(227-139) WK 33

14 FRIDAY • AUGUST • 2020

JULY 2

S	M	T	W	T	F
			1	2	3
5	6	7	8	9	10
12	13	14	15	16	17
19	20	21	22	23	24
26	27	28	29	30	31

Object Oriented Programming

A class is a template for an object, and an object is an instance of a class.

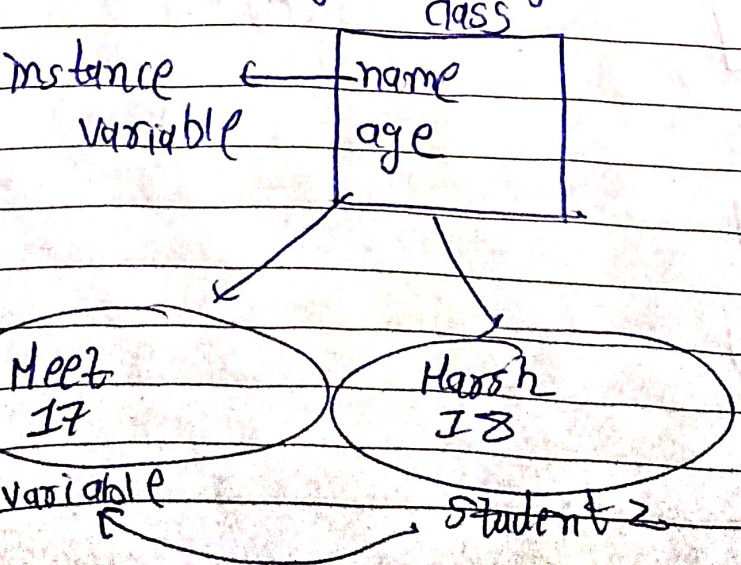
→ A class creates a new datatype that can be used to create objects.

→ Class ⇒ logical construct.
Object ⇒ Physical reality (occupies some space in memory)

Objects

- State → value from its datatype
- Identity → distinguishes one Object from another.
- Behaviour → effect of data-type operations

The dot operator links the name of the object with the name of an instance variable. Although commonly referred to as the dot operator, the formal specification for java categorizes the '.' as a separator



The 'new' keyword dynamically allocates (that is, allocates at runtime) memory for an object & returns a reference to it.

→ This reference is, more or less, the address in memory of the object allocated by new.
→ This reference is then stored in the variable.
→ Thus, in java, all class objects must be dynamically allocated.

```
Box mybox;   // declare reference to object
mybox = new Box(); // allocate a Box object
```

→ The first line declares my box as a reference to an object of type Box. At this point, mybox, does not yet refer to an actual object.

→ The next line allocates an object and assigns a reference to it to mybox. After the second line executes, you can use mybox as if it were a Box object. But in reality mybox simply holds, in essence, the memory address of the actual Box object.

→ The key to java's safety is that you cannot manipulate references as you can actual pointers.
→ Thus you cannot cause an object reference to point to an arbitrary memory location or manipulate it like an integer.

↳ are stored in stack memory
objects → heap memory

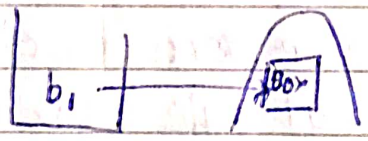
SUNDAY 16

class-name class-var = new classname ();

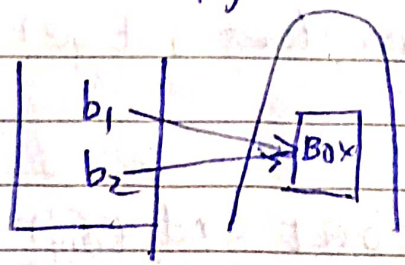
allocating memory at runtime.

constructor
↳ what occurs when an object of a class is created.

Box b_1 = new Box ();



Box b_2 = b_1;



Box b_1 = new Box ();

Compiler

JVM

The "this" Keyword :

→ Sometimes a method will need to refer to the object that invoked it. To allow this, Java defines the this keyword.

this can be used inside any method to the current object. That is this is always a reference to the object on which the method was invoked.

final Keyword:

A field can be declared as final. Doing so prevents its contents from being modified, making it essentially a constant.
→ This means that you must initialize a final field when it is declared.

→ It is a common coding convention to choose all uppercase identifiers for final fields:

```
final int FINAL = 2;
```

→ Unfortunately, final guarantees immutability only when instance variables are primitive types, not reference types.

→ If an instance variable of a reference type has the final modifier, the value of that instance variable (the reference to an object) will never change - it will always refer to the same object - but the value of the object itself can change.

The finalize() Method:

You can define specific actions that will occur when an object is just about to be reclaimed by the garbage collector.

Constructors :-

→ Once defined, the constructor is automatically called when the object is created, before the new operator completes.

→ Constructors look a little strange because they have no return type, not even void. This is because the implicit return type of a class constructor is the class type itself.