CS112

Objects and Classes (Part 1)

Lecture 02

Spring 2022 - 1443

College of Computer Science and Engineering

What do we mean by OO programming?

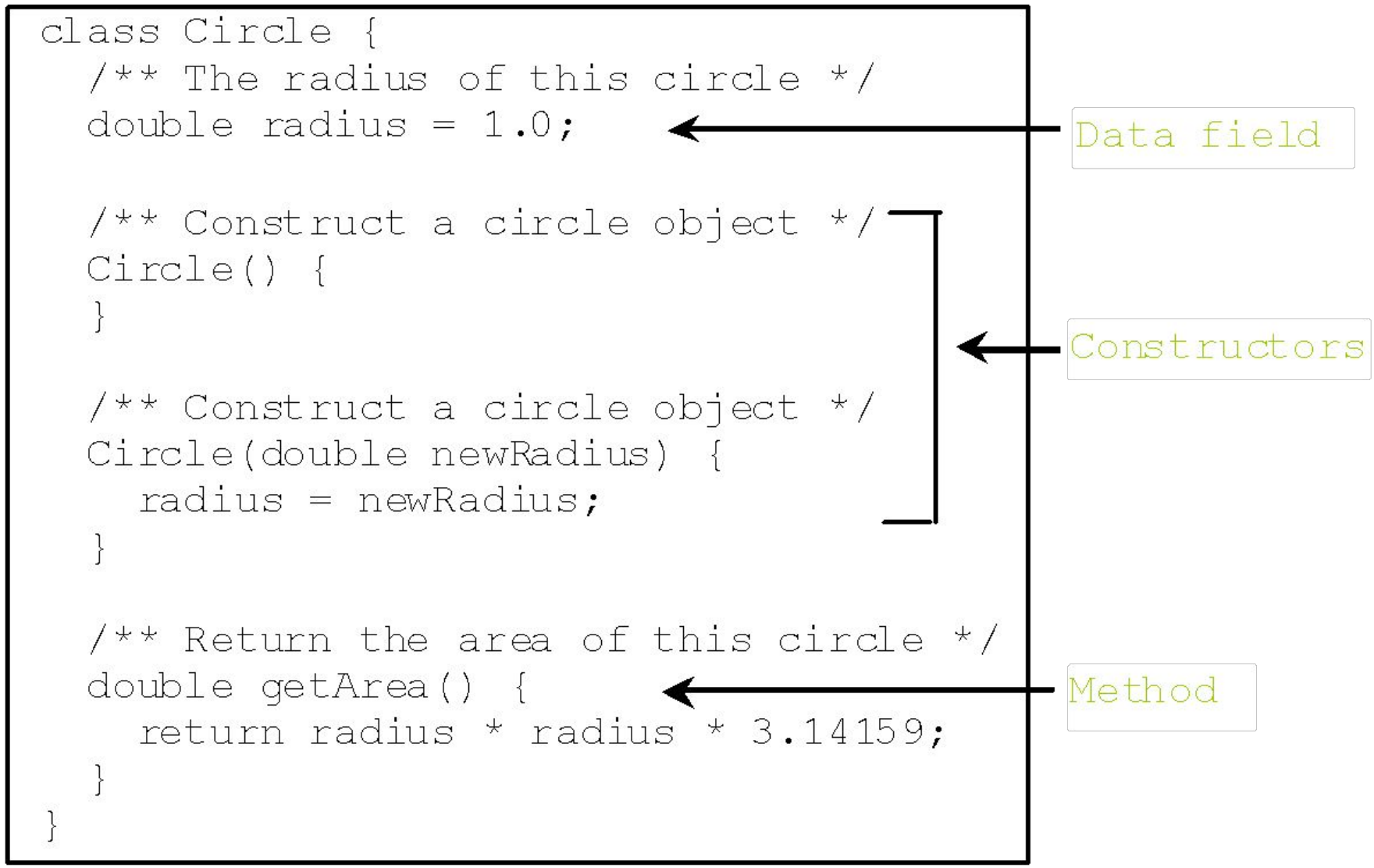
- Object-oriented programming (OOP) involves programming **using objects**
- An *object* represents an entity in the real world that can be distinctly identified
  - For example, a student, a desk, a circle, a button, and even a loan can all be viewed as objects.
- An object has:
  1. A unique identity
  2. A state which consists of a set of *data fields* (known as **properties**) with their current values
  3. Set of behaviors (known as **methods**)

