

Welcome to the

# Java Course

Module 2 – Day 01

# Content of the course

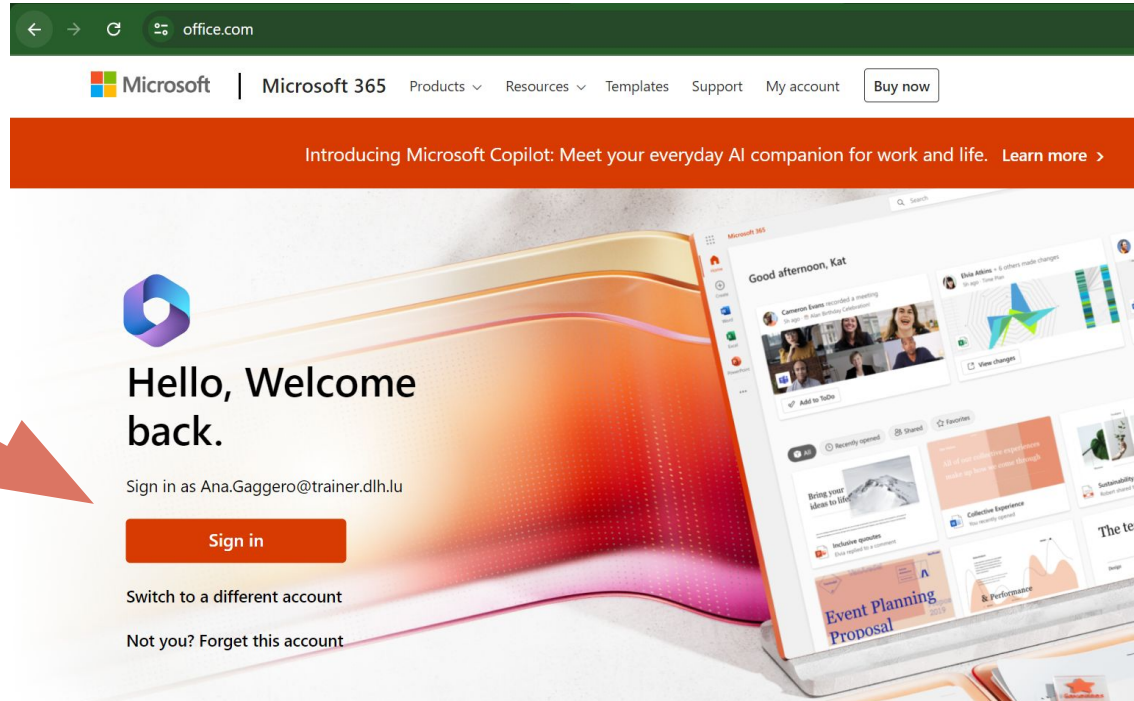
- Functions and procedures
- Arrays and lists
- Search and sorting algorithms
- Data structures
- Computational complexity

# Log into the local PC and to Microsoft Teams

Open Google Chrome and type: office.com

Step 1: Click on “Sign in”

