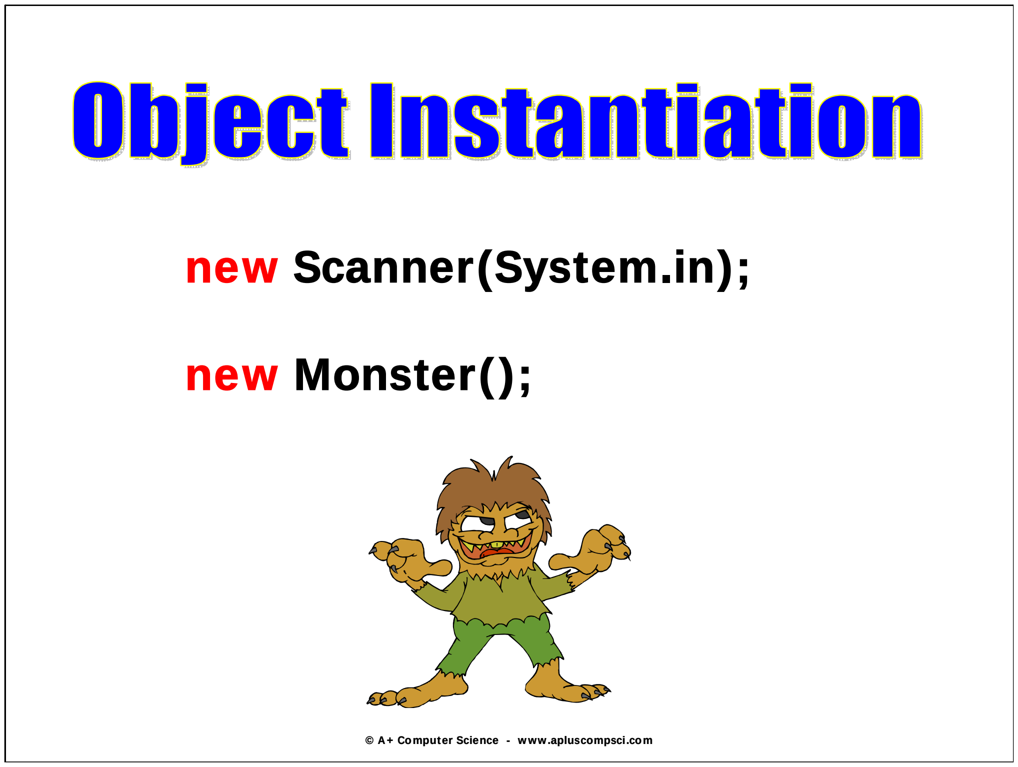
OOP return methods

Lab 03

© A+ Computer Science - www.apluscompsci.com

Objects

© A+ Computer Science - www.apluscompsci.com

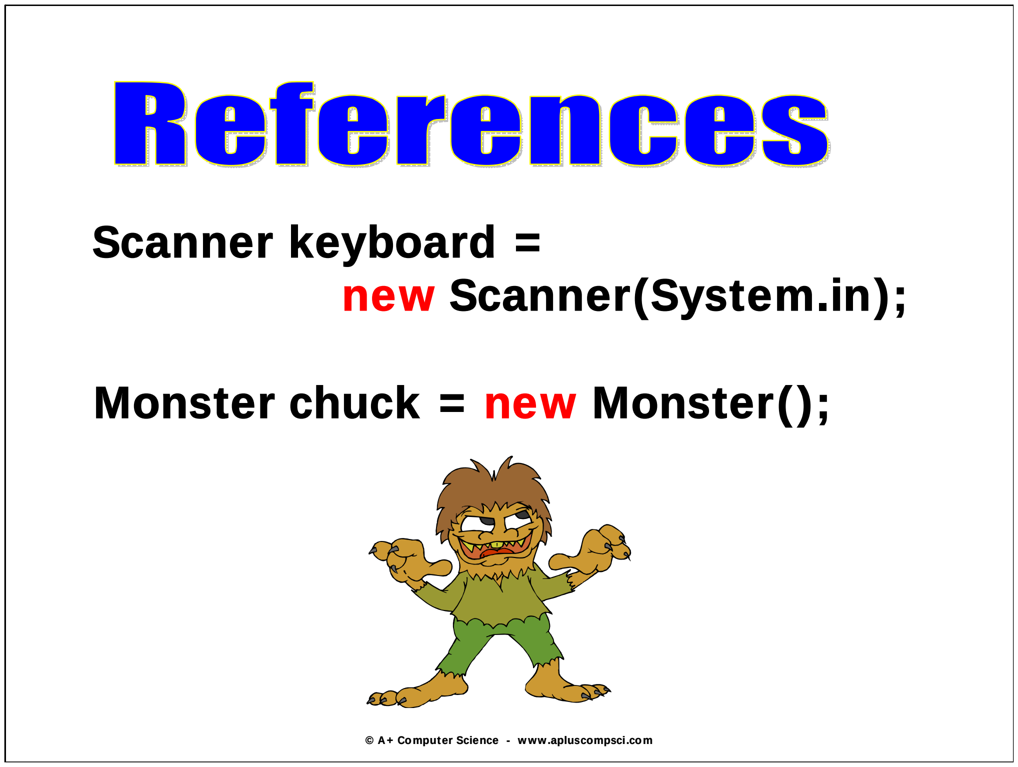


Object Instantiation

```
new Scanner(System.in);

new Monster();
```

© A+ Computer Science - www.apluscompsci.com

When the reserved word new is combined with a constructor call, a new Object is created in memory. This process of creating a new Object in memory is called instantiation.

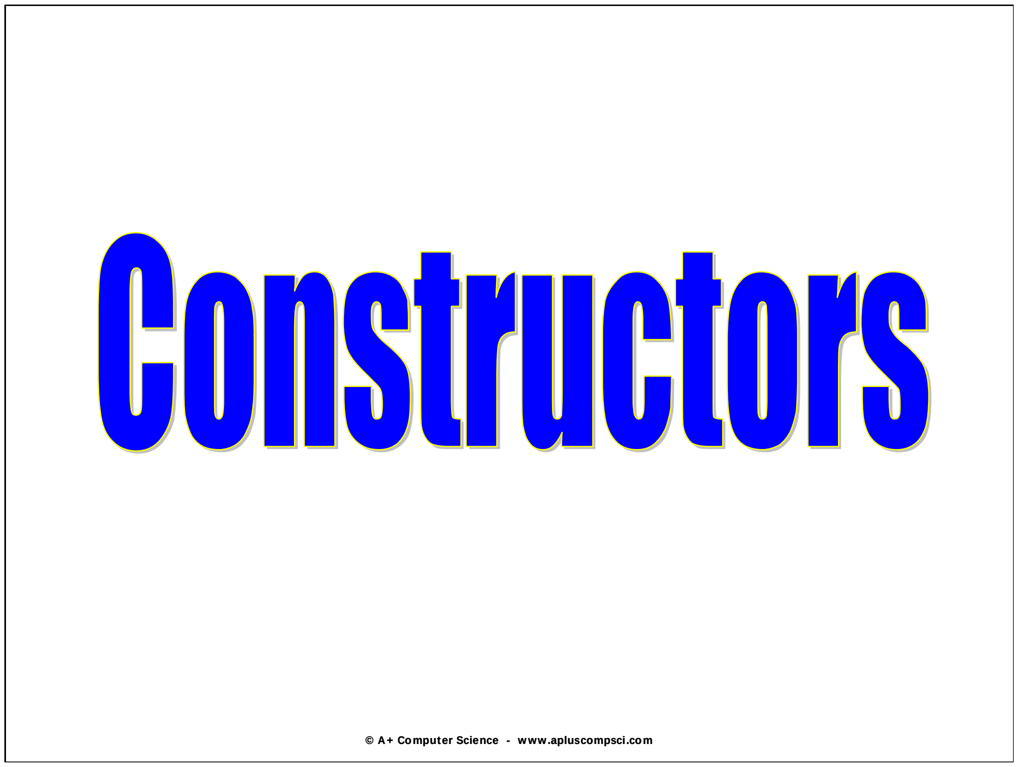


References

```
Scanner keyboard =
              new Scanner(System.in);

Monster chuck = new Monster();
```

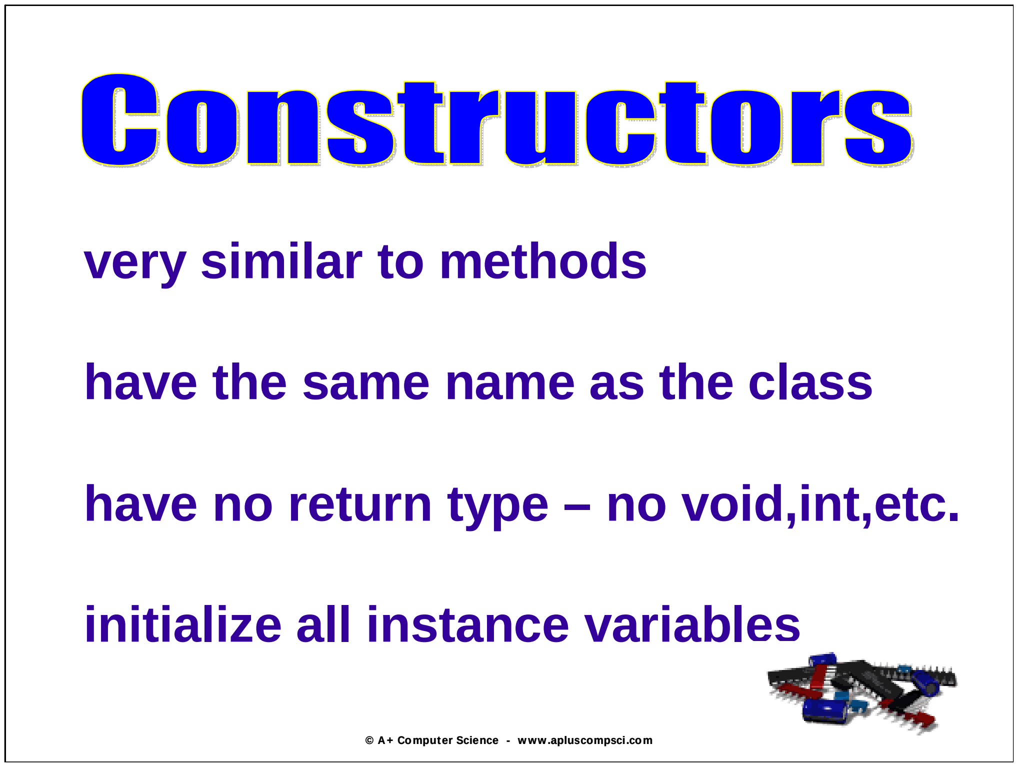
© A+ Computer Science - www.apluscompsci.com

Typically, a reference is used to store the location of the new Object. keyboard is a `Scanner` reference that is storing the location of the new `Scanner`.