Classes

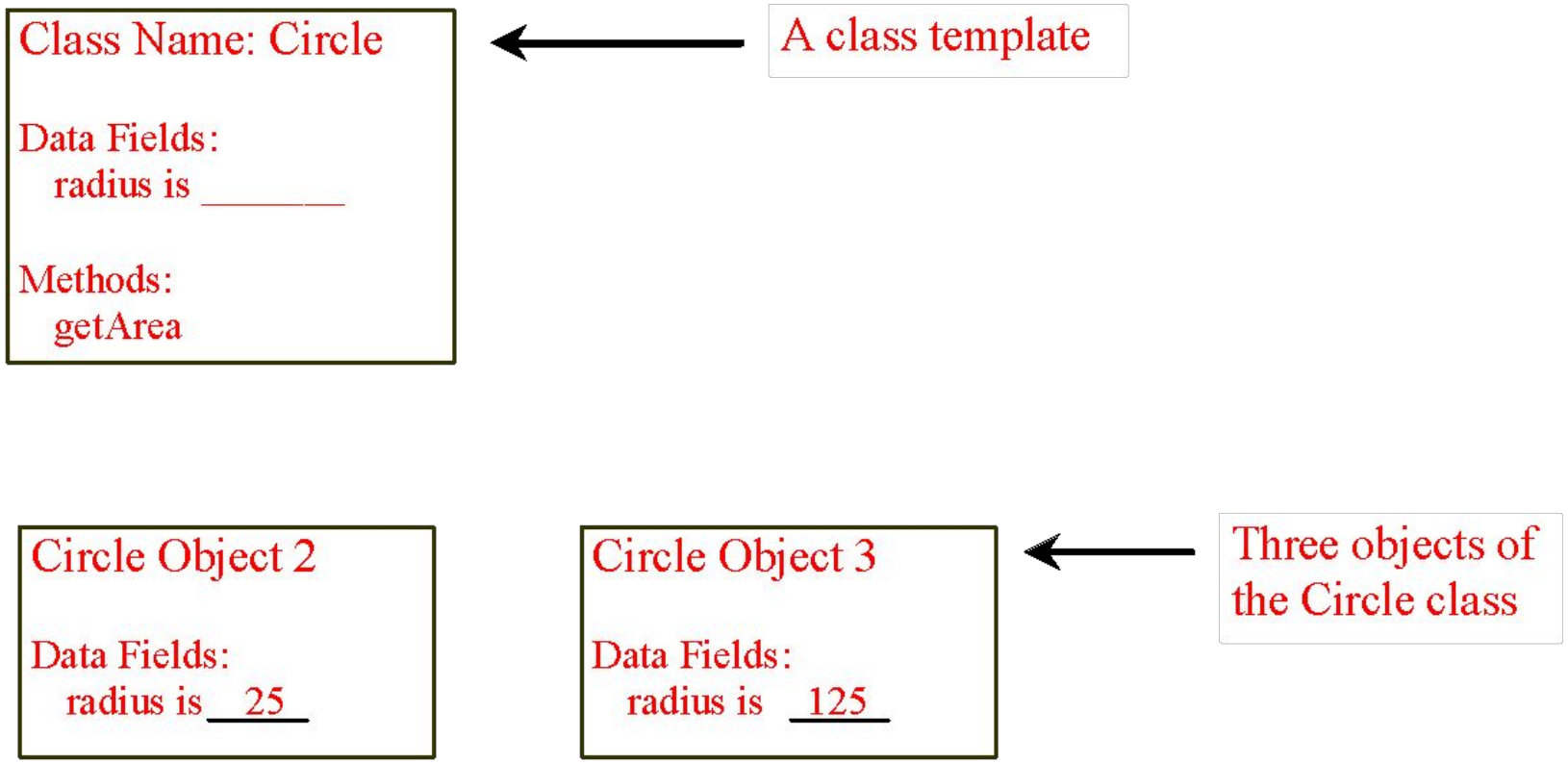
- Classes are constructs that define objects of the same type
- A Java class uses variables to define data fields and methods to define behaviors
- A class provides a special type of methods, known as constructors
  - Constructors are invoked to construct objects from the class

