Friday April 5

Review Lecture

Hashtable

- 2-column table
- keys contain no duplicates
- values may contain duplicates
- a key is used to identify a row

```
grades.put("Alan", "B");
grades.put("Alan", "C");
```

keys	values
"Alan"	~~"B"~~ "C"

name

Birthday

Hashtable<String, > book =

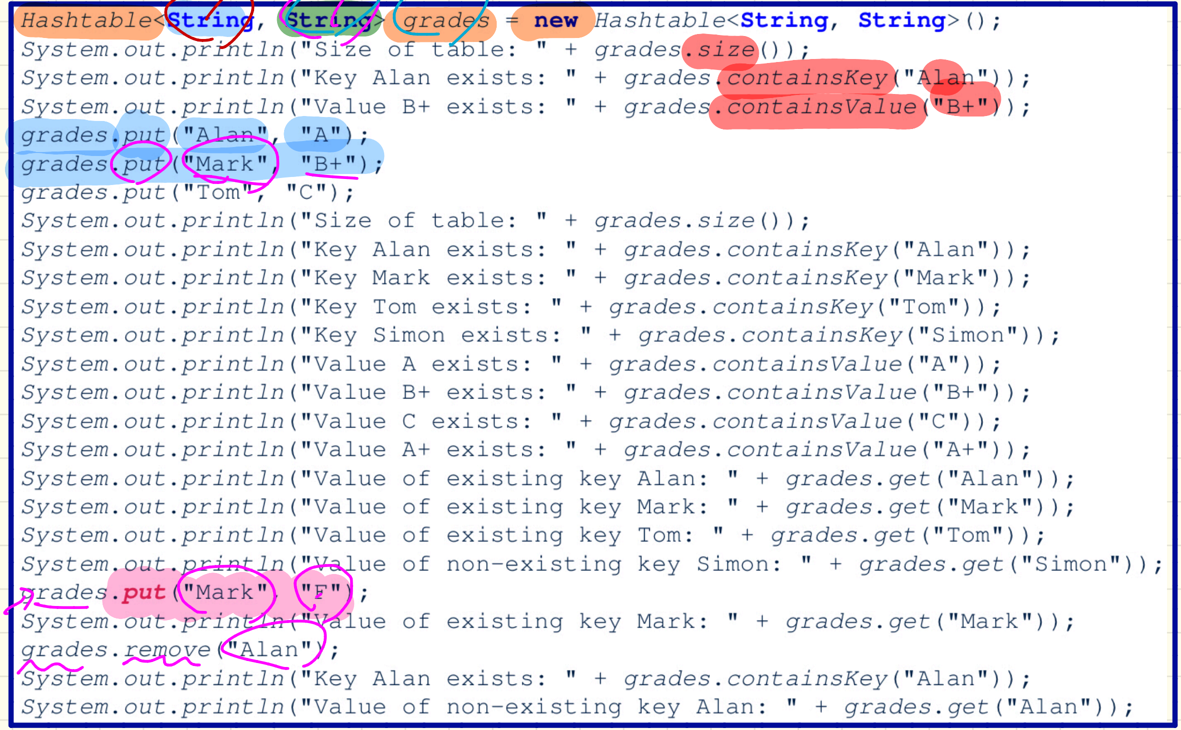
$ $

ks vs new

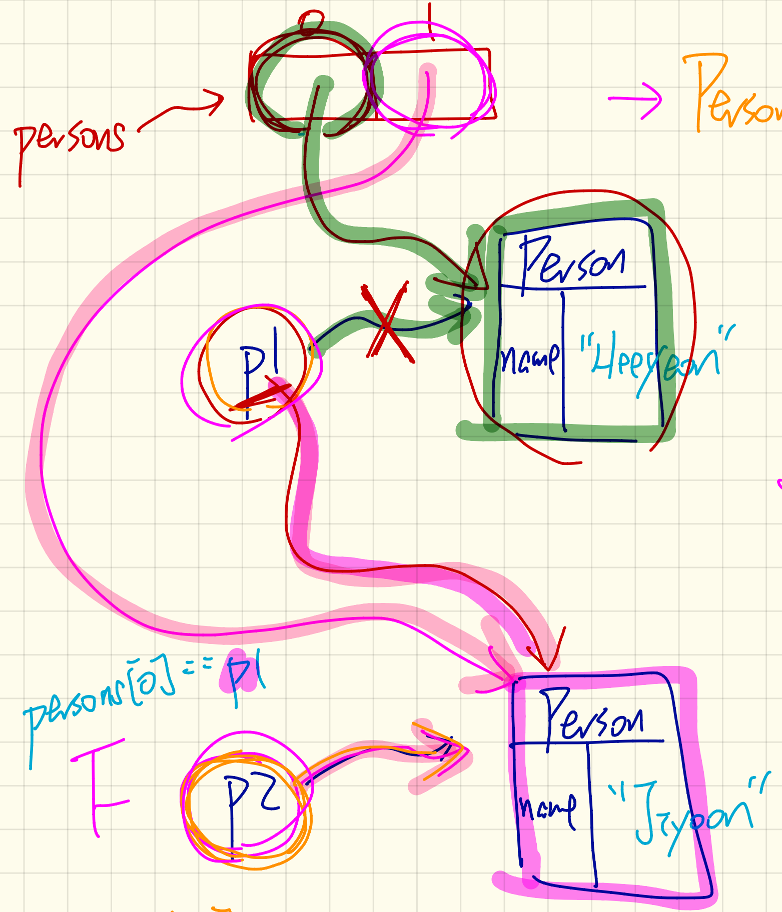
keys

values

Use of HashTable



```
Hashtable<String, String> grades = new Hashtable<String, String>();
System.out.println("Size of table: " + grades.size());
System.out.println("Key Alan exists: " + grades.containsKey("Alan"));
System.out.println("Value B+ exists: " + grades.containsValue("B+"));
grades.put("Alan", "A");
grades.put("Mark", "B+");
grades.put("Tom", "C");
System.out.println("Size of table: " + grades.size());
System.out.println("Key Alan exists: " + grades.containsKey("Alan"));
System.out.println("Key Mark exists: " + grades.containsKey("Mark"));
System.out.println("Key Tom exists: " + grades.containsKey("Tom"));
System.out.println("Key Simon exists: " + grades.containsKey("Simon"));
System.out.println("Value A exists: " + grades.containsValue("A"));
System.out.println("Value B+ exists: " + grades.containsValue("B+"));
System.out.println("Value C exists: " + grades.containsValue("C"));
System.out.println("Value A+ exists: " + grades.containsValue("A+"));
System.out.println("Value of existing key Alan: " + grades.get("Alan"));
System.out.println("Value of existing key Mark: " + grades.get("Mark"));
System.out.println("Value of existing key Tom: " + grades.get("Tom"));
System.out.println("Value of non-existing key Simon: " + grades.get("Simon"));
grades.put("Mark", "F");
System.out.println("Value of existing key Mark: " + grades.get("Mark"));
grades.remove("Alan");
System.out.println("Key Alan exists: " + grades.containsKey("Alan"));
System.out.println("Value of non-existing key Alan: " + grades.get("Alan"));
```



persons[0] == p1

F

```
        0     1
persons = {p1, p2};

p1 = p2;
```

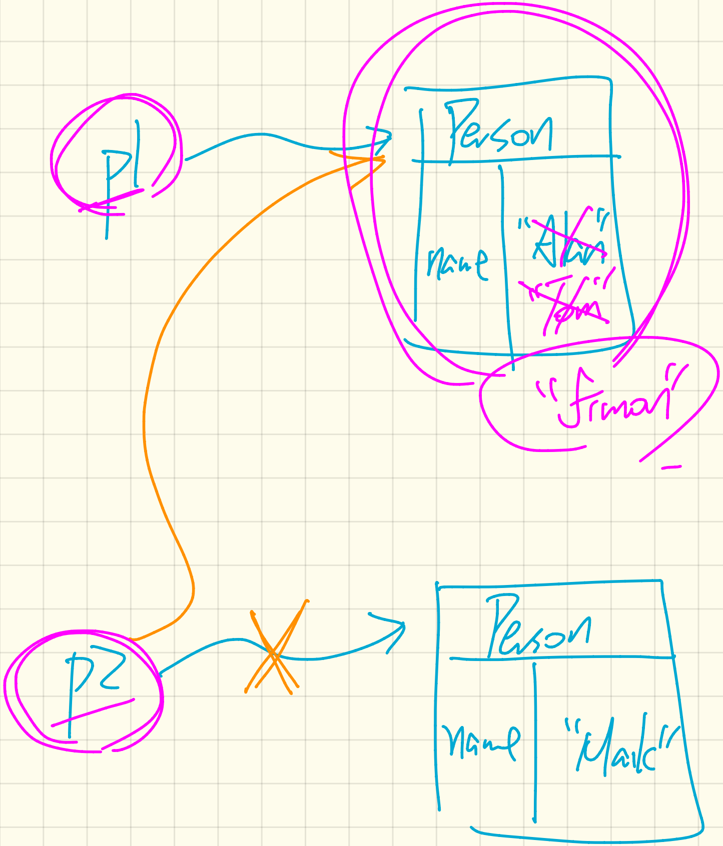
initializer

```
Person[] persons = new Person[2];
persons[0] = p1;
persons[1] = p2;
```

Store the address in p1 into index 0 of persons.

persons[1] == p2 T

persons[1] == p1 T



→ p1.setName("Tom")

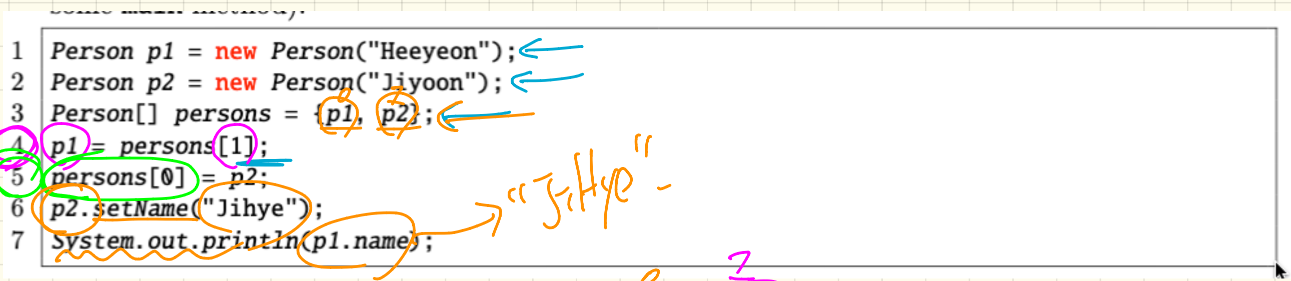
p2 = p1;

p1 == p2 T.

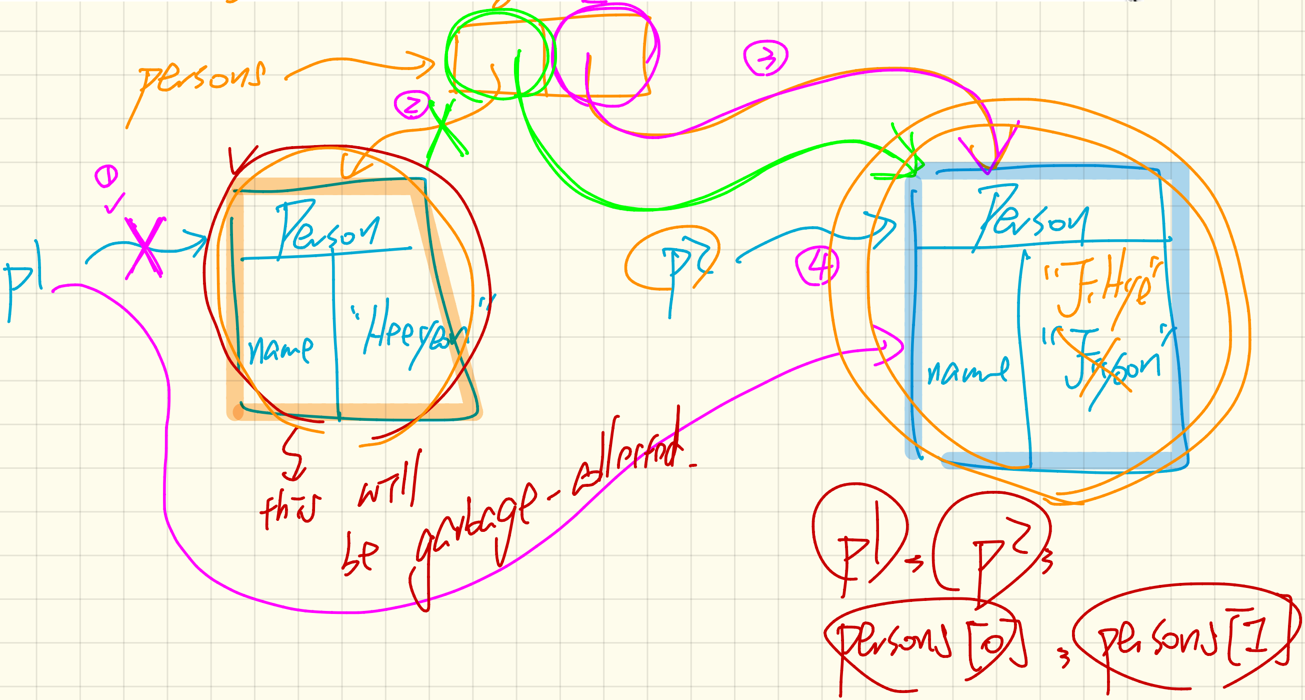
p1.setName("Simon")

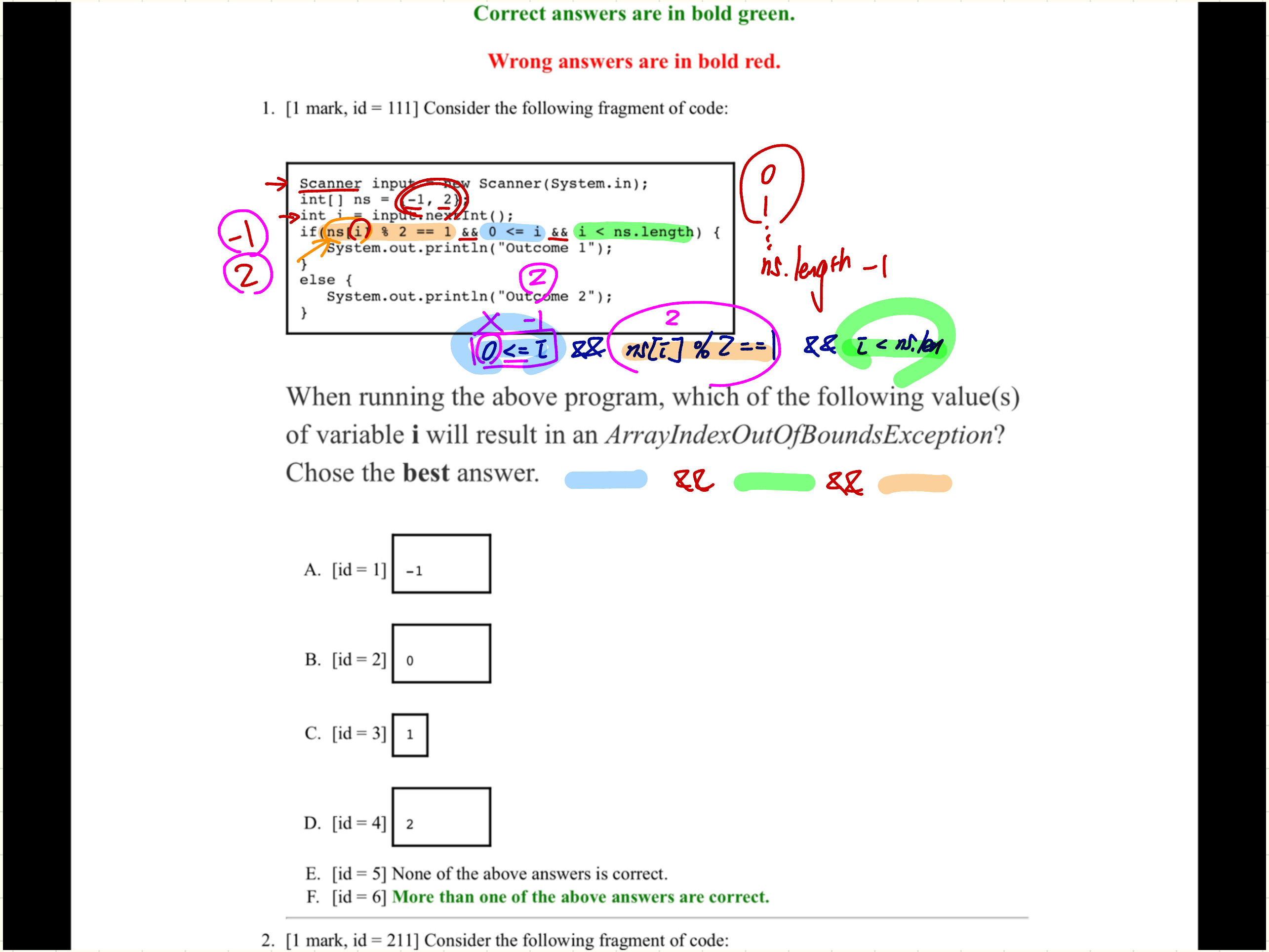
p2.getName()

↳ "Simon".



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





Correct answers are in bold green.

Wrong answers are in bold red.

1. [1 mark, id = 111] Consider the following fragment of code:

```
Scanner input = new Scanner(System.in);
int[] ns = {-1, 2};
int i = input.nextInt();
if(ns[i] % 2 == 1 && 0 <= i && i < ns.length) {
   System.out.println("Outcome 1");
}
else {
   System.out.println("Outcome 2");
}
```

-1

2

0

1

:

ns.length -1

2

-1

2

0 <= i && ns[i] % 2 == 1 && i < ns.len

When running the above program, which of the following value(s) of variable **i** will result in an *ArrayIndexOutOfBoundsException*? Chose the **best** answer.

🟦 && 🟩 && 🟧

A. [id = 1] `-1`

B. [id = 2] `0`

C. [id = 3] `1`

D. [id = 4] `2`

E. [id = 5] None of the above answers is correct.

F. [id = 6] **More than one of the above answers are correct.**

2. [1 mark, id = 211] Consider the following fragment of code:

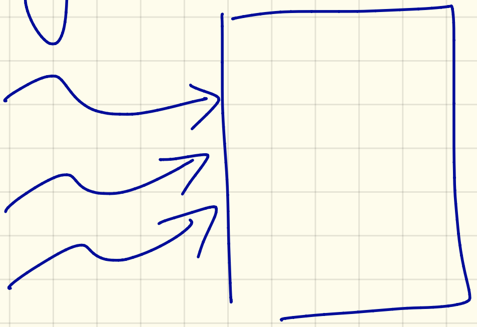
Written Questions (10%)

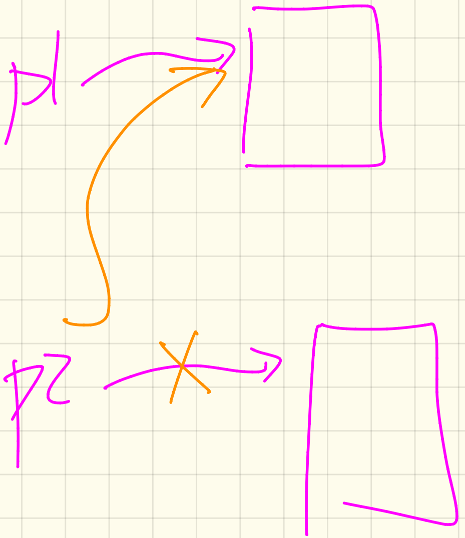
getFS(0) → {}

(2) → {1, 1}

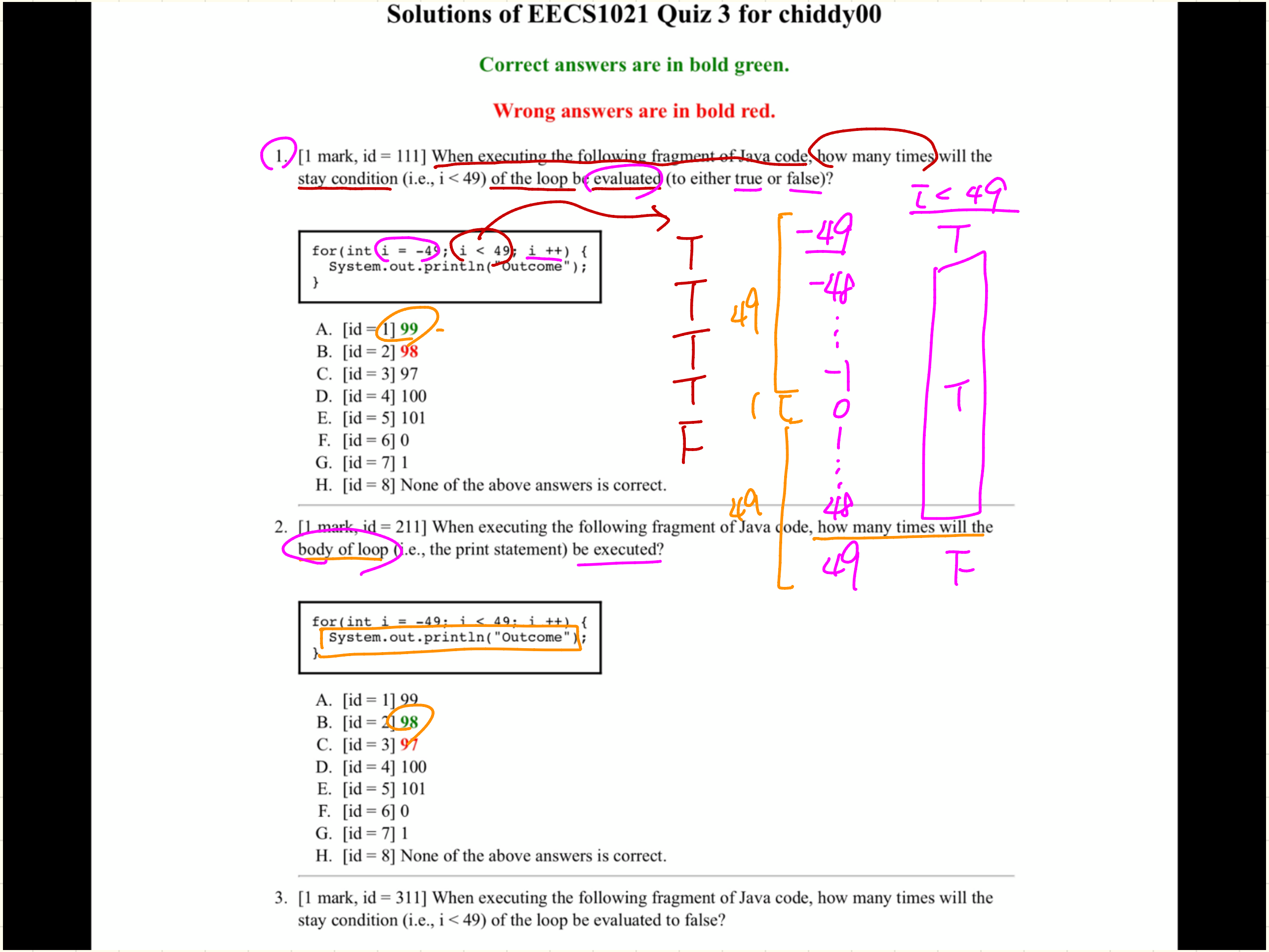
```
int[] getFibSeq(int i){
}
```

aliasing





```
Person p1 = new ...        ① p1 = p2
Person p2 = new ...
    p1 == p2  F            ② p2 = p1
```



Solutions of EECS1021 Quiz 3 for chiddy00

Correct answers are in bold green.

Wrong answers are in bold red.

1. [1 mark, id = 111] When executing the following fragment of Java code, how many times will the stay condition (i.e., i < 49) of the loop be evaluated (to either true or false)?

```
for(int i = -49; i < 49; i ++) {
  System.out.println("Outcome");
}
```

A. [id = 1] [✓ **99**]
B. [id = 2] **98**
C. [id = 3] 97
D. [id = 4] 100
E. [id = 5] 101
F. [id = 6] 0
G. [id = 7] 1
H. [id = 8] None of the above answers is correct.

T
T
T
T
F

i < 49

-49 T
-48
:
-1
0 T
1
:
48
49 F

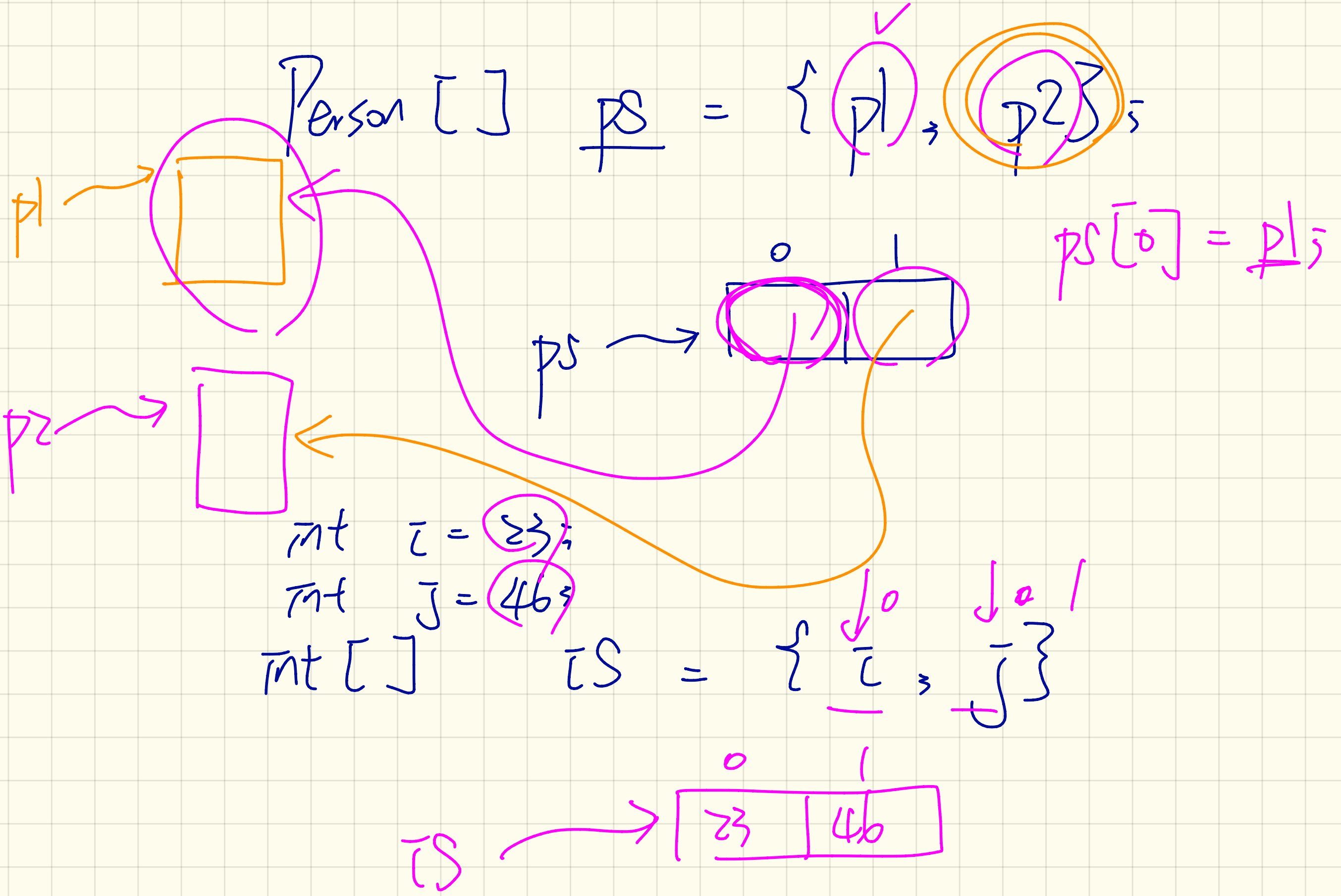
49
49

2. [1 mark, id = 211] When executing the following fragment of Java code, how many times will the body of loop (i.e., the print statement) be executed?

```
for(int i = -49; i < 49; i ++) {
  System.out.println("Outcome");
}
```

A. [id = 1] 99
B. [id = 2] [✓ **98**]
C. [id = 3] **97**
D. [id = 4] 100
E. [id = 5] 101
F. [id = 6] 0
G. [id = 7] 1
H. [id = 8] None of the above answers is correct.

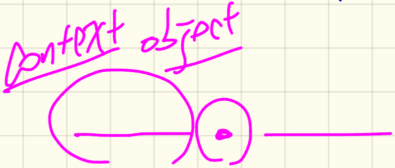