Constructors

© A+ Computer Science - www.apluscompsci.com



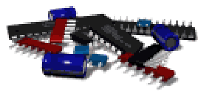
Constructors

very similar to methods

have the same name as the class

have no return type – no void,int,etc.

initialize all instance variables

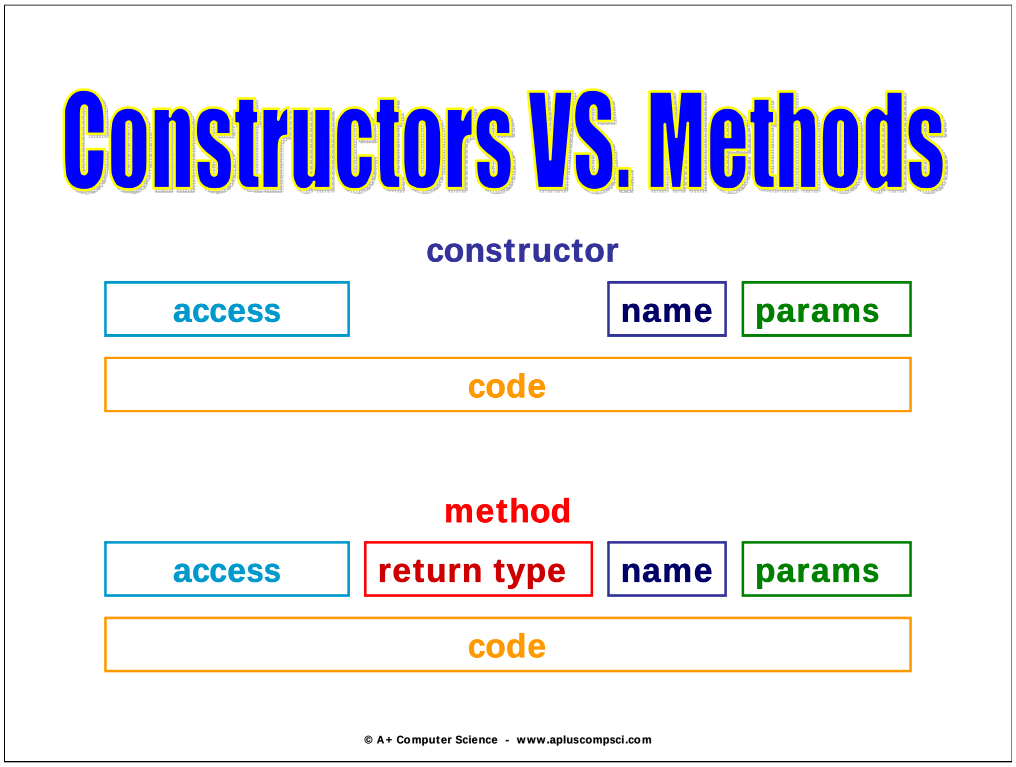


© A+ Computer Science - www.apluscompsci.com

Constructors are used to initialize all of the data. Typically, all variables are assigned a value in the constructor.

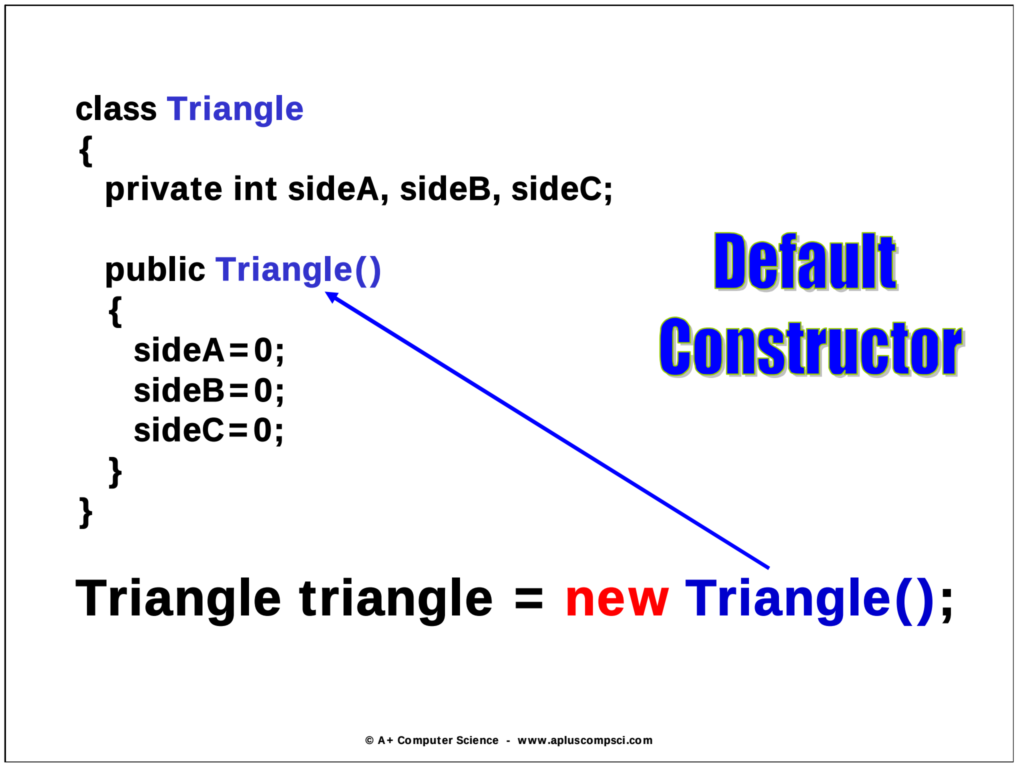
A constructor is very similar to a method, but it is technically not a method.

Constructors have no return type and are named the same as the class. This is slightly different from a method.



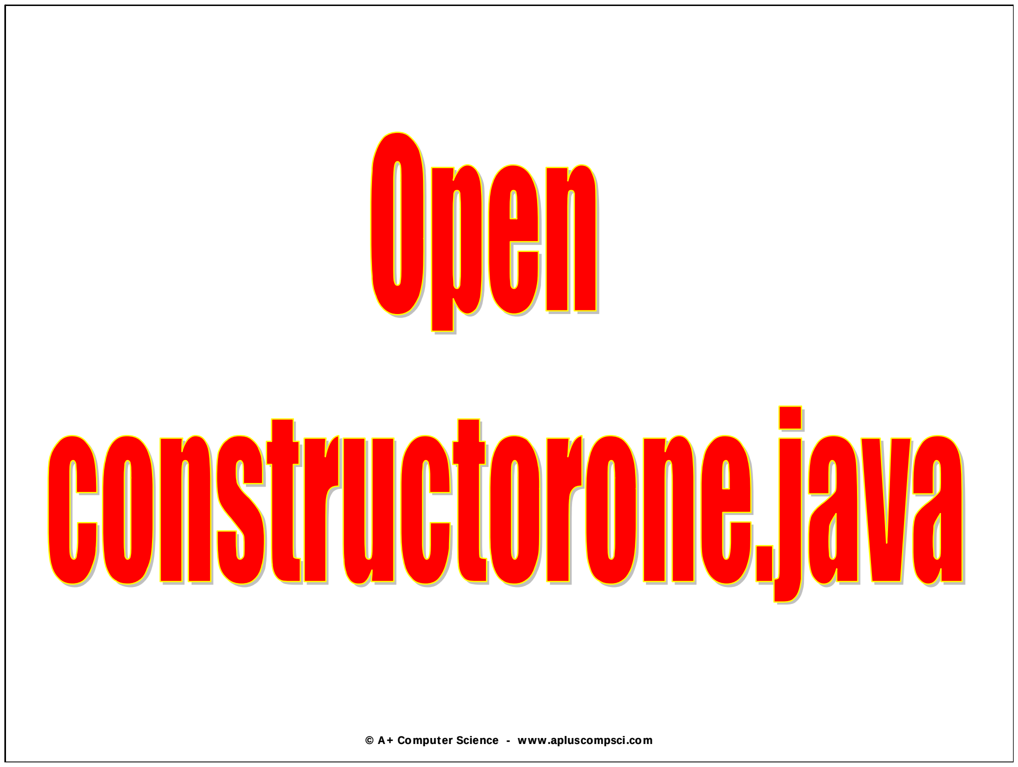
Constructors VS. Methods

Constructors and methods are more similar than different and sometimes constructors are referred to as constructor methods. Most of the differences are not visible. The only real visible differences exist in that the constructor has to be named the same name as the class and that the constructor has no return type. Methods and constructors both have a name, a list of parameters, and a block of code to execute. When you call a method or a constructor, the block of code associated with the name will be executed.



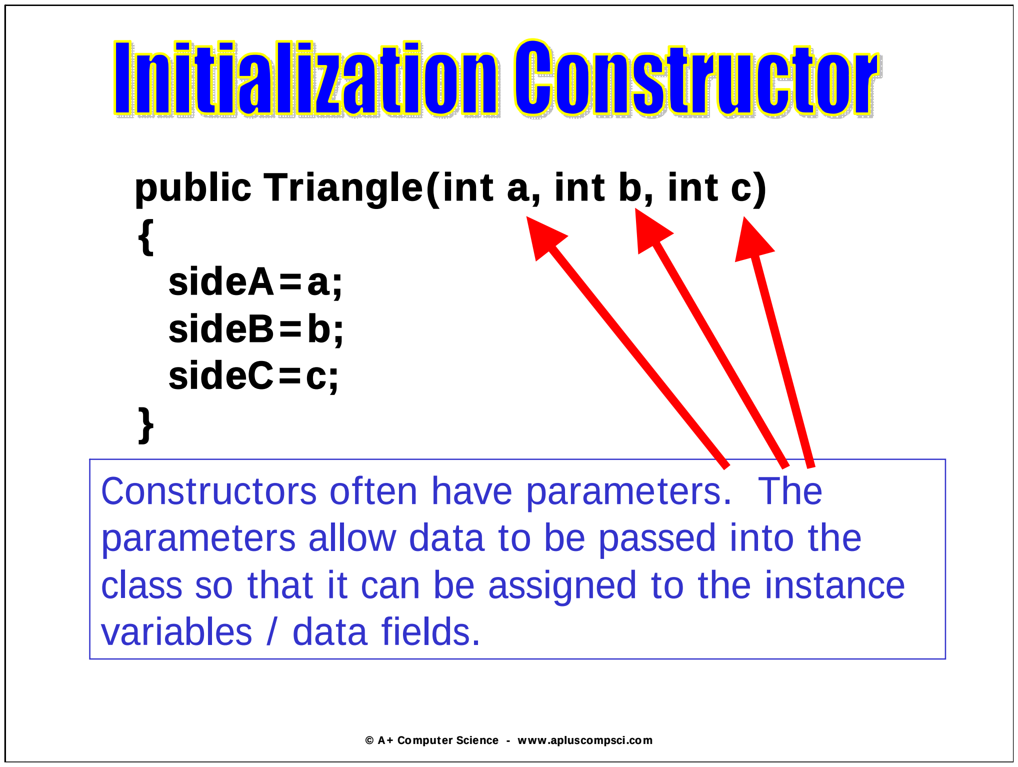
```
class Triangle
{
  private int sideA, sideB, sideC;

  public Triangle()
  {
    sideA=0;
    sideB=0;
    sideC=0;
  }
}

Triangle triangle = new Triangle();
```

Default constructors have no parameter list. When a default constructor is called, all instance variables / data fields are set to a zero value. If no constructors are provided for a class, Java will provide a default constructor that will initialize all data to a zero value.



