# Demo 4 / 6.10

In this demonstration you can download and import Ali.jar library. Further information on the implementation.

## Tasks

1. **Strings:** Create a program which works like following (a hint: check document for class `String`):

```
I lepeat what you said but I can't say l!
Give text >What is common between hiiri and rotta?
So, you said: What is common between hiili and lotta?
```

2. **Strings:** Did you pay attention to uppercase letters in the previous task? Modify the answer so that you make a function subroutine which returns a string and is called as follows:

```
String result = changeLetters(string,'r','l');
```

   The function will change both the upper- and lowercase letters (a hint: check document for class `Character`):

```
I repeat what you taid but I can't tay t!
Give text >Sister is in the tree
To, you taid: Titter it in the tree
```

   However, by changing two characters the program can be set to work like in the task 1.

3. **Comparison:** Create a program that asks user three integers and prints the highest and lowest of them. Create subroutines `highest(a,b,c)` and `lowest(a,b,c)` to help you. Think which values would be good for testing the subroutines.

4. **Strings:** Create required subroutines in order to get following main program to work (hints: `indexOf)`:

```
public static void main(String[] args) {
    String name = Syotto.kysy("Give last and first name");
    StringBuilder sb = new StringBuilder(name);
    deleteFirstWord(sb);
    System.out.println("Hi, your first name is " + sb);
    String lastName = giveFirstWord(name);
    System.out.println("and your last name is " + lastName);
}
```

5. **Loops:** Create a function subroutine `calculateCharacters(string, character)` which calculates the number of selected letters in a string. Test it with a main program that has calls like (make up more tests):

```
System.out.println(”%d%n”, calculateCharacters(”tree”,'e'));
System.out.println(”%d%n”, calculateCharacters(”tree”,'r'));
```

6. **Tables:** Get the example table presented on the lecture: Tables.java and modify it so that it has a subprogram called `printOver(table,limit)`, that prints all the numbers from

the table that exceed the given limit (it is for you to decide and comment on whether the limit number is included int those printed). Write a similar subprogram `sumOver(table,limit)`, that returns the sum of those numbers in the table that exceed the limit.

B1. Write a function subprogram that searches for the highest number in a double table. You can test this with a main program that has a table with preset values.

# GURU-tasks

G1-2 Write a program that has a double table in the main program. The cells of the table should be drawn as small circles. All circles are drawn in black except for the highest valued, which should be red.