CS159

Variables and Primitive Types

- Java Primitive Types:
  - `byte`, `short`, `int`, `long`, `float`, `double`, `boolean`, `char`

Example

```java
int a;
int b;
boolean aIsLess;

a = 3;
b = 4;
aIsLess = a < b;

if (aIsLess && b < a)
{
    System.out.println("Hello");
}
else
{
    System.out.println("Goodbye");
}
```

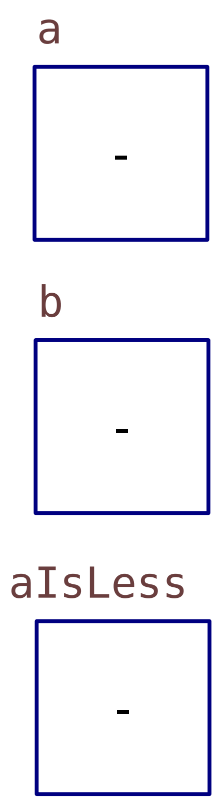
Is this code syntactically correct? What will be printed?

Example

```
int a;
int b;
boolean aIsLess;

a = 3;
b = 4;
aIsLess = a < b;

if (aIsLess && b < a)
{
    System.out.println("Hello");
}
else
{
    System.out.println("Goodbye");
}
```



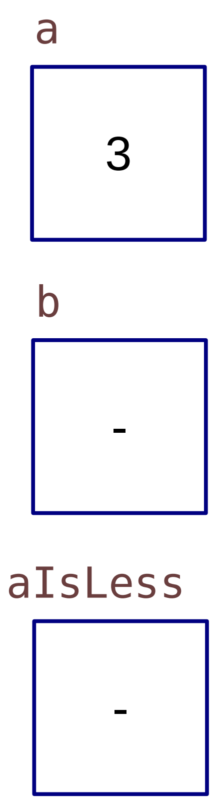
Note: “-” denotes an uninitialized variable.
It is a syntax error to access an uninitialized local variable.

Example

```
int a;
int b;
boolean aIsLess;

a = 3;
b = 4;
aIsLess = a < b;

if (aIsLess && b < a)
{
    System.out.println("Hello");
}
else
{
    System.out.println("Goodbye");
}
```

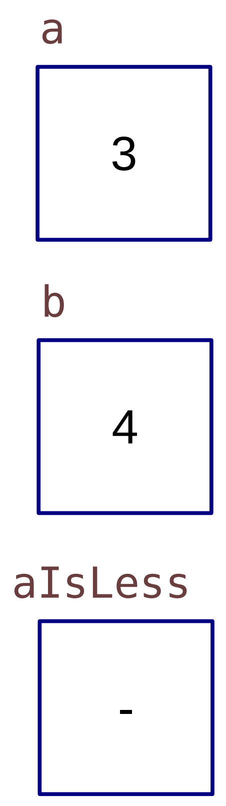


Example

```
int a;
int b;
boolean aIsLess;

a = 3;
b = 4;
aIsLess = a < b;

if (aIsLess && b < a)
{
    System.out.println("Hello");
}
else
{
    System.out.println("Goodbye");
}
```

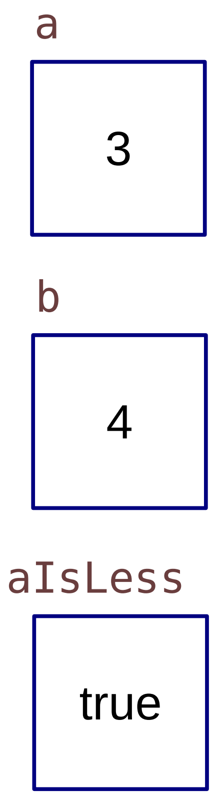


Example

```
int a;
int b;
boolean aIsLess;

a = 3;
b = 4;
aIsLess = a < b;

if (aIsLess && b < a)
{
    System.out.println("Hello");
}
else
{
    System.out.println("Goodbye");
}
```



Reference Types

- String, Arrays, all Classes.

Reference Example

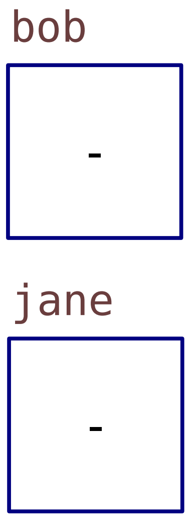
```
Person bob;
Person jane;
bob = new Person("Bob");
jane = bob;
jane.setName("Jane");
System.out.println(bob.getName());
```

Is this code syntactically correct? What will be printed?

(Assume that setName and getName perform the expected operations.)

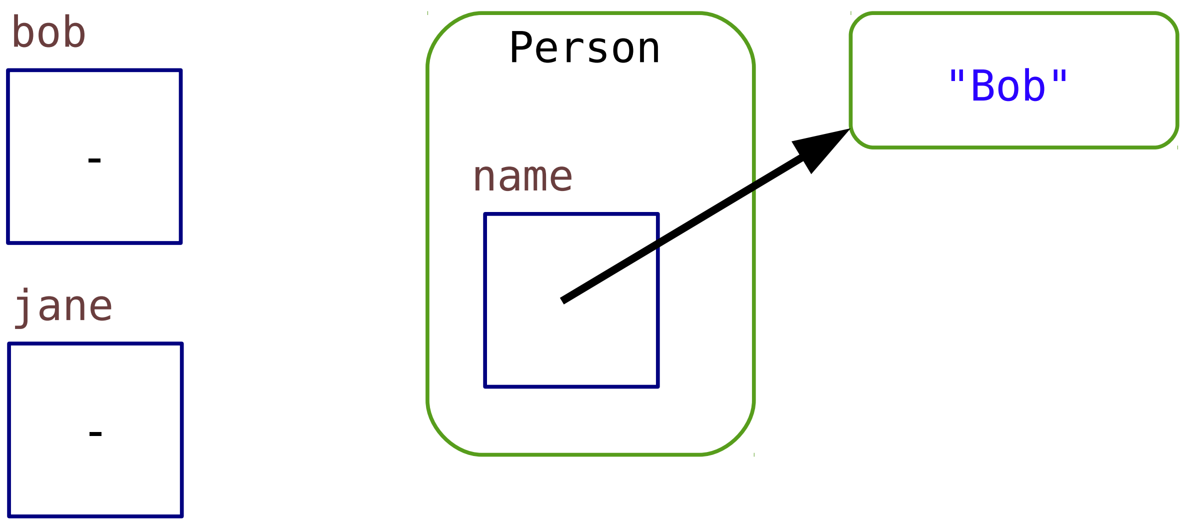
Reference Example

```
Person bob;
Person jane;
bob = new Person("Bob");
jane = bob;
jane.setName("Jane");
System.out.println(bob.getName());
```



