Shubham Gupta

Disc#114

Tuesdays 10-11 am

Wheeler 224

Lost

bit.ly/lost_disc2

Primitive	Reference
- byte, int, double, float, etc.	- everything Ex: object, Strings, Arrays.

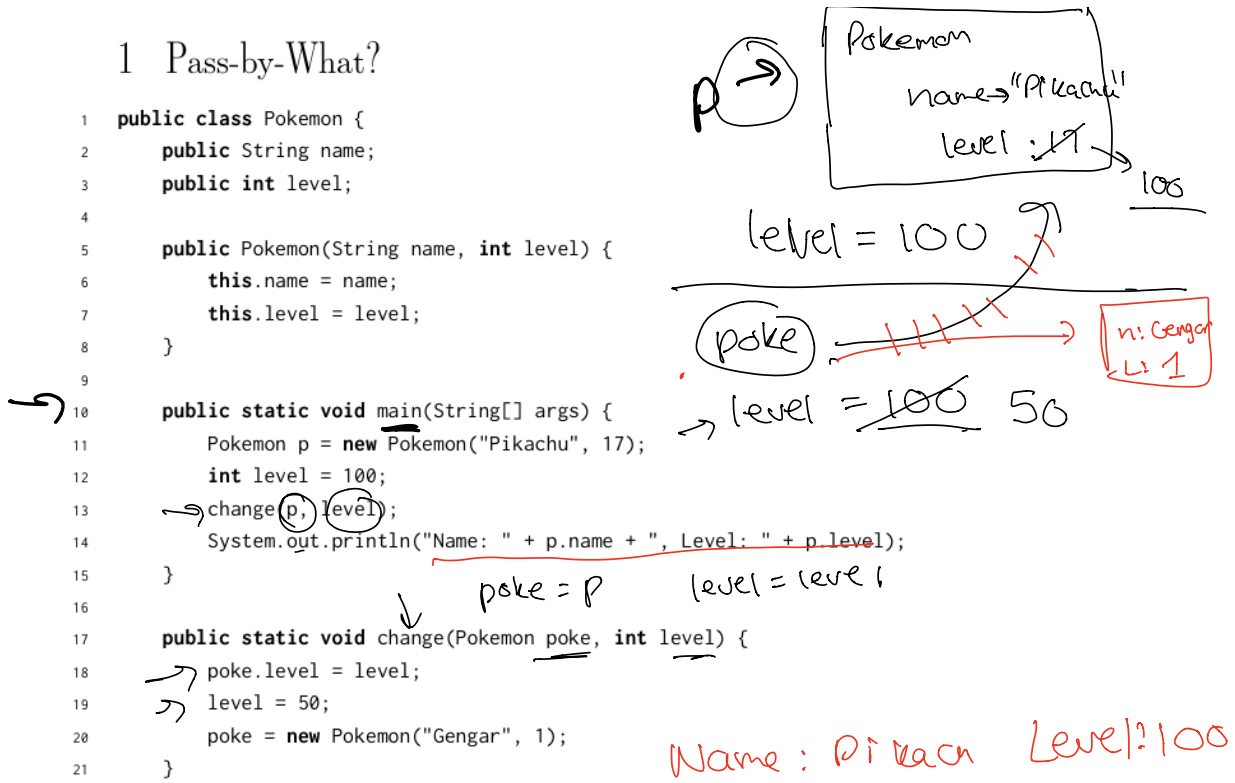
Golden Rule of =

- whenever you do "a = b" copy bits of data from b to a.
- Primitive Types: Copying the value
- reference Type: Copying memory address

shubscode.github.io/teaching

1 Pass-by-What?

```
1  public class Pokemon {
2      public String name;
3      public int level;
4
5      public Pokemon(String name, int level) {
6          this.name = name;
7          this.level = level;
8      }
9
10     public static void main(String[] args) {
11         Pokemon p = new Pokemon("Pikachu", 17);
12         int level = 100;
13         change(p, level);
14         System.out.println("Name: " + p.name + ", Level: " + p.level);
15     }
16
17     public static void change(Pokemon poke, int level) {
18         poke.level = level;
19         level = 50;
20         poke = new Pokemon("Gengar", 1);
21     }
22 }
```



1.1 (a) What would Java display?

(b) Draw the box-and-pointer diagram after Java evaluates the `main` method.

(c) On line 19, we set `level` equal to `50`. What `level` do we mean? An instance variable of the `Pokemon` class? The local variable containing the parameter to the `change` method? The local variable in the `main` method? Something else?

Static	Non Static

2 Static Methods and Variables

```
public class Cat {
   public String name;
   public static String noise;

   public Cat(String name, String noise) {
       this.name = name;
       this.noise = noise;
   }

   public void play() {
      System.out.println(noise + " I'm " + name + " the cat!");
   }

   public static void anger() {
      noise = noise.toUpperCase();
   }
   public static void calm() {
      noise = noise.toLowerCase();
   }
}
```

shared by everyone

this.name

Cat a = Cat(Al, "Meow"

Cat b = Cat(Bob, "bark")

a.noise

Static methods invoked via Cat class

ex. Cat.anger()

```
public class Dog {
    //instance varidbles
public int age;
public String breed;

    //constructer
    public Dog(String breed, int age){
        this.age = age;

    }
    public Dog( ) {
        age = 0;
        breed = "--";

    //methods
    public void play( ) {
```

Dog a = new Dog(--

name

return type

args

access modifier

static

Math.pow( )

Math.abs( )

Math a = new Math;

```
public Static void main(String[] args)
{
    System.*

}
```

java Hello

2 Static Methods and Variables

```java
public class Cat {
   public String name;
   public static String noise;

   public Cat(String name, String noise) {
       this.name = name;
       this.noise = noise;
   }

   public void play() {
      System.out.println(noise + " I'm " + name + " the cat!");
   }

   public static void anger() {
      noise = noise.toUpperCase();
   }
   public static void calm() {
      noise = noise.toLowerCase();
   }
}
```

Cat
noise: ~~"Meow"~~ ~~"Nyan"~~ ~~NYAN~~ nyan

a → Cat n: "Cream"

b → Cat n: "Tubbs"

Cat.play()

2.1 Write what will happen after each call of `play()` in the following method.

```java
public static void main(String[] args) {
    Cat a = new Cat("Cream", "Meow!");
    Cat b = new Cat("Tubbs", "Nyan!");
    a.play();
    b.play();
    Cat.anger();
    a.calm();
    a.play();
    b.play();
}
```

Nyan! I'm Cream the Cat!
Nyan! I'm Tubbs the Cat!

nyan! I'm Cream the Cat!
nyan! I'm Tubbs the Cat!

3 Practice with Linked Lists

3.1 Draw the box-and-pointer diagram that results from running the following code. A `StringList` is similar to an `IntList`. It has two instance variables, `first` and `rest`.

```
StringList L = new StringList("eat", null);
L = new StringList("shouldn't", L);
L = new StringList("you", L);
L = new StringList("sometimes", L);
StringList M = L.rest;
StringList R = new StringList("many", null);
R = new StringList("potatoes", R);
R.rest.rest = R;
M.rest.rest.rest = R.rest;
L.rest.rest = L.rest.rest.rest;
L = M.rest;
```

java visualizer

princeton