Example – Circle class

```
class Circle {
  /** The radius of this circle */
  double radius = 1.0;

  /** Construct a circle object */
  Circle() {
  }

  /** Construct a circle object */
  Circle(double newRadius) {
    radius = newRadius;
  }

  /** Return the area of this circle */
  double getArea() {
    return radius * radius * 3.14159;
  }
}
```

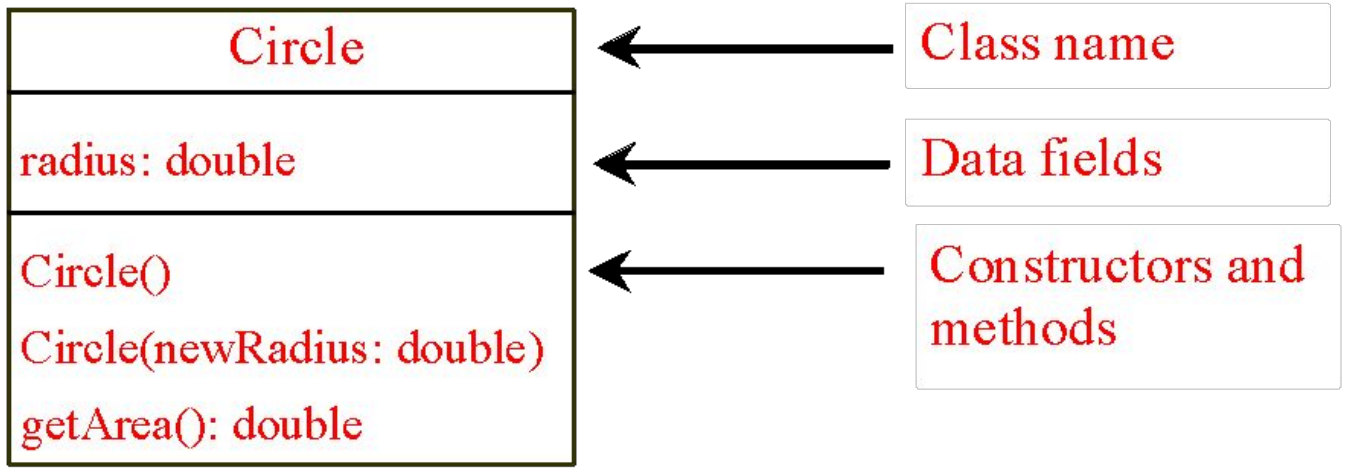


Example – Objects from Circle class



Example – UML Diagram

UML Class Diagram



Constructors (1)

- Constructors are a special kind of methods that are invoked to construct objects

```
Circle() {
}

Circle(double newRadius) {
  radius = newRadius;
}
```

Constructors (2)

- A constructor with no parameters is referred to as a *no-arg constructor*
- Constructors must have the same name as the class itself
- Constructors **do not have a return type—not even void**
- Constructors are invoked using the **new** operator when an object is created
- Constructors play the role of initializing objects

Creating Objects Using Constructors

```
new ClassName();
```

Example:

```
new Circle();
```

```
new Circle(5.0);
```

Default Constructor

- A class may be defined without constructors **A default constructor is defined automatically**
- In this case, a no-arg constructor with an empty body is implicitly defined in the class
  - This constructor, called *a default constructor*, is provided automatically *only if no constructors are explicitly defined in the class*

Declaring Objects Reference Variables

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

- To declare a reference variable, use the syntax:

```
ClassName objectRefVar;
```