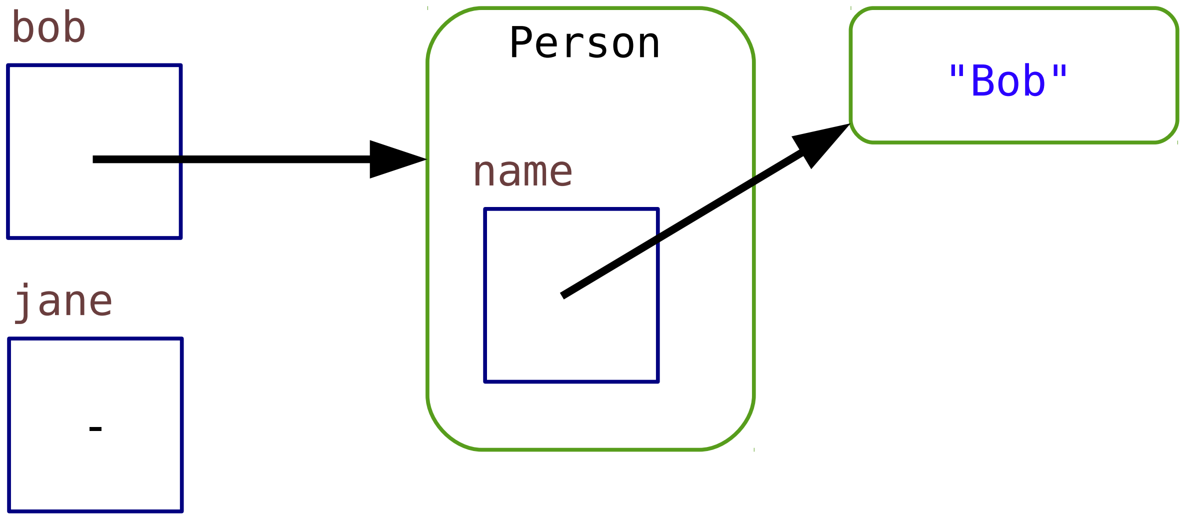
Reference Example

```
Person bob;
Person jane;
bob = new Person("Bob");
jane = bob;
jane.setName("Jane");
System.out.println(bob.getName());
```



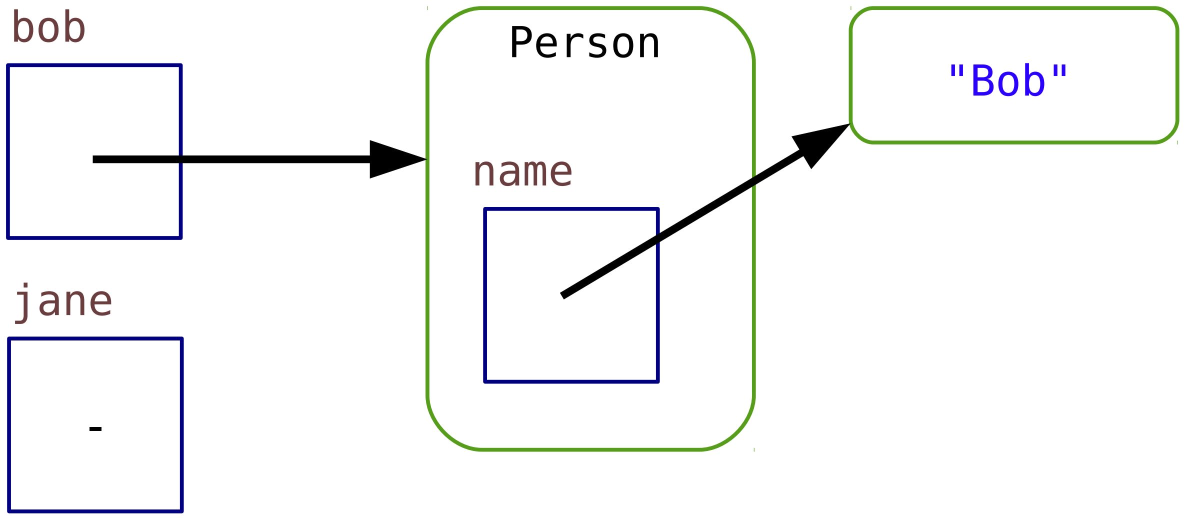
Reference Example

```
Person bob;
Person jane;
bob = new Person("Bob");
jane = bob;
jane.setName("Jane");
System.out.println(bob.getName());
```