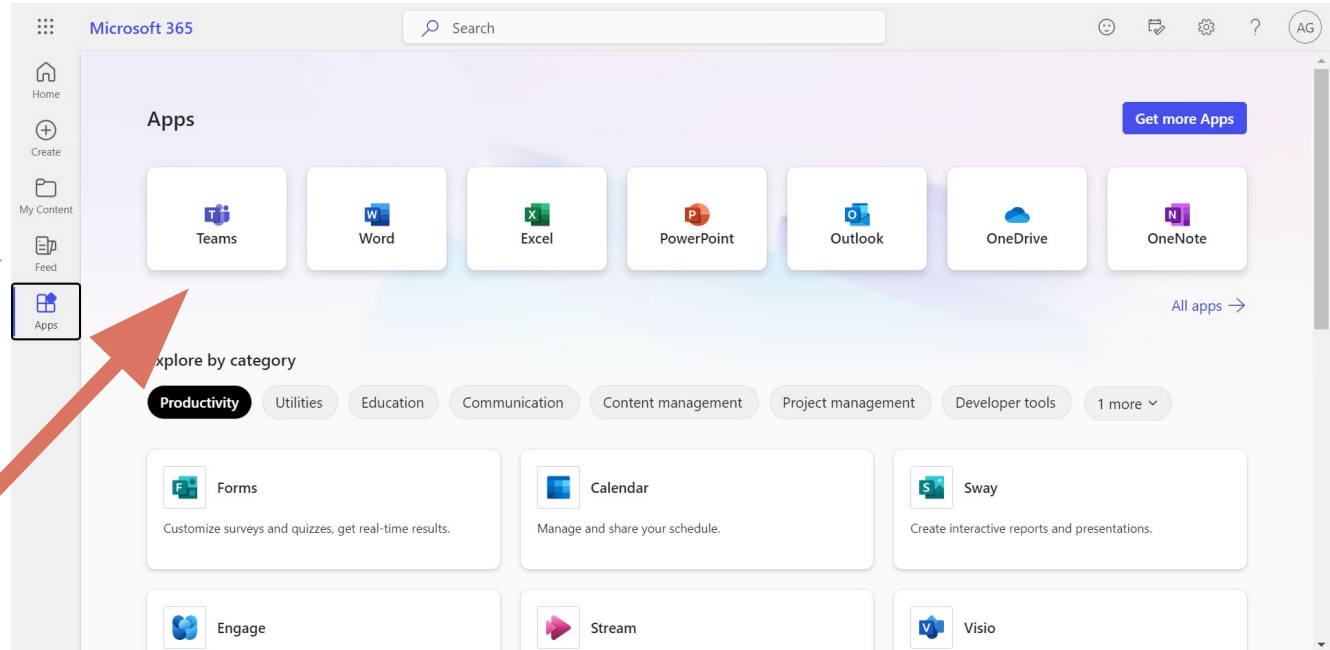
You should have received your password by email.



# Log into the local PC and to Microsoft Teams

Step 2: Click on “Teams”

