Wednesday February 27

Lecture 14

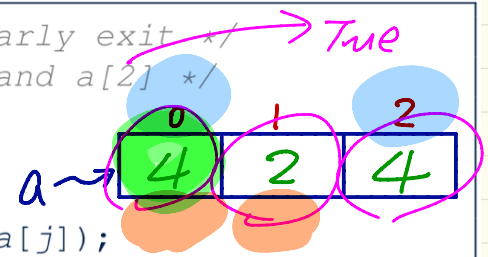
Finding Duplicates: No Duplicates, Redundant Scan

```
1  /* Version 1 with redundant scan */
2  int[] a = {1, 2, 3}; /* no duplicates */
3  boolean hasDup = false;
4  for(int i = 0; i < a.length; i ++) {
5    for(int j = 0; j < a.length; j ++) {
6      hasDup = hasDup || (i != j && a[i] == a[j]);
7    } /* end inner for */ } /* end outer for */
8  System.out.println(hasDup);
```

i	j	i != j	a[i]	a[j]	a[i] == a[j]	hasDup
0	0	false	1	1	true	false
0	1	true	1	2	false	false
0	2	true	1	3	false	false
1	0	true	2	1	false	false
1	1	false	2	2	true	false
1	2	true	2	3	false	false
2	0	true	3	1	false	false
2	1	true	3	2	false	false
2	2	false	3	3	true	false

Finding Duplicates : With Duplicates, Redundant Scan with no Early Exit

```
1  /* Version 1 with redundant scan and no early exit */
2  int[] a = {4, 2, 4}; /* duplicates: a[0] and a[2] */
3  boolean hasDup = false;
4  for(int i = 0; i < a.length; i ++) {
5    for(int j = 0; j < a.length; j ++) {
6      hasDup = hasDup || (i != j && a[i] == a[j]);
7    } /* end inner for */ } /* end outer for */
8  System.out.println(hasDup);
```



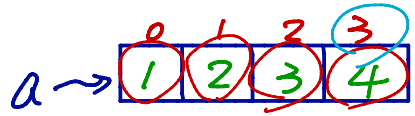
i	j	i != j	a[i]	a[j]	a[i] == a[j]	hasDup
0	0	false	4	4	true	false
0	1	true	4	2	false	false
0	2	true	4	4	true	true
1	0	true	2	4	false	true
1	1	false	2	2	true	true
1	2	true	2	4	false	true
2	0	true	4	4	true	true
2	1	true	4	2	false	true
2	2	false	4	4	true	true

N!

Finding Duplicates: No Duplicates, Non-Redundant Scan

have not found dup.

```
1  /* Version 3 with no redundant scan */
2  int[] a = {1, 2, 3, 4}; /* no duplicates */
3  boolean hasDup = false;
4  for(int i = 0; i < a.length && !hasDup; i ++) {
5    for(int j = i + 1; j < a.length && !hasDup; j ++) {
6      hasDup = a[i] == a[j];
7    } /* end inner for */ } /* end outer for */
8  System.out.println(hasDup);
```



0 1
0 2
0 3

1 2
1 3

(0,0) X 3 comb.

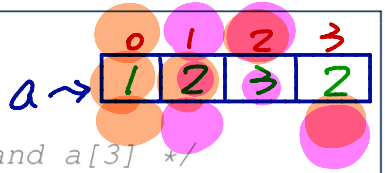
(1,0) X 2 comb.

1 comb.

i	j	a[i]	a[j]	a[i] == a[j]	hasDup
0	1	1	2	false	false
0	2	1	3	false	false
0	3	1	4	false	false
1	2	2	3	false	false
1	3	2	4	false	false
2	3	3	4	false	false

Finding Duplicates : With Duplicates, Non-Redundant Scan with Early Exit

```
1  /* Version 3 with no redundant scan:
2   * array with duplicates causes early exit
3   */
4  int[] a = {1, 2, 3, 2}; /* duplicates: a[1] and a[3] */
5  boolean hasDup = false;
6  for(int i = 0; i < a.length && !hasDup; i ++) {
7    for(int j = i + 1; j < a.length && !hasDup; j ++) {
8      hasDup = a[i] == a[j];
9    } /* end inner for */ } /* end outer for */
10 System.out.println(hasDup);
```



i	j	a[i]	a[j]	a[i] == a[j]	hasDup
0	1	1	2	false	false
0	2	1	3	false	false
0	3	1	2	false	false
1	2	2	3	false	false
1	3	2	2	true	true