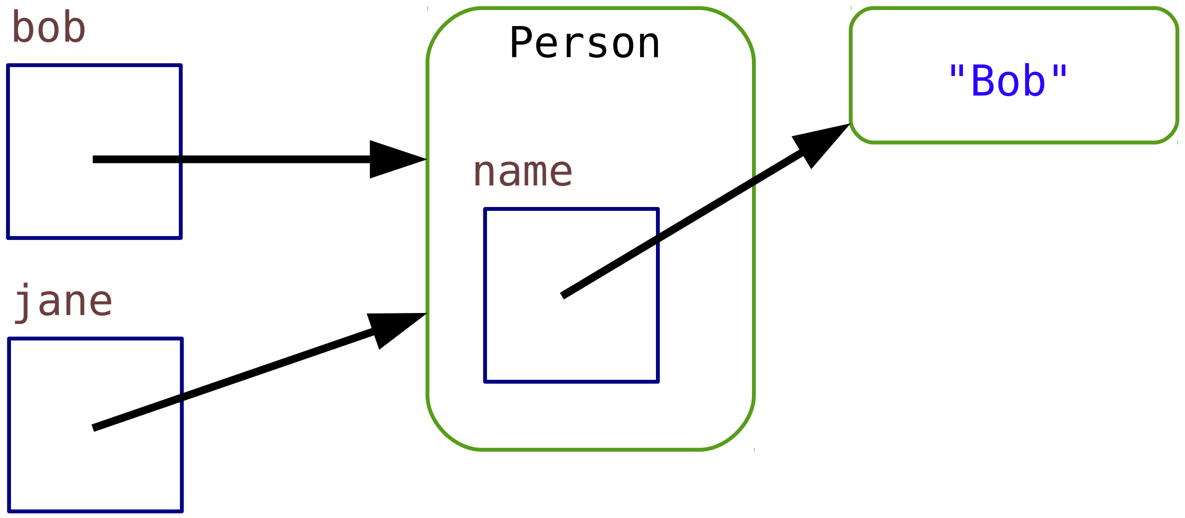
Reference Example

```
Person bob;
Person jane;
bob = new Person("Bob");
jane = bob;
jane.setName("Jane");
System.out.println(bob.getName());
```



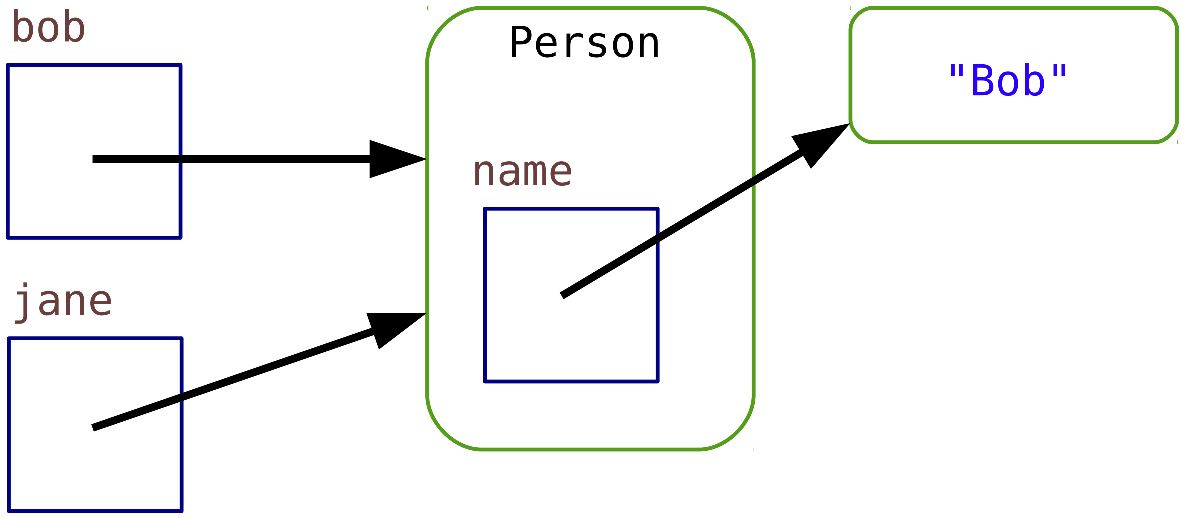
Reference Example

```
Person bob;
Person jane;
bob = new Person("Bob");
jane = bob;
jane.setName("Jane");
System.out.println(bob.getName());
```



Reference Example

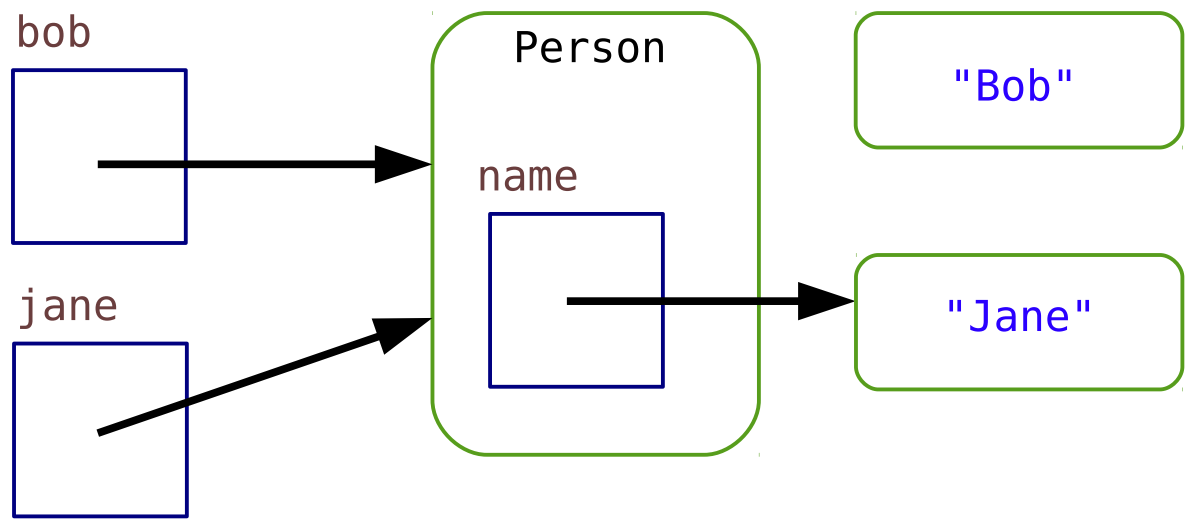
```
Person bob;
Person jane;
bob = new Person("Bob");
jane = bob;
jane.setName("Jane");
System.out.println(bob.getName());
```



The variables bob and jane are now “aliased”.

