Control Flow & Static Methods CS101

NIFR 2023

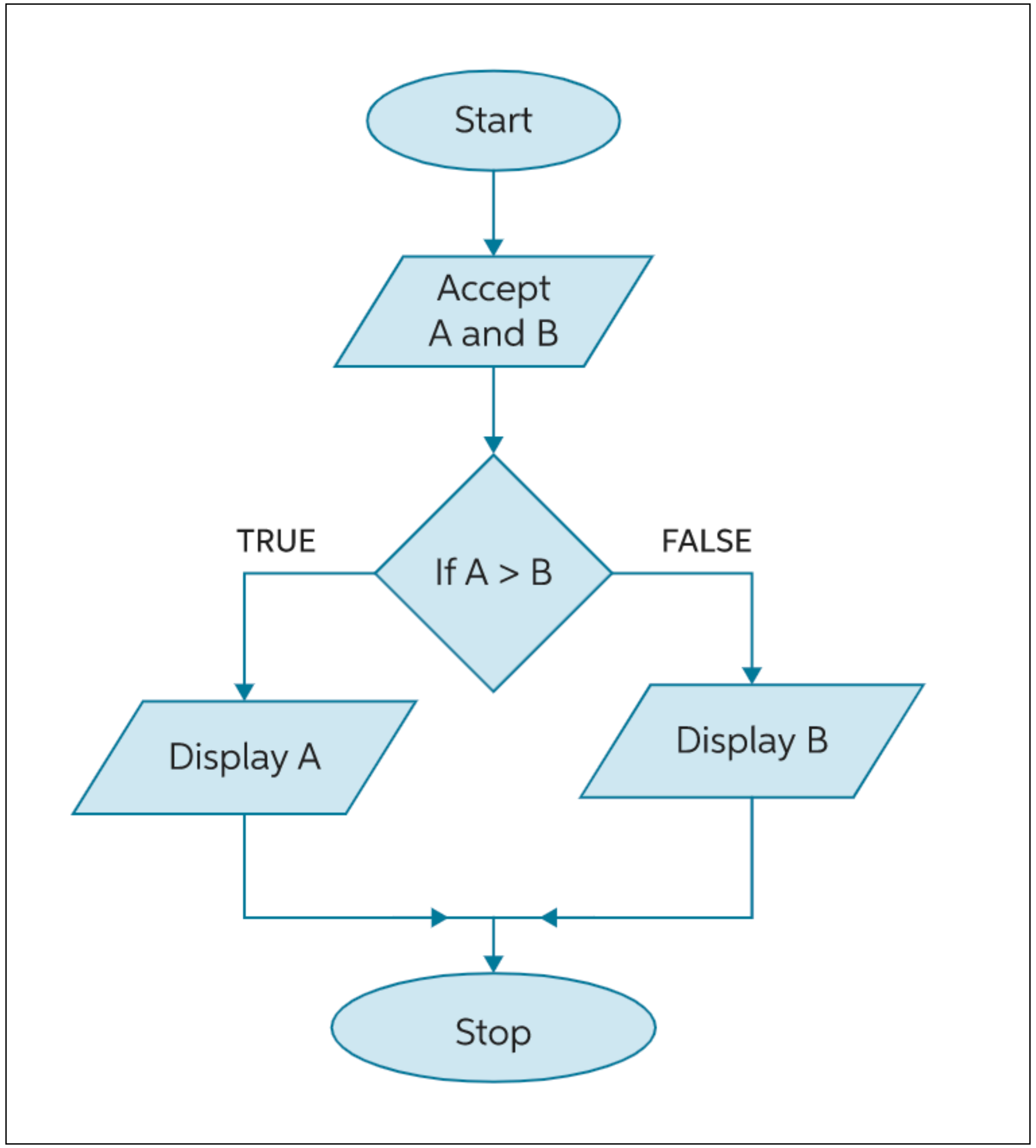
Agenda

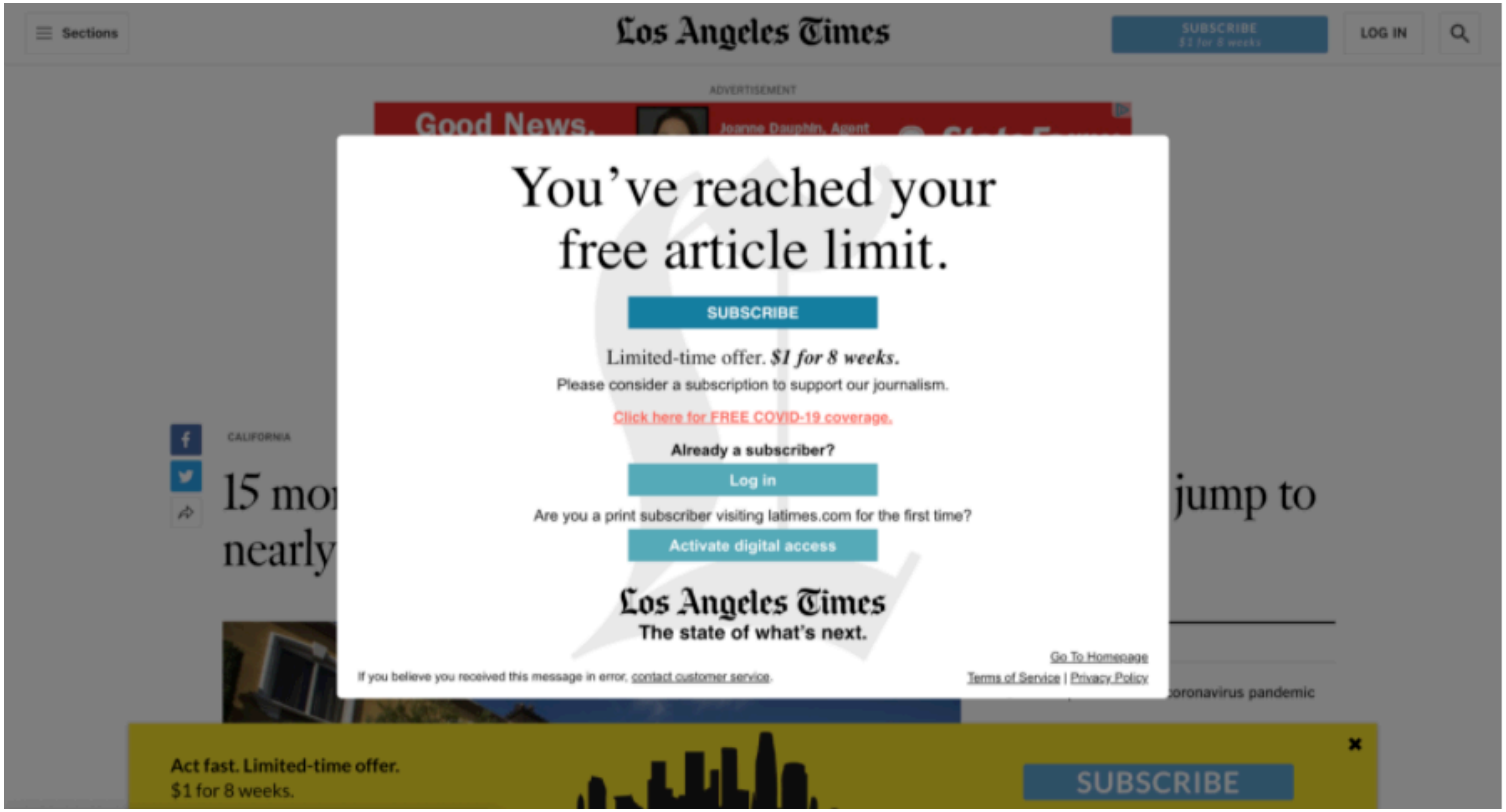
Application Development: CS101

- Conditionals
  - Switch statement
- Loops
- Static Methods
- Explainer
- Exercises
- Ugeopgave: spørgsmål

Examples:
https://github.com/nicklasdean/ita22-code-examples

Conditionals CS101


Start
Accept A and B
If A > B
TRUE
FALSE
Display A
Display B
Stop



15 mo jump to nearly

You've reached your free article limit.

SUBSCRIBE

Limited-time offer. ***$1 for 8 weeks.***

Please consider a subscription to support our journalism.

Click here for FREE COVID-19 coverage.

Already a subscriber?

Log in

Are you a print subscriber visiting latimes.com for the first time?

Activate digital access

Los Angeles Times

The state of what's next.

If you believe you received this message in error, contact customer service.

Go To Homepage

Terms of Service | Privacy Policy

coronavirus pandemic

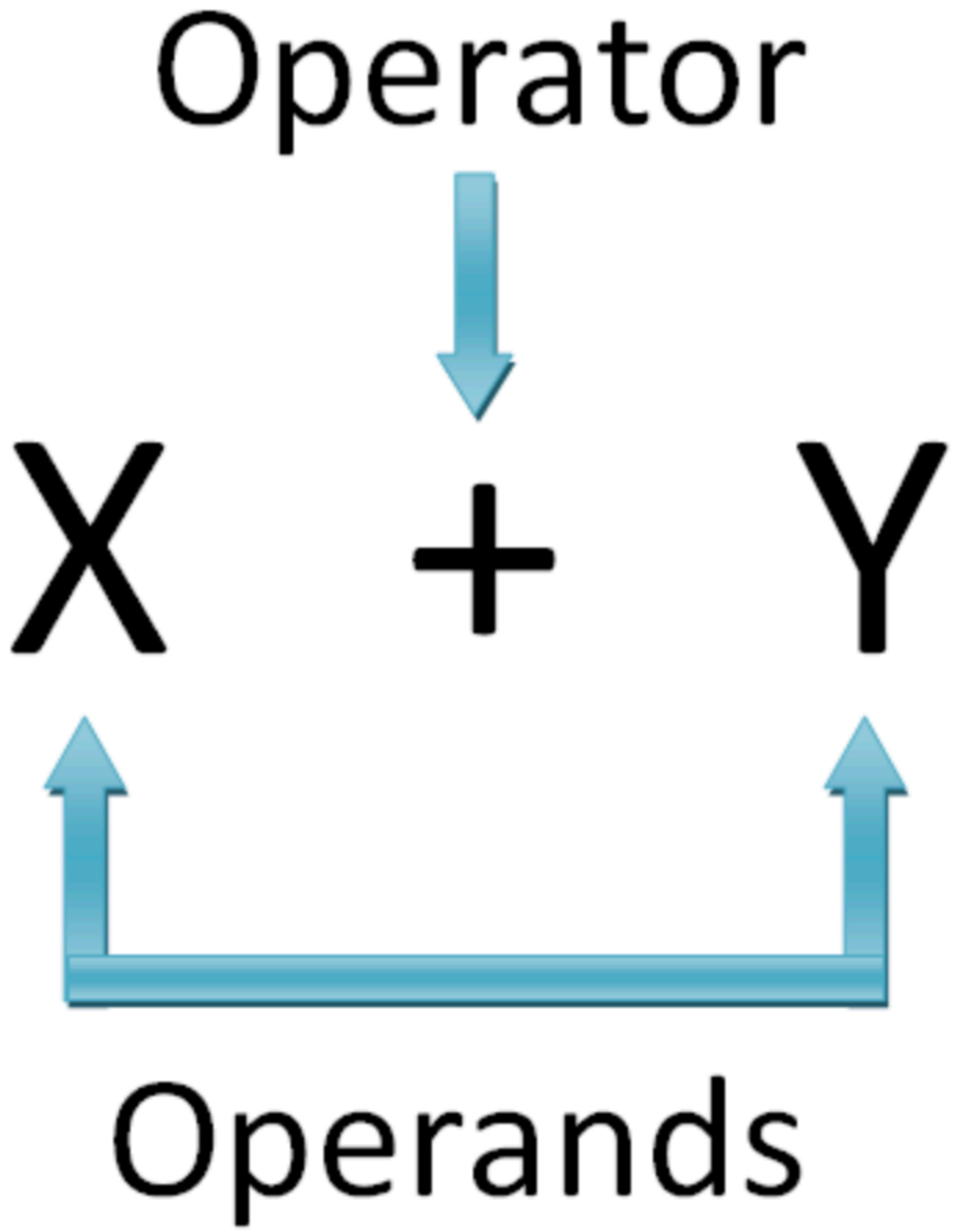
Act fast. Limited-time offer.
$1 for 8 weeks.

SUBSCRIBE

Anatomy of

```
if (){}
```

An expression



Operator	Name	Example
+	Addition	x + y
-	Subtraction	x - y
*	Multiplication	x * y
/	Division	x / y
%	Modulus	x % y
**	Exponentiation	x ** y
//	Floor division	x // y

Evaluating expressions

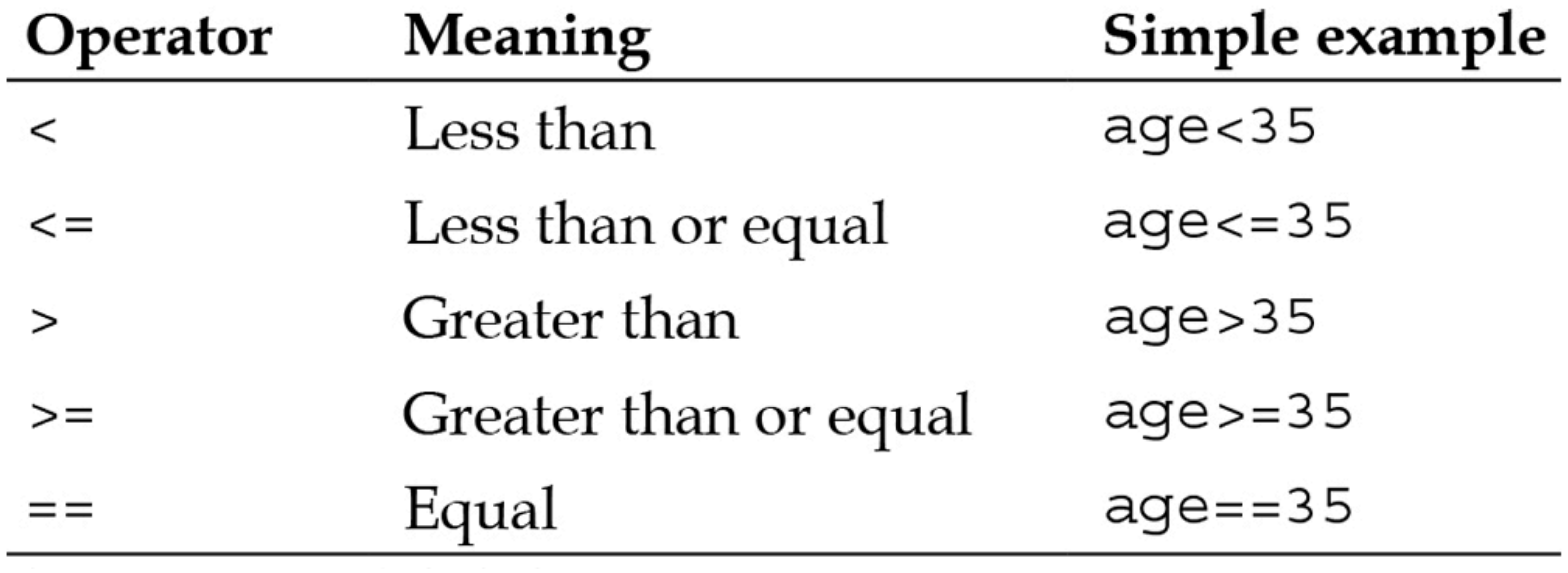
```
public class Example{
    public static void main(String[] args){´
    int heightFromUser;                 //Input from user
    final int HEIGHT_LIMIT = 160;    //Hard limit

    if(HEIGHT_LIMIT > heightFromUser){
        System.out.println("Sorry not tall enough to enter")
    }
}
```

Evaluating expressions

```
public class Example{
    public static void main(String[] args){´
    int heightFromUser;                //Input from user
    final int HEIGHT_LIMIT = 160;      //Hard limit

    if(HEIGHT_LIMIT > heightFromUser){
        System.out.println("Sorry not tall enough to enter")
    }
}
```

Operator, operand & expression

Relational operators



Operator	Meaning	Simple example
`<`	Less than	`age<35`
`<=`	Less than or equal	`age<=35`
`>`	Greater than	`age>35`
`>=`	Greater than or equal	`age>=35`
`==`	Equal	`age==35`

Else

```
public class Example{
    public static void main(String[] args){
    int heightFromUser;                 //Input from user
    final int HEIGHT_LIMIT = 160;    //Hard limit

    if(HEIGHT_LIMIT > heightFromUser){
        System.out.println("Sorry not tall enough to enter")
    }
    else{
        System.out.println("You can enter");
    }
    System.out.println("This is the end");
}
```

Else if

```
int time = 22;
if (time < 10) {
  System.out.println("Good morning.");
} else if (time < 20) {
  System.out.println("Good day.");
} else {
  System.out.println("Good evening.");
}
```

https://www.w3schools.com/java/java_conditions.asp

Logic operators

operations	and	or	not
operators	&&	||	!

a	*!a*
`true`	`false`
`false`	`true`

a	*b*	*a && b*	*a || b*
`false`	`false`	`false`	`false`
`false`	`true`	`false`	`true`
`true`	`false`	`false`	`true`
`true`	`true`	`true`	`true`

```
if(password.length > 8){
    if(password.contains("@")){
        //Acccepted
    }
    else{
        //Not acceptted
    }
}
else{
    //Not Accepted
}
```

```
if(password.length > 8 && password.contains("@")){
    //Acccepted
}
else{
    //Not Accepted
}
```

The switch statement

Anatomy of switch

```
public class Switch{
    public boolean switchExample() {
    switch(/*expression*/)
        case /*result 1*/:
            /*What to do*/
        case /*result 2*/:
            /*What to do*/
        case /*result 3*/:
            /*What to do*/
    }
}
```

Switch

- Multiple outcomes within the same expression
- if/else can create the same outcome
- Switch can be more readable
- Especially useful in compound results

```java
String dayOfTheWeek;

switch(getNumberOfWeek()){
    case 0:
        dayOfTheWeek = "monday";
        break;
    case 1:
        dayOfTheWeek = "tuesday";
        break;
    case 2:
        dayOfTheWeek = "wednesday";
        break;
    case 3:
        dayOfTheWeek = "thursday";
        break;
    case 4:
        dayOfTheWeek = "friday";
        break;
    case 5:
        dayOfTheWeek = "saturday";
        break;
    case 6:
        dayOfTheWeek = "sunday";
        break;
}
```

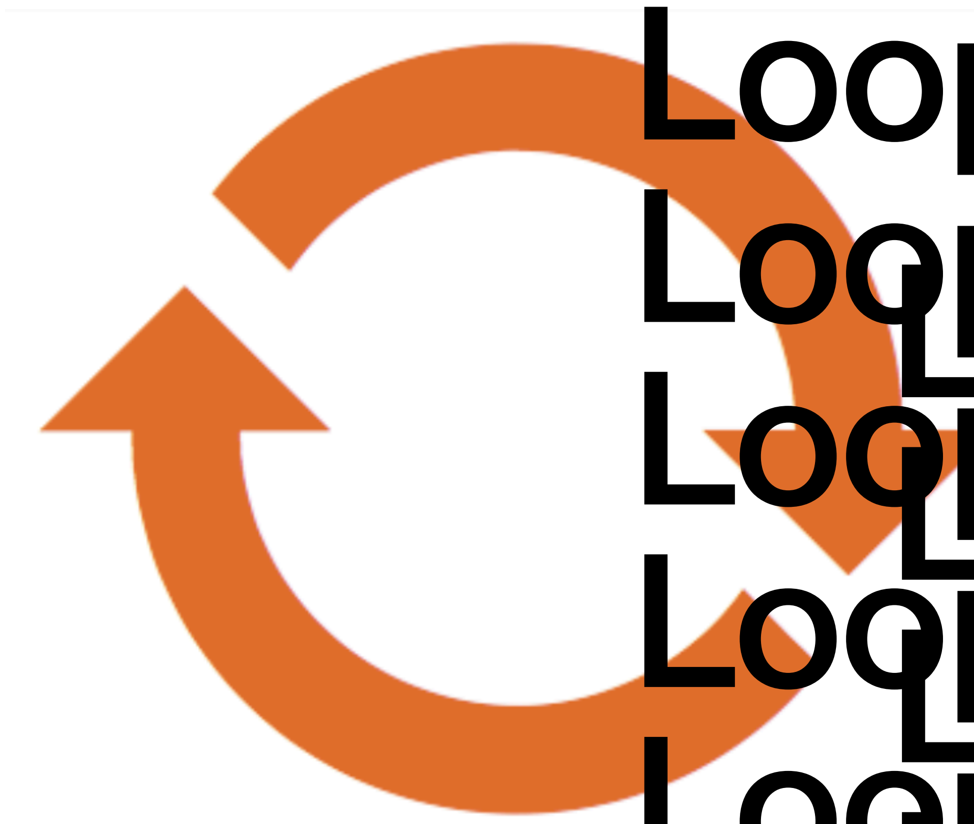
```
String season;

switch(getMonthInNumber()){
    case 11:
    case 0:
    case 1:
        season = "winter is coming";
        break;
    case 2:
    case 3:
    case 4:
        season = "Spring";
        break;
    case 5:
    case 6:
    case 7:
        season = "Summertime"
        break;
    case 8:
    case 9:
    case 10:
        season ="Fall"
}
```

Warmup exercise 1 15 min

- Request username and password from the user (Scanner)
- If the username is longer than 10 characters print out: “Too long”
- If the password contains “#” print out: “Invalid”
- Otherwise print out: “Accepted”

Loops CS101

What are loops?

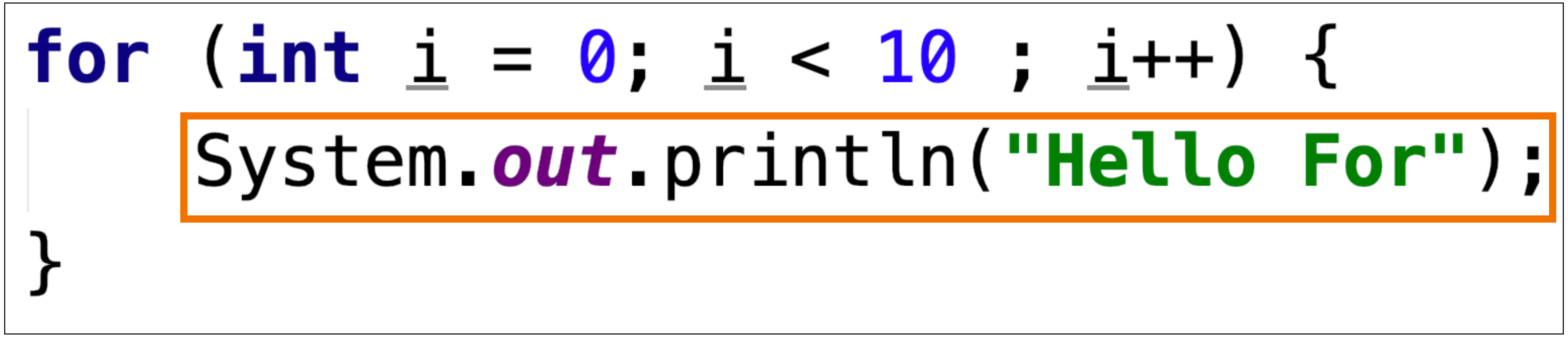


Loops
Loops
Loops
Loops
Loops
Loops
Loops
Loops
Loops
Loops
Loops
Loops
Loops
Loops
Loops

The program needs to do [something] a number of times

For loop

[something]

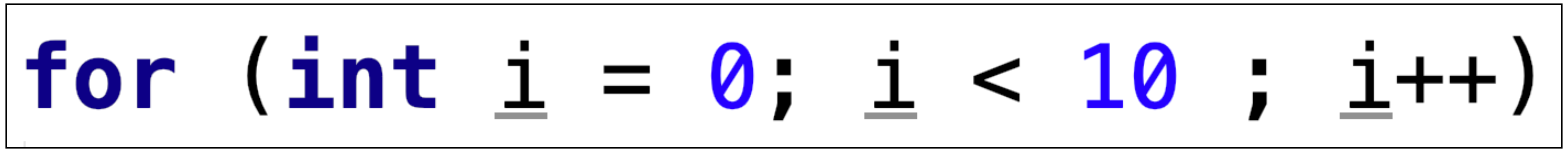


```
for (int i = 0; i < 10 ; i++) {
    System.out.println("Hello For");
}
```

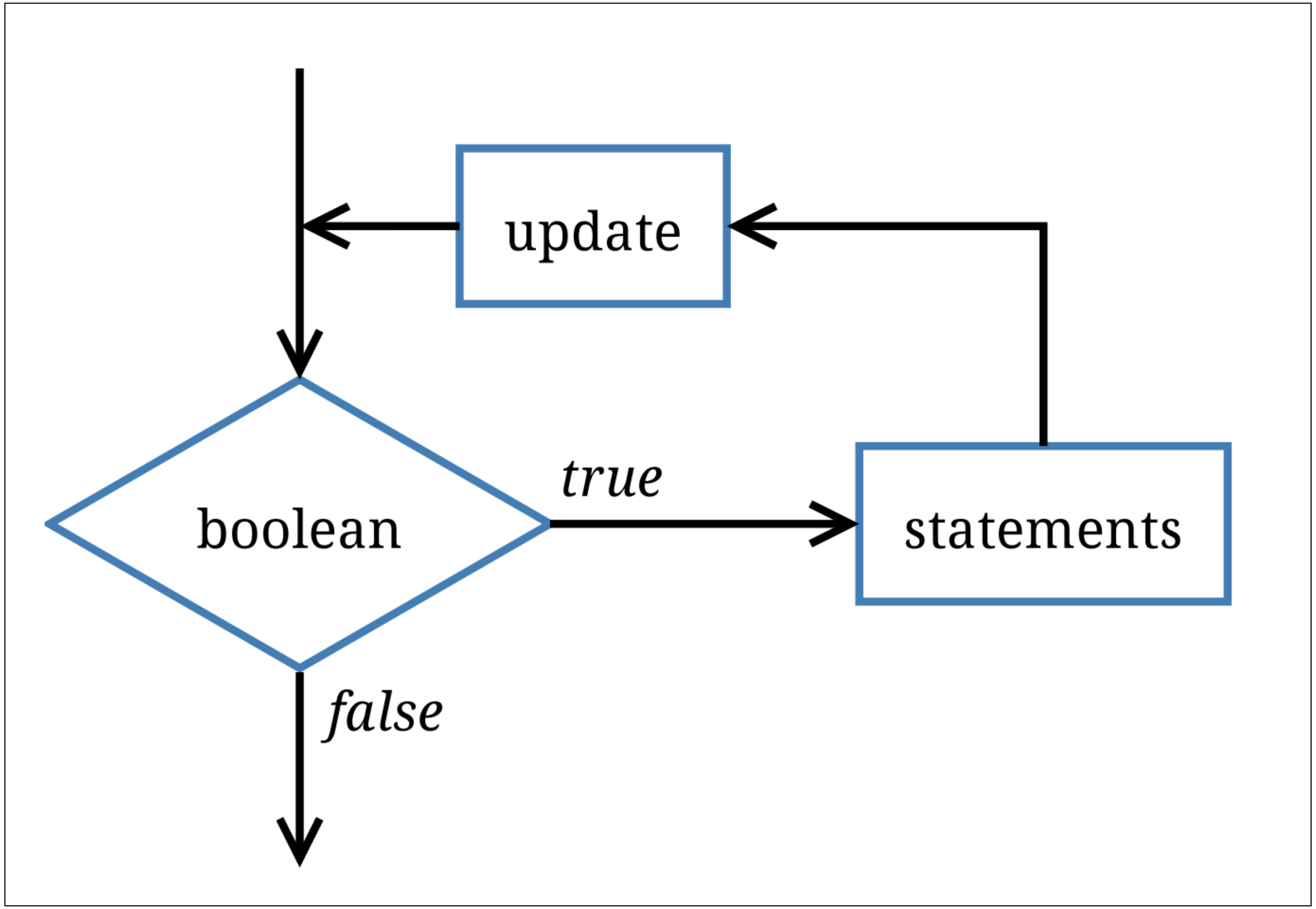
For loop “signature”

a number of times

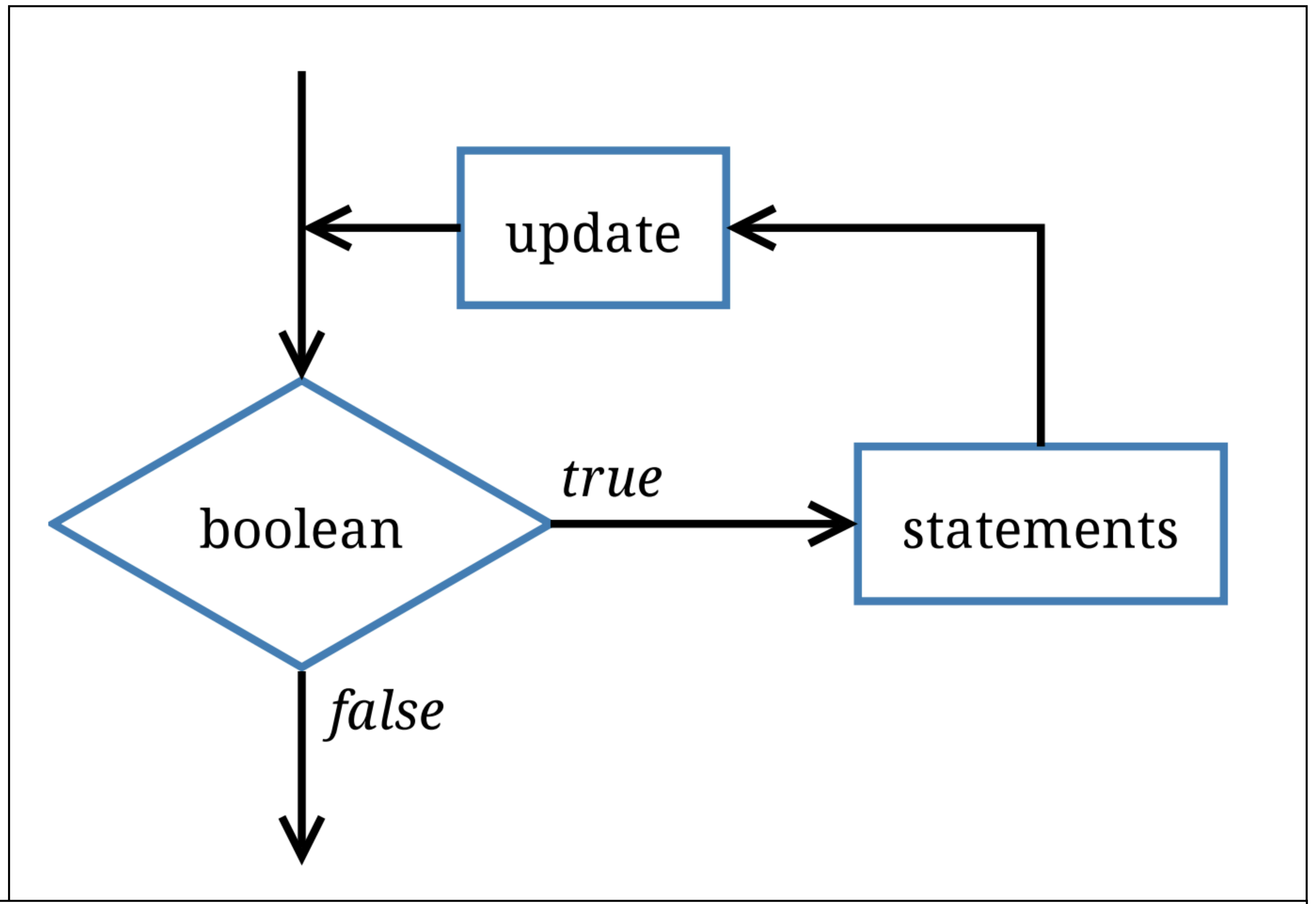
- For loops are used **when we know how** many iterations we need
- Incremental counter / iteration
- Can be decremental
- Counter variable can be used in the scope of the loop

