RAMAIAH
Institute of Technology

M.S. Ramaiah Institute of Technology
(Autonomous Institute, Affiliated to VTU)
Department of Computer Science and Engineering

Course Name: Object Oriented Programming
Course Code: CS36
Credits: 3:0:0

Term: September – December 2020

Faculty:
Dr. Geetha J
Hanumantha Raju

Object in Java

Object is the physical as well as logical entity

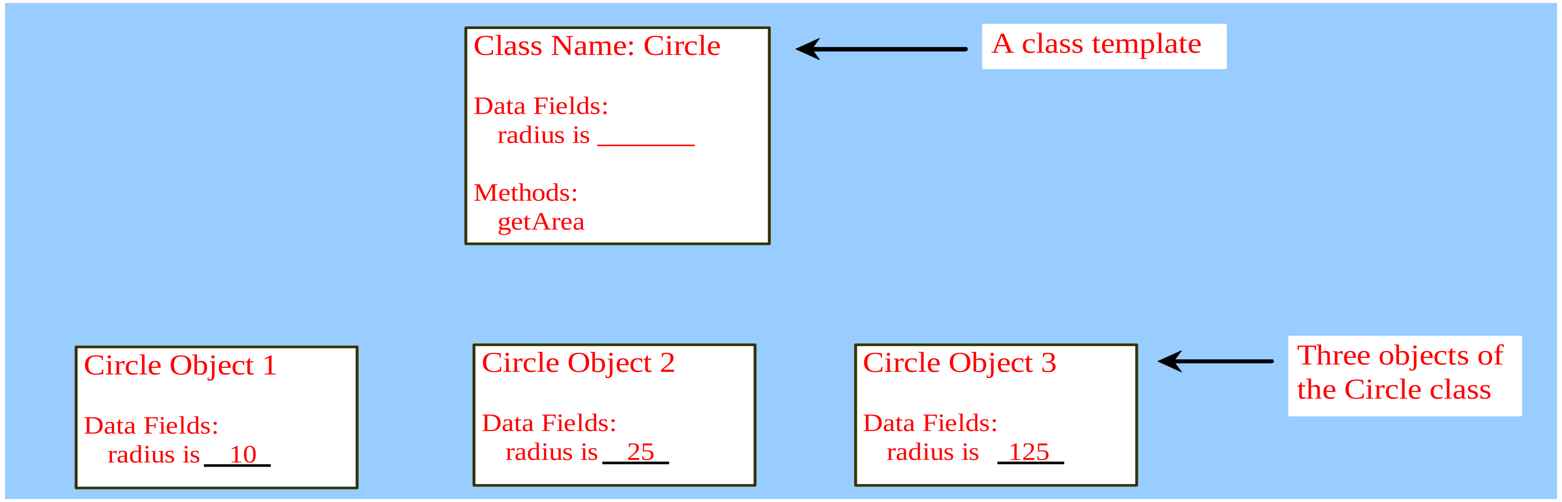
An entity that has state and behavior is known as an object e marker, pen, table, car etc. It can be physical or logical (tengil intengible). The example of integible object is banking systen

An object has three characteristics:

state: represents data (value) of an object.
behavior: represents the behavior (functionality) of an object such as deposit, withdraw etc.

identity: Object identity is typically implemented via a unique ID. The value of the ID is not visible to the external user. But, it is used internally by the JVM to identify each object uniquely.
For Example: Pen is an object. Its name is Reynolds, color is white etc. known as its state. It is used to write, so writing is its behavior.