Reference Example

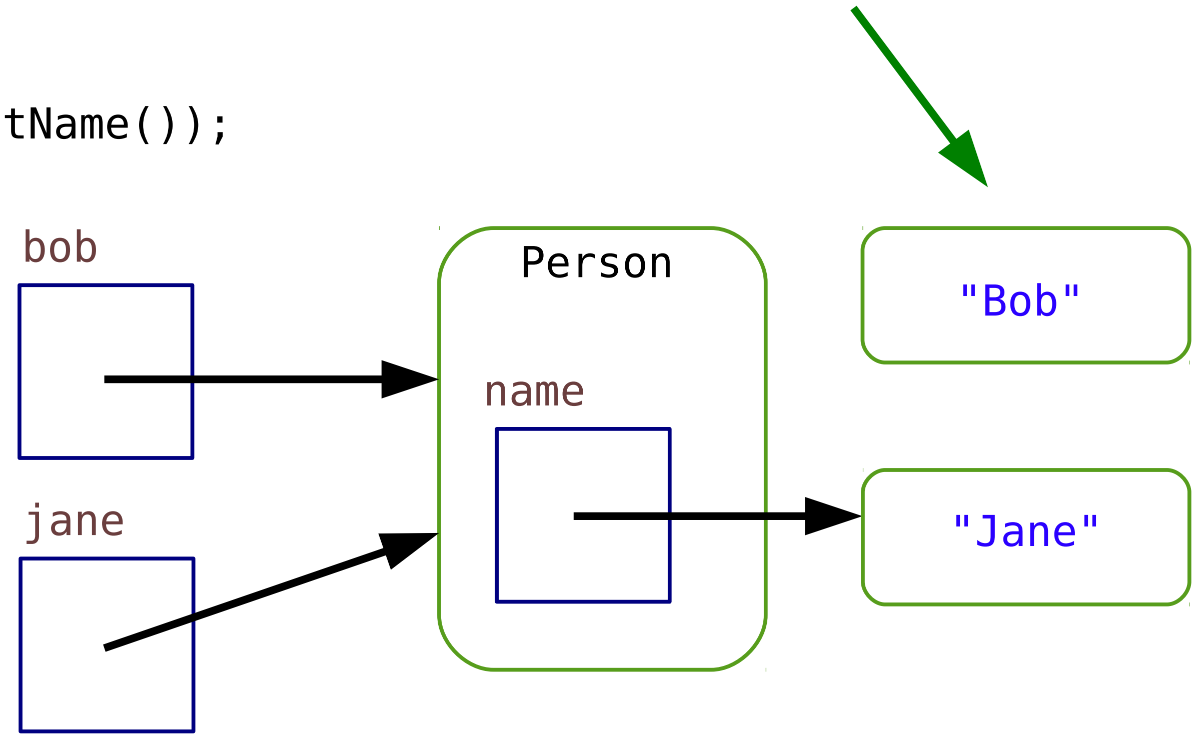
```
Person bob;
Person jane;
bob = new Person("Bob");
jane = bob;
jane.setName("Jane");
System.out.println(bob.getName());
```



Reference Example

```
Person bob;
Person jane;
bob = new Person("Bob");
jane = bob;
jane.setName("Jane");
System.out.println(bob.getName());
```

Quiz: What happens to this?

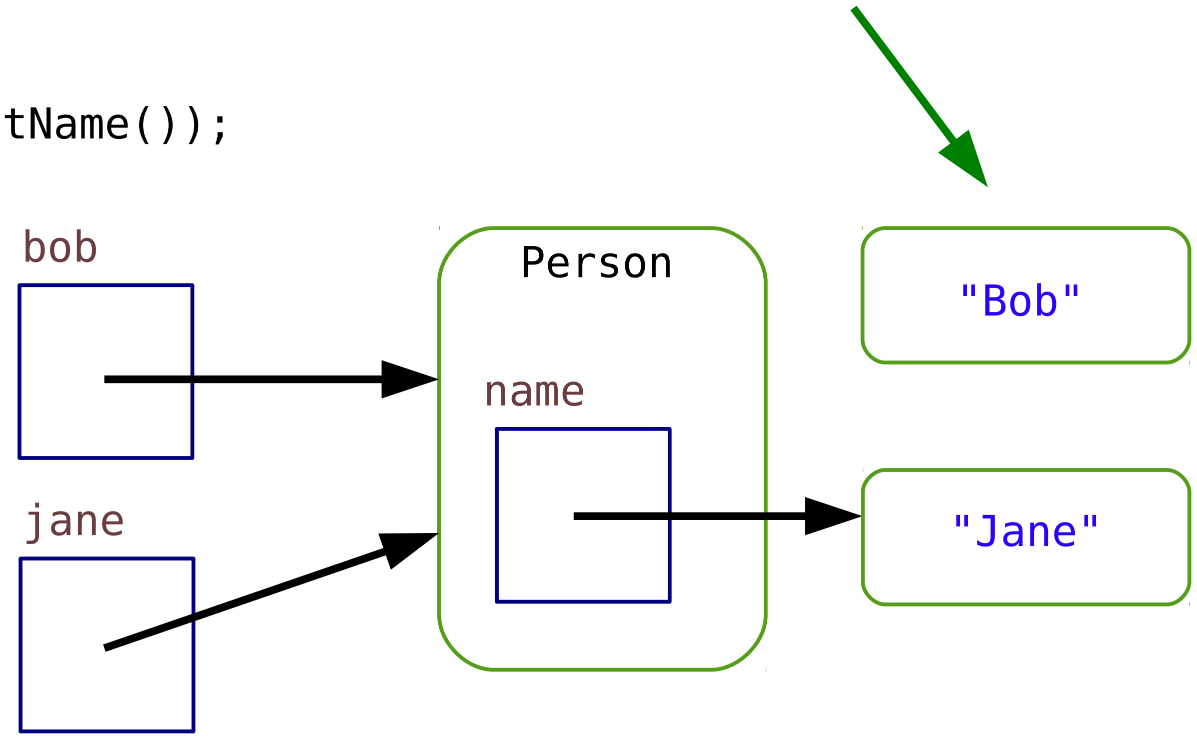


Reference Example

```
Person bob;
Person jane;
bob = new Person("Bob");
jane = bob;
jane.setName("Jane");
System.out.println(bob.getName());
```

Quiz: What happens to this?

Answer: garbage collection.



What is a Reference... *Really?*

- We can print the contents of `bob`:

```
System.out.println(bob);
```

  - Result is: `Person@6422b8ff`

What is a Reference... *Really?*

- We can print the contents of `bob`:

```
System.out.println(bob);
```

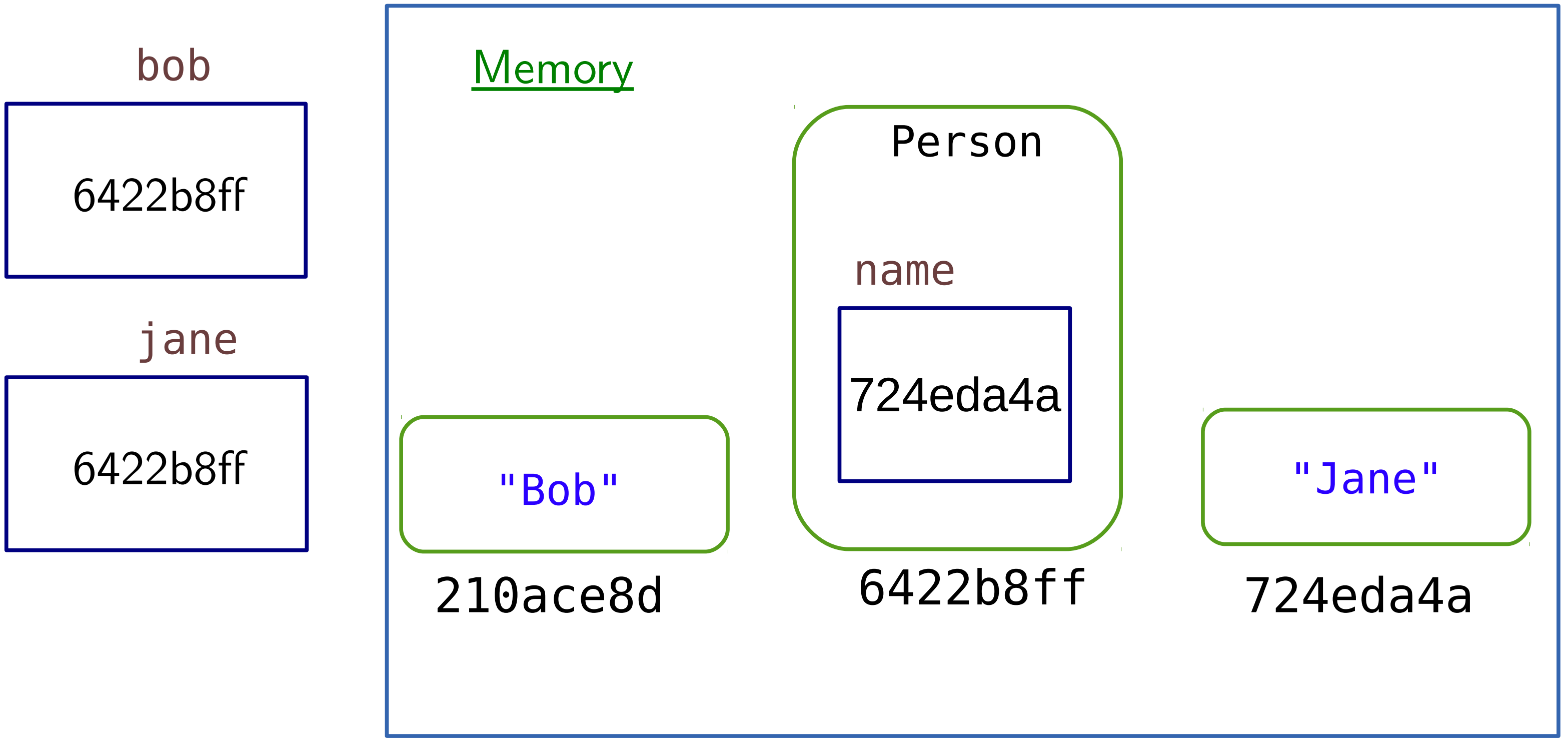
  - Result is: `Person@6422b8ff`

- Java is telling us that `bob` contains the memory location 6422b8ff*, where a Person object is stored.

*Not *quite* true in Java.

Reference Example Revisited

```
Person bob;
Person jane;
bob = new Person("Bob");
jane = bob;
jane.setName("Jane");
System.out.println(bob.getName());
```



The Moral...

- Reference variables aren't <u>so</u> different from primitive variables.
  - Both store one smallish piece of information (8-64 bits)
- Assignment (=) copies contents.
- Equality operator (==) compares contents.

Passing Arguments

```java
public static void makeChanges(Person person, int age)
{
    person.setName("Thor");
    age = age + 10;
}
```

```java
public static void argDemo()
{
    Person bob;
    int bobsAge;

    bob = new Person("Bob");
    bobsAge = 21;

    makeChanges(bob, bobsAge);
    System.out.println(bob.getName());
    System.out.println(bobsAge);
}
```

Is this code syntactically correct?
What will be printed?