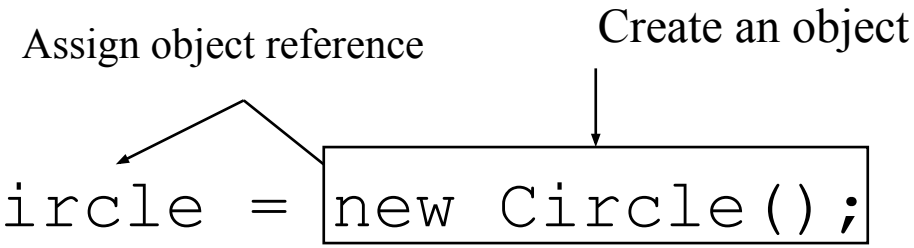
Example:

```
Circle myCircle;
```

Declaring/Creating Objects in a Single Step

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

Example:



Accessing Object's Members

- Referencing the object's data:

```
objectRefVar.data
```

e.g., `myCircle.radius`

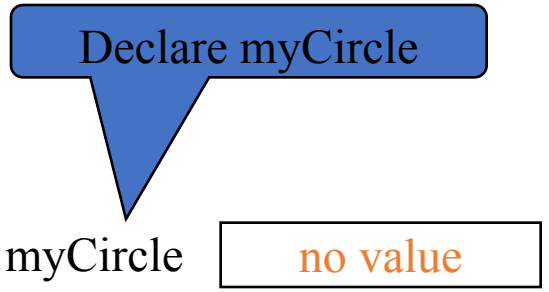
- Invoking the object's method:

```
objectRefVar.methodName(arguments)
```

e.g., `myCircle.getArea()`

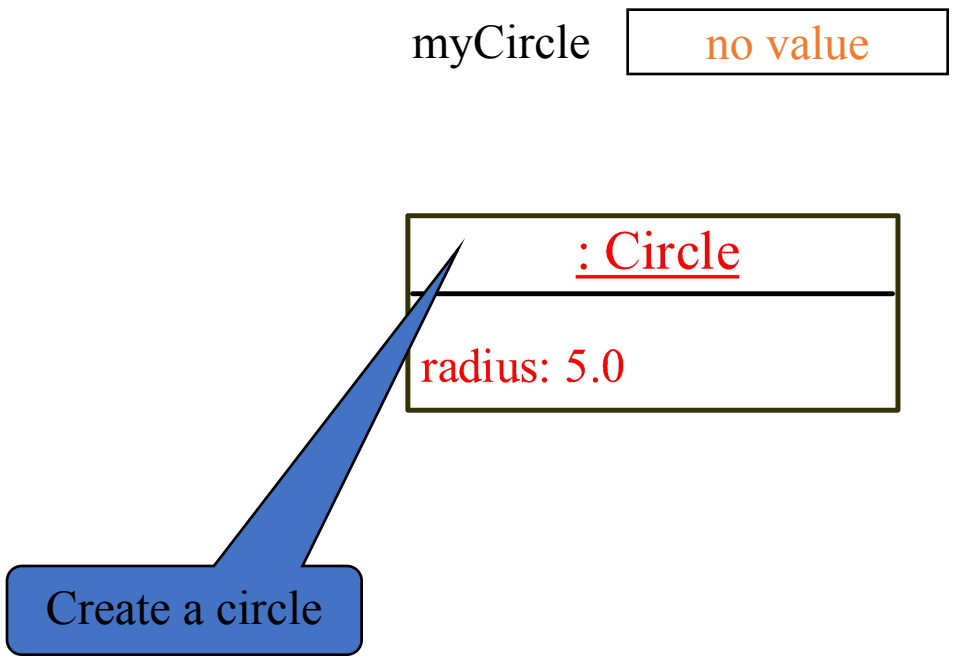
Trace Code – Circle class example (1)

```
Circle myCircle = new Circle(5.0);

Circle yourCircle = new Circle();

yourCircle.radius = 100;
```