Open constructorone.java

© A+ Computer Science - www.apluscompsci.com



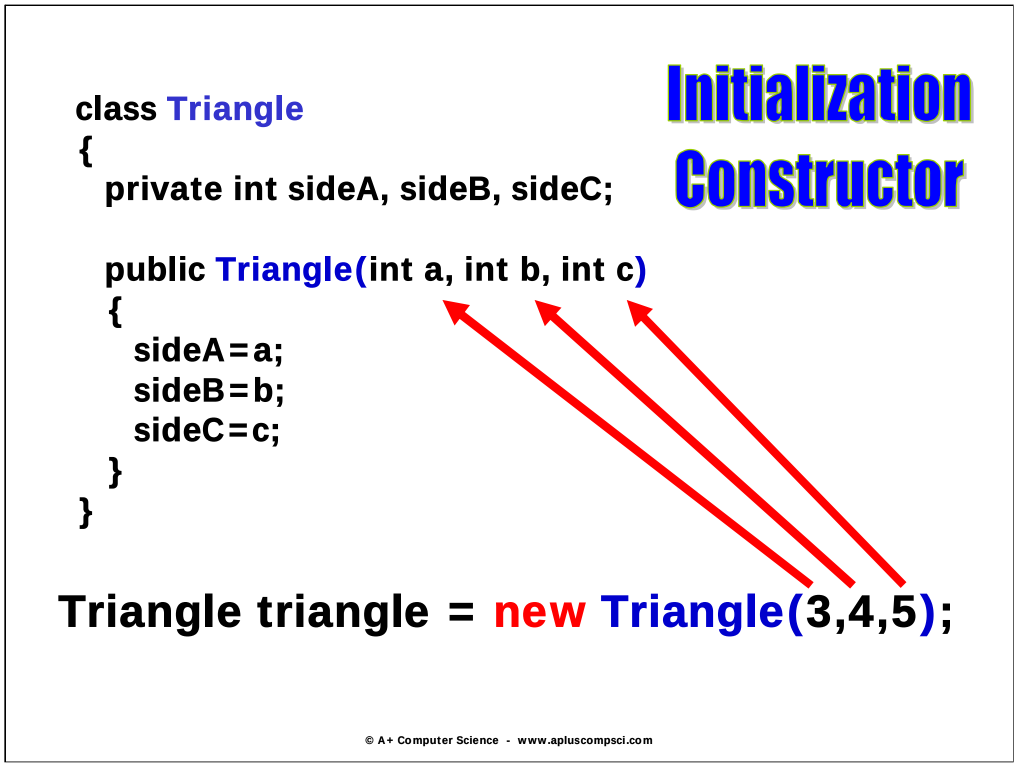
Initialization Constructor

```
public Triangle(int a, int b, int c)
{
  sideA=a;
  sideB=b;
  sideC=c;
}
```

Constructors often have parameters. The parameters allow data to be passed into the class so that it can be assigned to the instance variables / data fields.

© A+ Computer Science - www.apluscompsci.com

Initialization constructors have a parameter list and will receive parameters when called. The number of parameter and types passed in must match up with the parameter list following the method name.

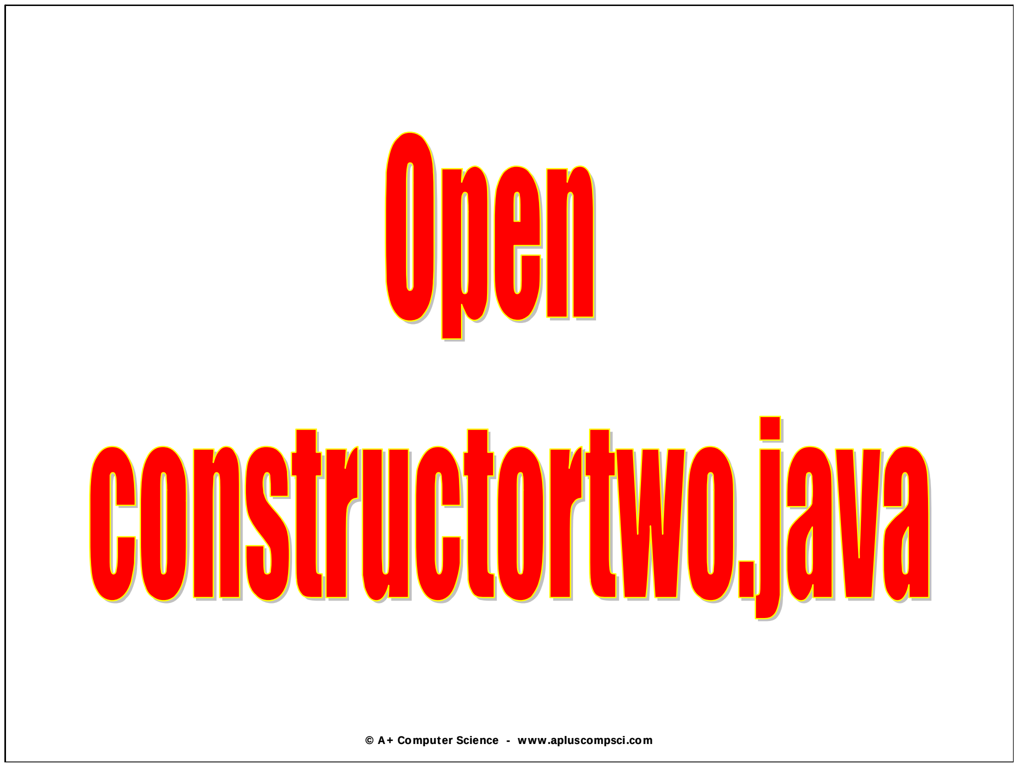


Initialization Constructor

```
class Triangle
{
  private int sideA, sideB, sideC;

  public Triangle(int a, int b, int c)
  {
    sideA=a;
    sideB=b;
    sideC=c;
  }
}

Triangle triangle = new Triangle(3,4,5);
```

© A+ Computer Science - www.apluscompsci.com

Initialization constructors have a parameter list and will receive parameters when called. The number of parameter and types passed in must match up with the parameter list following the method name.

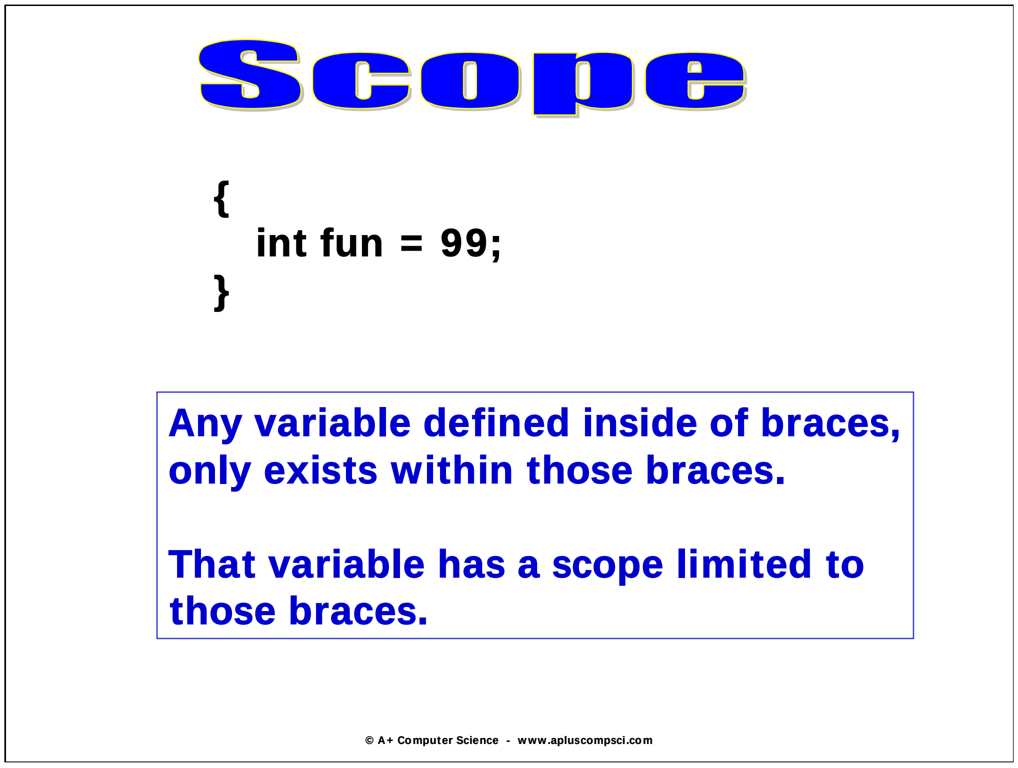


Open constructortwo.java

© A+ Computer Science - www.apluscompsci.com

Variable Scope

© A+ Computer Science - www.apluscompsci.com



Scope

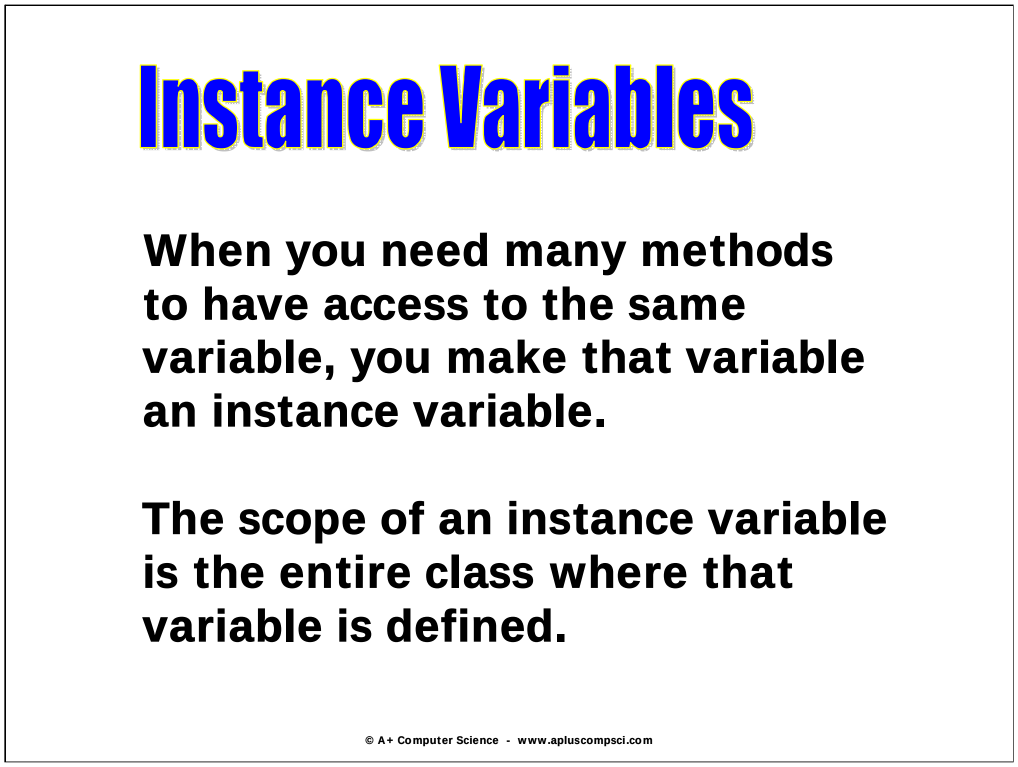
```
{
   int fun = 99;
}
```

Any variable defined inside of braces, only exists within those braces.