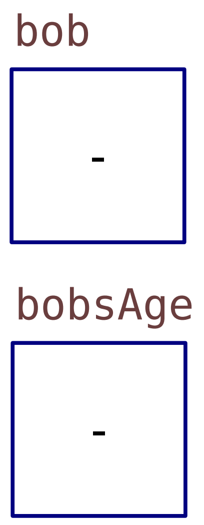
Passing Arguments

```
public static void makeChanges(Person person, int age)
{
    person.setName("Thor");
    age = age + 10;
}
```

```
public static void argDemo()
{
    Person bob;
    int bobsAge;

    bob = new Person("Bob");
    bobsAge = 21;

    makeChanges(bob, bobsAge);
    System.out.println(bob.getName());
    System.out.println(bobsAge);
}
```



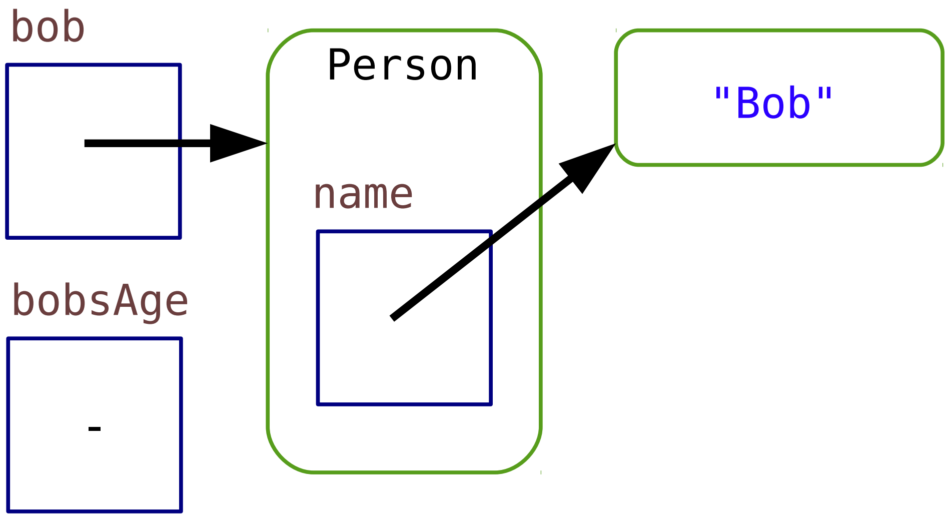
Passing Arguments

```
public static void makeChanges(Person person, int age)
{
    person.setName("Thor");
    age = age + 10;
}
```

```
public static void argDemo()
{
    Person bob;
    int bobsAge;

    bob = new Person("Bob");
    bobsAge = 21;

    makeChanges(bob, bobsAge);
    System.out.println(bob.getName());
    System.out.println(bobsAge);
}
```



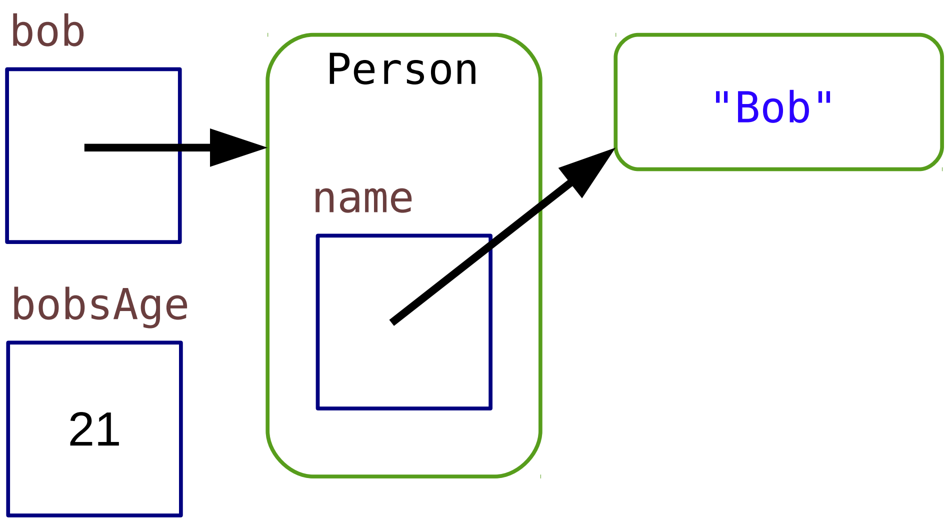
Passing Arguments

```
public static void makeChanges(Person person, int age)
{
    person.setName("Thor");
    age = age + 10;
}
```

```
public static void argDemo()
{
    Person bob;
    int bobsAge;

    bob = new Person("Bob");
    bobsAge = 21;

    makeChanges(bob, bobsAge);
    System.out.println(bob.getName());
    System.out.println(bobsAge);
}
```