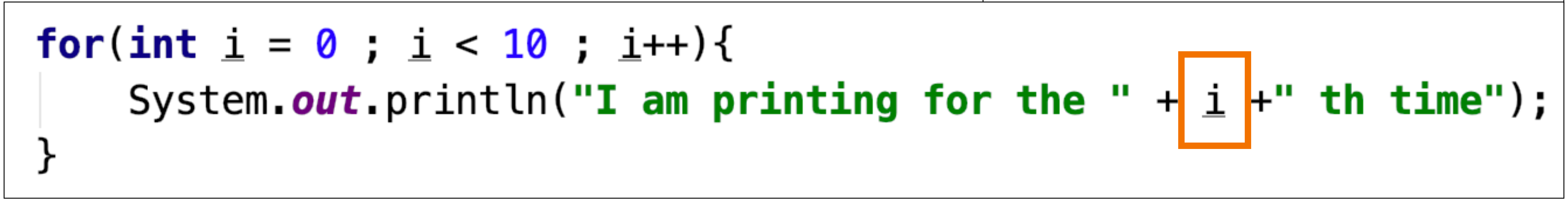


```
for (int i = 0; i < 10 ; i++)
```


update
boolean
true
statements
false

For loop





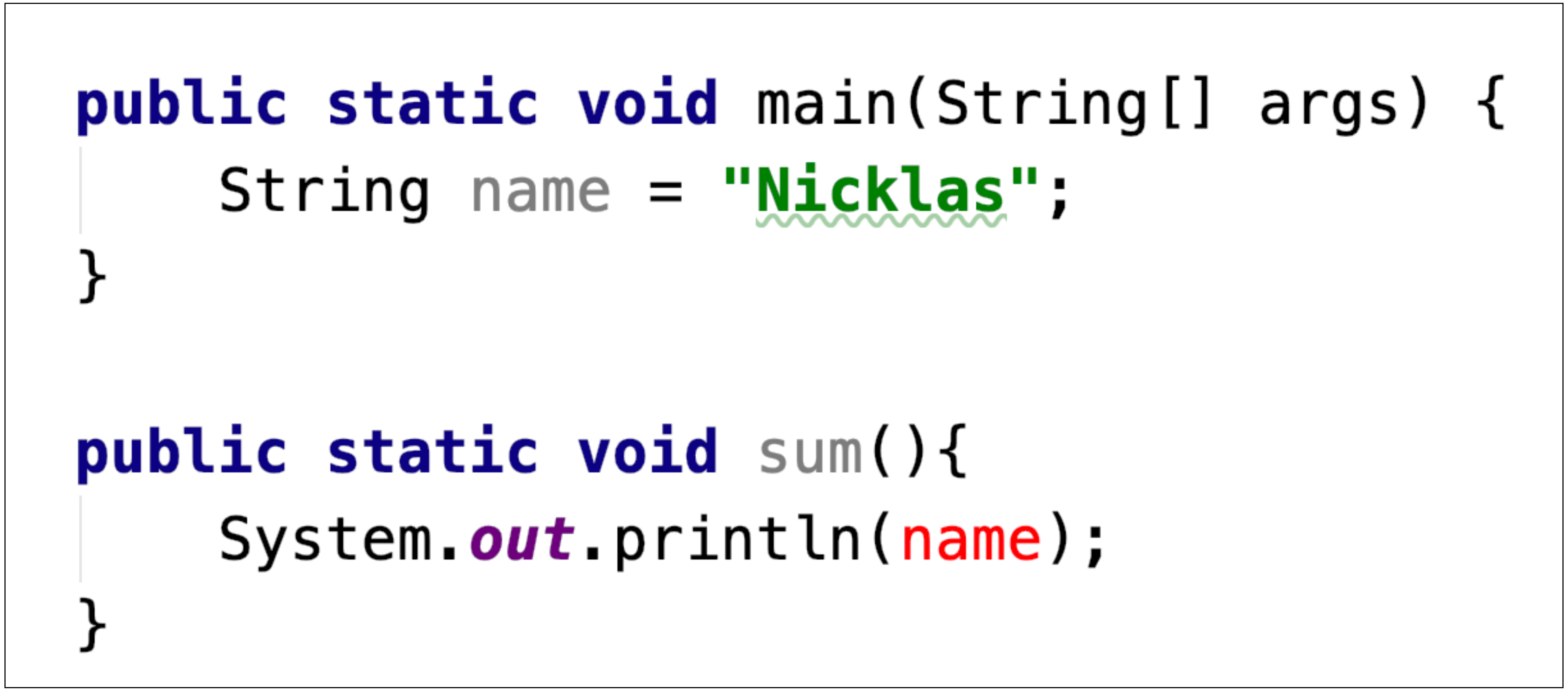
```
for(int i = 0 ; i < 10 ; i++){
    System.out.println("I am printing for the " + i +" th time");
}
```

Break

Methods CS101

```java
public class HelloWorld{
    public static void main(String[] args){
        //The main method will be executed first
        //It is at the bottom of the call stack
        System.out.println("Obligatory Hello World");
    }
}
```

Method scope



```
public static void main(String[] args) {
    String name = "Nicklas";
}

public static void sum(){
    System.out.println(name);
}
```

Defining the method

```
public static void main(String[] args){
    sumTwoNumbers(5,5);
}

public static void sumTwoNumbers(int first, int second){
    System.out.println("The result is: " + (first + second));
}
```

Calling the method

```java
public static void main(String[] args){
    sumTwoNumbers(5,5);
}

public static void sumTwoNumbers(int first, int second){
    System.out.println("The result is: " + (first + second));
}
```

Example: Returning the value

Why use methods?

- Writing reusable code
- DRY - Don’t repeat yourself
- Simpler and more accurate to test the sum of multiple small components than a single large

Explainer: Introduction

Exercises 1 CS101

Explaining a subject

Teacher/student exercise

- 20 min: Create a presentation on one or more subjects with a partner
- 5 min: Presentation & Active Critical Listening
- 5 min: Questions, common ground & differences

Explainer

Topics

a) What is the difference between a void method and a value method?

b) What is meant by the method scope?

c) What is method overloading?