we found a duplicate

Common Errors (1) Improper Initialization of Loop Counter

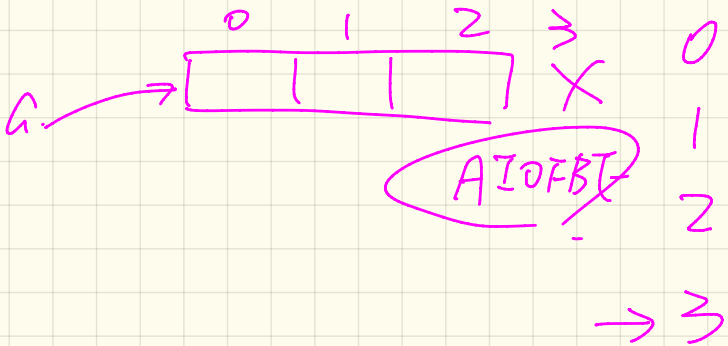
```
boolean userWantsToContinue = false;
while (userWantsToContinue) {
  /* some computations here */
  String answer = input.nextLine();
  userWantsToContinue = answer.equals("Y");
}
```

false by default

default value

Common Errors (2) Improper Stay Condition

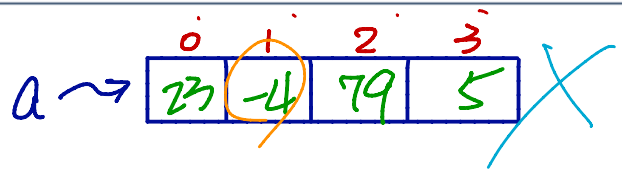
```
for (int i = 0; i <= a.length; i ++) {
  System.out.println(a[i]);
}
```



Common Errors (3) Improper Update to Loop Counter

Does the following loop print all slots of array `a`?

```
int i = 0;
while (i < a.length) {
    i ++;
    System.out.println(a[i]);
}
```



i	a[i]
→ 0	
1	a[1] -4
2	a[2] 79
3	a[3] 5
4	a[4] X

AIOBE.

Common Errors (4) Improper Update of Stay Condition

```
1 String answer = input.nextLine();
2 boolean userWantsToContinue = answer.equals("Y");
3 while (userWantsToContinue) { /* stay condition (SC) */
4   /* some computations here */
5   answer = input.nextLine();
6 }
```

"N"

userWantsToContinue = answer.equals("Y");

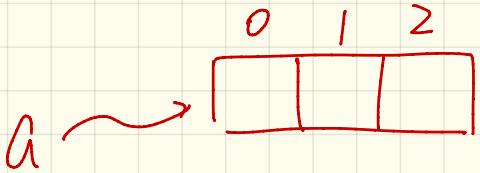
Common Errors (5) Improper Start Value of Loop Counter

```
int i = a.length - 1;
while (i >= 0) {
  System.out.println(a[i]); i --; }
while (i < a.length) {
  System.out.println(a[i]); i ++; }
```

2

when exiting the 1st loop i == -1

-1 < 3 T



i

2

1

0 → 0 >= 0 T

-1

a[-1] → AIOOBE

Common Errors (6) Misplaced Semicolon

Semicolon (;) in Java marks *the end of a statement* (e.g., assignment, `if` statement, `for`, `while`).

```
int[] ia = {1, 2, 3, 4};
for (int i = 0; i < 10; i ++); {
  System.out.println("Hello!");
}
```

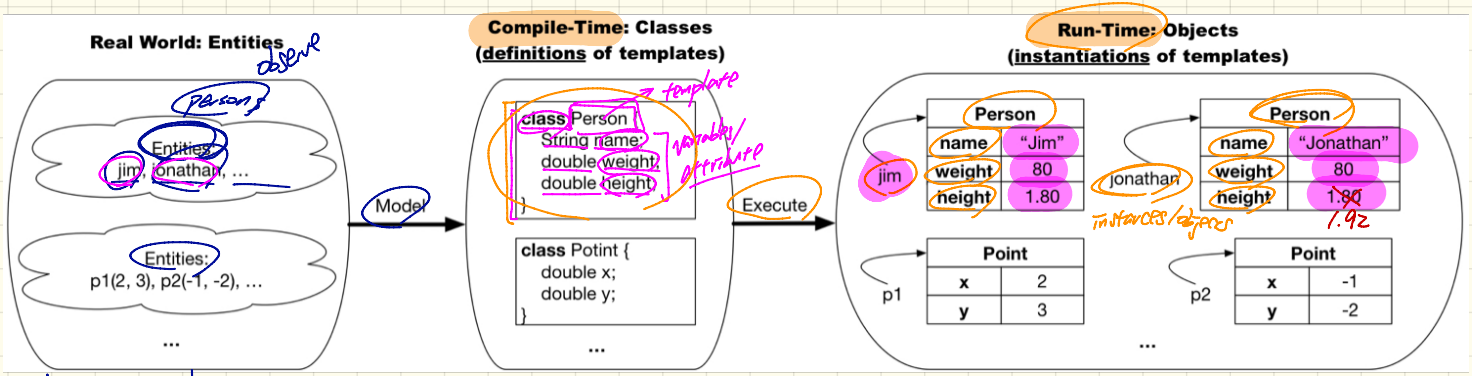
body of for loop is empty

not the body of for loop.

Output?

Hello.

Entities, Classes, Objects



object-oriented