Objects



Class Name: Circle

Data Fields:
radius is _______

Methods:
getArea

A class template

Circle Object 1

Data Fields:
radius is __10_

Circle Object 2

Data Fields:
radius is __25_

Circle Object 3

Data Fields:
radius is __125_

Three objects of the Circle class

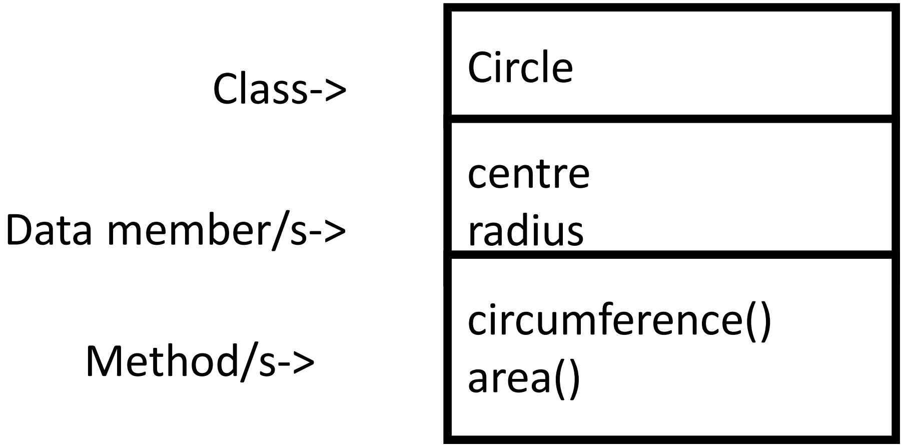
An object has both a state and behavior. The state defines the object, and the behavior defines what the object does

Class in Java

Object is an instance of a class. Class is a template or blueprint from which objects are created. So object is the instance(result) of a class.

A **class** is a group of objects that has common properties. It is a template or blueprint from which objects are created. A class in java can contain:

- **data member**
- **method**
- **constructor**
- **block**
- **class and interface**



