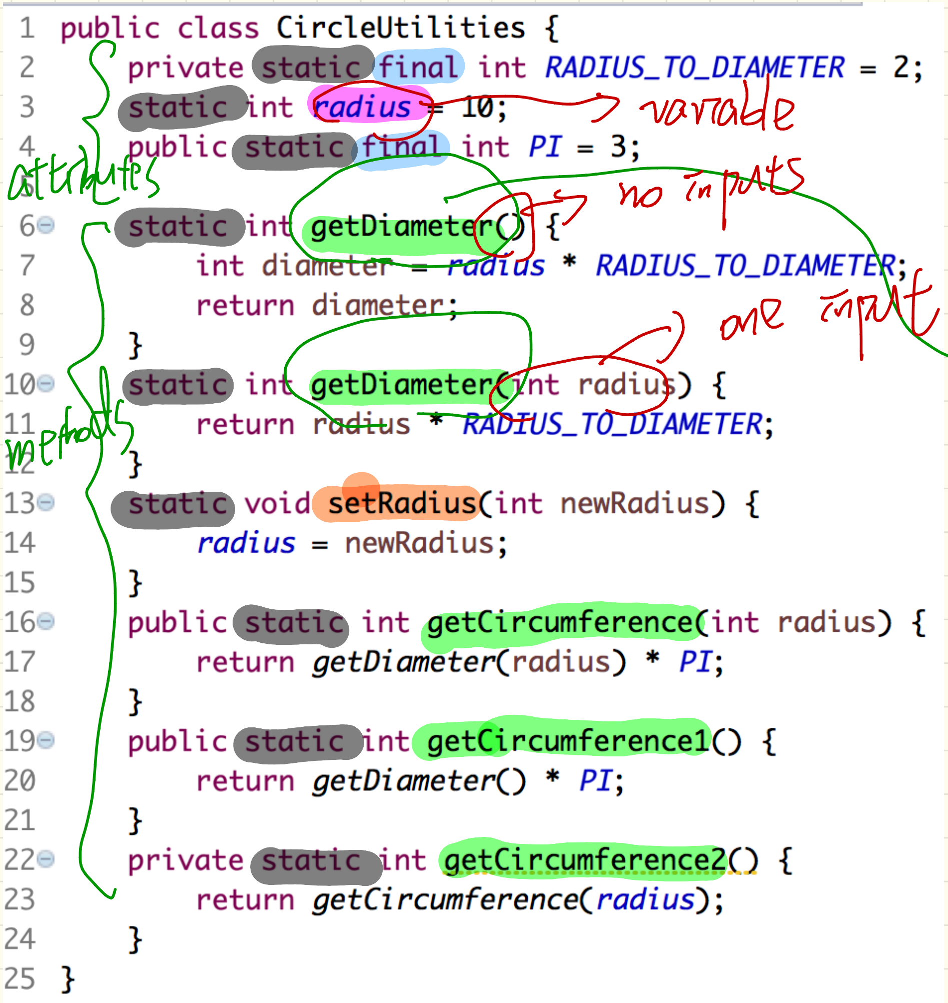
Tuesday September 12

Lecture 2



```
public class CircleUtilities {
    private static final int RADIUS_TO_DIAMETER = 2;
    static int radius = 10;
    public static final int PI = 3;

    static int getDiameter() {
        int diameter = radius * RADIUS_TO_DIAMETER;
        return diameter;
    }
    static int getDiameter(int radius) {
        return radius * RADIUS_TO_DIAMETER;
    }
    static void setRadius(int newRadius) {
        radius = newRadius;
    }
    public static int getCircumference(int radius) {
        return getDiameter(radius) * PI;
    }
    public static int getCircumference1() {
        return getDiameter() * PI;
    }
    private static int getCircumference2() {
        return getCircumference(radius);
    }
}
```

