Tutorial

Practice Exercises: Teaching Block 1

- Operators (including operator precedence)
- Control structures: selection and repetition
- The **`break`** statement
- Basic Java programs
- Simple problem specifications
- <u>Other practice exercise types</u>: “Fill in the gaps” + “Predict the question”

This set of exercises is in addition to those included directly in lecture slides (and extra reading materials), which you should also attempt.

Queen Mary University of London

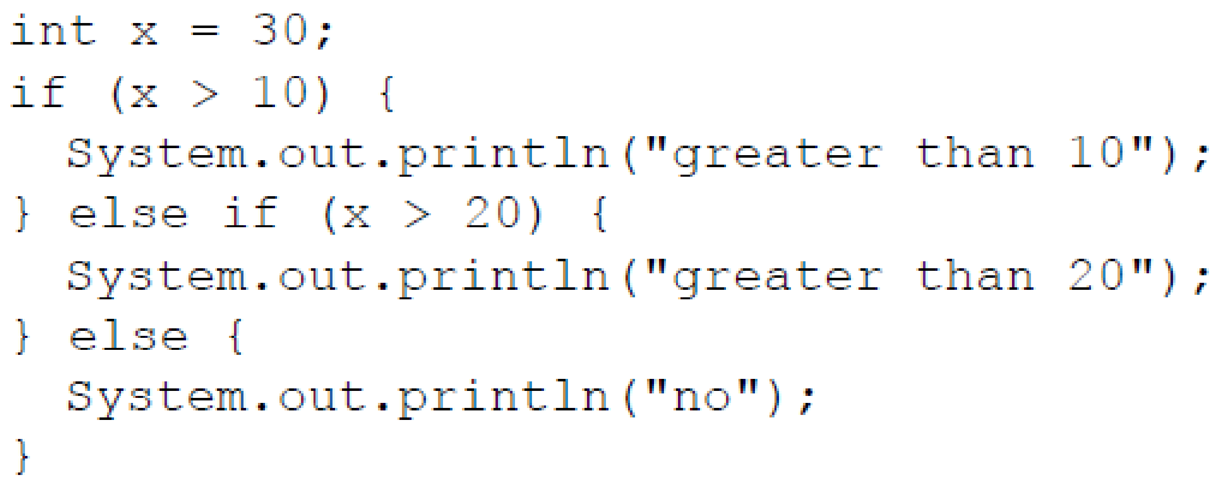
Question 1

- What is the output of this code fragment?

```
int x = 5;
if (x < 10) {
  System.out.println("less than 10");
} else if (x > 10) {
  System.out.println("greater than 10");
} else {
  System.out.println(“equals to 10 ");
}
```

Question 2

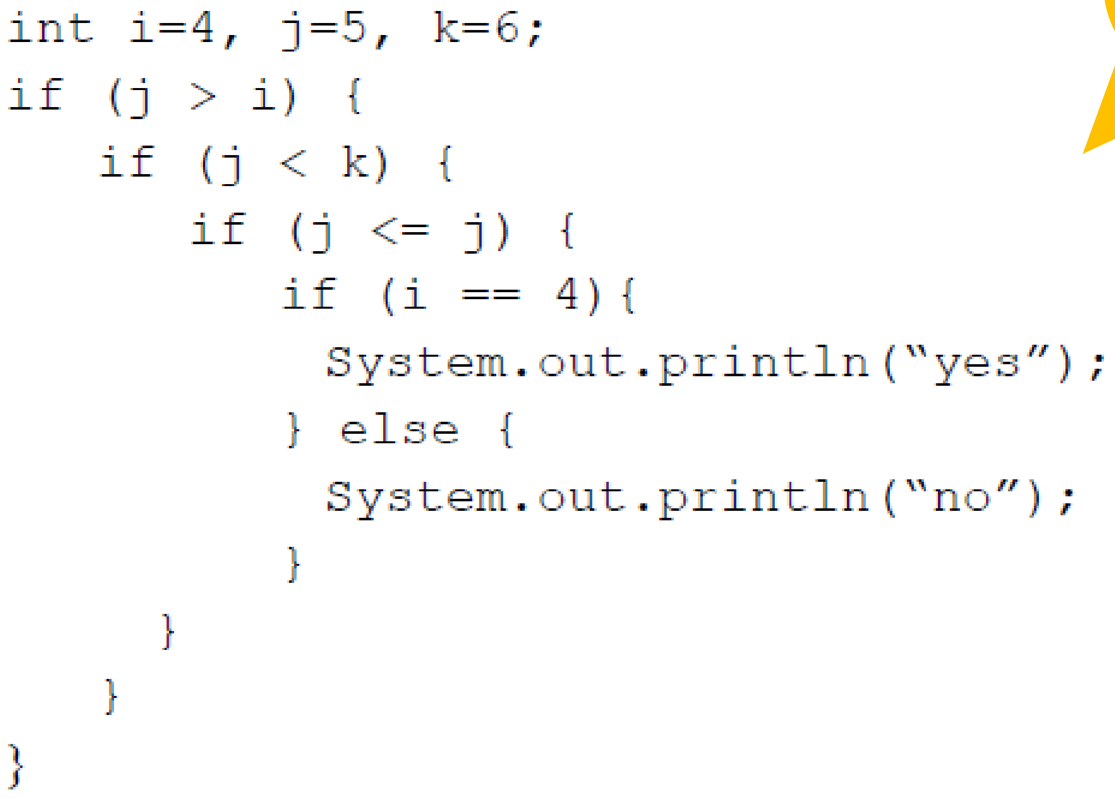
- What is the output of this code fragment?



```
int x = 30;
if (x > 10) {
  System.out.println("greater than 10");
} else if (x > 20) {
  System.out.println("greater than 20");
} else {
  System.out.println("no");
}
```