Class->

Data member/s->

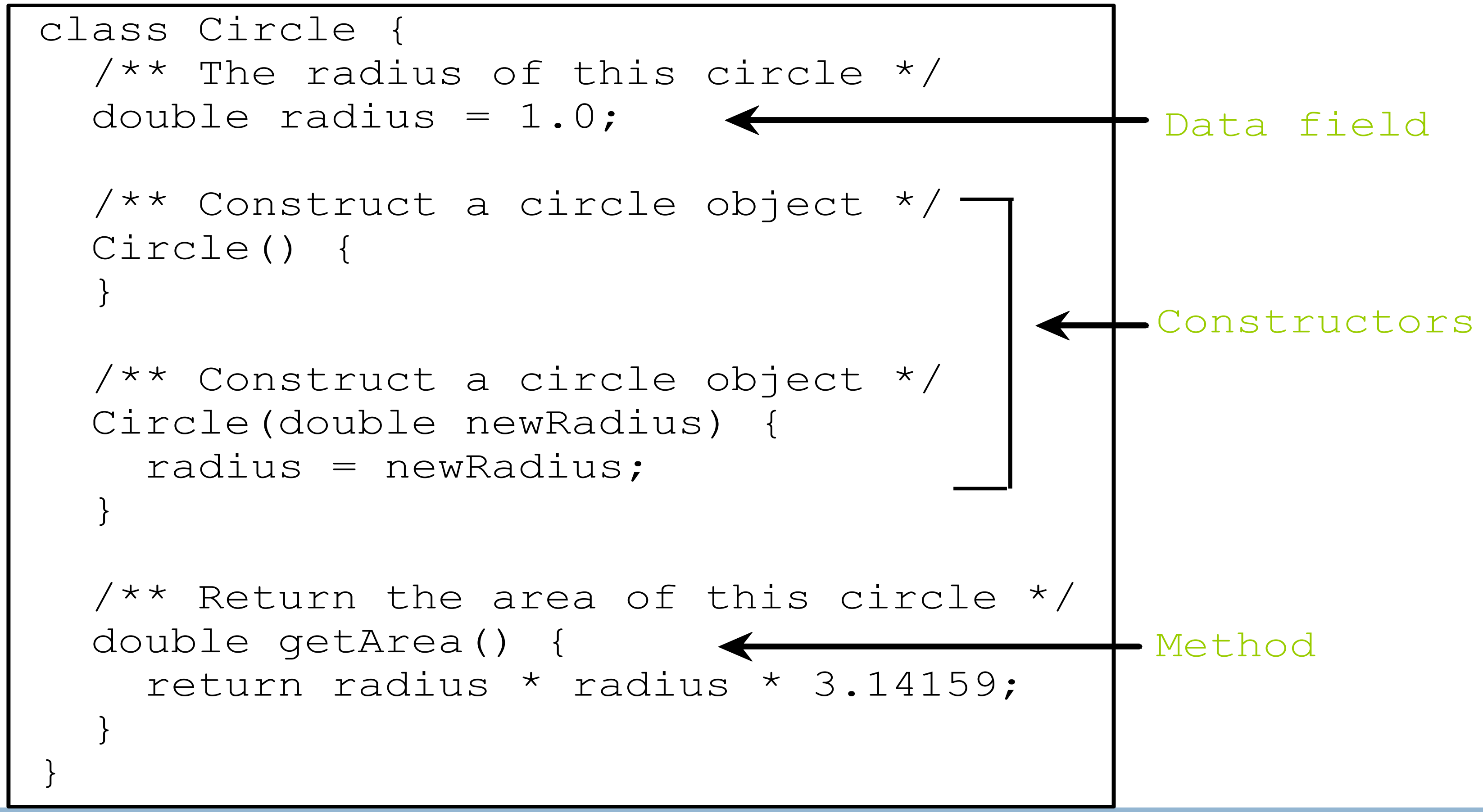
Method/s->

Circle

centre
radius

circumference()
area()

Classes



```
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;
  }
}
```

Methods in Java

What is System.out.println? It is a method: a collection of statements that performs a sequence of operations to display a message on the console. It can be used even without fully understanding the details of how it works. It is used by invoking a statement with a string argument. The string argument is enclosed within parentheses. In this case, the argument is "Welcome to Java!" You can call the same println method with a different argument to print a different message.

Adding Methods to Class Circle

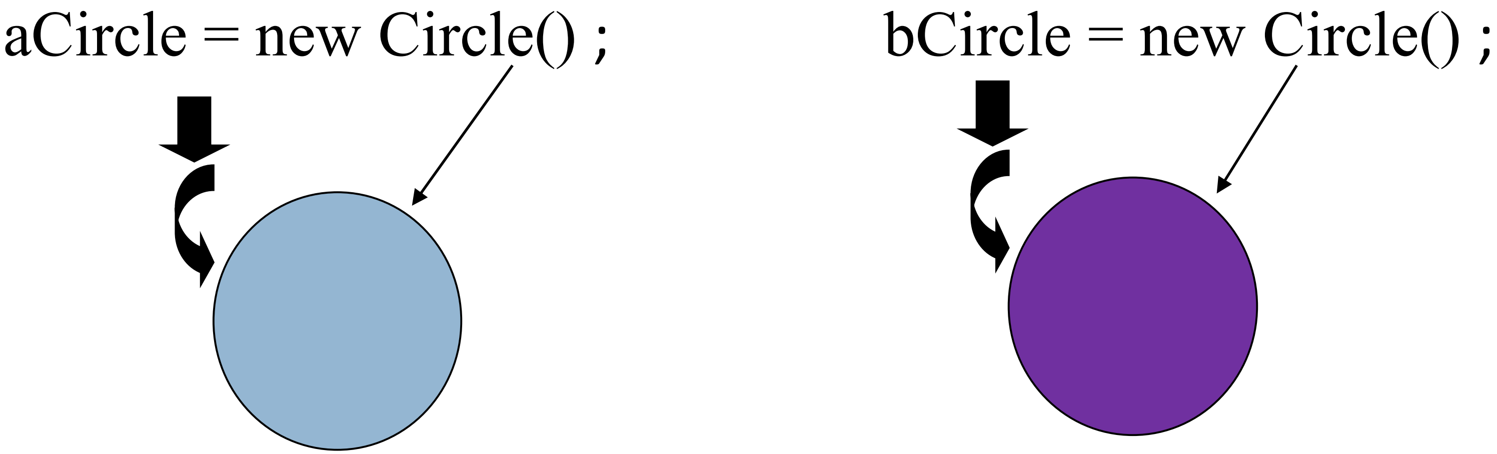
```
public class Circle {

    public double x, y; // centre of the circle
    public double r;    // radius of circle

     //Methods to return circumference and area
     public double circumference() {
              return 2*3.14*r;
     }
     public double area() {
              return 3.14 * r * r;
    }
}
```

Creating objects of a class

```
aCircle = new Circle();
bCircle = new Circle() ;
```

Objects are created dynamically using the *new* keyword.

aCircle and bCircle refer to Circle objects.

