CSE 121 – Lesson 14

Miya Natsuhara
Autumn 2023

Music: 121 23au Lecture Tunes 🐐

TAs:
- Trey
- Christina
- Sahej
- Vinay
- Kriti
- Sebastian
- Colton
- Anju
- Maria
- Minh
- Annie
- Janvi
- Jonus
- Shreya
- Vivian
- Jasmine
- Arkita
- Lydia
- Andy
- Nicole
- Christian
- Vidhi
- Luke
- Nicolas
- Simon
- Lucas
- Ritesh
- Andras
- Shayna
- Jessie
- Logan
- Hibbah
- Archit
- Hannah
- Lydia
- Jacob
- Julia
- Ayesha
- Aishah
- Yijia

sli.do #cse121
Announcements, Reminders

- **C3 deadline** extended to Tuesday, Nov 28!
- Gumball (& friends) Visit on Monday, December 11 1:00pm-3:00pm
- Next week:
  - Tuesday: section cancelled
  - Wednesday: lecture optional (Lead TA Nic will be teaching!)
  - Thursday, Friday: University holidays (campus is closed)
  - IPL: reduced staffing Mon, Tues, Wed (maybe partial day); closed Thurs-Sun
- Final Exam: **Wednesday, Dec 13 12:30pm-2:20pm**
What would the array `a` store at the end of this arrayMystery method if 
{\(-20, 20, 26, 32, 50, 3\)} was passed in?

public static void arrayMystery(int[] a) {
    for (int i = a.length - 1; i >= 1; i--) {
        if (a[i] > a[i - 1] + 10) {
            a[i - 1] = a[i - 1] + 5;
        }
    }
}
(PCM) Value Semantics vs. Reference Semantics

- Applies when working with primitive types
- Variables/parameters hold a *copy* of the actual value

- Applies when working with objects
- Variables/parameters hold a *reference* to the object

```java
age = 10;

name: age (int)
```

```java
int[] arr = new int[4];

name: arr (int[])
```
**Value Semantics vs. Reference Semantics**

```java
int a = 3;
int b = a;
a = 99;

int[] list1 = {4, 8, 15, 16, 23};
int[] list2 = list1;
list1[1] = 99;
```

```
age = 10;

name: age (int)

int[] arr = new int[4];

name: arr (int[])
```

Lesson 14 - Autumn 2023
boolean test = true;
flipValue(test);

public static void flipValue(boolean b) {
    b = !b;
}

boolean[] tests = {true, true, false, true, false, false};
flipValues(tests);

public static void flipValues(boolean[] b) {
    for (int i = 0; i < b.length; i++) {
        b[i] = !b[i];
    }
}
(PCM) null

The absence of a reference!
   Sort of like a "zero-equivalent" for references!
   Default value for object types

NullPointerExceptions occur when you try to ask a reference do something, but there isn't a reference there...it's null!
   "do something" such as get an element from an array, or call a method on an object)
public static void main(String[] args) {
    int x = 0;
    int[] a = new int[4];
    x++;

    mystery(x, a);
    System.out.println(x + " " + Arrays.toString(a));

    x++;
    mystery(x, a);
    System.out.println(x + " " + Arrays.toString(a));
}

public static void mystery(int x, int[] a) {
    x++;
    a[x]++;
    System.out.println(x + " " + Arrays.toString(a));
}
public static void main(String[] args) {
    int x = 0;
    int[] a = new int[4];
    x++;

    mystery(x, a);
    System.out.println(x + " " + Arrays.toString(a));

    x++;
    mystery(x, a);
    System.out.println(x + " " + Arrays.toString(a));
}

public static void mystery(int x, int[] a) {
    x++;
    a[x]++;
    System.out.println(x + " " + Arrays.toString(a));
}