That variable has a scope limited to those braces.

© A+ Computer Science - www.apluscompsci.com

When a variable is defined within a set of braces, that variable can only be accessed inside those braces.



Instance Variables

When you need many methods to have access to the same variable, you make that variable an instance variable.

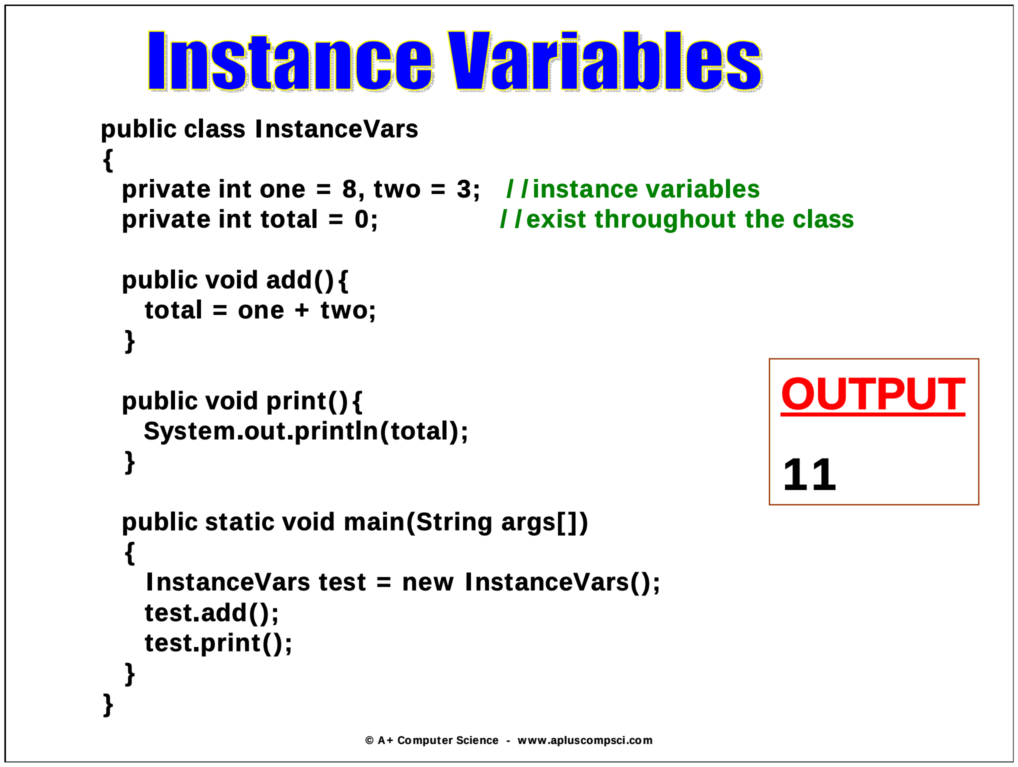
The scope of an instance variable is the entire class where that variable is defined.

© A+ Computer Science - www.apluscompsci.com

An instance variable is a variable tied to an instance of a class. Each time an Object is instantiated, it is given its own set of instance variables.

```
Monster x = new Monster();
```

x would refer to a new Monster that contains its own set of Monster instance variables.



Instance Variables

```
public class InstanceVars
{
  private int one = 8, two = 3;   //instance variables
  private int total = 0;          //exist throughout the class

  public void add(){
    total = one + two;
  }

  public void print(){
    System.out.println(total);
  }

  public static void main(String args[])
  {
    InstanceVars test = new InstanceVars();
    test.add();
    test.print();
  }
}
```

OUTPUT

```
11
```

© A+ Computer Science - www.apluscompsci.com

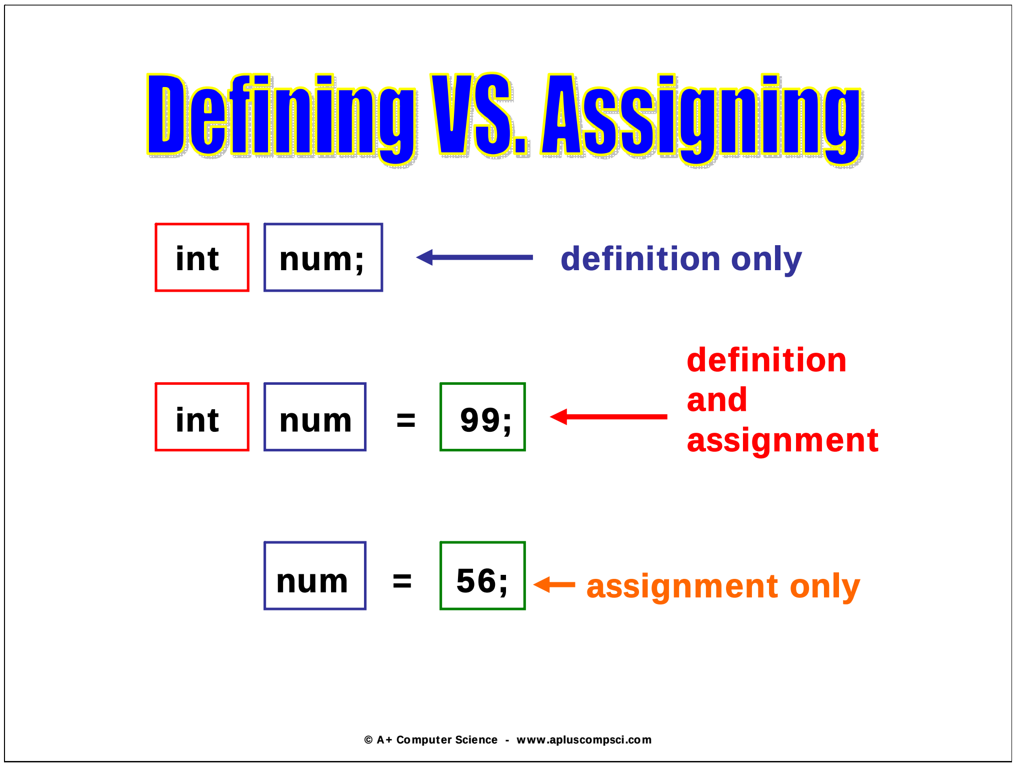
Class `InstanceVars` contains three instance variables : one, two, and total. Each time class `InstanceVars` is instantiated, a new set of instance variables is created.

```
InstanceVars test = new InstanceVars();
```

test refers to an `InstanceVars` Object that contains its own set of one, two, and three.

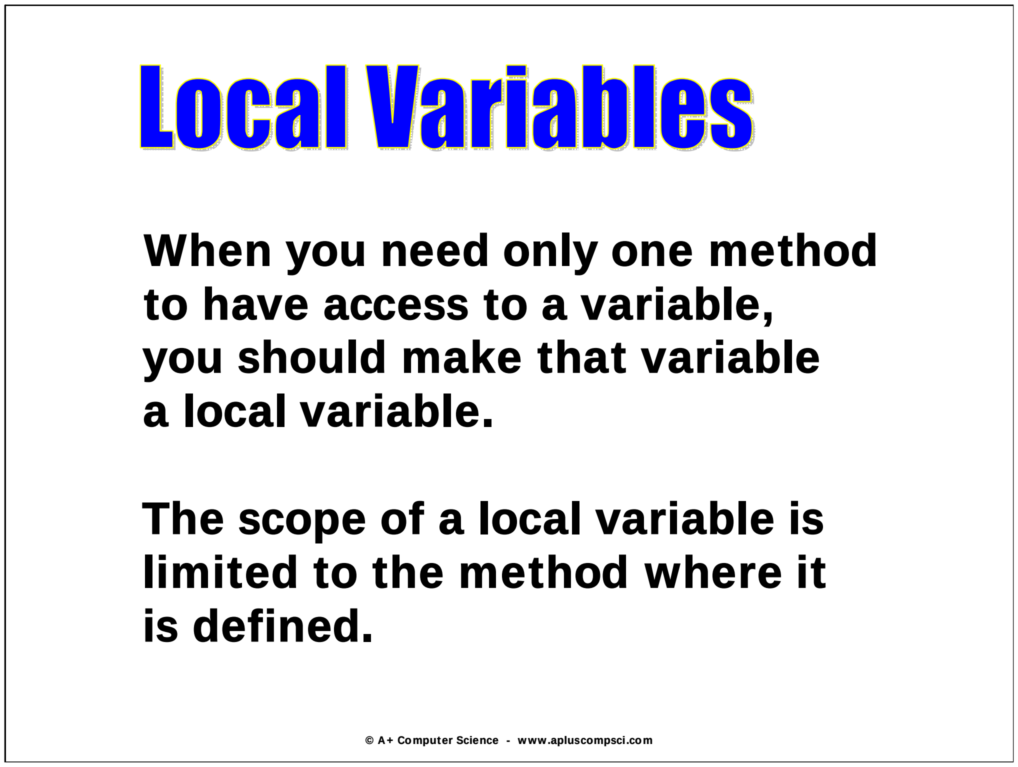
```
InstanceVars diff = new InstanceVars();
```

diff refers to an `InstanceVars` Object that contains its own set of one, two, and three.



Defining VS. Assigning

When defining a variable, the type must be listed and the name. When defining and assigning a variable, the type, name, and value must be listed. When assigning a variable only, the name and value must be listed.