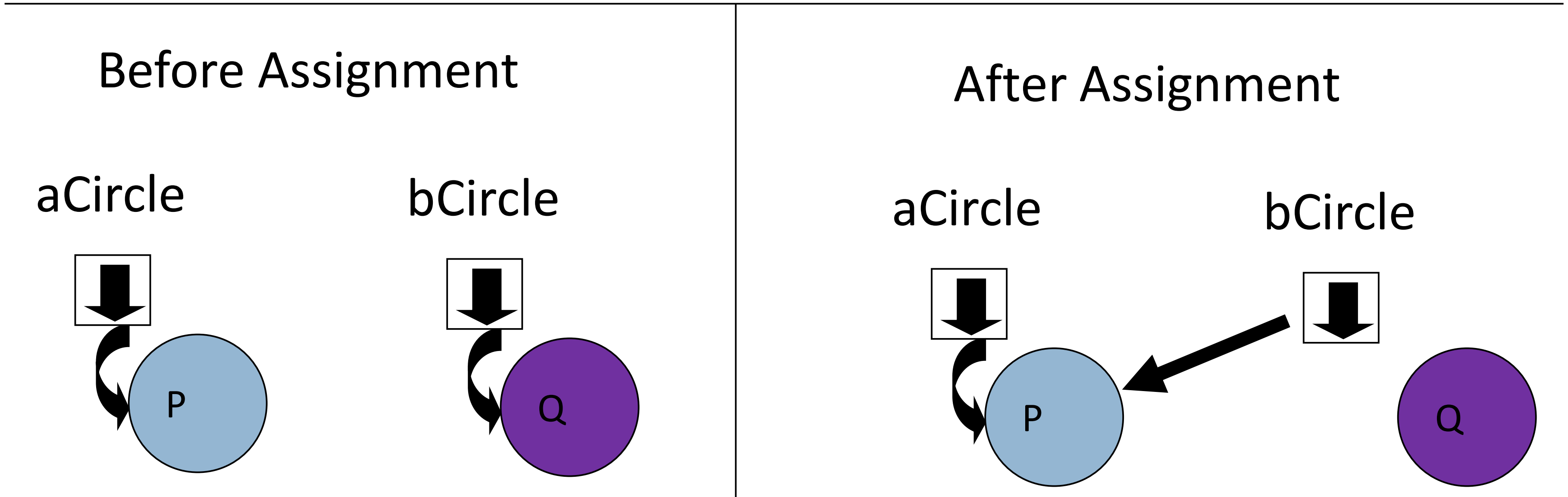


aCircle = new Circle() ;

bCircle = new Circle() ;

Creating objects of a class

```
aCircle = new Circle();
bCircle = new Circle() ;

bCircle = aCircle;
```



Before Assignment

aCircle

bCircle

P

Q

After Assignment

aCircle

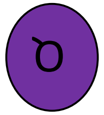
bCircle

P

Q

RAMAIAH
Institute of Technology

Automatic garbage collection



The object does not have a reference and cannot be used in future.

The object becomes a candidate for automatic garbage collection.

Java automatically collects garbage periodically and releases the memory used to be used in the future.

Accessing Object/Circle Data

- Similar to C syntax for accessing data defined in a structure.

```
Circle aCircle = new Circle();
aCircle.x = 2.0; // initialize center and radius
aCircle.y = 2.0;
aCircle.r = 1.0;
```

- Executing Methods in Object/Circle

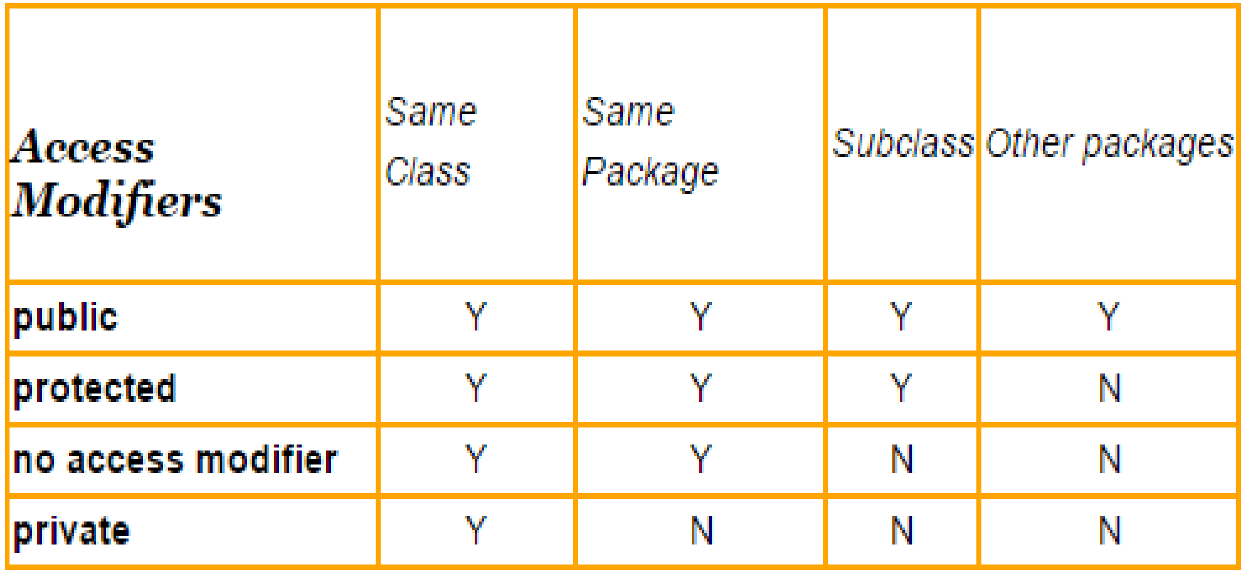
```
double area;
aCircle.r = 1.0;
area = aCircle.area();
```

