Lecture 5 - Sep 17

OOP Review

Exercise: Reference Aliasing & Arrays
Manual Management of Account IDs
Use of Static vs. Non-Static Variables

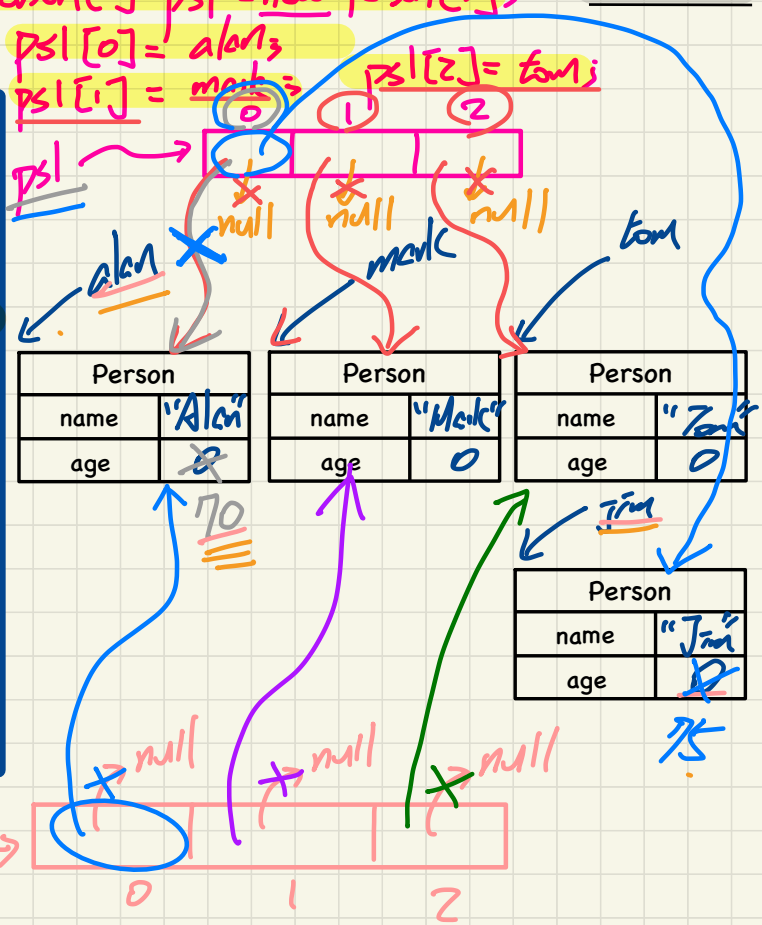
Announcements/Reminders

- Today's class: notes template posted
- Priorities:
  - + Review Lab0
  - + Complete Lab1
  - + Due: Next Tuesday (Sep 30)
- ProgTest0 (next Wednesday, Sep 24), Guide released
- Lab this Wednesday (Sep 17):
  - + Please go to your enrolled session.
  - + [≈ 40 min] Mock-up Programming Test 0
  - + [≈ 40 min] Q&As (TAs, Jackie)

Copying Reference Values: Aliasing


```
Person alan = new Person("Alan");
Person mark = new Person("Mark");
Person tom = new Person("Tom");
Person jim = new Person("Jim");
Person[] persons1 = {alan, mark, tom};
Person[] persons2 = new Person[persons1.length];
for(int i = 0; i < persons1.length; i ++) {
  persons2[i] = persons1[i]; }
persons1[0].setAge(70);
System.out.println(jim.getAge());
System.out.println(alan.getAge());
System.out.println(persons2[0].getAge());
persons1[0] = jim;
persons1[0].setAge(75);
System.out.println(jim.getAge());
System.out.println(alan.getAge());
System.out.println(persons2[0].getAge());
```

* Person[] ps1 = new Person[3];
ps1[0] = alan;
ps1[1] = mark;
ps1[2] = tom;



**

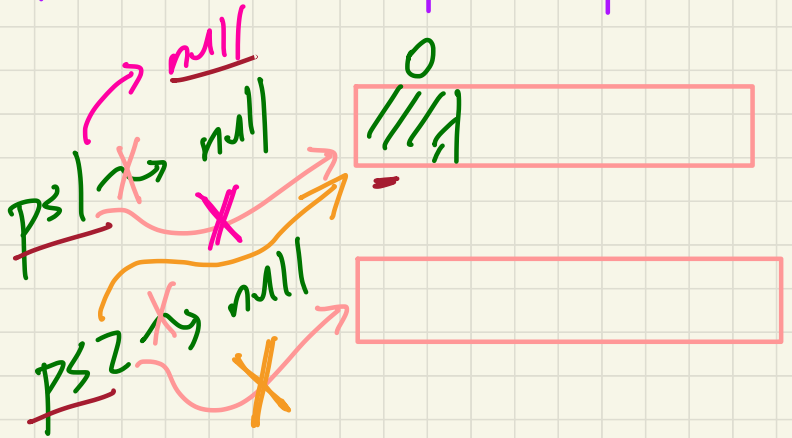
i	
0	ps2[0] = ps1[0];
1	ps2[1] = ps1[1];
2	ps2[2] = ps1[2];

