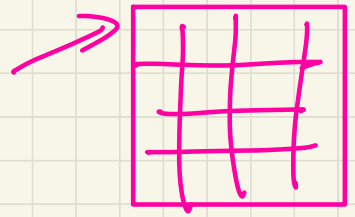
EECS1022 Programming for Mobile Computing (Winter 2021)

Q&A - Lectures W7

Monday, March 8

Weekly Tutorials



OOP. { W6, W7, W8

W9 → Lab7, Lab8

W9 → Mobile Computing. Android Studio

Eclipse → W10, W11 → 2d-array.

W12 → API.

- Practice Programming Test 3 (challenging).
- Practice Written Test 3 (aliasing)
- Lab6

For Constructor methods in a class, we are supposed to create different constructor method for each calling scenario (of course with different input types and their order).

Consider:

```
public class c_name{
  int c_var1;
  boolean c_var2 = false;
  public c_name(int num1, boolean bool2){
    c_var1 = num1;
    c_var2 = bool2;
  }
}
```

What if one calling wants to omit `bool2` input and give it a default value? Seems we need to create another constructor method involving only `int num1` parameter.

And also, what to do if the caller will possibly give arbitrary length of arguments to a class instance? Any syntax related to this scenario?

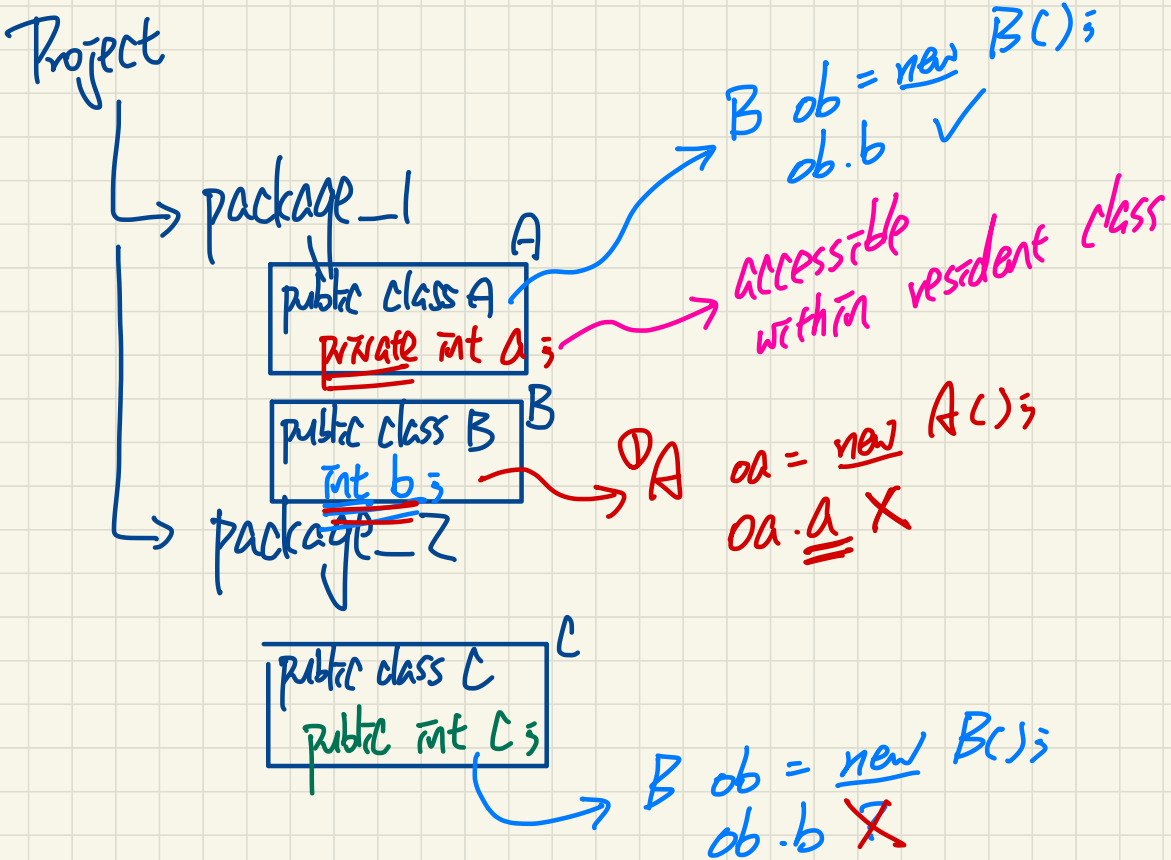
Thanks!

initialization of each attribute should occur in constructors.

public c_name(int num1){ then, why declare 2nd param in the first place.

overloading
- names same
- lists of parameter types distinct.

new c_name(___, ___, ___);