attributes

variable

no inputs

one input

methods

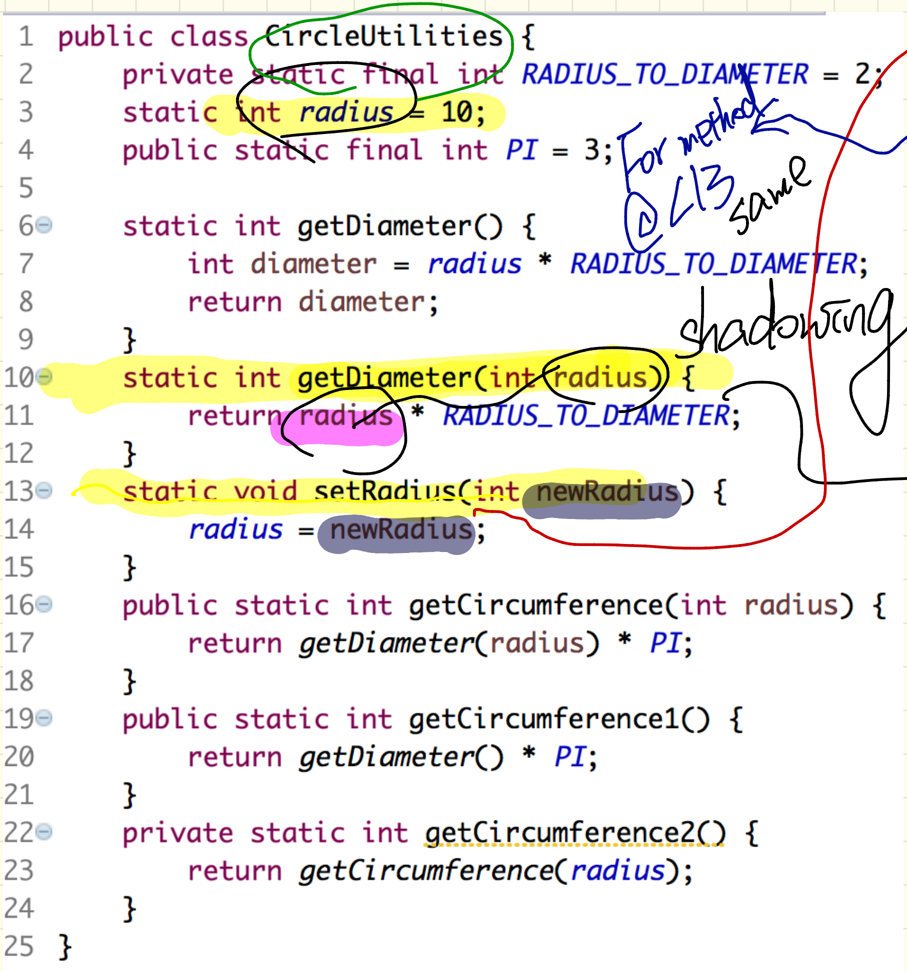
methods :

accessor : return
non-void

mutator : return
void

We overload the method getDiame.
with different parameter lists

input



```java
1  public class CircleUtilities {
2      private static final int RADIUS_TO_DIAMETER = 2;
3      static int radius = 10;
4      public static final int PI = 3;
5
6      static int getDiameter() {
7          int diameter = radius * RADIUS_TO_DIAMETER;
8          return diameter;
9      }
10     static int getDiameter(int radius) {
11         return radius * RADIUS_TO_DIAMETER;
12     }
13     static void setRadius(int newRadius) {
14         radius = newRadius;
15     }
16     public static int getCircumference(int radius) {
17         return getDiameter(radius) * PI;
18     }
19     public static int getCircumference1() {
20         return getDiameter() * PI;
21     }
22     private static int getCircumference2() {
23         return getCircumference(radius);
24     }
25 }
```

