Object-oriented programming

13th of February 2019
Ljubov Jaanuska
ljabov.jaanuska@ut.ee
Outline

• Introduction
  – to the course
    • What for?
    • How?
    • When?
    • If?
  – into Java
    • Java
    • First programs
    • ...
    • ...

Why OOP?

• Running on programming
  • Python
  • Java

• Broaden perspectives
  – objects, subjects
Programming language

• Java
  – popularity
    • https://www.tiobe.com/tiobe-index/
  – vacancies
    • http://www.cvkeskus.ee
    • http://www.cv.ee/too/infotehnoloogia/q-java
People are different

- take the course for fun
- programming fans
- hate Python
- participated in programming competitions
- got E for the *Computer Programming*
- OOP golden customers

- Do not be afraid to ask help:
  - Moodle forum
  - Google
  - ljubov.jaanuska@ut.ee
The *Computer Programming* course was...

- easy
- fair
- hard
- very hard
What grade did you get for the Computer Programming?

- A
- B or C
- D or E
- F
Course outline

• 6 EAP $\rightarrow$ 6*26 h = 156 h

• Sessions
  – Wednesdays 12.15 – 15.45

• **Flipped learning**
  – Before session: videos, text material, homework
  – During session: revision

• 2 tests
  – Time limited

• Project in pairs

• Exam
Pay attention to...

• Creativity
  – programming

• Focus
  – sessions create friendly environment for studies
  – a good deal of time for self-education

• Co-work
  – with students
    • sessions in pairs
    • project
    • forum
  – with the instructor
    • questions-answers

• Timing
Grading

- Home assignments
  - 1 point x 12
- In-class assignments
  - 1 point x 12
  - test 1 - 16 points
  - test 2 - 16 points
  - project: 5 points x 2
  - project public presentation - 3 points
- Exam - 33 points
  - To take the exam, a student must have:
    - at least 10 points for Test 1
    - at least 12 points for home assignments and group works
    - at least 4 points for in-class sessions
  - To pass the exam, at least 15 points must be obtained.
- Grading
  "E" 51-60, "D" 61-70, "C" 71-80, "B" 81-90, "A" 91-
What grade do you aim to get in this course?

- A
- B or C
- D or E
- F
On the completion of the course, the student:

* is able to define the essential concepts of object-oriented paradigm (encapsulation, abstraction, inheritance, polymorphism, overloading, overriding) and analyze the corresponding programs
* is able to describe different data structures (array, linked list, stack, queue, hash table) and their use
* is able to explain the value and nature of application programming interfaces (APIs) and find useful information from them
* is able to explain the essence of event-driven programming and exception handling and exemplify their use
* is able to design, implement, test, and debug programs in one object-oriented programming language with integrated development environment (IDE) implementing the above items
* is able to explain the essential elements of teamwork drawing on personal experience
Materials

• https://courses.cs.ut.ee/2019/oopn/spring/Main/HomePage

• https://moodle.ut.ee/course/view.php?id=8254

• http://docs.oracle.com/javase/tutorial/

• H. Schildt *Java: The Complete Reference*
Java

• James Gosling, *Sun Microsystems*
• *Oak, Green*
• 1995

• Name

• History
Java

- Easy
- Object-oriented
- Strongly typed language
- Secure
- OS independent
  - write once, run anywhere
  - bytecode
- Multithreads
- Dynamic
- Also
API, JDK, IDE

• **API** – *Application Programming Interface* – library, programmed Java objects for developing own applications
  – we are going to use Java SE 11

• **JDK** – *Java Development Toolkit* – software package for developing Java programs
  – we are going to use JDK 11

• **IDE** – *Integrated Development Environment* – programming environment
  – IntelliJ IDEA Community
  – Eclipse IDE
Java Technologies

• **Java SE**
  – Standard Edition

• **Java EE**
  – Enterprise Edition

• **Java Cloud**

• **Java Embedded**
  – Java SE Embedded
  – Java ME
    • Micro Edition
  – Java Card
  – Java TV

https://www.oracle.com/java/technologies/

• **JavaFX**
  – Client Application Platform

https://openjfx.io/
Which company developed Java?

- Dragon Ltd
- Smilers Computers
- Sun Microsystems
- Grape International
A package for developing Java programs is ...

