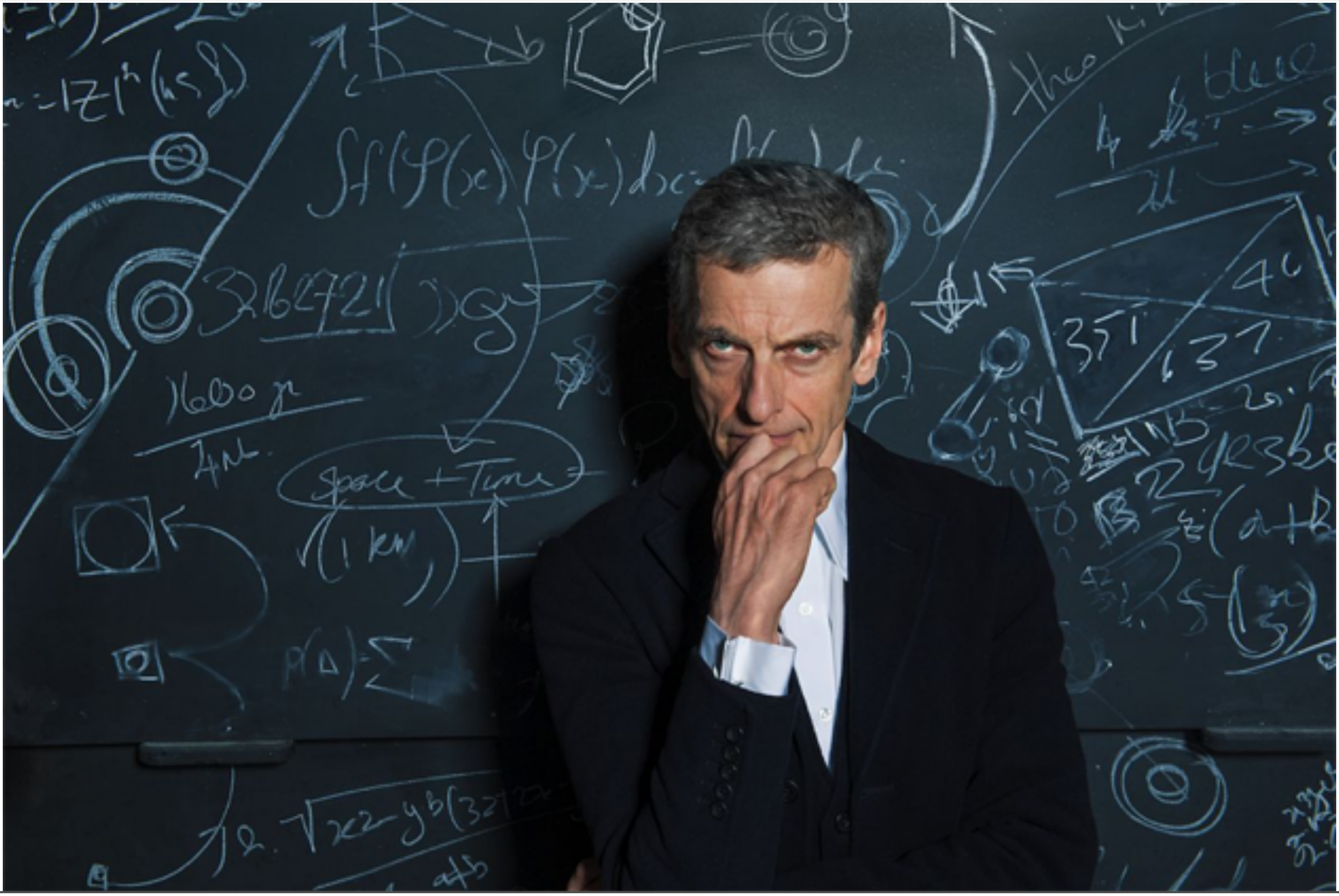
Java

PRAGMATIC IT Learning & Outsourcing Center

Lector: Milen Penchev
Skype: donald8605
E-mail: milen.penchev@gmail.com
Facebook: http://www.facebook.com/milen.penchev.39