For method @L13 same

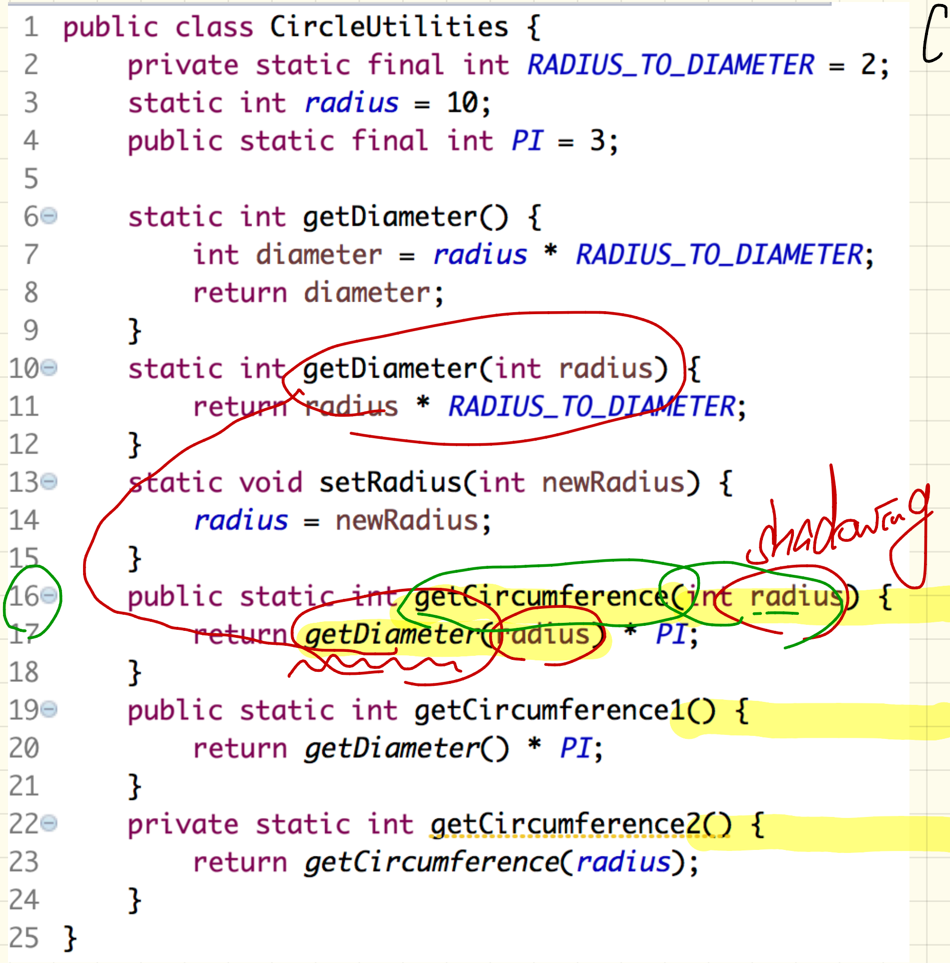
For method @L10 (shadowing) different

① radius

② CircleUtilities radius

shadowing

due to shadowing, if you want to refer to radius @ L3 ⇒ CircleUtilities . radius



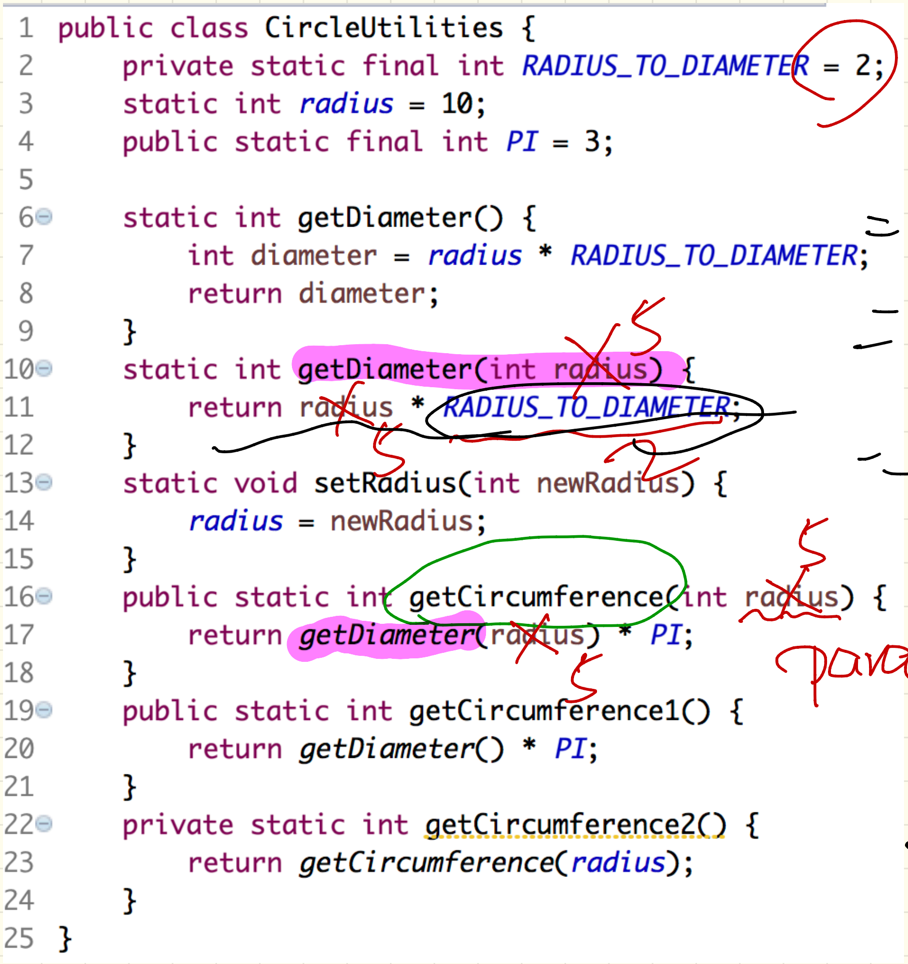
```java
public class CircleUtilities {
    private static final int RADIUS_TO_DIAMETER = 2;
    static int radius = 10;
    public static final int PI = 3;

    static int getDiameter() {
        int diameter = radius * RADIUS_TO_DIAMETER;
        return diameter;
    }
    static int getDiameter(int radius) {
        return radius * RADIUS_TO_DIAMETER;
    }
    static void setRadius(int newRadius) {
        radius = newRadius;
    }
    public static int getCircumference(int radius) {
        return getDiameter(radius) * PI;
    }
    public static int getCircumference1() {
        return getDiameter() * PI;
    }
    private static int getCircumference2() {
        return getCircumference(radius);
    }
}
```

