Programming Course C++
Winter Semester 2016/17

- Dr. Stefan Harfst -

Exercise Sheet 10

- These exercises should be done in groups of two.
- After completing the exercises, create a tar-file using e.g. the command

```
tar -zcvf Exercise08_yourgroup.tgz <list of files>
```

  and then upload the resulting file in Stud.IP as shown in the lecture.
- This week's exercise sheet is due on Jan 16th, 2:15pm (i.e. before the next lecture).

What is the aim of this exercise:

- Learn how to define a derived class.
- Make use of the libraries from the C++ Standard Library, in particular `<vector>` and `<algortihm>`, to solve complex tasks.
- How to measure performance of an algorithm and estimate its scaling.

Problem 10.1: Random Distributions *(4 + 4 bonus points)*

Write a class `Uniform` which **inherits** from the `class LCG` from exercise 7.2 as a base. You can use the provided solution or your own implementation. The new class is intended for generating random numbers in a given range $[r_{\min}..r_{\max}]$ with the defaults $r_{\min} = 0$ and $r_{\min} = 1$.

- Implement two constructors to either use the default range or to set a chosen range. Do not forget to also initialize the members of the base class, which is done best calling the default constructor.
- You may modify the privacy of members from the base class using `protected`.
- Overload the `operator()` so that it returns a random number in the given range.
- Include a little program that prints ten random numbers in the range $[-3..5]$.

For the bonus points

- Following the same principle implement a class `Gaussian` (inheriting from `LCG`) to generate normal distributed RNs.
- The constructors should be used to set the mean (default 0) and standard deviation σ (default 1).
- Write a program that generates 100 random values with a standard Gaussian distribution and print the fraction of values that are within 1σ of the mean.

Problem 10.2: Using the C++ Standard Library *(5 points)*

The provided file `names_and_score.txt` contains a few thousands lines. Each line contains two entries: a name and a (seemingly random) number (score).

Write a program that sort the lines by the alphabetical order of the names. It should then print out the 2500th name and the total score e.g. as follows:

```
The 2500th name in the sorted list is: ......

total name score = 8........
```

The total score s is defined as

$$s = \sum (\text{score of name}) \cdot (\text{position in ordered list}) \tag{1}$$

where the summation is done for all names. For example, the name `COLIN` has a score of 53 and is the 938th name in the sorted list, therefore it adds $53 \cdot 938 = 49714$ to the total score s.

Solve this by using any of the libraries from the `C++` Standard Library, in particular `<vector>` and `<algorithm>` should prove useful. Follow these instructions:

- Define a `class NameScore` to store name and score in a single object.
- Read in the data from the file and store it in a `vector<NameScore>`.
- To read and store individual lines you may overwrite `istream& operator>> (istream& is, NameScore& ns)` as a friend of the class. Alternatively, use a constructor (remember problem 5.2).
- The `sort`-function from the `C++` SL requires `bool operator< (const NameScore& ns)` to be overloaded. Alphabetic comparison for `string` is already implemented in `<string>`.

The problem was inspired by Problem 22 at Project Euler[1].

[1] `http://projecteuler.net/`