Recap



Recap

- Скоростта на интернета в залата

Recap

Recap

1995

Recap

James Gosling

Recap

Recap

- Object-Oriented

Recap

- Object-Oriented
- Compiles .java into .class

Recap

- Object-Oriented
- Compiles .java into .class
- Runs in JVM

Recap

- Object-Oriented
- Compiles .java into .class
- Runs in JVM
- How to install Java / Eclipse

Recap

- Object-Oriented
- Compiles .java into .class
- Runs in JVM
- How to install Java / Eclipse
- First program (Hello from the other side)

Recap

- Object-Oriented
- Compiles .java into .class
- Runs in JVM
- How to install Java / Eclipse
- First program (Hello from the other side)
- Class name rules

Recap

- Object-Oriented
- Compiles .java into .class
- Runs in JVM
- How to install Java / Eclipse
- First program (Hello from the other side)
- Class name rules
- Instance, class and local variables; arguments

Recap

- Object-Oriented
- Compiles .java into .class
- Runs in JVM
- How to install Java / Eclipse
- First program (Hello from the other side)
- Class name rules
- Instance, class and local variables; arguments
- Primitives

Recap

- Object-Oriented
- Compiles .java into .class
- Runs in JVM
- How to install Java / Eclipse
- First program (Hello from the other side)
- Class name rules
- Instance, class and local variables; arguments
- Primitives
- Arithmetic, Unary, Equality, Conditional Operators

Recap

- Object-Oriented
- Compiles .java into .class
- Runs in JVM
- How to install Java / Eclipse
- First program (Hello from the other side)
- Class name rules
- Instance, class and local variables; arguments
- Primitives
- Arithmetic, Unary, Equality, Conditional Operators
- AND, OR, NOT

Recap

- Object-Oriented
- Compiles .java into .class
- Runs in JVM
- How to install Java / Eclipse
- First program (Hello from the other side)
- Class name rules
- Instance, class and local variables; arguments
- Primitives
- Arithmetic, Unary, Equality, Conditional Operators
- AND, OR, NOT
- If-Else

Lecture 2 - Overall

- Loops
- while
- for
- do-while
- Switch
- Keywords - break and continue

Problem to solve

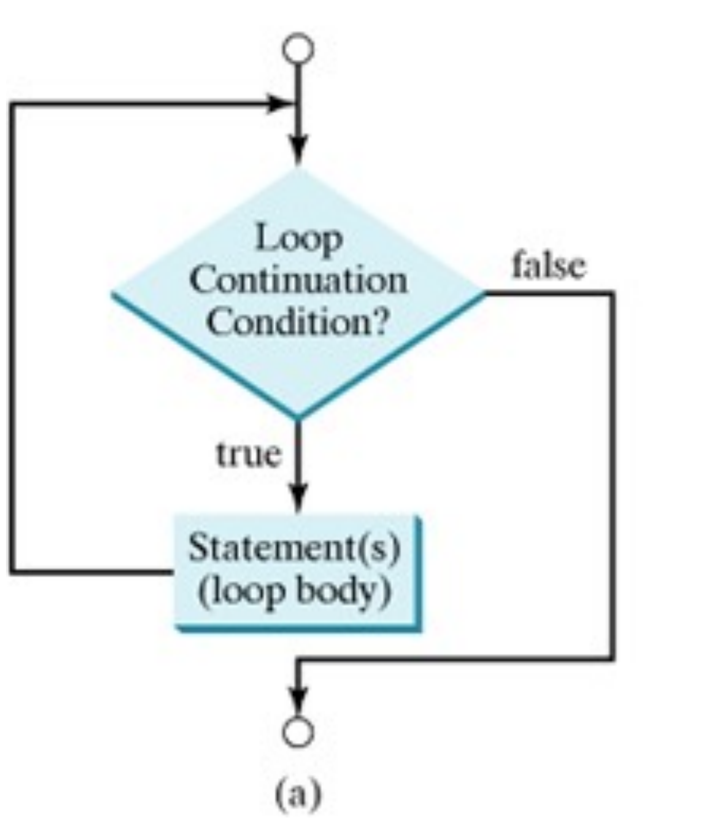
- Print all the numbers
  - From 1 to 5
  - From 1 to 1000
  - From 1 to n
  - From n to m

What is loop?