Copying Arrays

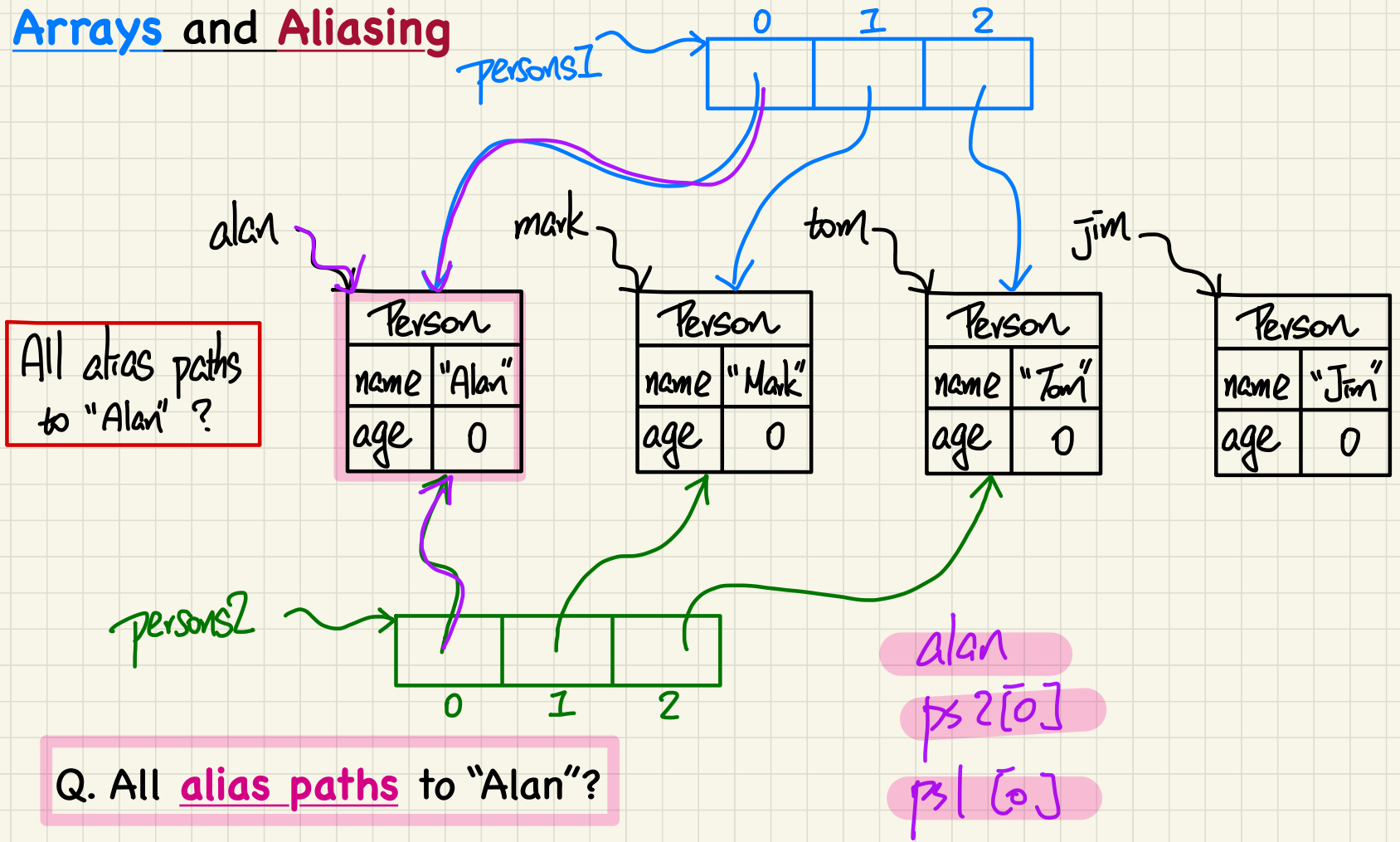
```
Person[] ps1;
Person[] ps2;        ①

ps1 = new ...

ps2 = new ...        ②

ps2 = ps1;           ③

ps1 = null;          ④
```

③ creating alias at the array level.

	①	②	③	④
ps1 == null	T	F	F	T
ps2 == null	T	F	F	F
ps1 == ps2	T	F	T	F.



Arrays and Aliasing



Managing Account IDs: Manual


```
public class Account {
  private int id;
  private String owner;
  public int getID() { return this.id; }
  public Account(int id, String owner) {
    this.id = id;
    this.owner = owner;
  }
}
```

goal: id's among Account objects are unique.

explicit parameter, presumably should not clash with existing id's.

Goal:
- auto. ID generation & management

```
class AccountTester {
  Account acc1 = new Account(1, "Jim");
  Account acc2 = new Account(1, "Jeremy");
  System.out.println(acc1.getID() != acc2.getID());
}
```

1. burden on the Account user to specify unique id's.

2. when id's are duplicated, not a compilation error (logical error)

Declaring Global Variables among Objects

* Each Counter object has its own copy (instance-specific value).

** All Counter objects share a single/unique copy (global var.).

```
public class Counter {
    private int l;
    static int g = 0;
    public Counter() {
        this.l = 0;
    }
    public int getLocal() {
        return this.l;
    }

    public void incrementLocal() {
        this.l ++;
    }

    public void incrementGlobal() {
        g ++;
    }
}
```

```
public class CounterTester {
    public static void main(String[] args) {
        Counter c1 = new Counter();
        Counter c2 = new Counter();

        System.out.println("c1's local: " + c1.getLocal());
        System.out.println("c2's local: " + c2.getLocal());
        System.out.println("Global accessed via c1: " + c1.g);
        System.out.println("Global accessed via c2: " + c2.g);
        System.out.println("Global accessed via Counter: " + Counter.g);

        c1.incrementLocal();
        c2.incrementLocal();
        c1.incrementGlobal();
        c2.incrementGlobal();
        Counter.g = Counter.g + 1; // Counter.global ++;
    }
}
```

Accessing static variables: ClassName.nameOfStaticVar

e.g. Counter.g