Question 3

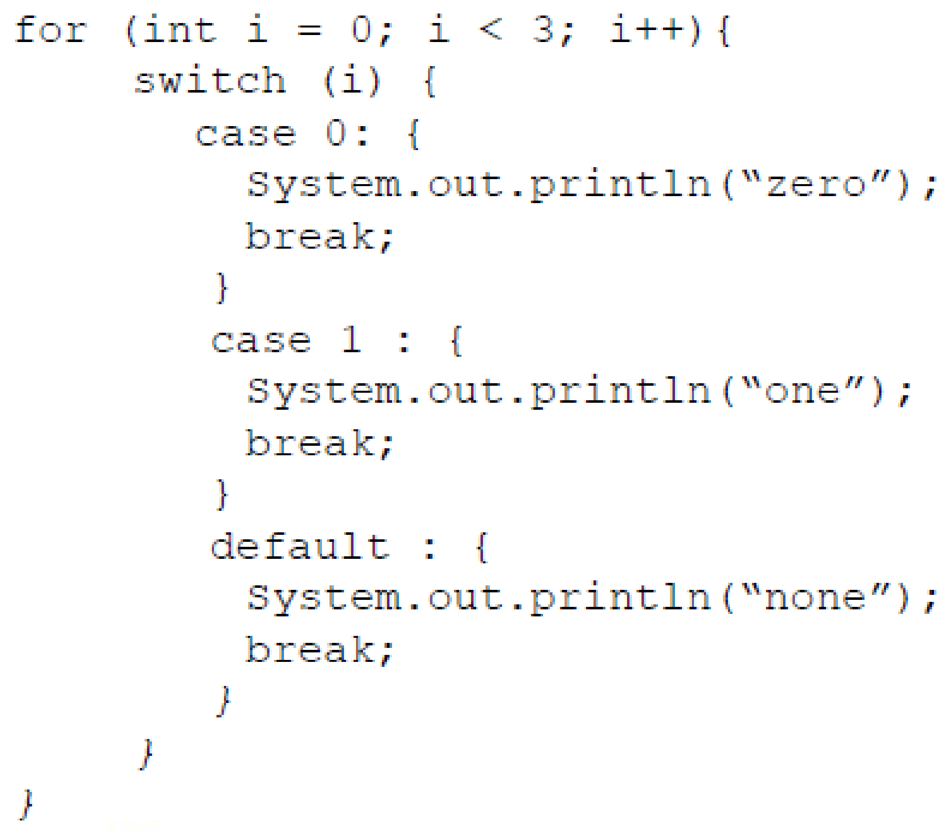
- What is the output of this code fragment?



```
int i=4, j=5, k=6;
if (j > i) {
    if (j < k) {
        if (j <= j) {
            if (i == 4){
              System.out.println("yes");
            } else {
              System.out.println("no");
            }
        }
    }
}
```

Question 4

- What is the output of this code fragment?



```
for (int i = 0; i < 3; i++){
      switch (i) {
         case 0: {
           System.out.println(“zero”);
           break;
          }
          case 1 : {
           System.out.println(“one”);
           break;
          }
          default : {
           System.out.println(“none”);
           break;
          }
      }
}
```