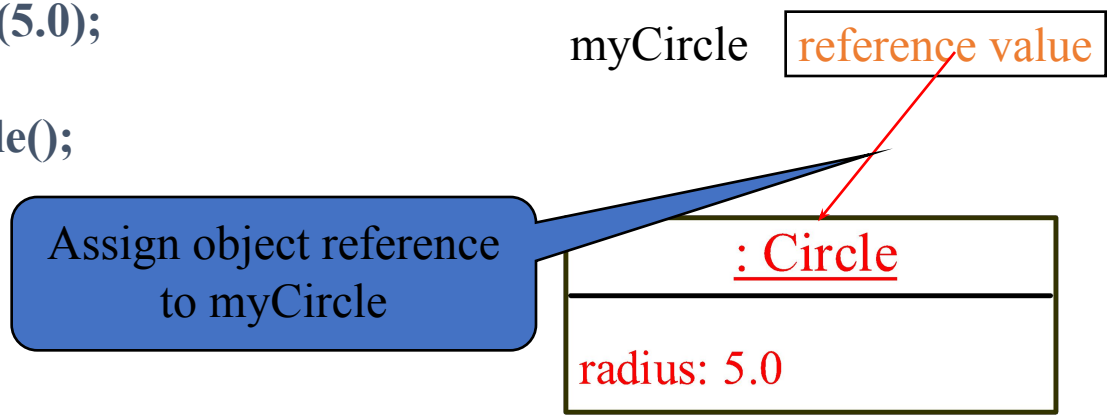
Trace Code – Circle class example (2)

```
Circle myCircle = new Circle(5.0);

Circle yourCircle = new Circle();

yourCircle.radius = 100;
```



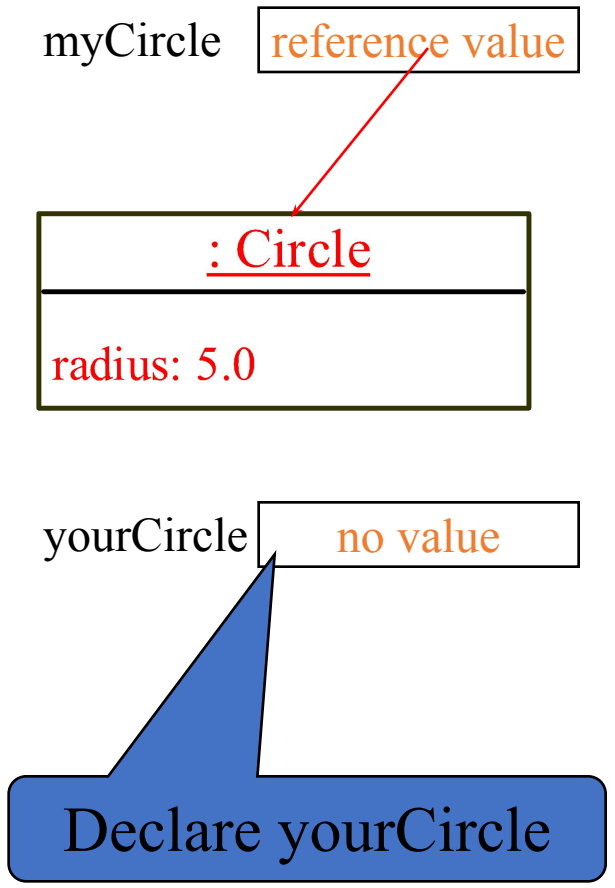
Trace Code – Circle class example (3)

```
Circle myCircle = new Circle(5.0);

Circle yourCircle = new Circle();

yourCircle.radius = 100;
```