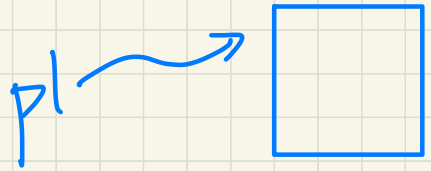

Project
package_1
A
public class A
private int a;
B ob = new B();
ob.b ✓
accessible within resident class
B
public class B
int b;
A oa = new A();
oa.a ✗
package_2
C
public class C
public int C;
B ob = new B();
ob.b ✗

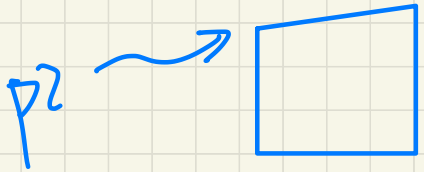
Written Test 2 - Practice Questions on Aliasing

1. Assume that a **Person** class is already defined, and it has an attribute **name** and a constructor that initializes the person's name from the input string. Consider the following fragment of Java code (inside some **main** method):

```
1  Person p1 = new Person("Heeyeon");
2  Person p2 = new Person("Jiyoon");
3  System.out.println(p1 != p2);
```

True.





Written Test 2 - Practice Questions on Aliasing

2. Assume that a **Person** class is already defined, and it has an attribute **name** and a constructor that initializes the person's name from the input string. Consider the following fragment of Java code (inside some **main** method):

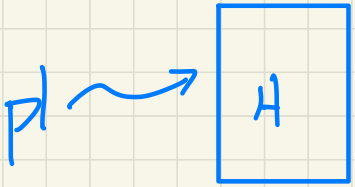
```
1  Person p1 = new Person("Heeyeon");
2  Person p2 = new Person("Jiyoon");
3  Person[] persons = new Person[2];
4  System.out.println(persons[persons.length()] != null);
```

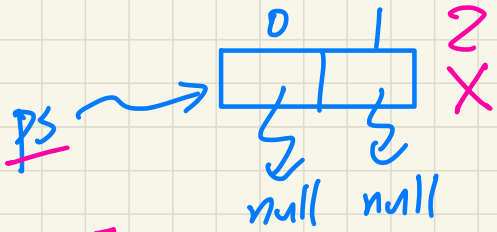
2 → compilation error!

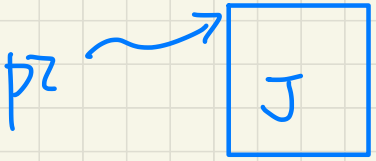
→ -NPE
→ -AIOBE-

ps[1].getName() → NPE

Tip 12 - SCE & Array Indexing.







ps[2]

Written Test 2 - Practice Questions on Aliasing

3. Assume that a **Person** class is already defined, and it has an attribute **name** and a constructor that initializes the person's name from the input string. Consider the following fragment of Java code (inside some **main** method):

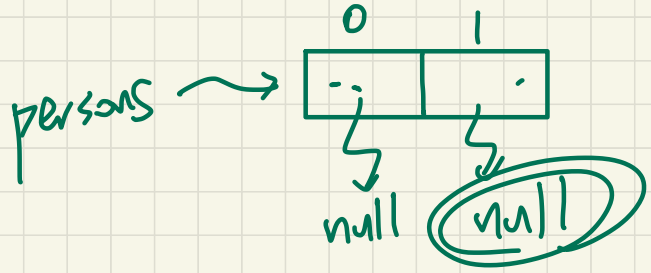
```
1  Person p1 = new Person("Heeyeon");
2  Person p2 = new Person("Jiyoon");
3  Person[] persons = new Person[2];
4  System.out.println(persons[persons.length] != null);
```

ARRAY OUT OF BOUNDS.

Written Test 4 - Practice Questions on Aliasing

4. Assume that a **Person** class is already defined, and it has an attribute **name** and a constructor that initializes the person's name from the input string. Consider the following fragment of Java code (inside some **main** method):

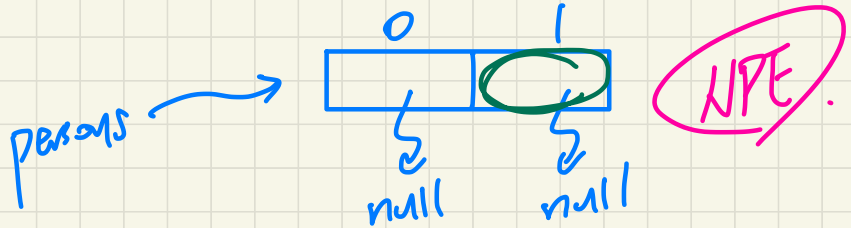
```
1  Person p1 = new Person("Heeyeon");
2  Person p2 = new Person("Jiyoon");
3  Person[] persons = new Person[2];
4  System.out.println(persons[persons.length - 1] != null);
```