1



2

# Log into the virtual machine

Step 1



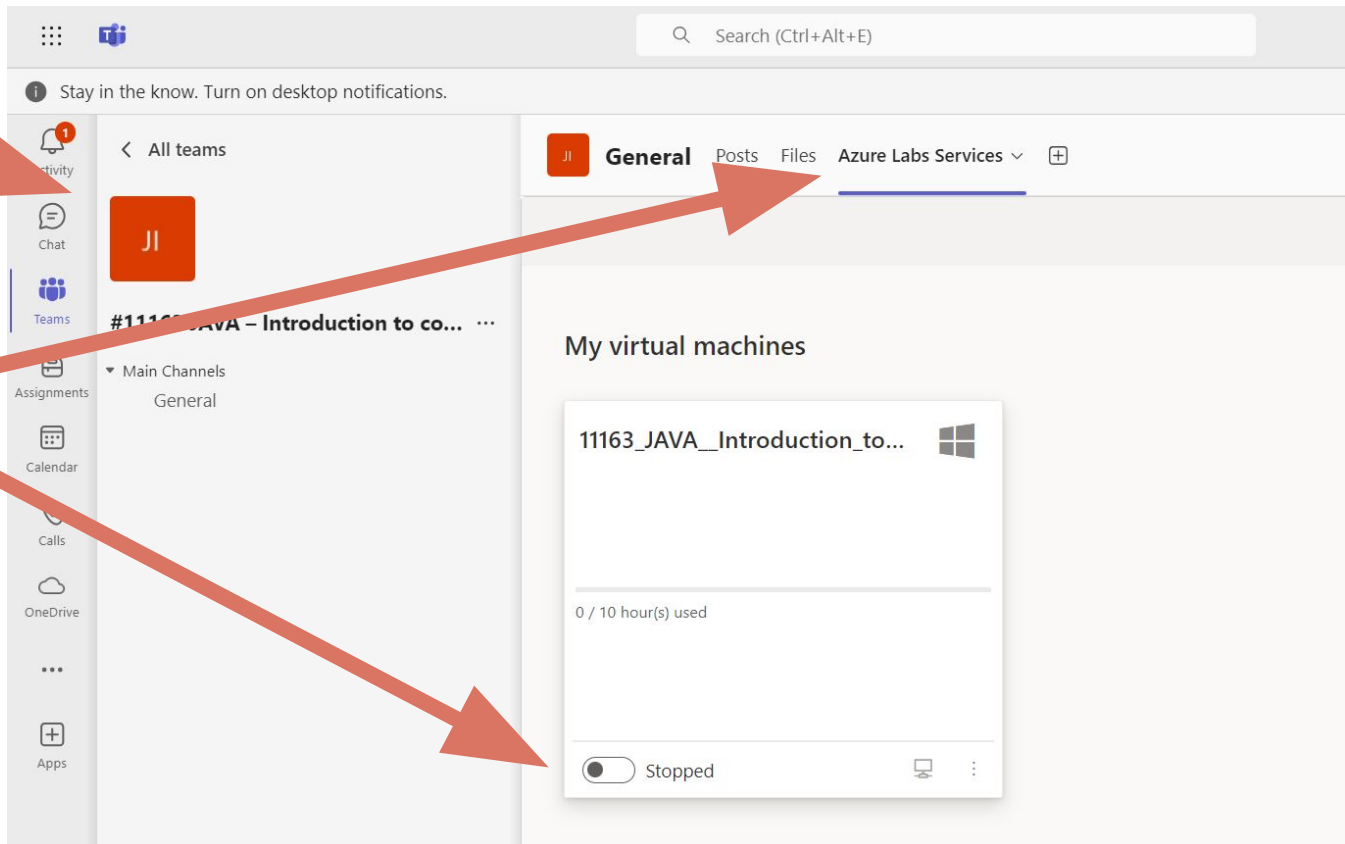
Step 2



Step 3



The virtual machine will be already on. If not, turn it on, it can take up to 5 minutes to start.



# Log into the virtual machine

**Username:** student

**Password:** StudentDLH2024

# The project

Continue to write a program that already gathers information from students and validates it. This week, we will be storing this data, allowing a user to search and visualize them in a certain way.

1. Student Registration
2. Data Validation
3. **Data storage**
4. **Data search**
5. **Data visualization**

# Code Editor

To develop in a programming language, you need a code editor that understands and executes the language. In this course, we will mainly use a very common one:

Visual Studio Code





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

**Let's START!**

# Functions

A function is a block of code that performs a specific task and returns a value.

The diagram illustrates the components of a Java function. A red rectangular box contains the function code. Above the box, labels with arrows point to specific parts of the function signature: 'Access modifier' points to 'public', 'Type of return value' points to 'int', 'Name' points to 'addNumbers', and 'Parameters' points to '(int num1, int num2)'. Inside the box, a lighter red rectangle highlights the return statement 'return num1 + num2;'. An arrow labeled 'Return value' points to the expression 'num1 + num2'.

Access modifier

Type of return value

Name

Parameters


```
public static int addNumbers(int num1, int num2)
{
    return num1 + num2;
}
```

Return value

# Procedures

A special type of function that does not return a value, known as a void method.

Access modifier      Type of return value      Name      Parameters

The diagram consists of four labels positioned above a red rectangular box containing Java code. Four red arrows point from the labels to specific parts of the code: 'Access modifier' points to 'public', 'Type of return value' points to 'void', 'Name' points to 'concat', and 'Parameters' points to '(String str1, String str2)'.

public void concat(String str1, String str2)

{

str1 += str2;

}

# Keywords

- return → exit a method
- void → no return value
- final → used to declare constants

```
1 public static final int NUM_STUDENTS = 100;
```

- static → belong to class, not object

```
1 public static final int MIN_VALUE = 0x80000000;
```

```
1 Integer.MIN_VALUE;
```

**Now YOUR TURN !**

Let's do exercises 1.1 and 1.2

# Single-line comments

```
// This is a comment
```

```
System.out.println("Hello World");
```

```
System.out.println("Hello World"); // This is a comment
```

# Multi-line comments

Multi-line comments start with `/*` and ends with `*/`.

```
/* The code below will print the words Hello World  
to the screen, and it is amazing */
```

```
System.out.println("Hello World");
```



# JavaDoc

Javadoc is a documentation generator for Java language. It will generate API documentation in HTML format from Java source code.