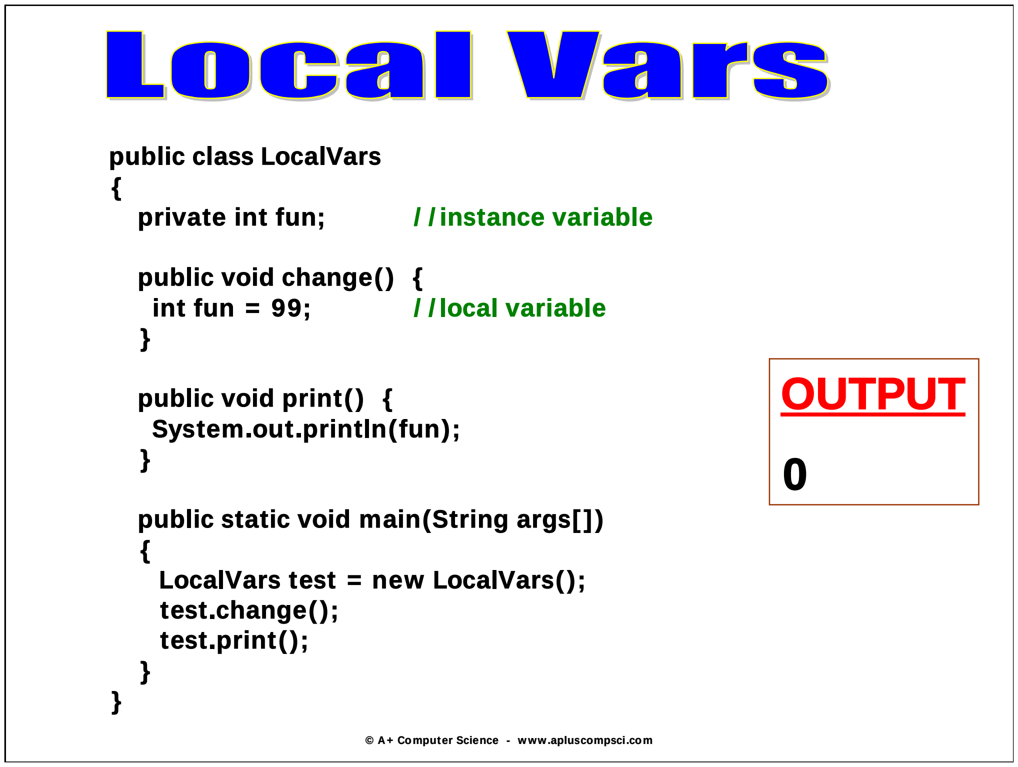


Local Variables

When you need only one method to have access to a variable, you should make that variable a local variable.

The scope of a local variable is limited to the method where it is defined.

© A+ Computer Science - www.apluscompsci.com



Local Vars

```
public class LocalVars
{
   private int fun;          //instance variable

   public void change()  {
     int fun = 99;            //local variable
   }

   public void print()  {
     System.out.println(fun);
   }

   public static void main(String args[])
   {
     LocalVars test = new LocalVars();
     test.change();
     test.print();
   }
}
```

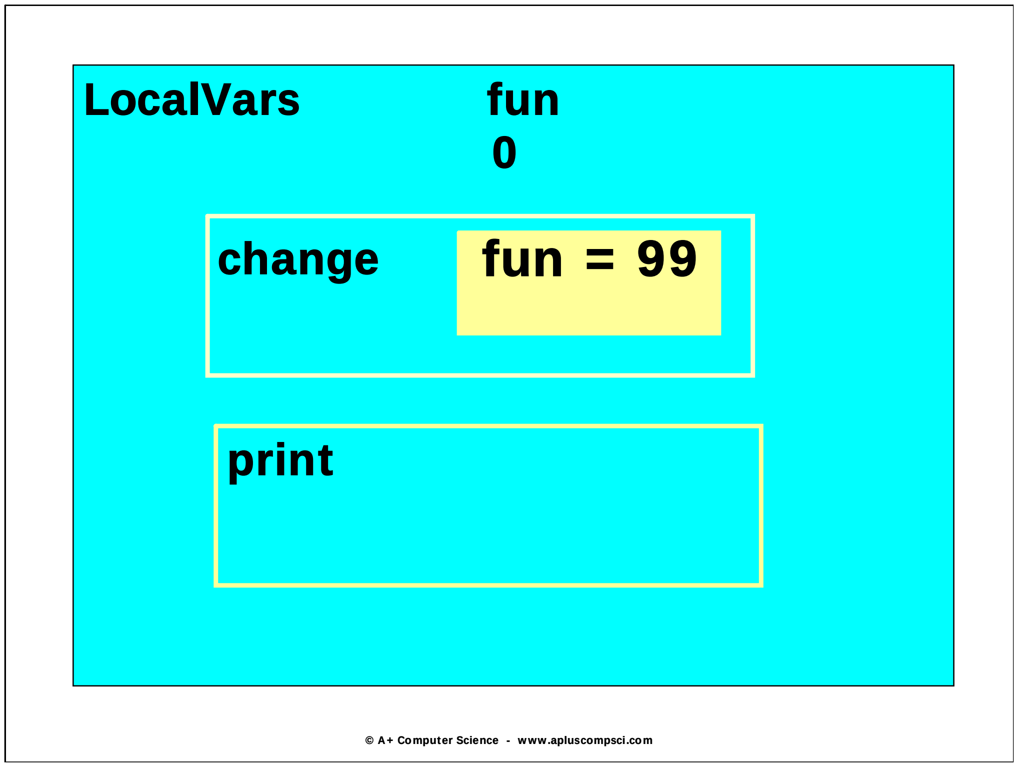
OUTPUT

```
0
```

© A+ Computer Science - www.apluscompsci.com

Class `LocalVars` has an one instance variable named fun.

Method change contains a local variable named fun. Local variables take precedence over instance variables. Local variables can only be used inside the method in which they are defined.



Class `LocalVars` has an one instance variable named fun.

Instance variable fun can be used anywhere in class LocalVars.

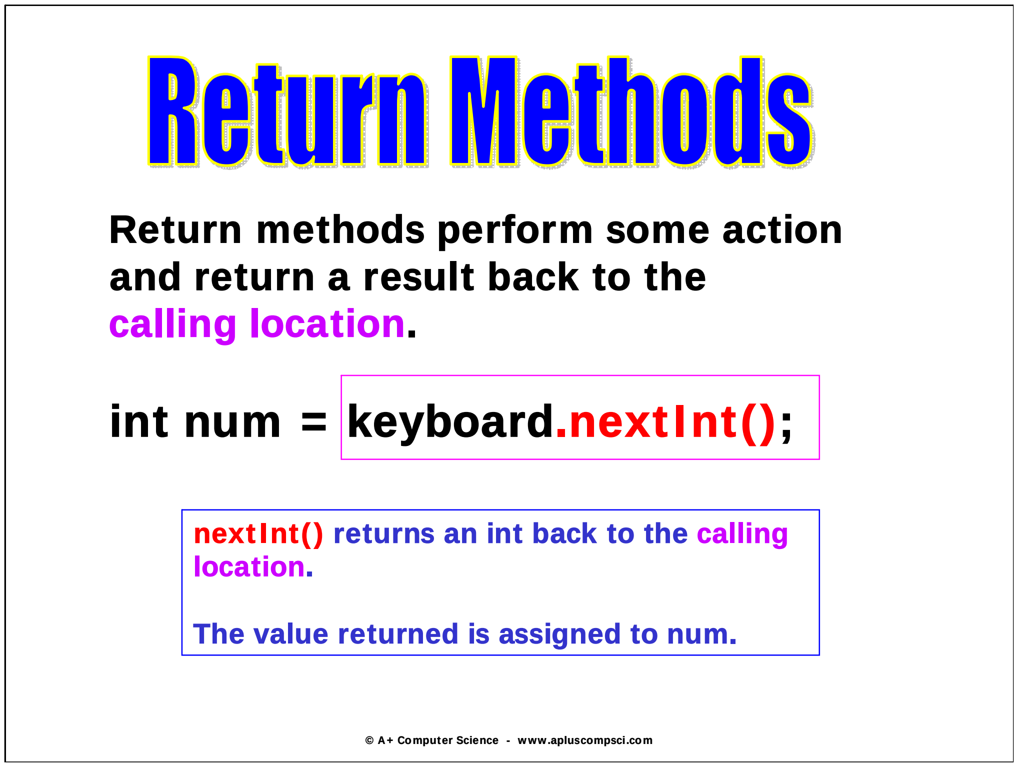
Method change contains a local variable named fun. Method change does not make any changes to the instance variable fun. Local variable fun can only be used in method change.

open localvars.java

© A+ Computer Science - www.apluscompsci.com

return methods

© A+ Computer Science - www.apluscompsci.com



Return Methods

Return methods perform some action and return a result back to the calling location.

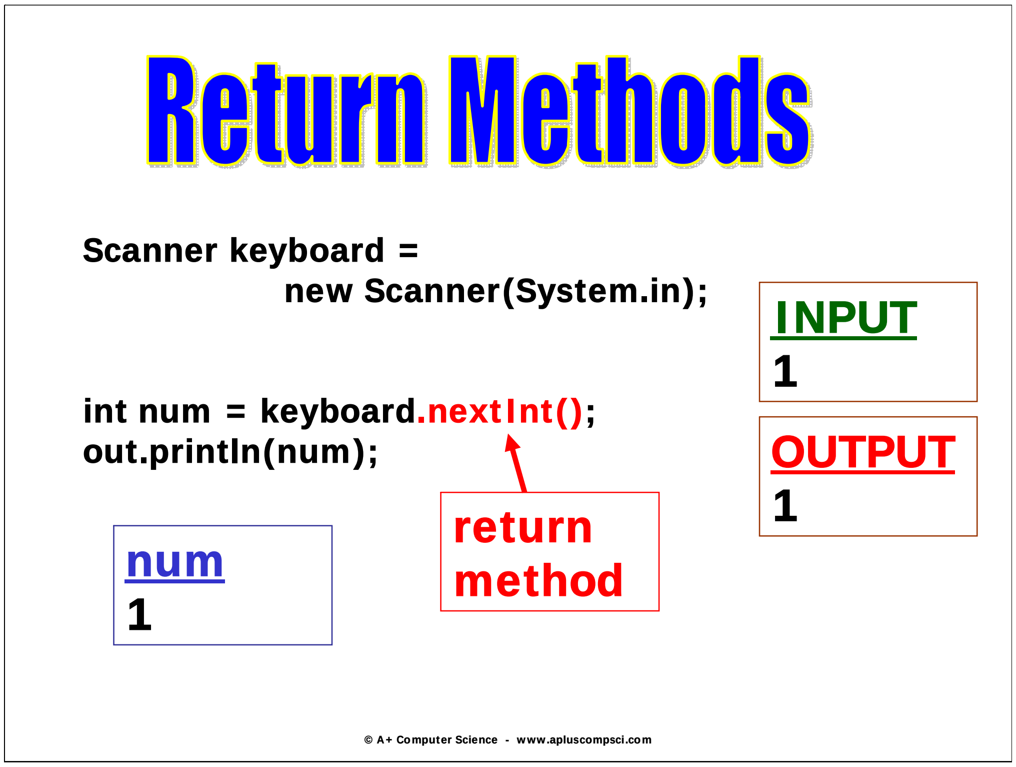
```
int num = keyboard.nextInt();
```

nextInt() returns an int back to the calling location.

