Assignment 2

Problems?

Do not hesitate to ask your teaching assistant at the practical meetings (or Jonas at the lectures) if you have any problems. You can also post a question in the assignment forum in Moodle.

Prepare Eclipse for Assignment 2

Create a new *package* with the name YourLnuUserName_assign2 inside the Java project 1DV506 and save all program files for this assignment inside that package.

VG Exercises

Exercises 7-9 and 13 are VG tasks that is only needed to do if you aspire for a higher grade (A or B).

Lecture 4 - Control Statements

The ability to combine different control statements to solve a problem is very important. Therefore, there is quite a large number of (short) tasks on this topic.

- **Exercise 1**
  Write a program CountA.java that reads a string from the keyboard and then prints how many 'a' and 'A' the string contains. An example of what an execution might look like:

  ```
  Provide a line of text: All cars got the highest safty grading A.
  Number of 'a': 3
  Number of 'A': 2
  ```

- **Exercise 2**
  Write a program Backwards.java that reads an arbitrary string from the keyboard and then prints it backwards. An example of an execution:

  ```
  Provide a line of text: Anakin Skywalker
  Backwards: reklawykS nikanA
  ```

  Notice: You are supposed to use control statements to iterate over the input text backwards.

- **Exercise 3**
  Write a program Divisible.java that prints all the numbers from 100 to 200, ten per line, that are divisible by 4 or 5, but not both. The numbers in a printed line are separated by exactly one space.

  Notice: The program should not take any user input. It should just print the numbers formatted as described above.

- **Exercise 4**
  Write a program HighLow.java, implementing the game *High and Low*. The computer chooses a random integer between 1 and 100 and lets the user guess the value. After each guess, the user is given a clue of the type "higher" or "lower". An example of an execution:

  ```
  Guess 1: 67
    Clue: higher
  Guess 2: 82
    Clue: lower
  Guess 3: 77
    Correct answer after only 3 guesses – Excellent!
  ```

  After 10 guesses, the program ends with a proper comment.

- **Exercise 5**
  Write a program Triangle.java reading a positive odd integer N from the keyboard, and then prints two triangles. First a *right-angled* triangle, then an *isosceles* triangle. See example below to see what we mean by right-angled and isosceles triangles.

  ```
  Provide an odd positive integer: 7
  Right-Angled Triangle:
  *******
   ******
    *****
     ****
  ```

```
    ***
     **
      *

Isosceles Triangle:
     *
    ***
   *****
  *******
```

The program should end with an error message if the input N is not an odd postive integer.

- **Exercise 6**
  Write a program SecondLargest that reads 10 integers from the keyboard and then prints the second largest one. An example of an execution:

  ```
  Provide 10 integers: 67 -468 36 1345 -7778 0 34 7654 45 -666
  The second largest is: 1345
  ```

  Try to design the program such that changing the number of integers to be read (10) is easy.
  Recommendation: Use a smaller value than 10 while developing the program.
  Notice: You are not allowed to use arrays or any other data structure for storing all the integers.

- **Exercise 7 (VG-exercise)**
  Write a program CountDigits.java that for an arbitrary positive integer N (read from the keyboard) prints the number of zeros, odd digits, and even digits. An example of an execution:

  ```
  Provide a positive integer: 6789500
  Zeros: 2
  Odd: 3
  Even: 2
  ```

  Notice: We consider 0 to be neither odd nor even.

- **Exercise 8 (VG-exercise)**
  Write a program BirthdayCandles that computes how many boxes of candles a person needs to buy each year for his birthday cake. You can assume that the person reaches an age of 100, the number of candles used each year is the same as the age, that you save non-used candles from one year to another, and that each each box contains 24 candles. Also, at the end, we want you to print the total number of boxes one has to buy, and how many candles that are available after having celebrated the 100th birtday. An example of an execution:

  ```
  Before birthday 1, buy 1 box(es)
  Before birthday 7, buy 1 box(es)
  Before birthday 10, buy 1 box(es)
  Before birthday 12, buy 1 box(es)
  Before birthday 14, buy 1 box(es)

  ...

  Before birthday 95, buy 3 box(es)
  Before birthday 96, buy 4 box(es)
  Before birthday 97, buy 5 box(es)
  Before birthday 98, buy 4 box(es)
  Before birthday 99, buy 4 box(es)
  Before birthday 100, buy 4 box(es)

  Total number of boxes: 211, Remaining candles: 14
  ```

  Notice: In our example we only have a print-out of those birthdays where you must buy boxes. In the non-printed years (e.g. 2-6 and 8-9) you can handle the birthdays without having to buy any more candles.