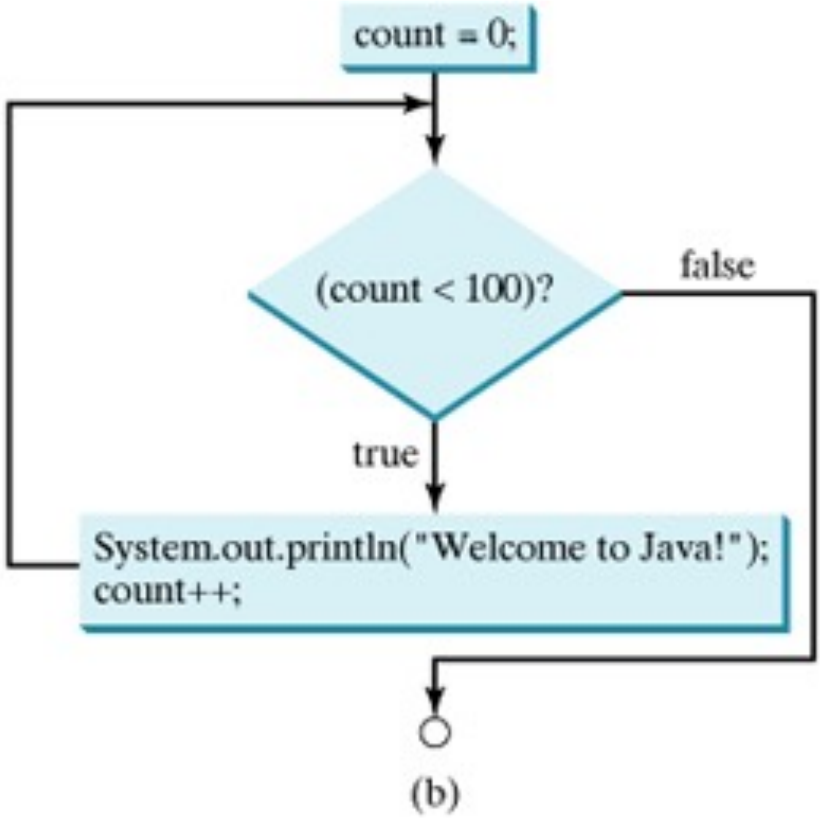
- A loop is a structure that allows sequence of statement to be executed more times in a row
- Loops have a boolean condition and a block of code for execution. While the condition is true, the block is being executed.
- A loop that never ends is called an infinite loop

While loop

- While the condition is true, the block is being executed.



(a)



(b)

While loop

- While loop example:

Counter initialization

Boolean condition.
If i > 100, the block will NOT be executed

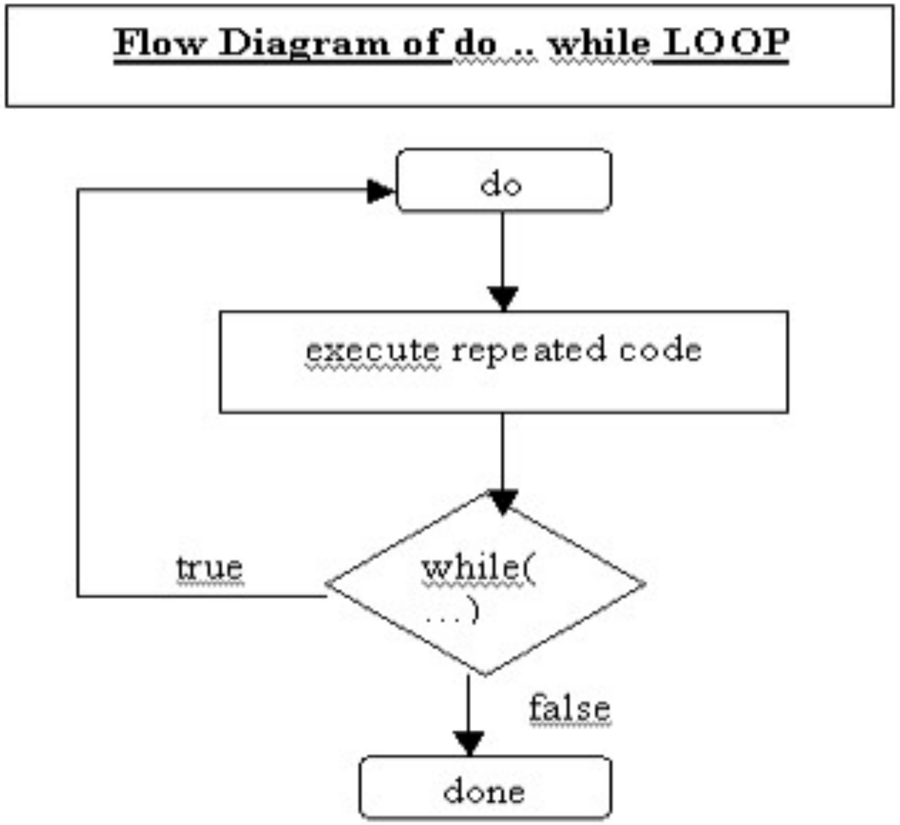
Block of code for repeatable execution

```
        int i = 1;
        while (i <= 100) {
                System.out.println(i);
                i++;
        }
```