Passing Arguments

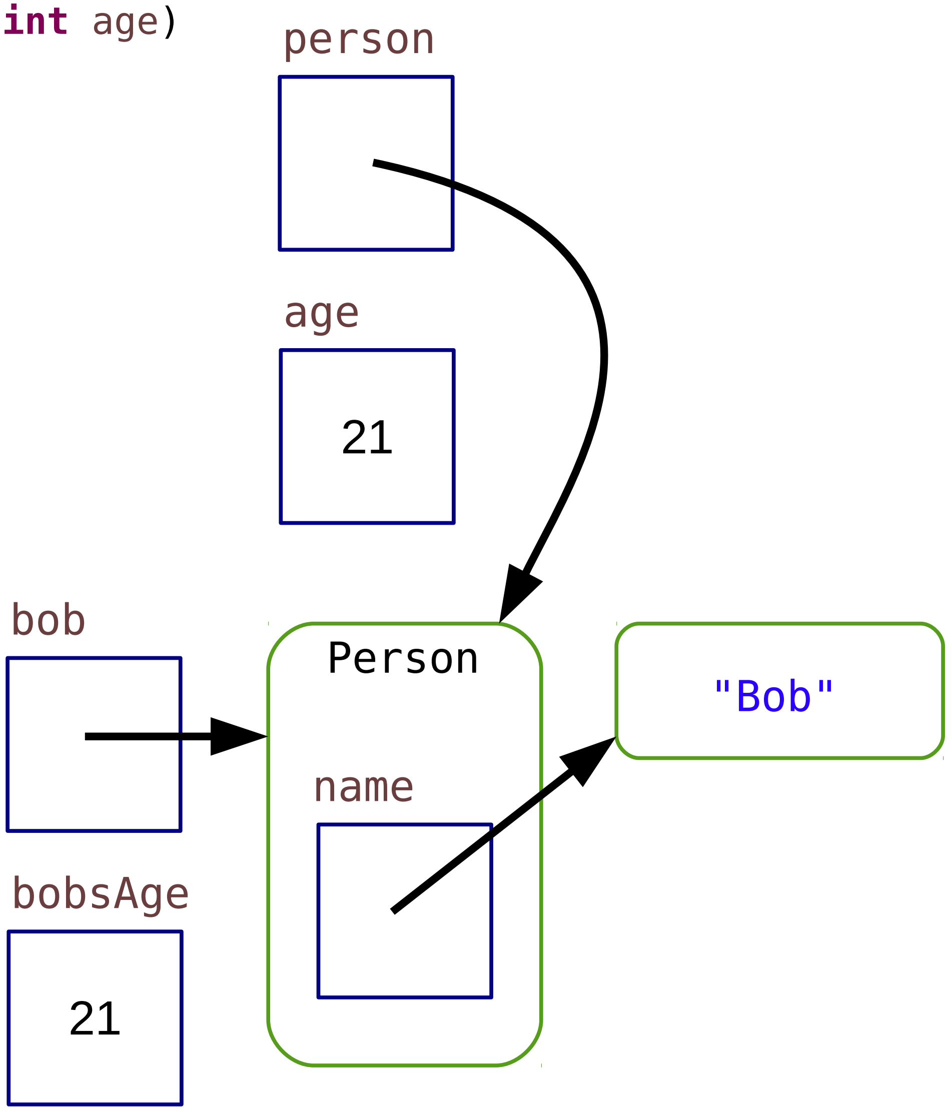
```
public static void makeChanges(Person person, int age)
{
    person.setName("Thor");
    age = age + 10;
}



public static void argDemo()
{
    Person bob;
    int bobsAge;

    bob = new Person("Bob");
    bobsAge = 21;

    makeChanges(bob, bobsAge);
    System.out.println(bob.getName());
    System.out.println(bobsAge);
}
```



Passing Arguments

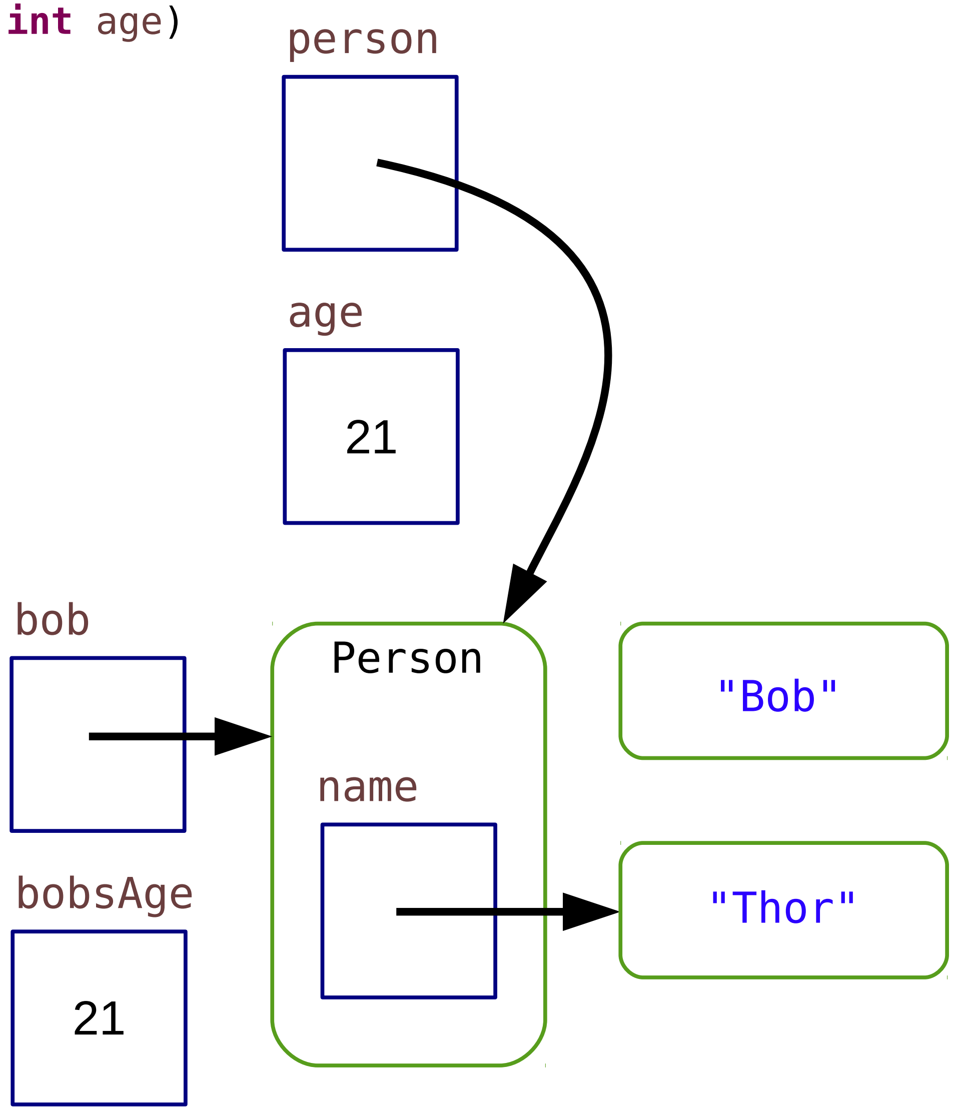
```
public static void makeChanges(Person person, int age)
{
    person.setName("Thor");
    age = age + 10;
}



public static void argDemo()
{
    Person bob;
    int bobsAge;

    bob = new Person("Bob");
    bobsAge = 21;

    makeChanges(bob, bobsAge);
    System.out.println(bob.getName());
    System.out.println(bobsAge);
}
```