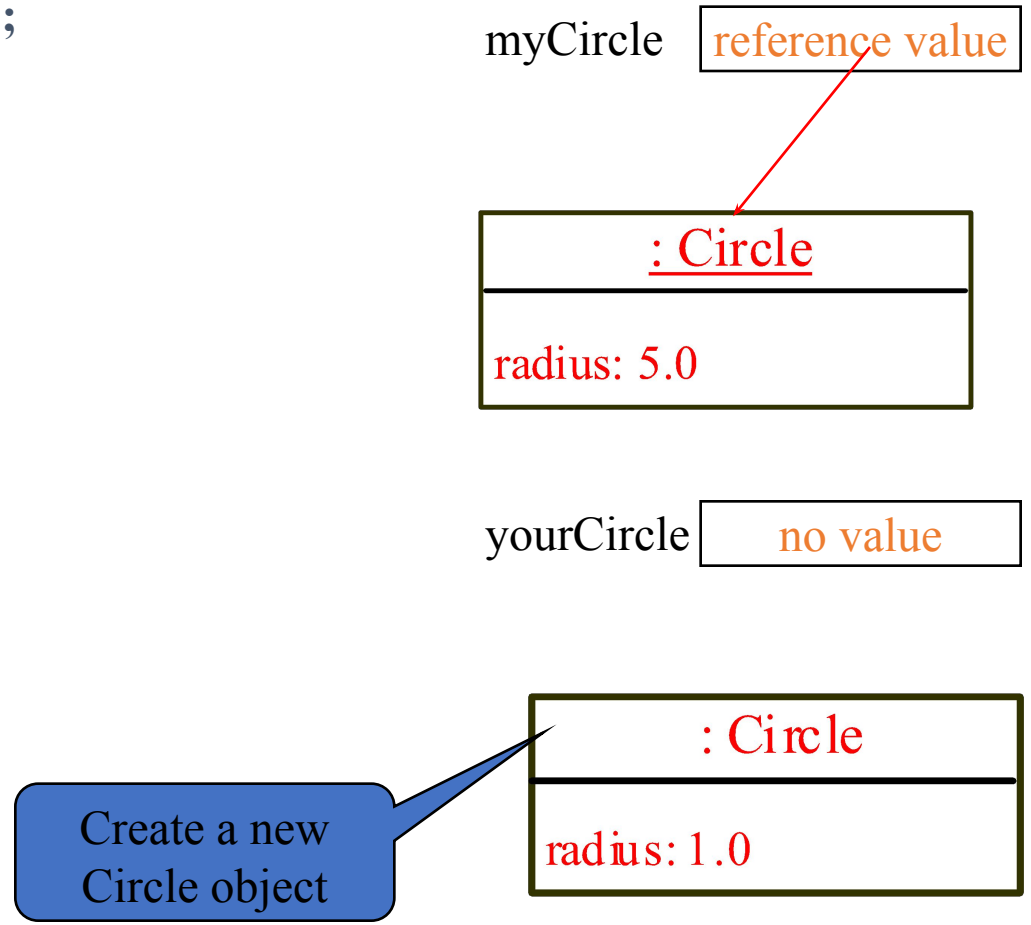
Trace Code – Circle class example (4)

```
Circle myCircle = new Circle(5.0);

Circle yourCircle = new Circle();

yourCircle.radius = 100;
```