The format used by Javadoc is the de facto industry standard for documenting Java classes.

# JavaDoc

```
1 /**
2  * A simple program for student registration.
3  * @author Lisa Hennecart
4  */
5 public class StudentRegistration {
```

# JavaDoc

```
1 /**
2  * Calculates the sum of two numbers.
3  *
4  * @param a the first number
5  * @param b the second number
6  * @return the sum of a and b
7  */
8 public int sum(int a, int b) {
9     return a + b;
10 }
```

**Now YOUR TURN !**

Let's do exercises 2.1 and 2.2

# Arrays and Lists

Both used to store collections of elements but differ in:

- Fixed Size vs. Dynamic Size
- Primitives vs. Objects
- Direct vs. Indirect Access
- Performance
- Length vs. Size

# Arrays

```
1 // Array declaration and initialization
2 int[] array = new int[5];
3
4 // Adding elements
5 array[0] = 1;
6
7 // Accessing elements
8 int elementFromArray = array[0];
9
10 // Size/Length
11 int lengthOfArray = array.length;
12
13 // create an array with all same values
14 int[] zeroArray = new int[5];
15 Arrays.fill(zeroArray, 0);
16
17 // initialize array with values
18 int[] myArray = {1, 2, 3, 4, 5};
```

# Lists

ArrayList<>

```
1 // ArrayList declaration and initialization
2 ArrayList<Integer> arrayList = new ArrayList<>();
3
4 // Adding elements
5 arrayList.add(1);
6
7 // Accessing elements
8 int elementFromArrayList = arrayList.get(0);
9
10 // Size/Length
11 int sizeOfArrayList = arrayList.size();
12
13 // Creates an ArrayList with 5 elements, all set
   to 10
14 ArrayList<Integer> zeroList = new ArrayList<>
   (Collections.nCopies(5, 0));
15
16 // initialize array with values
17 ArrayList<Integer> myList = new ArrayList<>
   (Arrays.asList(1, 2, 3, 4, 5));
```

# Polymorphic variables

Ability to hold different data types into a variable

```
1 List<String> myList = new ArrayList<>();
```



# Wrapper variables

- `int` → `Integer`
- `char` → `Character`
- `Double` → `Double`

```
1 List<Integer> myList = new ArrayList<>();
```

**Now YOUR TURN !**

Let's do exercises 3.1 and 3.2

# Project Students - Step 6

```
How many students do you want to register? 3
>>> Student 1 <<<
Enter first name: Ana
Enter last name: Gaggero
Enter birthday (day of month): 22
Enter birth month: 10
Enter birth year: 1982
Enter course registered: Java
>>> Student 2 <<<
Enter first name: Carol
Enter last name: Muller
Enter birthday (day of month): 12
...
```

# Project Students - Step 6

```
...
>>> Student 3 <<<
Enter first name: Tom
Enter last name: Grass
Enter birthday (day of month): 7
Enter birth month: 1
Enter birth year: 1980
Enter course registered: Java

List of registered students:
Ana Gaggero born the 22 of October 1982. Registered to Java
Carol Muller born the 12 of April 1990. Registered to Python
Tom Grass born the 7 of January 1980. Registered to Java
```

# Project Students - Step 6

Modify the program:

- Split the code into methods (functions and procedures). At least one to convert the month from number to text, one to request each student's information and one to print the students information.
- Store the students information in a List instead of storing all students together in one String.