What is an object?

An object is an encapsulation of one or more values of primitive types and user defined types.

What is a primitive type? Give an example.

What is a user-defined type (programmer-defined type)? Give an example.

So, an object is an encapsulated entity with state information where state information is the collective sum of the values of all of its fields.

For example, if we have a class named Account2 with three fields (inside Account2.java):

```java
private int balance;
private int accountNumber;
private String owner;
```

then, an instance (object) of Account2, say jimsAccount, could be created as follows (inside the main of UseAccount2.java, so jimsAccount is a local variable in main):

```java
Account2 jimsAccount = new Account2(2000, 46, "Jim Jones");
```

assuming that a constructor with three parameters were included in the definition of the Account2 class, it would then create an object consisting of the following values graphically represented as follows:

```
jimsAccount:
+---------------+-----------------+
| balance | 2000 |
+---------------+-----------------+
+---------------+-----------------+
| accountNumber | 46 |
+---------------+-----------------+
+---------------+-----------------+
| owner | "Jim Jones" |
+---------------+-----------------+
```

Here, three values (2000, 46, "Jim Jones") collectively represent the state of the object named jimsAccount at one point in time until something changes again in the object.

The state information can be updated as you interact with the object. For example, you can ’deposit’ into the account object or ’withdraw’ from it. As you do, the state information will change accordingly.

You can create as many objects as you wish using the same exact class definition (remember I called it a blueprint?). Say, we want to create another one, say, named JanesAccount. We would do the following to create it:

```java
Account2 JanesAccount = new Account2(3000, 47, "Jane Smith");
```

And it would look like the following. I am going to draw it more accurately this time than I did with jimsAccount above:
In this drawing, I am assuming that the value (3000) of the first field, i.e., `balance`, is located in memory location 568, an arbitrary memory address that I took out of the hat. The value (47) of the second field (`accountNumber`) will follow immediately after the first value, and the value of the reference into the string "Jane Smith" following immediately after the second value. You know how many bits it would take to represent the value 3000 (it is an `int`), and how many for 47 (it is also an `int`), and then the reference into the string "Jane Smith" (it is an object reference). Remember that a string value in Java is an object too. So, you would not see the value of "Jane Smith" in the location where owner field is supposed to be in, directly – you will instead see an object reference and you would have to follow the reference to find the actual string value. I am assuming that the actual value of "Jane Smith" is located starting in memory address 688. Actually, a string is represented as an array of `char`s. So, more accurate picture would look like this: