Algorithms and computational complexity

Laboratory 1: A rough repetition on Java programming.

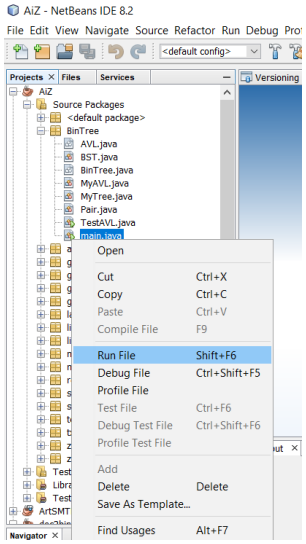
1. Classes.

As Java is an object-oriented programming language, classes are building blocks of the programs. The whole code must be contained within classes. Each public class must be stored in a file named exactly like the class, with the extension "java". E.g. MyApp class should be placed in the file MyApp.java:

```
1.  public class MyApp {
2.      //inside we declare attributes (variables) and methods (functions) which can be private or public
3.      private int x = 0;
4.
5.      public int getX() {
6.          return x;
7.      }
8.
9.
10.     / **
11.     @param args the command line arguments
12.         The main method is the entry point of the program
13.     */
14.     public static void main(String[] args) {
15.         // TODO code application logic here
16.         MyApp app = new MyApp();
17.         app.x++;
18.         System.out.println("Hello! The value of x is " + app.getX());
19.     }
20. }
```

2. Compilation and run of the application.

If you are using an IDE type tool, simply select the appropriate option. For example in NetBeans:



However, compilation from the command line is performed using the javac command, e.g.

javac MojaAplikacja.java

As a result of compilation files with the *class* extension are created. Running the application from the command line is implemented using the java command, e.g.

java. My application

Remeber that the class to be run must contain a static main method with a signature as in the above example.

3. An example application. Reading input from the keyboard and writing to the console.

```
1.  // using a class from external library
2.  import java.util.Scanner;
3.
4.  public class MyApp {
5.
6.    public static void main (String [] args) {
7.          My application app = new MyApp ();
8.          app.hello ();
9.    }
10.
11.   private void hello () {
12.         // using an instance of Scanner one can conveniently read from keyboard (or from files and streams)
13.         Scanner sc = new Scanner (System.in);
14.         // print a message
15.         System.out.println ("Enter your name");
16.         // read a line of text from the keyboard
17.         String s = sc.nextLine ();
18.         // concatenation and writing to the screen
19.         System.out.println ("Hello" + s);
20.   }
21.
22. }
```

4. Loops:

```
1.  while (condition) {
2.
3.      instructions ...
4.
5.   }
6.
7.   // in contrast to the above, the instructions are always executed at least once!
8.   do {
9.
10.     instructions ...
11.
12.  } while (condition);
13.
14.
15.  for (int i = 0; i <10; i ++) {
16.
17.     instructions ...
18.
19.  }
```

5. Arrays

Declaration of a five-element array containing integers:

```
int tab [5];
```

Arrays are indexed from 0. Operations on array elements:

```
1.  x = tab [0];
2.  tab [1] = tab [2] + 3;
3.  tab [4] = 7;
```

6. Methods

The methods are functions defined inside classes and interfaces. A method definition is as follows:

modifiers returnedValueType **methodName** (argumentList) {
    instructions ...
}

For example:

```
1.  public class Xyz{
2.    public void message () {  // this method returns nothing, so the return type is void
3.       System.out.println ( "Note");
4.    }
5.
6.    private double square (double x) {
7.       return x * x;
8.    }
9.  }
```

7. Tasks

1) Draw a block diagram and write a program finding the largest common divisor for two natural numbers. Use Euclidean algorithm with subtraction. For example, for 4 and 12 the result is 4.

2) Draw a block diagram and write a program to check if the given natural number is a prime number.

3) Draw a block diagram and write a program that converts a binary number to its decimal representation. For example, $(1001)_2 = 9$; $(10111)_2 = 23$

4) Write a program that finds prime numbers using the Eratosthenes sieve method.

5) Write a program calculating the sum of digits for a given number. Stop if we get a single digit number. Otherwise calculate the sum of the digits of the result until we get a one-digit number. For example sum(1234567)= 28, sum(28)=10, sum(10) = **1**.

In your program define at least three methods: a method that reads the number, a method for calculating the sum of digits and a method writing the result.

8. Recursion

An example method using recursion that calculates the largest common divisor:

```
1.  private int lcd (int a, int b) {
2.     if (a == b) {
3.         return a;
4.     }
5.     else if (a> b) {
6.         return lcd (a-b, b);
7.     }
8.     else {
9.         return lcd (b-a, a);
10.    }
11. }
```

<u>Tasks:</u>

6) Write a program that recursively calculates the factorial (n!) of the given natural number n.

fac(0) = 1;

fac(n) = n * fac(n-1), for n > 0;

7) Write a program calculating the recurrent nth word of the Fibonacci sequence.

f (0) = 0, f (1) = 1,

f (n) = f (n-1) + f (n-2), for n> 1

8) Write a program that calculates the Newton symbol recursively:

$$\binom{n}{k} = \begin{cases} 1 & \text{for k=0 or k=n,} \\ \binom{n-1}{k-1} + \binom{n-1}{k} & \text{for } 0 < k < n \end{cases}$$

<u>Next week:</u>

Arrays and files. Collections. Representation of Sets. Operations on sets (union, product, subtraction).