CSE 121 – Lesson 4

Elba Garza & Matt Wang

Winter 2024

Music: 121 24wi Lecture Beats :D

TAs: Abby Aishah Anju Annie Archit Ayesha Christian
Hannah Heather Hibbah Jacob James Janvi Jasmine
Jonus Julia Lucas Luke Maria Nicole Shananda
Shayna Trey Vidhi Vivian

sli.do #cse121-4

Announcements, Reminders

- C1 releasing later today (due Tuesday, January 23rd)
- Feedback for Creative Project 0 released yesterday!
  - Use the Grade Checker tool to track your grades!
- Resubmission form for Resubmission Cycle 0 (R0) releasing tomorrow (due Thursday, January 25th)
- Remember – IPL is in person! (For async help, use the message board)
- Quiz 0: Thursday, February 1st during your quiz section.
  - Bring device to take quiz on! (e.g., laptop, surface)

Resubmissions

In general, you may revise resubmit a given Programming Assignment or Creative Project each week based on the feedback you receive with **no penalty**. The grade of your resubmission will completely replace your previous grades for that assignment.

Logistics:

- One resubmission per week
  - An assignment is only eligible for resubmission the 3 cycles after its feedback is posted
- There are 8 resubmission cycles this quarter (all listed on the course calendar)
- To use a resubmission, you will need to:
  - Set the submission you want to be graded as "Final"
  - Fill out a form listing some information (Name, assignment, what you changed, etc.)

Last Time…

- Variables
  - Container that stores a specific data type
  - Must declare & initialize!
  - Manipulate, modify, reuse ♻️
- Strings
  - Sequence of characters treated as one, yet can be indexed as individual parts
  - char, represents a single character

```
// declare AND initialize
int version = 5;
```

L	a	u	f	e	y	!
0	1	2	3	4	5	6

🐞 Debugging

We also started to think about **debugging** with a couple recent activities in class (Duck Heights & `BuggyMadLibs.java`)

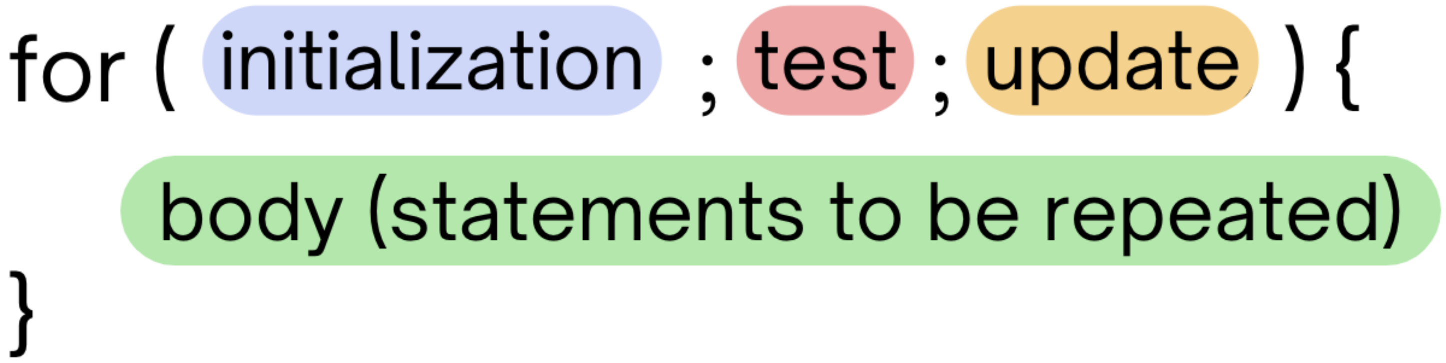
Bugs happen – debugging is a natural part of programming!

Ways to approach debugging?

(PCM) for loops!

For loops are our first *control structure*

A syntactic structure that *controls* the execution of other statements.

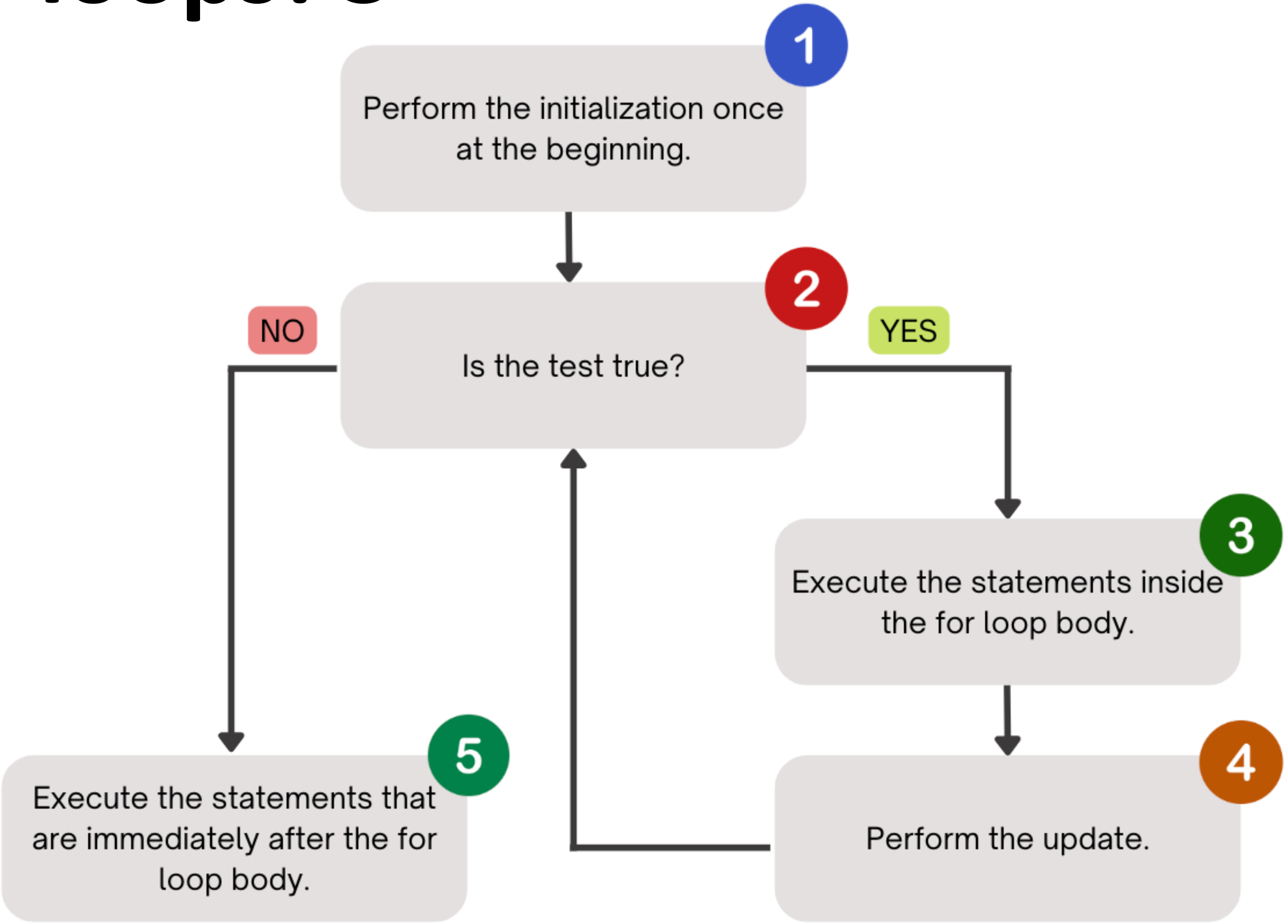


```
for ( initialization ; test ; update ) {
    body (statements to be repeated)
}
```

(PCM) for loops! 2

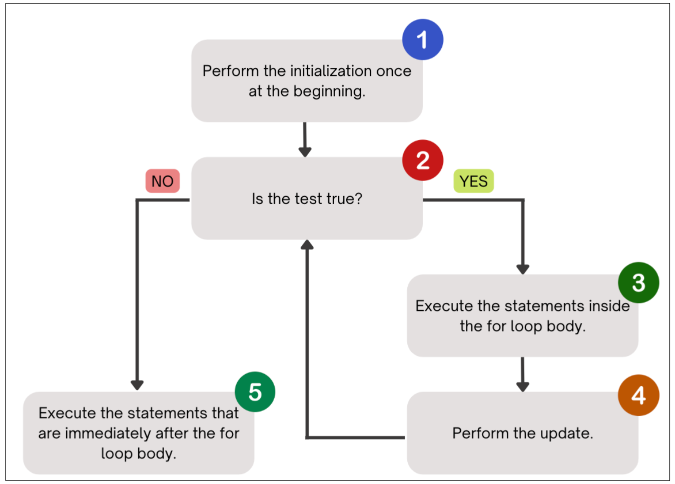
```
for (int counter = 1; counter <= 5; counter++) {
    System.out.println("I love CSE 121!");
}
```

(PCM) for loops! 3



1
Perform the initialization once at the beginning.
2
Is the test true?
NO
YES
3
Execute the statements inside the for loop body.
4
Perform the update.
5
Execute the statements that are immediately after the for loop body.

(PCM) for loops! 4


Perform the initialization once at the beginning.

Is the test true?

Execute the statements inside the for loop body.

Perform the update.

Execute the statements that are immediately after the for loop body.

```
                1                   2                4
for (int counter = 1; counter <= 5; counter++) {
3  System.out.println("I love CSE 121!");
}
5
```

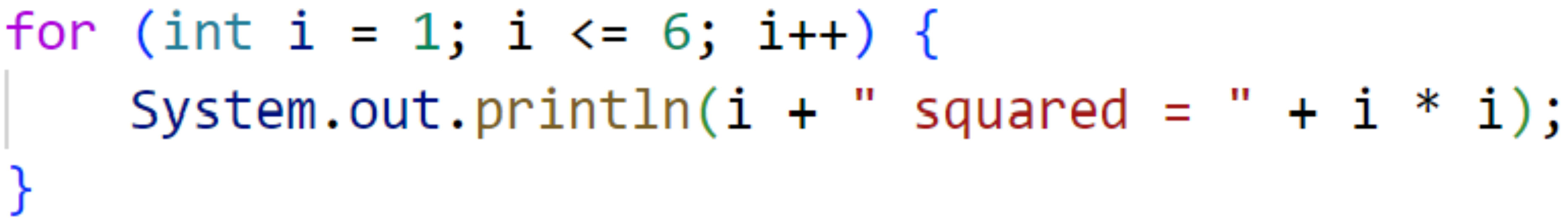
```
I love CSE 121!
I love CSE 121!
I love CSE 121!
I love CSE 121!
I love CSE 121!
```

counter

6

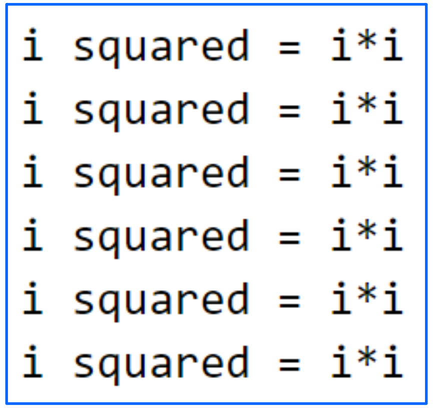
Poll in with your answer!

What output does the following code produce?



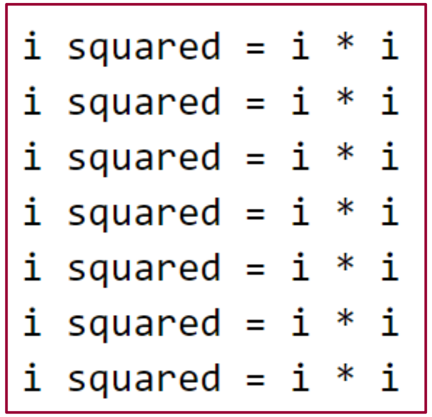
```
for (int i = 1; i <= 6; i++) {
    System.out.println(i + " squared = " + i * i);
}
```

A.



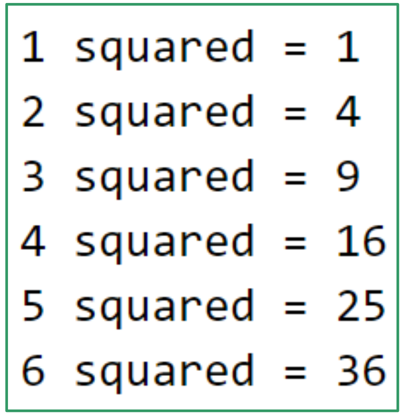
```
i squared = i*i
i squared = i*i
i squared = i*i
i squared = i*i
i squared = i*i
i squared = i*i
```

B.



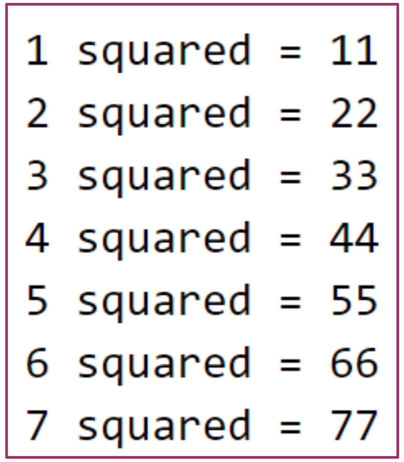
```
i squared = i * i
i squared = i * i
i squared = i * i
i squared = i * i
i squared = i * i
i squared = i * i
i squared = i * i
```

C.



```
1 squared = 1
2 squared = 4
3 squared = 9
4 squared = 16
5 squared = 25
6 squared = 36
```

D.



```
1 squared = 11
2 squared = 22
3 squared = 33
4 squared = 44
5 squared = 55
6 squared = 66
7 squared = 77
```

(PCM) String traversals

```
// For some String s
for (int i = 0; i < s.length(); i++) {
    // do something with s.charAt(i)
}
```

Fencepost Pattern 1

Some task where one piece is repeated *n* times, and another piece is repeated *n-1* times and they alternate

```
h-u-s-k-i-e-s
```

Fencepost Pattern 2

Some task where one piece is repeated *n* times, and another piece is repeated *n-1* times and they alternate

`h-u-s-k-i-e-s`