```
public class SquareExample {

	public static void foo(Square s1, Square s2){
		s1.side += 5;
		s2.side += 5;
	}

	public static void main(String[] args) {
		Square a = new Square (10);
		Square b = new Square (20);
		System.out.println(a.side+" "+b.side);
		foo(a,b);
	}
}
```

<table>
<tr>
<td>Stack

Method: foo
Method is called
s1 → 1
s2 → 2

After s1.side += 5
s1 → 1 (value of reference in heap is changed)

After s2.side += 5
s2 → 2 (value of reference in heap is changed)

Stack frame for foo collapses after foo returns</td>
<td rowspan="2">Heap
Square a = new Square (10);
1 → 10

Square b = new Square (20);
2 → 20

After s1.side += 5
1 → 15

After s2.side += 5
2 → 25</td>
</tr>
<tr>
<td>Method: main
Reference a
Square a = new Square (10);
a → 1
Reference b
Square b = new Square (20);
b → 2

foo is called
foo returns

References remain the same in main
a → 1
b → 2

So values are now 15 and 25.</td>
</tr>
</table>

Time line for foo call

```
public class SquareExample {
	public static void bar(Square s1, Square s2){
		s1.side += 5;
		s2 = new Square(50);
		s2.side += 5;
	}

	public static void main(String[] args) {
		Square a = new Square (10);
		Square b = new Square (20);
		System.out.println(a.side+" "+b.side);
		bar(a,b);
		System.out.println(a.side+" "+b.side);
	}
}
```

<table>
<tr>
<td>Stack

Method: bar
Method is called
s1 → 1
s2 → 2

After s1.side += 5
s1 → 1 (value of reference in heap is changed)

<code>s2 = new Square(50);</code>
s2 → 3 (new reference)

After s2.side += 5
s2 → 3 (ref value changed)

Stack frame for bar collapses after bar returns</td>
<td rowspan="2">Heap
<code>Square a = new Square (10);</code>
1 → 10

<code>Square b = new Square (20);</code>
2 → 20

After s1.side += 5
1 → 15

<code>s2 = new Square(50);</code>
3 → 50

After s2.side += 5
3 → 55

Notice that object 2 is unchanged.</td>
</tr>
<tr>
<td>Method: main
Reference a
<code>Square a = new Square (10);</code>
a → 1
Reference b
<code>Square b = new Square (20);</code>
b → 2

bar is called
bar returns

References remain the same in main
a → 1, b → 2
So values are now 15 and 20.</td>
</tr>
</table>

Time line for bar call

```
class Password
{
  public String password;

  public Password(String initPassword)  { password = initPassword;  }
}

public class PasswordTester
{
  public static void main(String[] args)
  {
    String s = "TopSecret";
    Password p = new Password("NoneOfYourBusiness");
    swap(s, p);
    System.out.println(s + "\n" + p.password);
  }

  public static void swap(String a, Password b)
  {
    String temp = a;
    a = b.password;
    b.password = temp;
  }
}
```

Stack	Heap

Method: main
 	
`String s = "TopSecret";`
s → 1
 	
`Password p = new Password("NoneOfYourBusiness");`
p → 2
 	
p.password → 3.
 	
swap is called
 	
swap returns
 	
s is still 1.
p is still 2.
 	
But p.password is in heap and is changed to 1.
 	
p.password → 1 and s → 1
 	
p.password = s = `"TopSecret"`
	`String s = "TopSecret";`
	1 → `"TopSecret"`
	
	`Password p = new Password("NoneOfYourBusiness");`
	
	2 → Password object p.
	
	Constructor for password is called and initPassword = `"NoneOfYourBusiness"` is assigned as the new password.
	
	3 → `"NoneOfYourBusiness"`
	
	Construct returns. Now 2 → Password object p, has a reference in heap as
	
	p.password → 3

Method: swap
a → 1
b → 2
 	
`String temp = a;`
 	
temp → 1
 	
`a = b.password;`
`We know that, b.password` → 3
So a takes reference to the same object
 	
a → 3
 	
`b.password = temp;`
 	
b.password → 1
 	
Method returns
	b.password reference is in heap. Now it is changed to 1.
	
	So p.password changes from 3 to 1. (“TopSecret”)
	
	p.password → 1

Call stack/heap for swap

What happens when the below code is executed?

```
public class PasswordMixer
{
  public static void main(String[] args)
  {
    Password p1 = new Password("Test1");
    Password p2 = new Password("Test2");
    System.out.println("Before swap");
    System.out.println("P1 = "+p1.password + "\n" +"P2 = "+ p2.password+"\n");
    swap(p1, p2);
    System.out.println("P1 = "+p1.password + "\n" +"P2 = "+ p2.password);
  }

  public static void swap(Password a, Password b)
  {
    String temp = a.password;
    a.password = b.password;
    b.password = temp;
  }
}
```