Passing Arguments

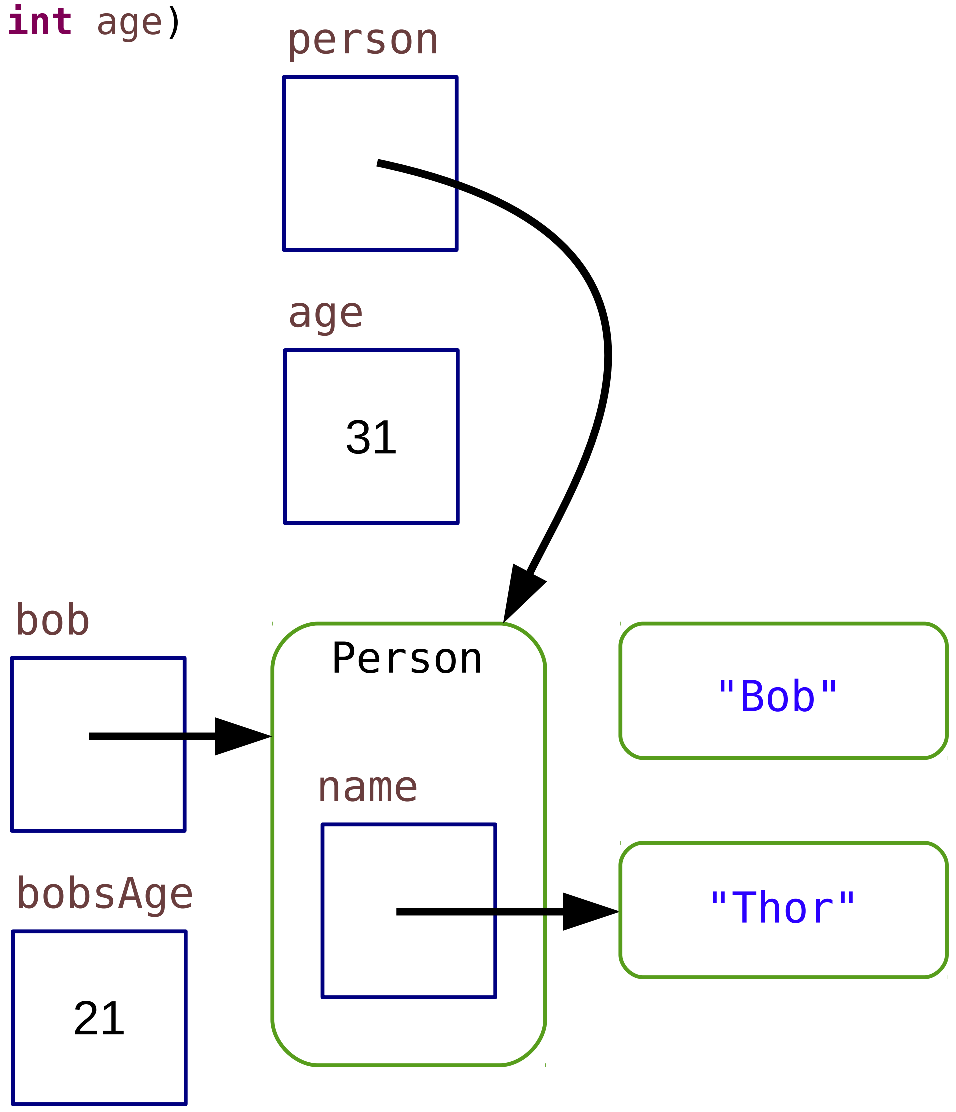
```
public static void makeChanges(Person person, int age)
{
    person.setName("Thor");
    age = age + 10;
}




public static void argDemo()
{
    Person bob;
    int bobsAge;

    bob = new Person("Bob");
    bobsAge = 21;

    makeChanges(bob, bobsAge);
    System.out.println(bob.getName());
    System.out.println(bobsAge);
}
```



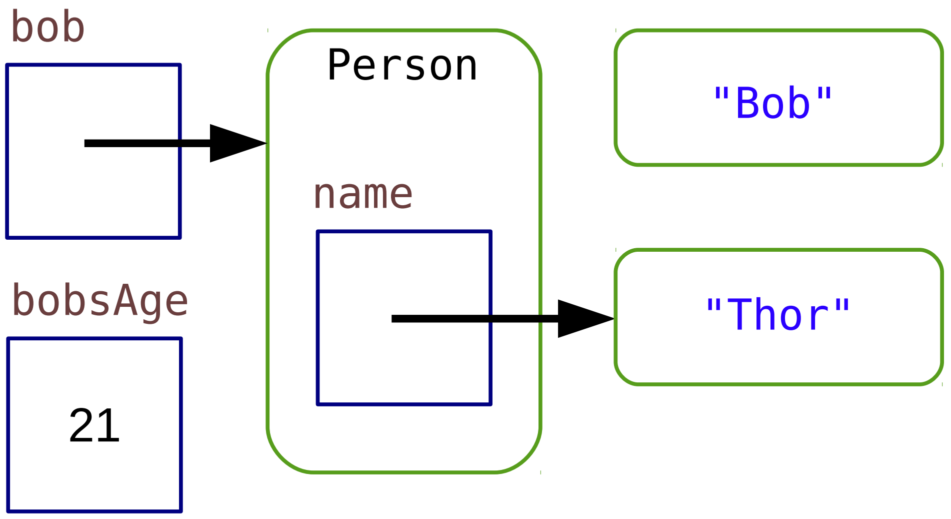
Passing Arguments

```
public static void makeChanges(Person person, int age)
{
    person.setName("Thor");
    age = age + 10;
}
```

```
public static void argDemo()
{
    Person bob;
    int bobsAge;

    bob = new Person("Bob");
    bobsAge = 21;

    makeChanges(bob, bobsAge);
    System.out.println(bob.getName());
    System.out.println(bobsAge);
}
```


bob
Person
"Bob"
name
bobsAge
21
"Thor"