Written Test 3 - Practice Questions on Aliasing

5. Assume that a **Person** class is already defined, and it has an attribute **name** and a constructor that initializes the person's name from the input string. Consider the following fragment of Java code (inside some **main** method):

```
1  Person p1 = new Person("Heeyeon");
2  Person p2 = new Person("Jiyoon");
3  Person[] persons = new Person[2];
4  System.out.println(persons[persons.length - 1].name.equals("Jiyoon"));
```

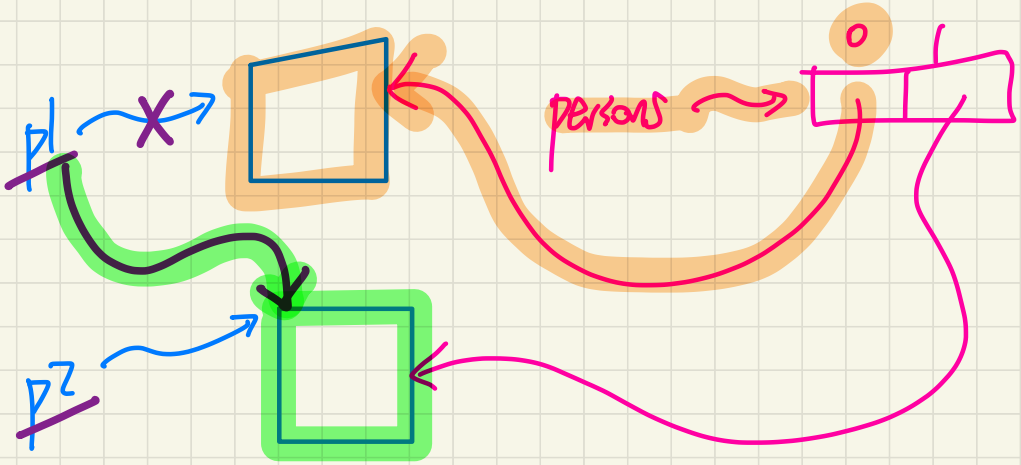


Written Test 3 - Practice Questions on Aliasing

6. Assume that a **Person** class is already defined, and it has an attribute **name** and a constructor that initializes the person's name from the input string. Consider the following fragment of Java code (inside some **main** method):

```
1 Person p1 = new Person("Heeyeon");
2 Person p2 = new Person("Jiyoon");
3 Person[] persons = {p1, p2};
4 p1 = p2;
5 System.out.println(persons[0] == p1);
```

0 1 False.

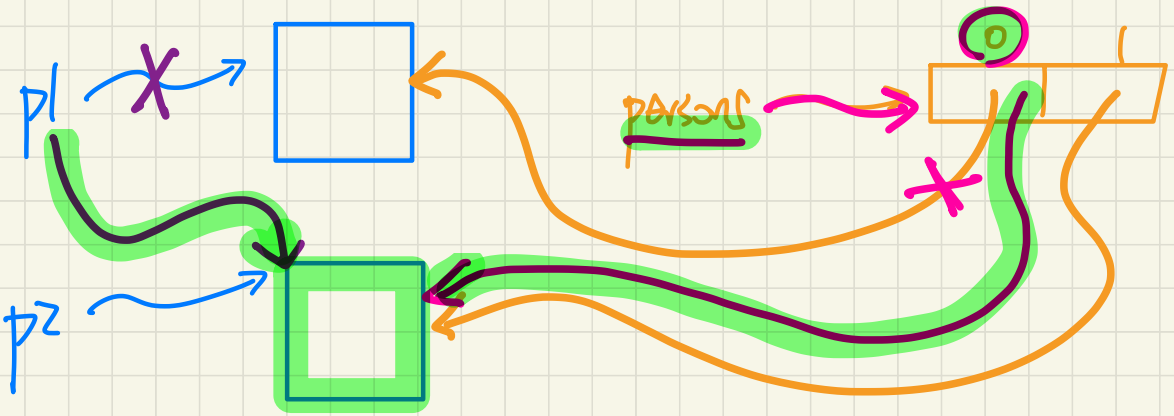


Written Test 3 - Practice Questions on Aliasing

7. Assume that a **Person** class is already defined, and it has an attribute **name** and a constructor that initializes the person's name from the input string. Consider the following fragment of Java code (inside some **main** method):

```
1  Person p1 = new Person("Heeyeon");
2  Person p2 = new Person("Jiyoon");
3  Person[] persons = {p1, p2};
4  p1 = p2;
5  persons[0] = p2;
6  System.out.println(persons[0] == p1);
```

Fake? True



Written Test 3 - Practice Questions on Aliasing

8. Assume that a **Person** class is already defined, and it has an attribute **name**, a constructor that initializes the person's name from the input string, and a mutator method **setName** that changes the person's name from the input string. Consider the following fragment of Java code (inside some **main** method):

```
1 Person p1 = new Person("Heeyeon");
2 Person p2 = new Person("Jiyoon");
3 Person[] persons = {p1, p2};
4 p1 = persons[1];
5 persons[0] = p2;
6 p2.setName("Jihye");
7 System.out.println(p1.name);
```