The value returned is assigned to num.

© A+ Computer Science - www.apluscompsci.com

Return methods typically take in some type of data, do something to the data, and then send back a result. `nextInt()` is a Scanner return method. `nextInt()` retrieves the next integer value from the keyboard and sends it back to num.



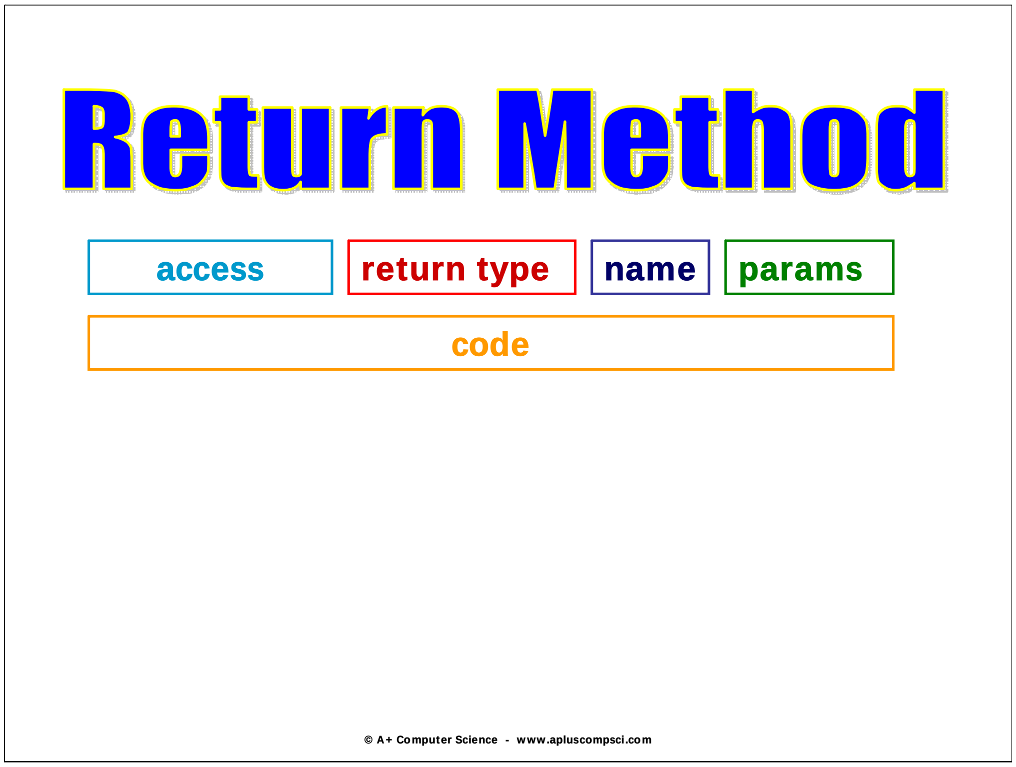
Return Methods

```
Scanner keyboard =
            new Scanner(System.in);

int num = keyboard.nextInt();
out.println(num);
```

INPUT
1
OUTPUT
1
return
method
num
1

© A+ Computer Science - www.apluscompsci.com

`nextInt()` gets the next int entered on the keyboard and returns it. The int returned by `nextInt()` is placed in num.



Return Method

access | return type | name | params

code

© A+ Computer Science - www.apluscompsci.com

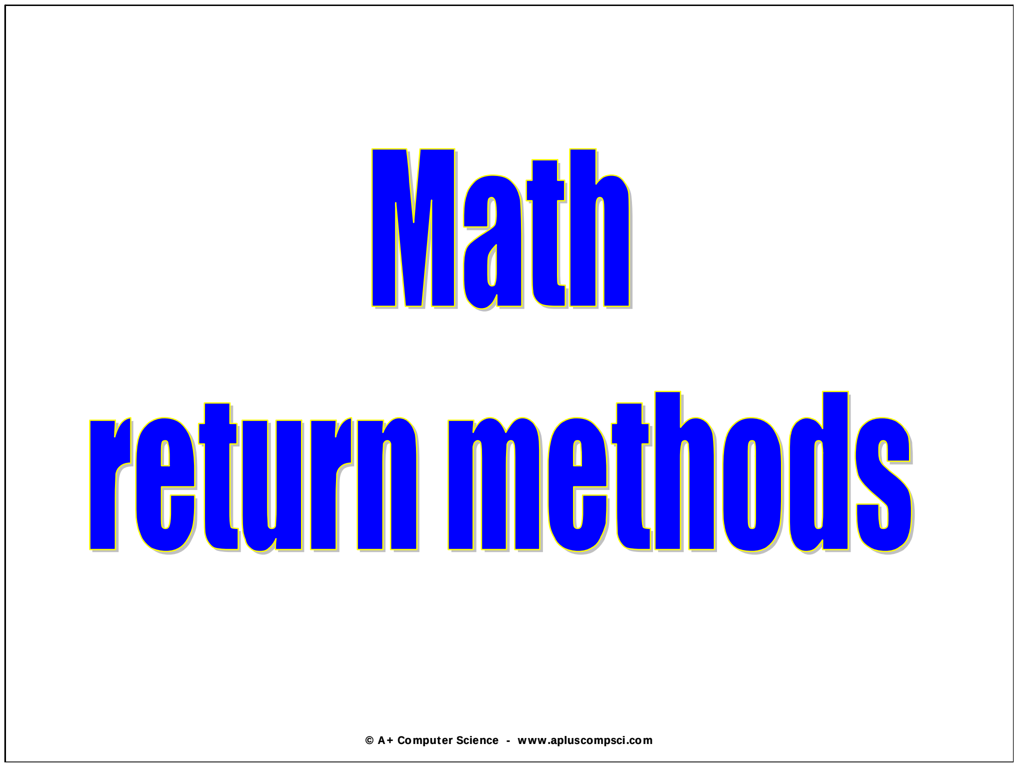
Return methods have a return type specified before the method name.

`public int getNum()`

`getNum()` returns an integer value.

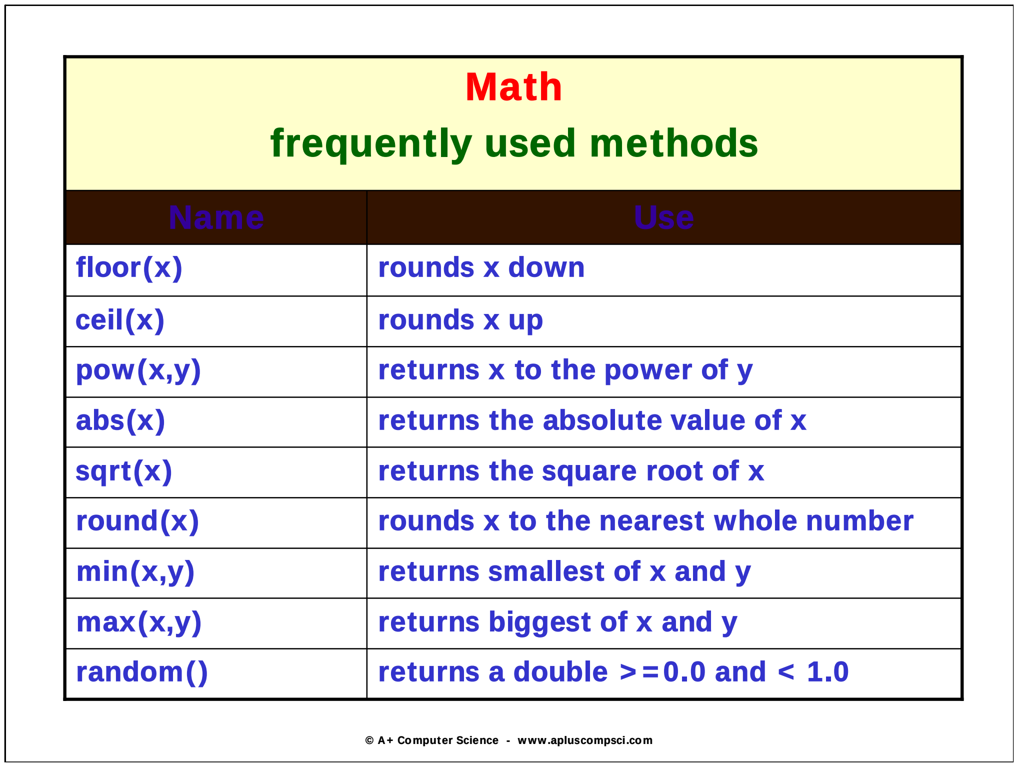
`public double getStuff()`

`getStuff()` returns a double/decimal value.



Math return methods

© A+ Computer Science - www.apluscompsci.com

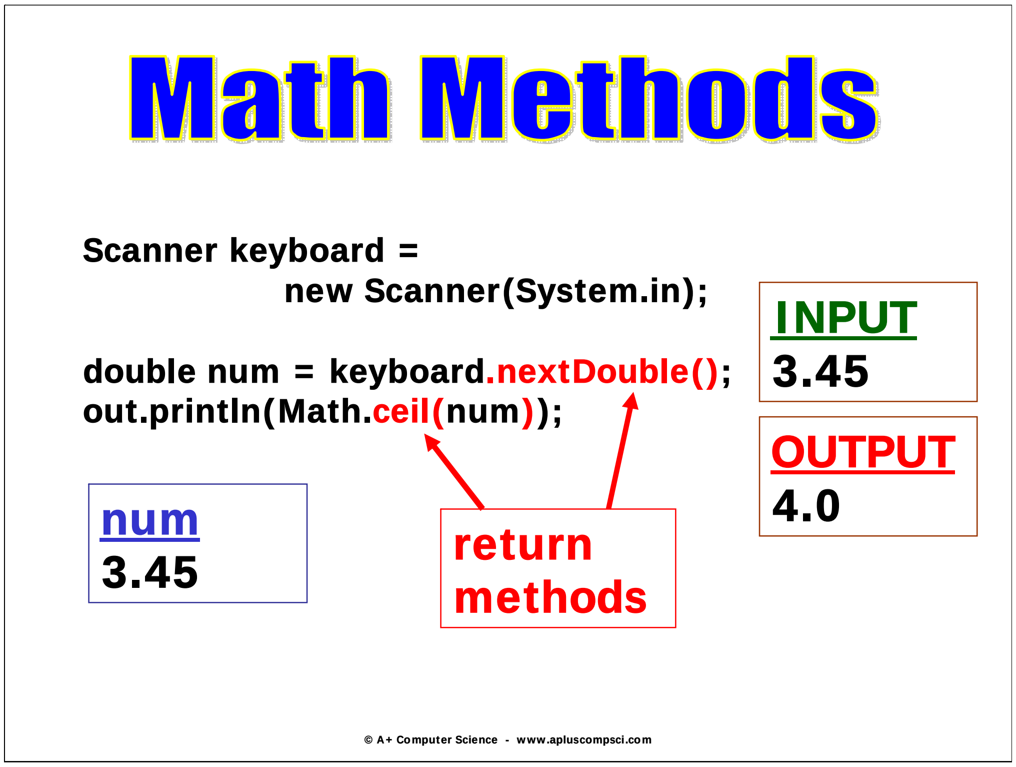


Math
frequently used methods

Name	Use
floor(x)	rounds x down
ceil(x)	rounds x up
pow(x,y)	returns x to the power of y
abs(x)	returns the absolute value of x
sqrt(x)	returns the square root of x
round(x)	rounds x to the nearest whole number
min(x,y)	returns smallest of x and y
max(x,y)	returns biggest of x and y
random()	returns a double >=0.0 and < 1.0

© A+ Computer Science - www.apluscompsci.com

The Math class contains many useful math related methods.