Using Circle Class

```
// Circle.java:  Contains both Circle class and its user class

//Add Circle class code here

class MyMain{

    public static void main(String args[])    {

        Circle aCircle;  // creating reference

        aCircle = new Circle(); // creating object

        aCircle.x = 10;  // assigning value to data field

        aCircle.y = 20;

        aCircle.r = 5;

        double area = aCircle.area(); // invoking method

        double circumf = aCircle.circumference();

        System.out.println("Radius="+aCircle.r+" Area="+area);

        System.out.println("Radius="+aCircle.r+" Circumference ="+circumf);

    } }
```

Access Modifiers In Java

Access modifiers specifies who can access them. There are four access modifiers used in java. They are public, private, protected, no modifier (declaring without an access modifier).



Access Modifiers	*Same Class*	*Same Package*	*Subclass*	*Other packages*
public	Y	Y	Y	Y
protected	Y	Y	Y	N
no access modifier	Y	Y	N	N
private	Y	N	N	N

Variables

1) Instance variables

Instance variables are variables that are declare inside a class but outside any method, constructor or block.

```
class Student { String name; int age; } // name, age are instant variables
```

2) Static variables

Static are class variables declared with static keyword. Static variables are initialized only once and by default it initialized to 0. Static variables are also used in declaring constant along with final keyword.

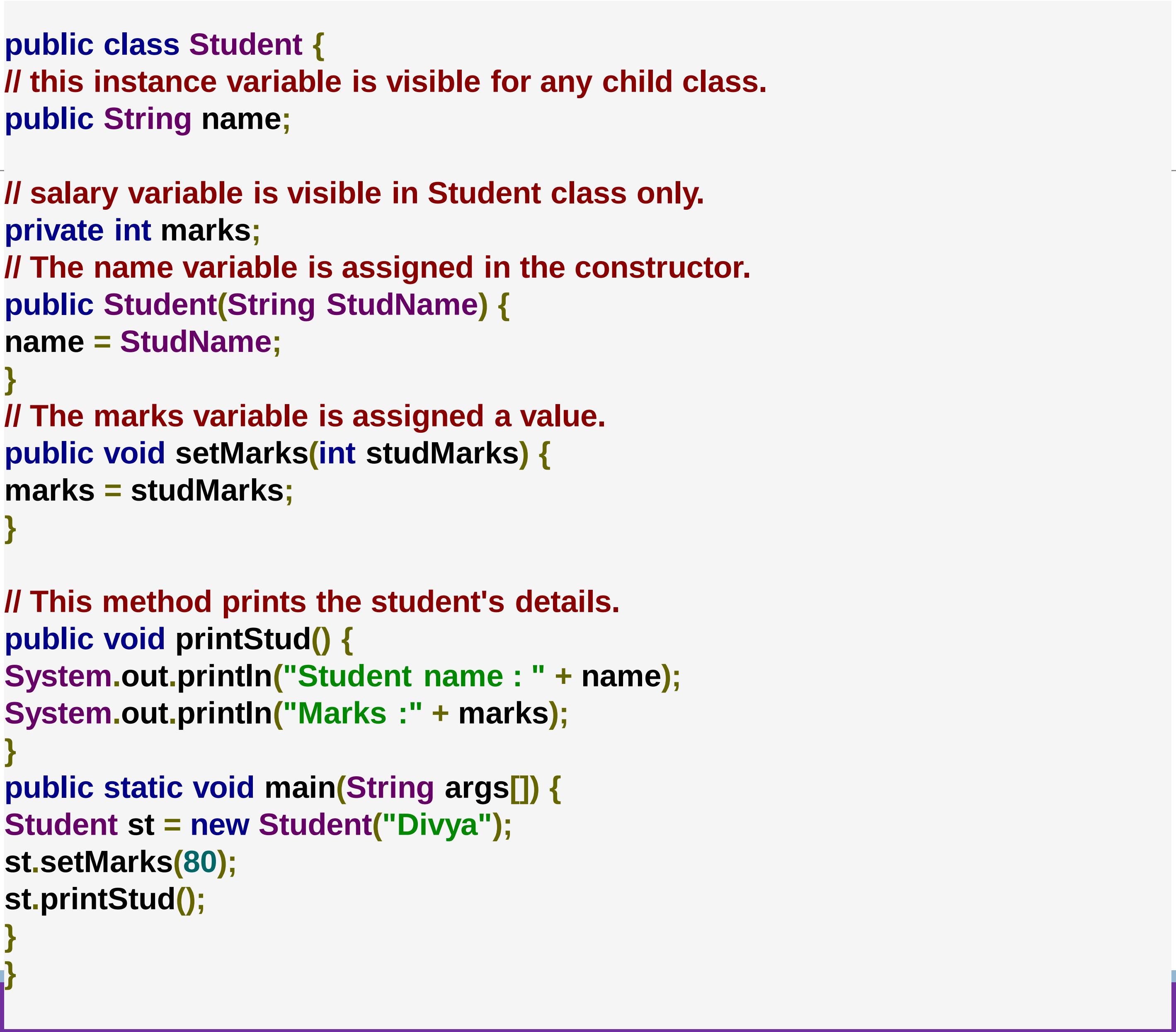
```
class Student { String name; int age; static int instituteCode=101; }
```