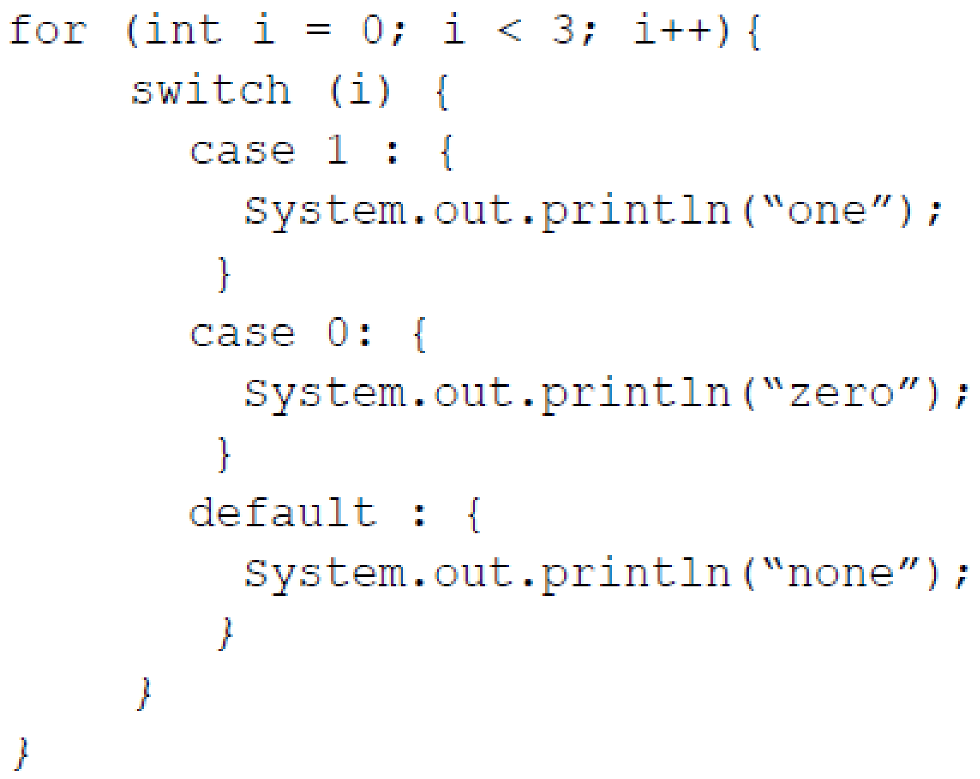
Question 5

- What is the output of this code fragment?

```
for (int i = 0; i < 3; i++){
     switch (i) {
        case 0: {
          System.out.println(“zero”);
         }
         case 1 : {
          System.out.println(“one”);
         }
         default : {
          System.out.println(“none”);
         }
     }
}
```

Question 6

- What is the output of this code fragment?



```
for (int i = 0; i < 3; i++){
     switch (i) {
        case 1 : {
          System.out.println(“one”);
         }
        case 0: {
          System.out.println(“zero”);
         }
        default : {
          System.out.println(“none”);
         }
     }
}
```

Questions 7+8

- Write a Java program that calculates the sum of integers in the range **1** to **100** (*inclusive*).

- Write a Java program that produces a multiplication table, showing the results of multiplying the integers **1** through to **3**. The output of your program should look as follows:

```
1  2  3
2  4  6
3  6  9
```

Questions 9+10

- What will be printed out using the following code?

```java
class Question2c {
  public static void main(String[] args) {
    int i=8, j=9;
    boolean test;
    test=i>7&&j-- > i++;
    System.out.println(i);
    System.out.println(j);
    System.out.println(test);
  }
}
```

⇔

```java
test=(i>7)&&(j>i);
j=j-1;
i=i+1;
```

- Write a block of code that calculates the sum of all the integers divisible by **3**, in the range **1** to **99** (*inclusive*). You are not required to write a complete program.

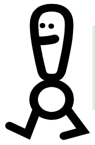
Exercise: Fill in the Gaps

- Consider the incomplete Java program `StarsTriangle`; it displays the pattern below, when it is compiled and run.
- Your challenge is to use the collection of statements on the far right, together with some extra right brackets (`}`), to complete the program `StarsTriangle`.

```
public class StarsTriangle {
  public static void main(String[] args) {
    // code missing
  }
}
```

```
*
**
***
****
*****
******
*******
********
```

```
col = col + 1;
int col = 0;
int row = 0;
int size = 8;
row = row + 1;
row = 0;
System.out.print('*');
System.out.println('*');
System.out.println();
while (col <= row) {
while (col < row) {
while (row < size) {
while (row <= size) {
col = 0;
```



Some of the statements may not be necessary!

Exercise: Predict the Question

- Determine the question that should result in the following possible answers:

Answer 1

"This happens when a class has multiple methods with the same name but different lists of parameters. It helps ensure consistency when naming methods."

Answer 2

"Methods in the same class that share the same name but accept different variable types as arguments. Such methods give programmers the flexibility to call a similar method with different types of data."