- JFK
- API
- JDK
- ACTA
First program

- Task 1
  
  https://courses.cs.ut.ee/2019/oopn/spring/Main/s1
First program

```java
/*
Long comment
*/

//Short comment

public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, world!");
    }
}

File name: HelloWorld
Compile: >javac HelloWorld.java
Execute: >java HelloWorld
```
The source file by convention is named after the class within it and has the .java extension.

Access modifier. Both the class and method are **public** so can be accessed by anyone.

The class keyword used for specifying a class.

Opening curly brace of the class.

Opening curly brace of the the main () method.

This is known as a return type and all methods are required to have one. The **void** keyword denotes that this method doesn't return a value.

Name of method.

Method parameters go between these parentheses.

Opening curly brace of the the main () method.

The **static** keyword allows the main () method to be called before any objects of the class have been created.

The main () method contains one or more statements which must be declared within the curly braces of the main () method and end in a semi-colon.

Closing curly brace of the class.

Closing curly brace of the the main () method.

This denotes the arguments to the main () method. The main () method must declare an array of objects of type String. An array is a collection of similar objects or primitives. The name of the array is called args but you can call it whatever you like. More on arrays and the other stuff mentioned here will be gone into later in the lessons.
use any text editor to create your program

type javac HelloWorld.java to compile your program

type java HelloWorld to execute your program

editor → HelloWorld.java → compiler → HelloWorld.class → JVM → "Hello, World"

your program (a text file)

Error

computer-language version of your program

Error

output

Unexpected result
Bytecode

• Language for JVM (Java Virtual Machine)

H:\oop19> javap -c HelloWorld
Compiled from "HelloWorld .java"
public class HelloWorld {
    public HelloWorld();
    Code:
        0:  aload_0
        1:  invokespecial  #1 //Method java/lang/Object."<init>":()V
        4:  return

    public static void main(java.lang.String[]);
    Code:
        0:  getstatic   #2 //Field java/lang/System.out:Ljava/io/PrintStream;
        3:  ldc   #3 //String Hello, world
        5:  invokevirtual  #4 //Method java/io/PrintStream.println:(Ljava/lang/String;)V
        8:  return
}
Java executes lines of code…?

- From top to bottom
- All at once
- In stages from less complex to more complex
- From top to bottom including jumps if it encounters a method
Components of the first program

• Comments
• Keyword
  – public, class, static, void
• Access modifier
  – public, static
• Statements
  – Generally end with a semi-colon (;)
• Blocks
  – {     }
• Classes
• Methods
  – Method main
Reserved keyword

abstract assert boolean break byte case catch char class const*
continue default double do else enum exports extends final finally float
for goto* if implements import instanceof int interface long module
native new package private protected public requires return short static
strictfp super switch synchronized
this throw throws transient try void volatile while

• Literals (not keywords)
  false null true
Variables and data types

Every Java variable has its data type

– types
  • primitive or object-oriented
– name
– value

int x = 17;
Primitive data types

• Whole numbers
  – byte
  – short
  – int
  – long
• Fractions
  – float
  – double
• Boolean
  – boolean
• Symbols
  – char
Whole numbers

- **byte**
  - 8 bytes
  - -128 .. 127

- **short**
  - 16 bytes
  - -32 768 ... 32 767

- **int**
  - 32 bytes
  - -2 147 483 648 ... 2 147 483 647

- **long**
  - 64 bytes
  - -9 223 372 036 854 775 808 ... 9 223 372 036 854 775 807
    (quintillion)
Which is not a primitive data type?

- int
- float
- double
- String
What is the output?

```java
int n = 1000000;
System.out.println(n*n);
```

- 1000000000000
- -727379968
- error
- something else
Fractions

• IEEE 754
  – float
    • 32 bytes
  – double
    • 64 bytes

• double radius = 7.8;
Boolean

- boolean
  - true
  - false

- boolean tv = true;
Symbols

• **char**
  – 16 byte for the *Unicode symbols*
  – '\u0000' (0) ... '\uffff' (65535)

• `char ch1 = 'a';`
• `char ch2 = 65;`
• `System.out.print("ch1 and ch2: ");`
• `System.out.println(ch1 + " " + ch2);`
Names

• Case sensitive
• Can contain numbers, letters, _, $
• Cannot begin with a number
• Cannot be a reserved word
• Cannot be true, false, null
Which of the following are valid names for Java *classes* according to the convention?

- Hello World
- HelloWorld
- helloWorld
- Hello-World
Which one of the following is NOT a correct variable name?