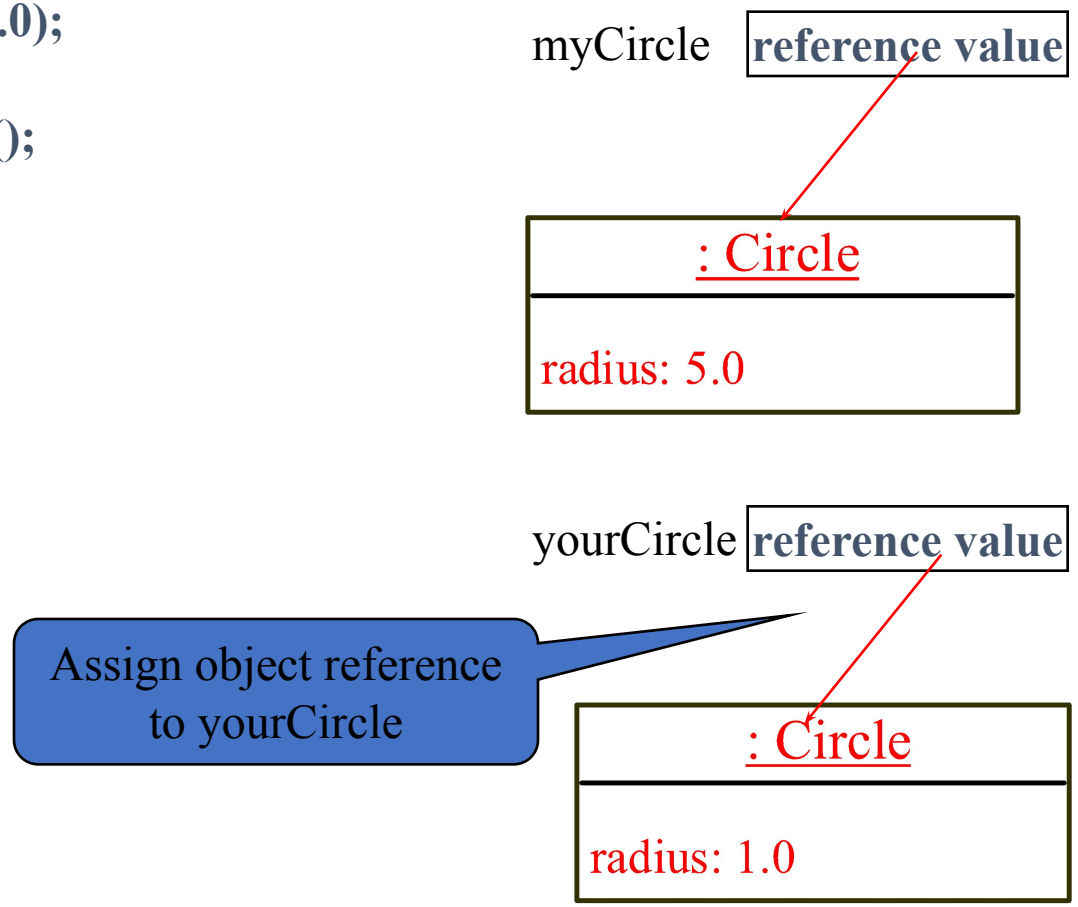
Trace Code – Circle class example (5)

```
Circle myCircle = new Circle(5.0);

Circle yourCircle = new Circle();

yourCircle.radius = 100;
```