shadowing

CU. getCircumference(5)

helper methods

a method is a block of code which can be reused by referring to its name.



```java
public class CircleUtilities {
    private static final int RADIUS_TO_DIAMETER = 2;
    static int radius = 10;
    public static final int PI = 3;

    static int getDiameter() {
        int diameter = radius * RADIUS_TO_DIAMETER;
        return diameter;
    }
    static int getDiameter(int radius) {
        return radius * RADIUS_TO_DIAMETER;
    }
    static void setRadius(int newRadius) {
        radius = newRadius;
    }
    public static int getCircumference(int radius) {
        return getDiameter(radius) * PI;
    }
    public static int getCircumference1() {
        return getDiameter() * PI;
    }
    private static int getCircumference2() {
        return getCircumference(radius);
    }
}
```

5

5 2

5

5

parameter

CU. getCircumference(5)

= CU. getDiameter(5) * 3

= 5 * 2 * 3

= 30.

argument value for replacing parameter radius

parameter

fac(x) = x * (x-1) * ...

fac(5) fac(4) ... 1

arguments

modifiers -

1. Visibility { Project / package / class   public / private

variable

?

you must access this using its class name.

2. Constant/Variable { double myPI = 3.14; ✓   myPI = 6.28;

final double PI = 3.14;

~~PI = 6.28;~~   it's constant

final static int FOO = 2

~~FOO = 4;~~

```
class MyMath {
    public static int Foo = 3;

}
```

It's guaranteed only one copy of Foo will exist at runtime

We can only use the class name to access this attribute ∵ it's static

MyMath . Foo

MyMath	
Foo	3

```
class MyMathUser {
    ________ main( ____ ) {
        MyMath. Foo = 4;
    } }
```

```
class MyMath2 {
    public int foo = 4;
}
```

non-static attribute

obj1.foo

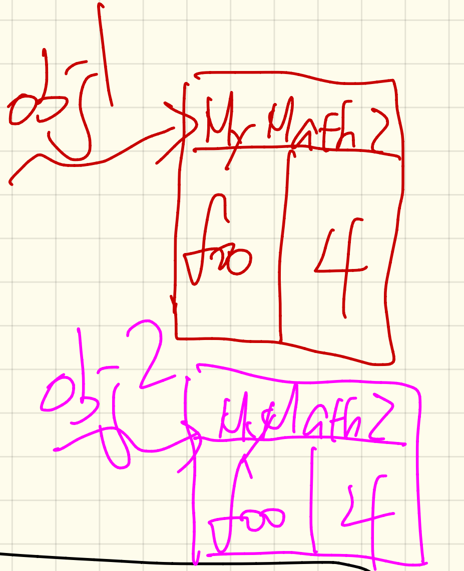
obj2.foo

~~MyMath2.foo~~

ambiguous

∵ (two copies of MyMath2

```
class MyMath2App {
    ________ main ( ) {
        MyMath2 obj1 = new MyMath2();
        MyMath2 obj2 = new MyMath2();
    }
}
```


obj1
MyMath2
foo
4
obj2
MyMath2
foo
4