- **Exercise 9 (VG-exercise)**
  Write a program Palindrome.java, testing if a line of text (read from the keyboard) is a palindrome. A palindrome is a text consisting of the same sequence of characters read backwards, as if read from the front. Ignore all characters that are not letters, and consider an upper case letter to be equal to the corresponding lower case letter. Examples of palindromes:

  ```
  "Anna"  "x"  "Ff"   "A1 n2%}=3N{[a]"  "Was it a rat I saw?"
  ```

  Hint: The two static methods Character.isLetter(char c) and Character.toLowerCase(char c) might be useful!

Lecture 5 - Arrays and ArrayList

- **Exercise 10**
  Complete the program Reverse.java below such that:
  1. It first prints the content of the array text in one line.
  2. Reorder the array elements backwards (in opposite order) by shifting place of first and last

character, second and second to last character, etc.

3. Then print the array content one more time.

```java
public class Reverse {
  public static void main(String[] args) {
    char[] text = { '!', 'y', 's', 'a', ' E', ' ', 's', 'a', 'w', ' ',
        's', 'i', 'h', 'T' };

    // Continue here ...
  }
}
```

Notice: You should not only print them backwards. You should also swap places of the elements in the array.

- **Exercise 11**
  Create a program ReverseOrder.java that reads an arbitrary number of positive integers from the keyboard and then prints them in reverse order. The reading stops when the user inputs a negative number. An example of an execution:

```
Enter positive integers. End by giving a negative integer.
Integer 1: 5
Integer 2: 10
Integer 3: 15
Integer 4: 20
Integer 5: -7

Number of positive integers: 4
In reverse order: 20, 15, 10, 5
```

Note: The user is not supposed to give the number of integers to enter.

- **Exercise 12**
  Write a program TwoDices.java that simulates that you are rolling two dices 10000 times. At the same time, keep track of the number of times you get the result (adding the dice values) 2, 3, ..., 11, 12. (Use an array to store the numbers.) After the simulation, present the frequencies for the different numbers. An example of an execution:

```
Frequency table (sum,count) for rolling two dices 10000 times
2 267
3 555
4 833
5 1101
6 1346
7 1690
8 1399
9 1083
10 862
11 580
12 284
```

Use the Random class to simulate the dices.

- **Exercise 13 (VG-exercise)**
  When the union is reporting about the latest salary negotiations they are presenting the *average salary*, the *median salary*, and the *salary gap* for the workers that they represent. Write a program SalaryRevision.java that reads an arbitrary number of salaries (integers) and then reports the median and average salaries, and the salary gap. All of them should be integers (correctly rounded off).

  By *salary gap* we mean the difference between the highest and lowest saleries. The *median salary* is the middle salary (or average of the two middle salaries) when all saleries have been sorted. The easiest way to sort an ArrayList is to use the static sort method in class java.util.Collections.

  Two different executions might look like this:

```
Provide salaries (and terminate input with 'X'): 21700 28200 26300 25100 22600 22800 19900 X
Median: 22800
Average: 23800
Gap: 8300

Provide salaries (and terminate input with 'X'): 22100 29800 27300 25400 23100 22300 X
Median: 24250
Average: 25000
Gap: 7700
```

Notice, the following programming pattern can be used to read an arbitrary number of integers from the keyboard.

```
Scanner scan = new Scanner(System.in);
...
```

```
while (scan.hasNextInt()) {
    int salary = scan.nextInt();
    ...
}
```

Submission

All exercises should be handed in and we are only interested in your .java files. (Notice that the VG exercises 7, 8, 9, and 13 are not mandatory.) Hence, zip the directory named YourLnuUserName_assign2 (inside directory named src) and submit it using the Moodle submission system.