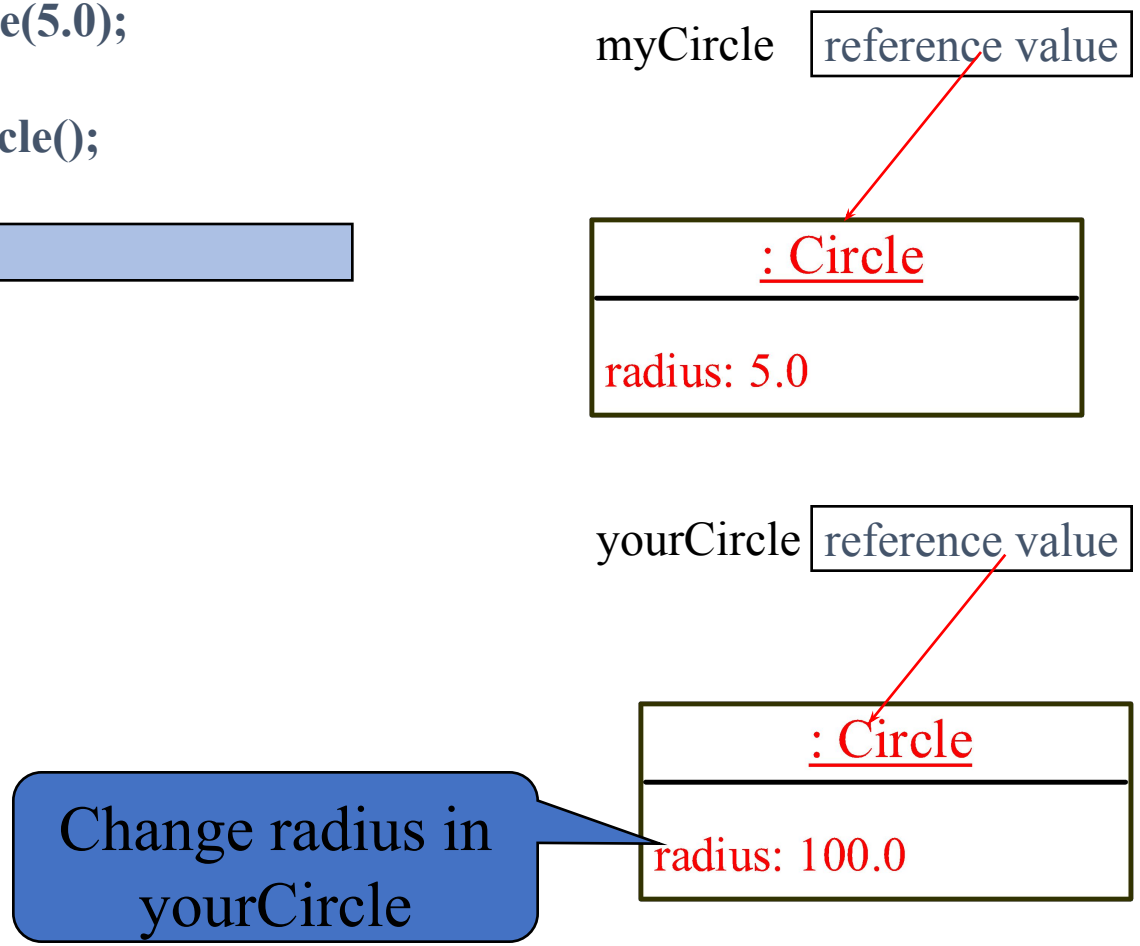
Trace Code – Circle class example (6)

```
Circle myCircle = new Circle(5.0);

Circle yourCircle = new Circle();

yourCircle.radius = 100;
```



Trace Code – Circle class example (7)

```
Circle myCircle = new Circle(5.0);

Circle yourCircle = new Circle();

yourCircle.radius = 100;
```



Reference Data Fields

- The data fields can be of reference types
- For example, the following Student class contains a data field *name* of the String type

```
public class Student {
  String name; // name has default value null
  int age; // age has default value 0
  boolean isScienceMajor; // isScienceMajor has default value false
  char gender; // c has default value '\u0000'
}
```

The null Value

- If a data field of a reference type does not reference any object, the data field holds a special literal value, null

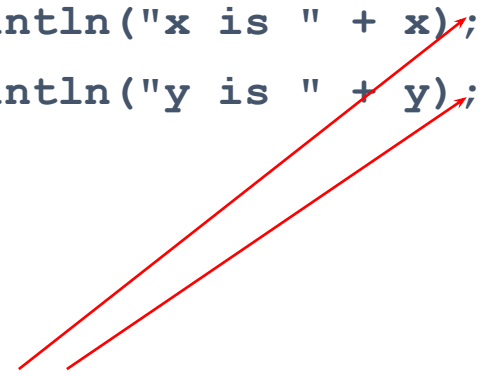
Default Value for a Data Field (1)

- The default value of a data field is:
  - null for a reference type
  - 0 for a numeric type
  - false for a boolean type
  - '\u0000' for a char type.

Default Value for a Data Field (2)

- Java assigns no default value to a local variable inside a method.

```
public class Test {
  public static void main(String[] args) {
    int x; // x has no default value
    String y; // y has no default value
    System.out.println("x is " + x);
    System.out.println("y is " + y);
  }
}
```



Compile error: variable not initialized

Exercise

- Write a program that contains two classes a Main class and a Student class:
  - The Student class has three data fields which are: name, GPA, and SID. It has a constructor that takes SID as a parameter.
  - The Main class is used to declare objects from the Student class.
  - The Main method should take student's data from the console, store them in an object of type Student then display student's data in the console.