- 2bad
- zero
- theLastValueButOne
- year2000
Will the following line cause an error: `int Public = 50000`?

• Yes
• No
Declaration and assignment

char a = 'e';
double radius;
int x, y, z, k;
radius = 3.4;
x = y = z = 1;
int i = 1, j = 2;
k = 1 + 5;
x = y + z;
long code = 48007140350L;
Which one of the following declarations is NOT correct?

- double duty;
- float loan;
- boolean value = 12;
- int start = 34, end = 99;
Operators and precedence

| ++  | --  | !  | unary operators |
| *   | /   | %  | multiplicative  |
| +   | -   | additive |
| >   | >=  | <  | <= relational   |
| ==  | !=  | equality |
| &&  | logical AND |
| ||  | logical OR |
| =   | +=  | -=  | *=  | /=  | %=  | assignment |
# Increment and decrement

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>++var</code></td>
<td>The value is incremented by one and then the expression is calculated using the new value of the variable</td>
</tr>
<tr>
<td><code>var++</code></td>
<td>The expression is calculated using the old value of the variable and then the variable value is incremented by one</td>
</tr>
<tr>
<td><code>--var</code></td>
<td>The value is decremented by one and then the expression is calculated using the new value of the variable</td>
</tr>
<tr>
<td><code>var--</code></td>
<td>The expression is calculated using the old value of the variable and then the variable value is decremented by one</td>
</tr>
</tbody>
</table>
### Math class

<table>
<thead>
<tr>
<th>Method</th>
<th>Example Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>double random()</td>
<td>double x = Math.random();</td>
</tr>
<tr>
<td>double abs(double x)</td>
<td>double x = Math.abs(-12.5);</td>
</tr>
<tr>
<td>double floor(double x)</td>
<td>double x = Math.floor(4.2);</td>
</tr>
<tr>
<td>long round(double x)</td>
<td>long x = Math.round(0.66);</td>
</tr>
<tr>
<td>double ceil(double x)</td>
<td>double x = Math.ceil(4.2);</td>
</tr>
<tr>
<td>double sin(double radians)</td>
<td>double x = Math.sin(Math.PI / 2.0);</td>
</tr>
</tbody>
</table>
What is the output?

```java
int i = 5;
int uus = 5 * i++;
System.out.println(uus);
```

- 25
- 26
- 30
- something else
What is the output?

```java
int i = 5;
int uus = 5 * ++i;
System.out.println(uus);
```

- 25
- 26
- 30
- something else
What is the output?

```java
int i = 5;
int j = (5 * i)++;
System.out.println(j);
```

- 25
- 26
- 30
- something else
Is it allowed to use in Java:

```java
int x = 5_000_000;
```

- Yes
- No
What is the result of the following expression: $1 + 0 + "0" + 5 * 10$?

- 100510
- 1050
- 1015
- 51
What is the result of the following expression: \((\text{int}) (10.0/3.0)\) ?

- 3
- 3.3333
- 1
- 3.0
Ask data from user

```java
//Import the package to work with the console input
import java.util.Scanner;

public class Data_Input {
    public static void main(String[] args) {
        //Create own Scanner object with a name (e.g. "scan") and
        //tell it to read text from the keyboard System.in
        Scanner scan = new Scanner(System.in);

        System.out.println("Enter your name");
        //Get a string from the input
        String userName = scan.nextLine();

        System.out.println("Enter an integer");
        //Get an integer from the input
        int myInt = scan.nextInt();

        System.out.println("Enter a double");
        //Get a double from the input
        double myDouble = scan.nextDouble();

        System.out.println(userName + " your entered integer: " + myInt);
        System.out.println(" and double: " + myDouble);
    }
```
Which line creates a scanner object?

- `input = Scanner(System.in);`
- `Scanner input = Scanner(input);`
- `Scanner input = new Scanner(System.in);`
- `Scanner input;`
Which is the correct way to allow a user to enter a String with spaces using the Scanner?

- `input.next()`;
- `input.nextInt()`;
- `input.nextDouble()`;
- `input.nextLine()`;
• Task 2, 3, 4

https://courses.cs.ut.ee/2019/oopn/spring/Main/s1
For next Wednesday

• Control statements
• Loops
• Methods
• Arrays

• https://courses.cs.ut.ee/2019/oopn/spring/Main/b2
See you in a week!