- WhileExample.java ,WhileExample2.java, WhileExample3.java, Example.java in code examples
- Numbers.java – nested while loop in code examples

do-while

- Gets executed at least once
- Condition is after the execution



Example of do-while

- An example of a do-while loop:

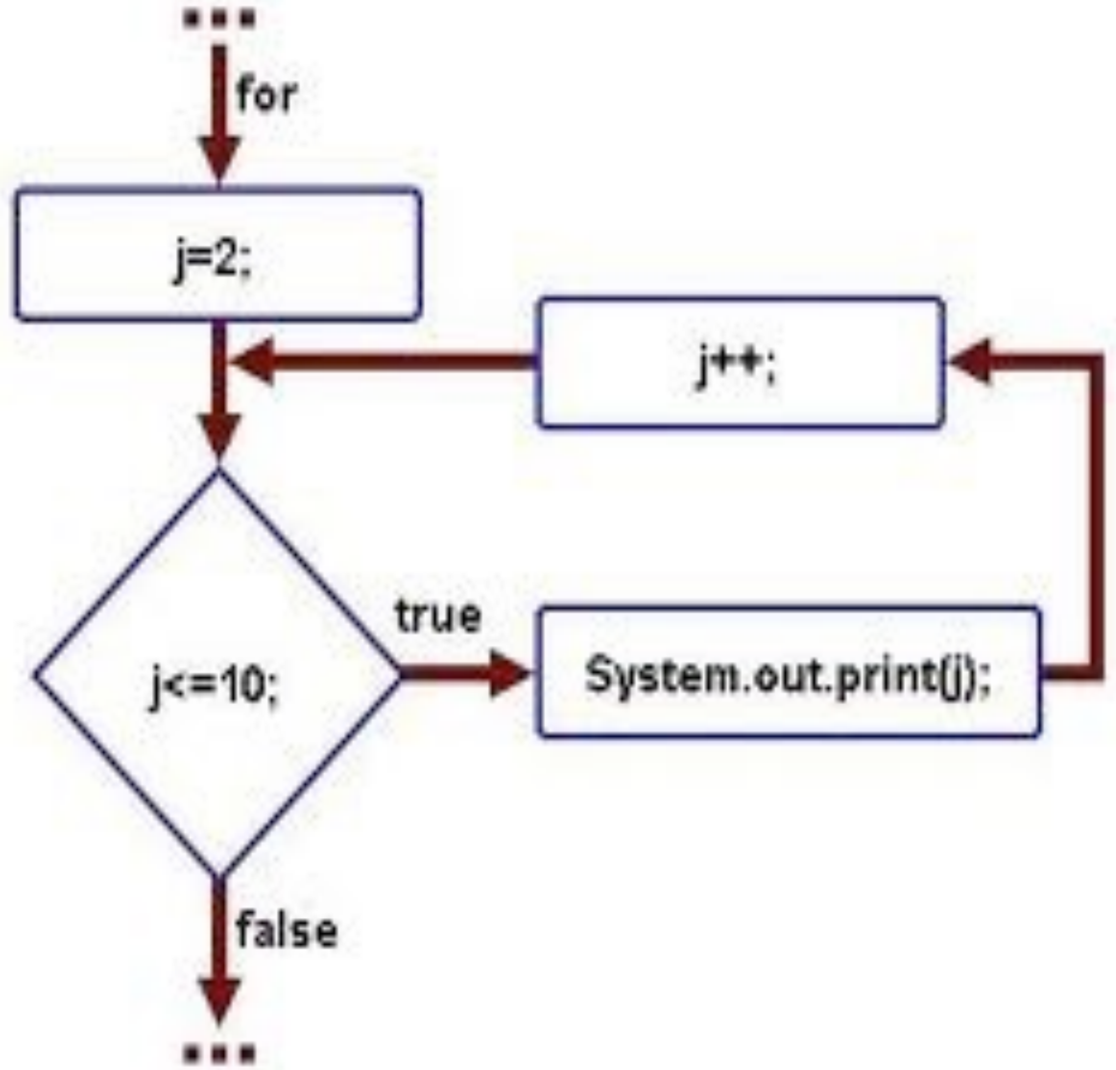
The code block that gets executed

if i<=1000(TRUE), execute the block once again

```
do {
        System.out.println(i);
        i++;
} while (i <= 1000)
```

For loop

- FOR loop:
  - Initialization
  - Condition
  - Increment
  - Body



Example of for loop

An example of for loop:

```
for (int i = 0; i < 100; i++) {
      System.out.println(i);
}
```

- Initialization: `int i = 0;`
- Condition: `i < 100;`
- Increment: `i++)`
- Body: `{`

```
        System.out.println(i);
}
```

ForExample.java, ForExample1.java, Fibonacci.java, Factorial.java, Sum.java in code examples

Problem

- Try to quit a for-loop during the execution of the repeatable block
- One possible to solution is to set the counter to a value which will make the boolean condition quit the loop.... but there is a much more proper way

Break

- Break is a keyword
- A statement by itself
- It doesn't require anything else
- It stops the execution of the loop

- BreakExample.java in code examples

```
for (int i = 0; i < 50; i++) {
        if (i == 7) {
                break;
        }
}
```

The loop will quit when i = 7

Problem

- Try to omit specific block of code in the body – for example sum all numbers between 1 and 100 but omit all numbers between 51 and 74
- Encapsulating the code in if-else statements may be used. Although for more complicated structures should be used for more complicated cases

Continue

- Continue is a keyword
- A statement by itself
- It doesn't require anything else
- It stops the current iteration of the loop, but doesn't stop the loop
- ContinueExample.java in code examples