3) Local variables

Local variables are declared in method constructor or blocks. Local variables are initialized when method or constructor block start and will be destroyed once its end. Local variable reside in stack. Access modifiers are not used for local variable.

```
float getDiscount(int price) {

 float discount; discount=price*(20/100); return discount;

} //here discount is a local variable
```

Example -1(Instance Variable)



```
public class Student {
// this instance variable is visible for any child class.
public String name;

// salary variable is visible in Student class only.
private int marks;
// The name variable is assigned in the constructor.
public Student(String StudName) {
name = StudName;
}
// The marks variable is assigned a value.
public void setMarks(int studMarks) {
marks = studMarks;
}

// This method prints the student's details.
public void printStud() {
System.out.println("Student name : " + name);
System.out.println("Marks :" + marks);
}
public static void main(String args[]) {
Student st = new Student("Divya");
st.setMarks(80);
st.printStud();
}
}
```

```
public class College {
 // salary variable is a private static variable
        private static int resultPercent;

        // DEPARTMENT is a constant
        public static final String BRANCH =
"Computer Science";

        public static void main(String args[]) {
                resultPercent = 76;
                System.out.println(BRANCH + "
Result %:" + resultPercent);
        }
}
```

```
class PO {
      private static int POCount;
      public static void main(String [] s  )
     {
        PO  po1 = new   PO();
        po1.updatePOCount();
       po1.updatePOCount();
System.out.println(POCount);
       }

      public void updatePOCount() {
      POCount++;  }
}

Output:
2
```

Static Key word with methods

```
class Math {
 public static double sqrt(double x)  {
          // calculate
          return  result;
  }
}
class MyApp {
 public static void main(String [] s ) {
          double dd;
          dd = Math.sqrt(7.11);
 }
}
```

Java static method

if you apply static keyword with any method, it is known as static method.

A static method belongs to the class rather than object of a class.

A static method can be invoked without the need for creating an instance of a class.

static method can access static data member and can change the value of it.

Example

```
//Program of changing the common property of all objects(static field).

class Student9{
    int rollno;
    String name;
    static String college = ”RVCE";

    static void change(){
    college = ”MSRIT";
    }

    Student9(int r, String n){
    rollno = r;
    name = n;
    }

    void display (){System.out.println(rollno+" "+name+" "+college);}

   public static void main(String args[]){
   Student9.change();

   Student9 s1 = new Student9 (111,“Kiran");
   Student9 s2 = new Student9 (222,"Aryan");
   Student9 s3 = new Student9 (333,“Sneha");

   s1.display();
   s2.display();
   s3.display();
   }
}
```

Scope of Variables

As in C/C++, scope is determined by the placement of curly braces {}.