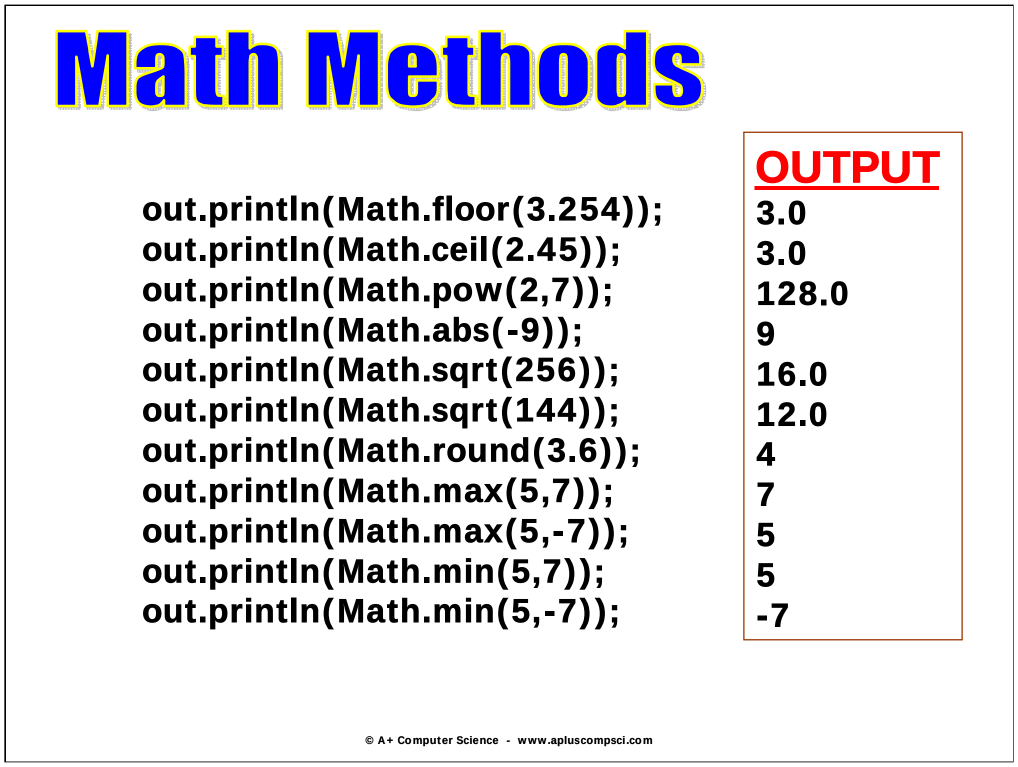
Math Methods

```
Scanner keyboard =
          new Scanner(System.in);

double num = keyboard.nextDouble();
out.println(Math.ceil(num));
```

INPUT
3.45
OUTPUT
4.0
num
3.45
return methods

© A+ Computer Science - www.apluscompsci.com

Math methods are return methods. First, a value is passed to a math method. Second, the math method performs some action. Finally, the math method returns a result.



Math Methods

```
out.println(Math.floor(3.254));
out.println(Math.ceil(2.45));
out.println(Math.pow(2,7));
out.println(Math.abs(-9));
out.println(Math.sqrt(256));
out.println(Math.sqrt(144));
out.println(Math.round(3.6));
out.println(Math.max(5,7));
out.println(Math.max(5,-7));
out.println(Math.min(5,7));
out.println(Math.min(5,-7));
```

OUTPUT

```
3.0
3.0
128.0
9
16.0
12.0
4
7
5
5
-7
```

© A+ Computer Science - www.apluscompsci.com

`floor(val)` returns val decreased to the nearest interger.

`ceil(val)` returns val increased to the nearest interger.

`pow(x,y)` reutrns x raised to the power of y

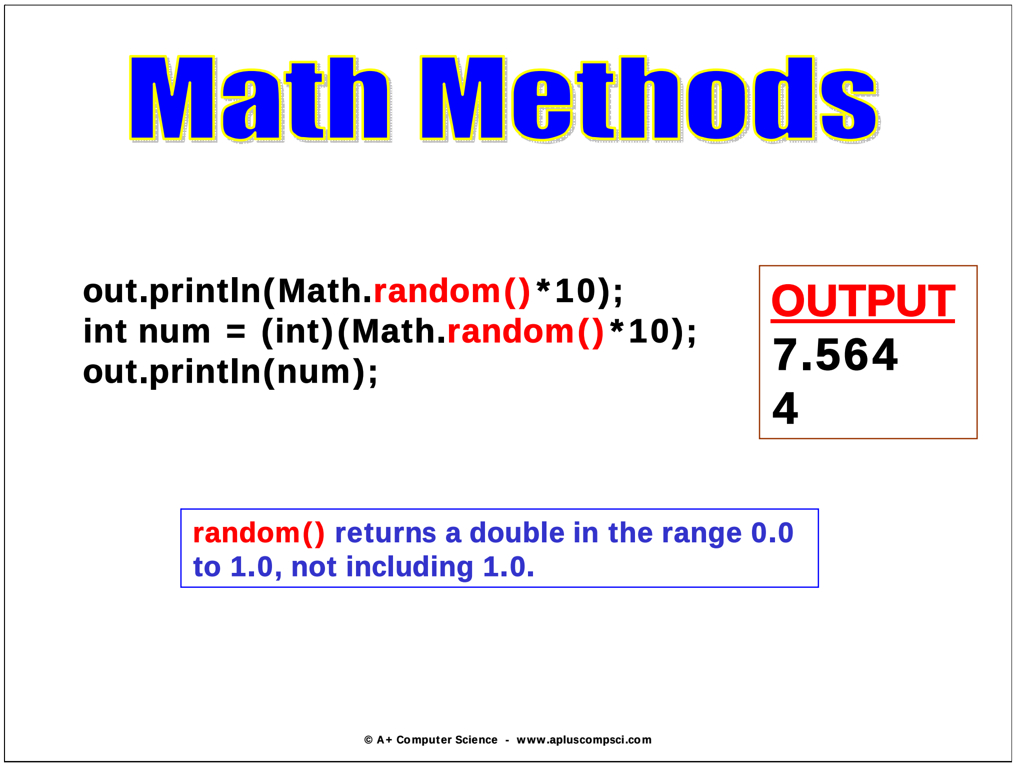
`abs(val)` returns the absolute value of val

`sqrt(val)` returns the square root of val

`round(val)` returns val rounede to the nearest integer

`max(one, two)` returns the largest of one and two

`min(one, two)` returns the smallest of one and two



Math Methods

```
out.println(Math.random()*10);
int num = (int)(Math.random()*10);
out.println(num);
```

OUTPUT

```
7.564
4
```

random() returns a double in the range 0.0 to 1.0, not including 1.0.

© A+ Computer Science - www.apluscompsci.com

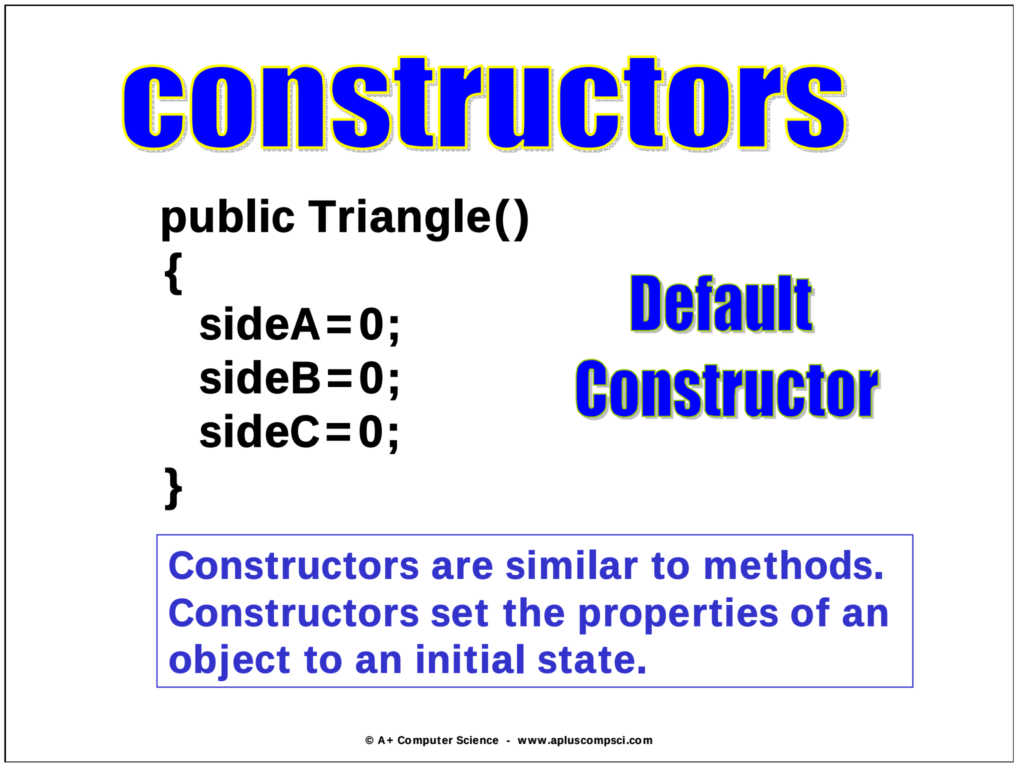
`random()` returns a random number between 0.0 and 1.0 not including 1.0.