if it is between 51 and 71, it will skip everything that is after continue

Switch statement

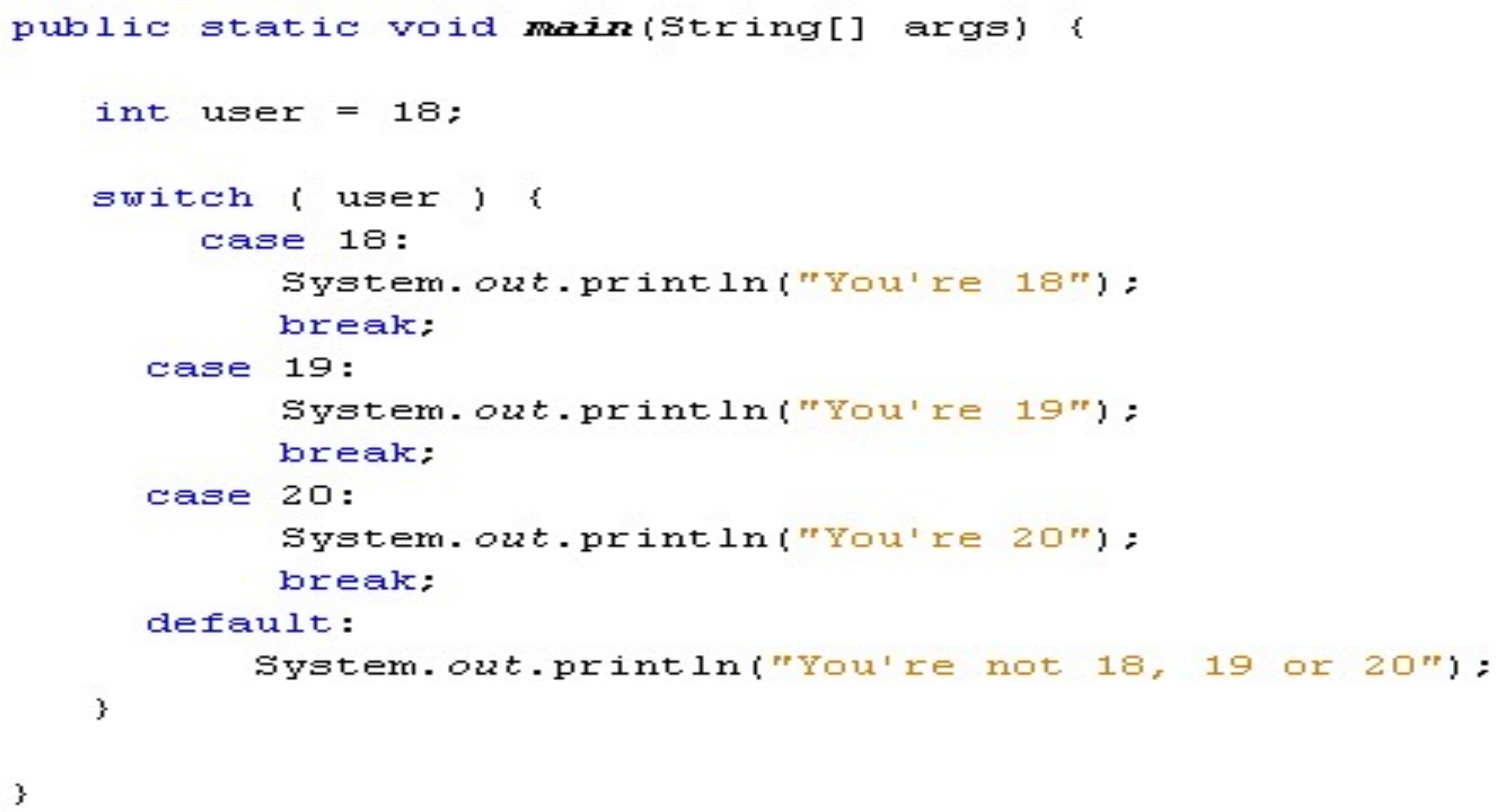
- Unlike *if-then* and *if-then-else* statements, the *switch* statement can have a number of possible execution paths

- A switch works with the byte, short, char, and int primitive data types. It also works with String class, and a few special classes that wrap certain primitive types: Character, Byte, Short, and Integer

Switch example (part 1)

- The body of a switch statement is known as a *switch block*. A statement in the switch block can be labeled with one or more case or default label. The switch statement evaluates its expression, then executes all statements that follow the matching case label.

Switch example (part 2)



```
public static void main(String[] args) {

    int user = 18;

    switch ( user ) {
        case 18:
            System.out.println("You're 18");
            break;
      case 19:
            System.out.println("You're 19");
            break;
      case 20:
            System.out.println("You're 20");
            break;
      default:
           System.out.println("You're not 18, 19 or 20");
    }

}
```

- SwitchDemo.java in the code examples

Switch – break & default

- Another point of interest is the break statement. Each break statement terminates the enclosing switch statement. Control flow continues with the first statement following the switch block. The break statements are necessary because without them, statements in switch blocks *fall through*: All statements after the matching case label are executed in sequence, regardless of the expression of subsequent case labels, until a break statement is encountered.
- The default section handles all values that are not explicitly handled by one of the case sections.
- SwitchDemoFallThrough.java in the code examples

How to ‘for each’ in Java

- You’ll get to know in the lecture related to Arrays and collections

Cheers! ☺

Summary

- Why do we use loops?
- What does a loop consist of?
- Difference between *while and do-while?*
- How to use *for – loop?*
- How to terminate a loop?
- How to stop the current iteration?