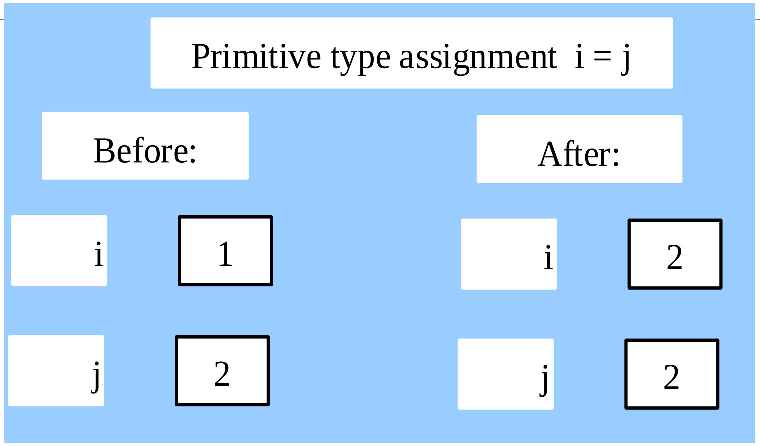
The scope of instance and static variables is the entire class. They can be declared anywhere inside a class.

The scope of a local variable starts from its declaration and continues to the end of the block that contains the variable. A local variable must be initialized explicitly before it can be used.

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

```
{
 int x = 12;
   /* only x available */
      { int q = 96;
         /* both x and q availab
*/
      }
      /* only x available */
      /* q “out of scope” */
    }
```

Copying Variables of Primitive Data Types and Object Types



Primitive type assignment i = j

Before:

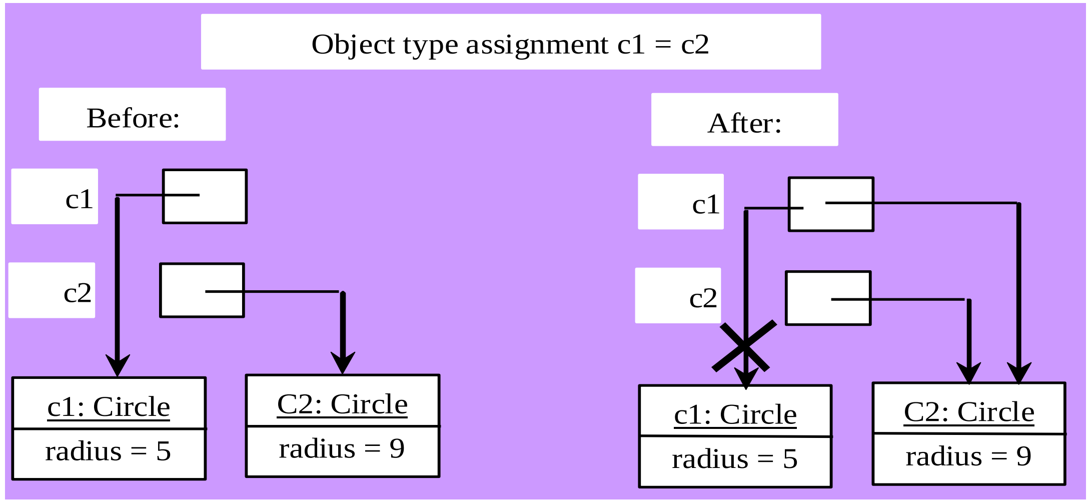
i 1

j 2

After:

i 2

j 2



Object type assignment c1 = c2

Before:

c1

c2

c1: Circle

radius = 5

C2: Circle

radius = 9

After:

c1

c2

c1: Circle

radius = 5

C2: Circle

radius = 9

Garbage Collection

As shown in the previous figure, after the assignment statement c1 = c2, c1 points to the same object referenced by c2. The object previously referenced by c1 is no longer referenced. This object is known as garbage. Garbage is automatically collected by JVM.

Exercises

Write a Java class Student to meet the following specification.

- The class should be able to support a 5 digit student ID, student name, marks for 3 subjects. You should have methods to set and get each of the attributes, and calculate the average for the student. Write a tester program to test your class. You should create 2 or 3 students and write code to test the class. Aim - Understand how to define a class and create objects of the class.

For the above student Class display total number of students you have entered using static variable.

Write a java program to get cube of a given number by static method.

Thank you