Open mathmethods.java randomone.java

© A+ Computer Science - www.apluscompsci.com

Pieces of the OOP Puzzle Part Two

© A+ Computer Science - www.apluscompsci.com



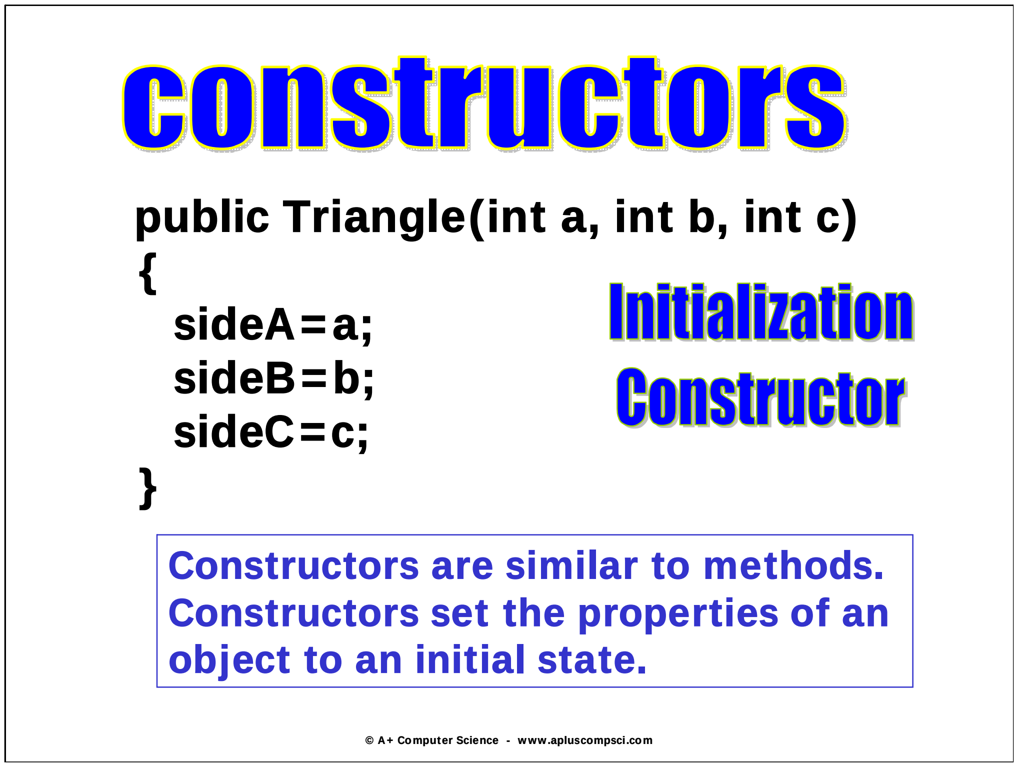
constructors

```
public Triangle()
{
  sideA=0;
  sideB=0;
  sideC=0;
}
```

Default Constructor

Constructors are similar to methods. Constructors set the properties of an object to an initial state.

© A+ Computer Science - www.apluscompsci.com



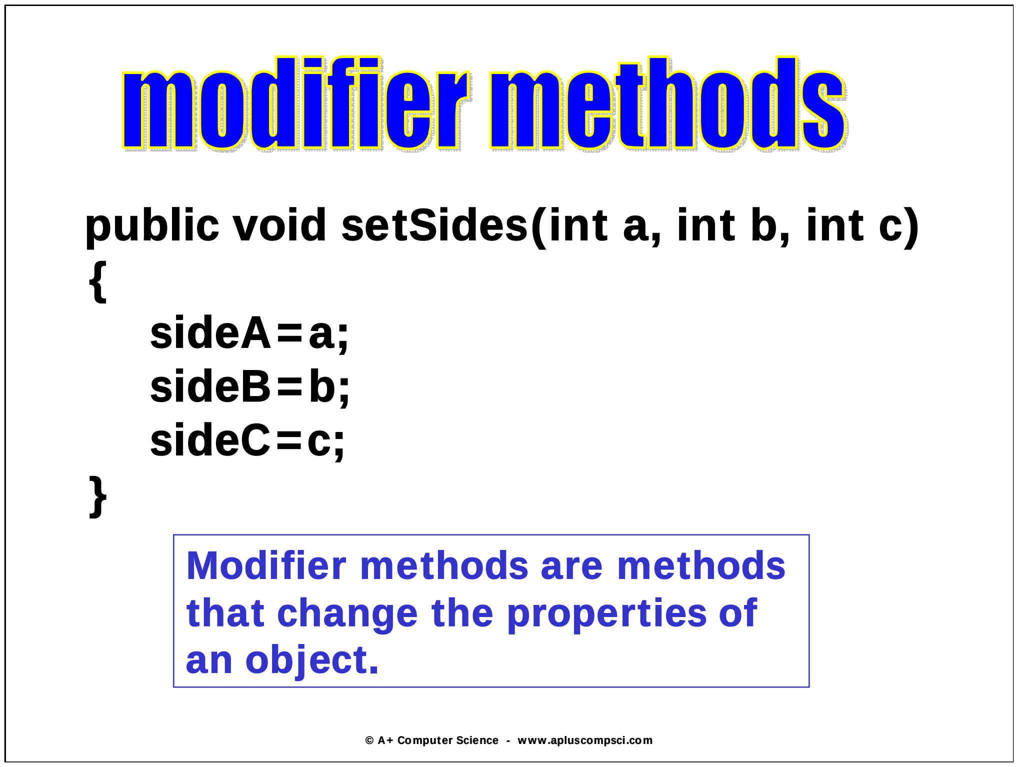
constructors

```
public Triangle(int a, int b, int c)
{
  sideA=a;
  sideB=b;
  sideC=c;
}
```

Initialization Constructor

Constructors are similar to methods. Constructors set the properties of an object to an initial state.

© A+ Computer Science - www.apluscompsci.com

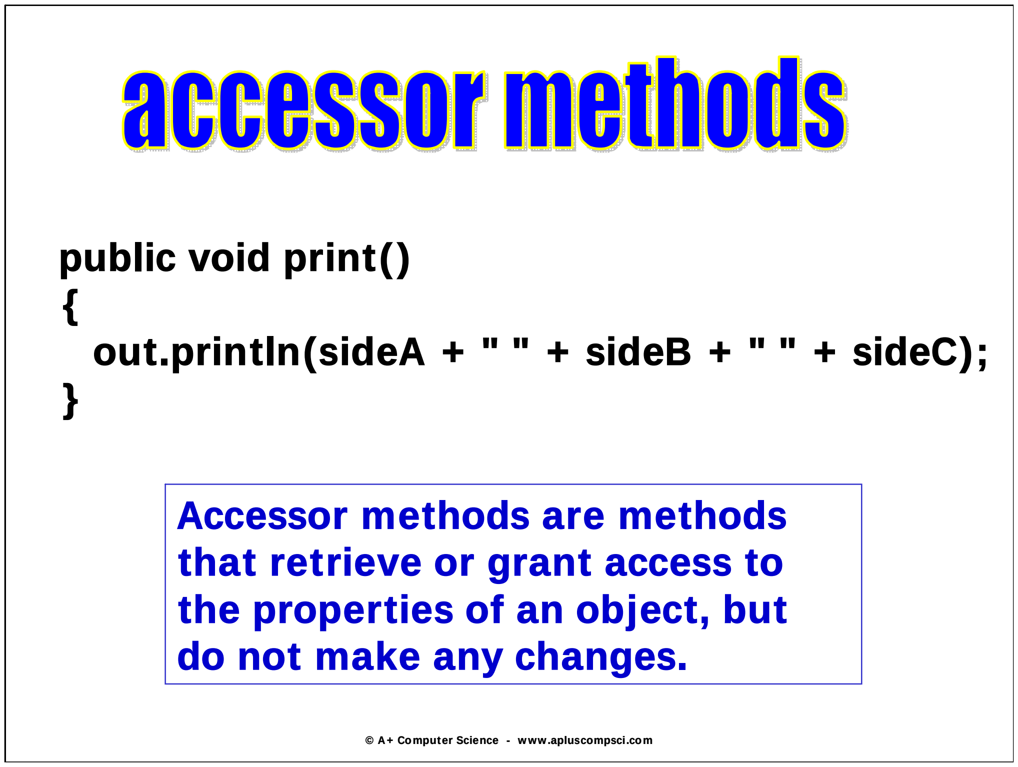


modifier methods

```
public void setSides(int a, int b, int c)
{
    sideA=a;
    sideB=b;
    sideC=c;
}
```

Modifier methods are methods that change the properties of an object.

© A+ Computer Science - www.apluscompsci.com

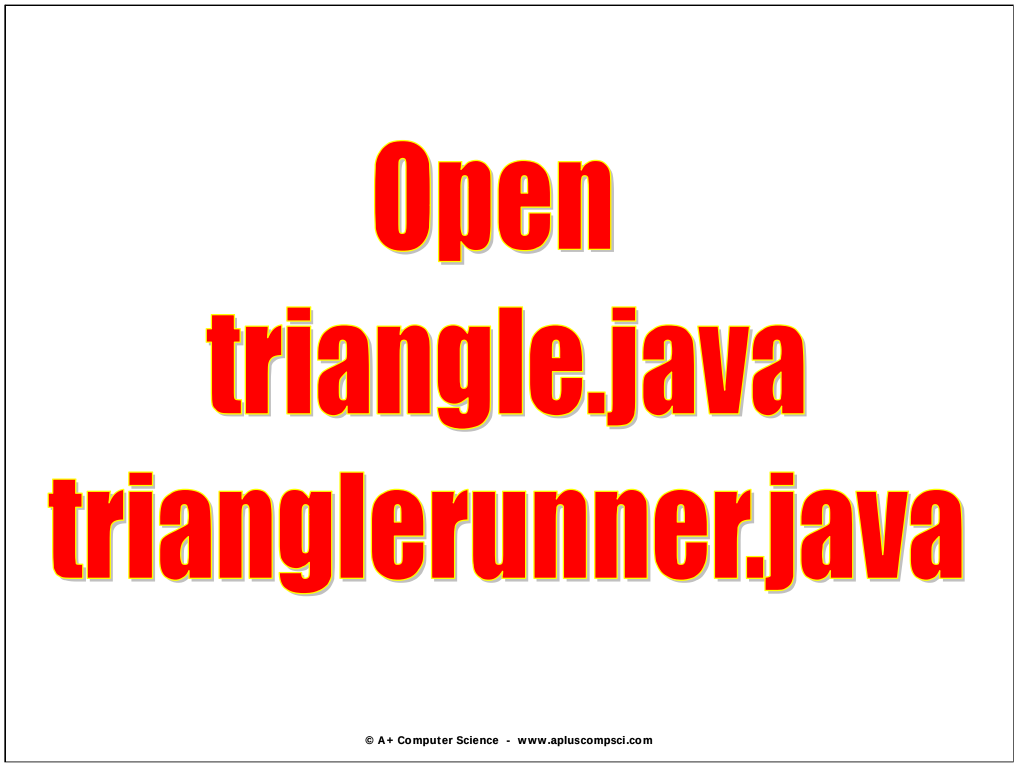


accessor methods

```
public void print()
{
  out.println(sideA + " " + sideB + " " + sideC);
}
```

Accessor methods are methods that retrieve or grant access to the properties of an object, but do not make any changes.

© A+ Computer Science - www.apluscompsci.com



Open triangle.java trianglerunner.java

© A+ Computer Science - www.apluscompsci.com

Continue work on Lab 03

© A+ Computer Science - www.apluscompsci.com