3. [1 mark, id = 311] When executing the following fragment of Java code, how many times will the stay condition (i.e., i < 49) of the loop be evaluated to false?

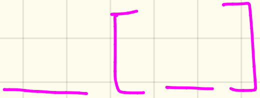


```
Person[] ps = {p1, p2};
ps[0] = p1;
int i = 23;
int j = 46;
int[] is = {i, j}
```

NullPointerException

ArrayIndexOutOfBoundEx.



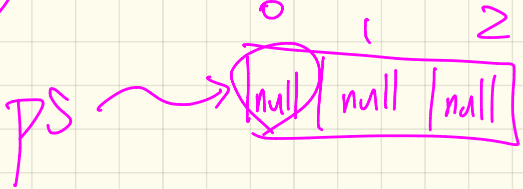


```
Person[] ps = new Person[3];
```

```
ps[0].getName()
```

NPE

null



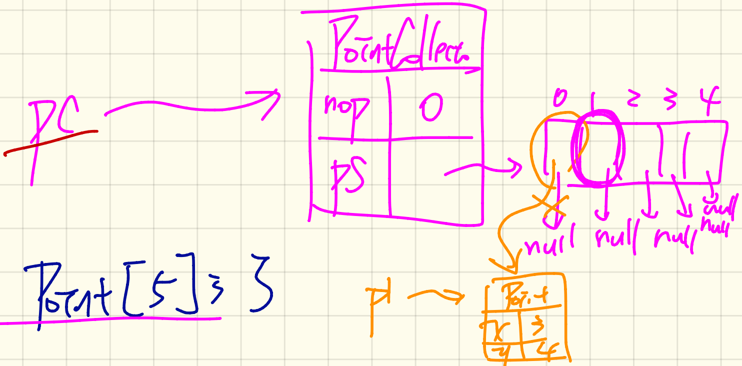
```
class PointCollector {
    Point[] ps;
    int nop;
    PointCollector() { ps = new Point[5]; }
    void addPoint(Point p){
        ps[nop] = p;
        nop++;
    }

    String getDesc() {
        String s = "";
        for (int i = 0; i < ~~ps.length~~; i++){
            s += ps[i].getX() + "," + ps[i].getY();
        }
        return s;
    }
}
```

this.nop

nop

2nd iteration: i == 1 NPE



```
PointCollector pc = new ...
Point p1 = new Point(3,4);
pc.addPoint(p1);

pc.getDesc()
```