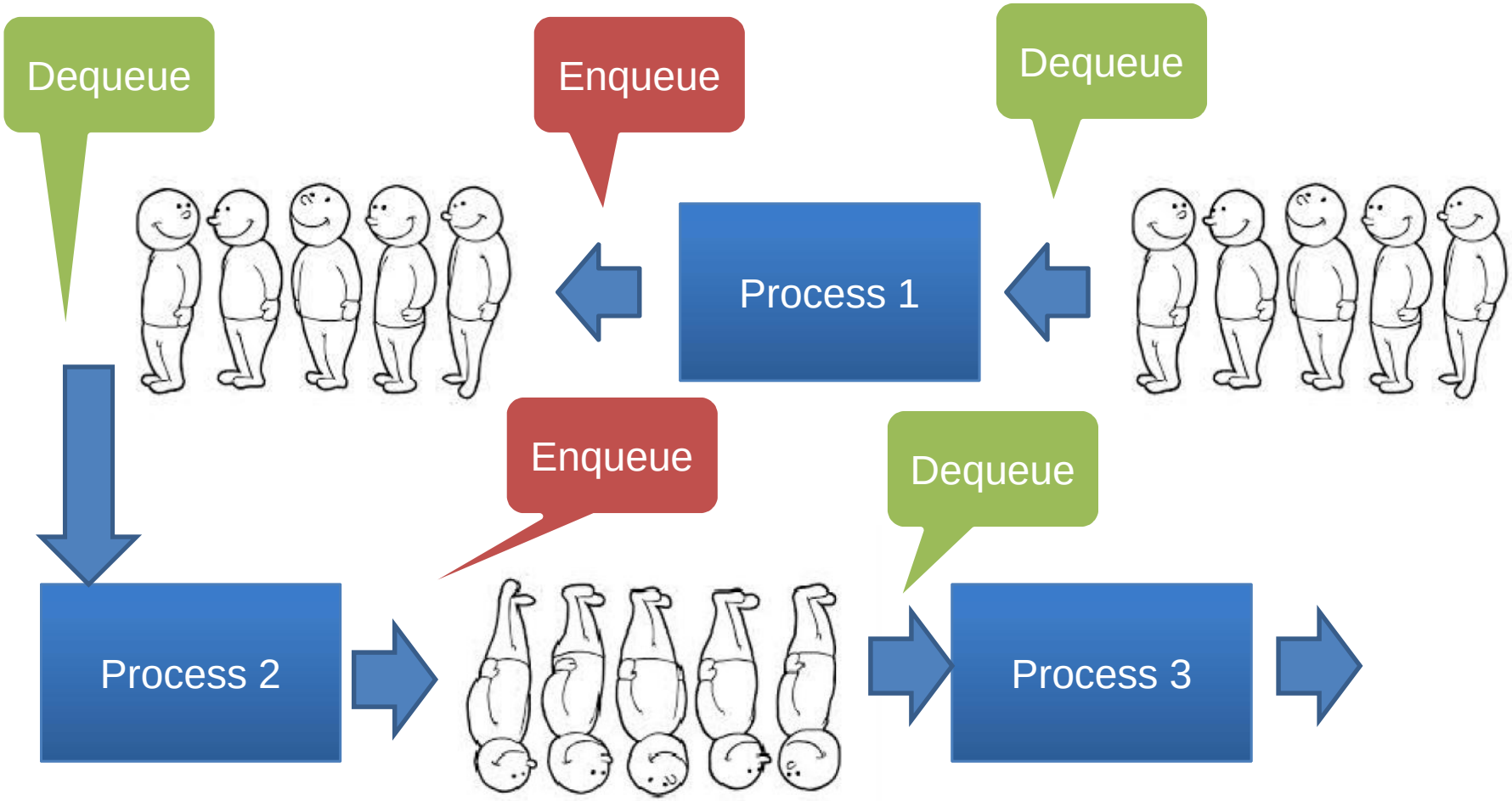
Getter method

- Get the value of the instance variable

Example: getting radius of circle

```
public int getRadius() {
        return radius ;
}
```

Queue Application



Array implementation

```
public class TestQueue {

public static void main(String[] args) {
 QueueArray q1= new QueueArray ();
 QueueArray q2= new QueueArray ();
 QueueArray q3= new QueueArray ();

q1.enqueue(2);
q1.enqueue(5);

q2.enqueue(q1.dequeue());
q2.enqueue(q1.dequeue());

q3.enqueue(q2.dequeue());
q3.enqueue(q3.dequeue());

 }//end main
}//end class
```

Queue Class

```
public class QueueArray {
    int maxSize=100;
    int [] queueArray =new int [100];
    int front=0;
    int size=0;
public void enqueue(int v) {
    if (size == maxSize) {
        System.out.println("Full");
    } else {
        queueArray[rear] = v;
        rear = (rear + 1) % maxSize;
        size++;
    }
}//end enqueue
```

```
public int dequeue() {
        if (size == 0) {
        System.out.println("Empty");
        return -1;
    } else {
        int v = queueArray[front];
        front = (front + 1) % maxSize;
        size--;
        return v;
    }
}//end dequeue
public void print() {
    int index = front;
    for(int i=0;i<size;i++)
    {
      System.out.print(queueArray[index] + "-");
        index = (index + 1) % maxSize;
    }
        System.out.println("");
}//end print

}//end class
```

Queue Application Homework

Enter The following numbers in first queue:

- 2
- 5
- 4
- 7
- 1
- 9

Then pass it to other queues