TECHNISCHE UNIVERSITEIT EINDHOVEN

Examination Programming (2IP91)
Tuesday 29 January 2013, 14:00–17:00 uur

This exam consists of 3 questions on 2 pages.

Grading: At each question, the number of points maximally to be awarded is mentioned in the margin. The grade g for this examination is the total number of points achieved divided by 11. The final grade is the result of the following formula rounded to the nearest intger number: $0.6 \cdot g + 0.4 \cdot h$. Here g is the grade for this exam and h is the average of the 5 highest grades for the homework assignments (an assignment that was not submitted is graded with a 0). For a passing final result, the grade for this examination has to be 5 or more.

---

1 Statistics

Put the methods of this question in a class `Statistics` and submit this code in a file named Statistics.java .

15 1. Write a method `double sum(double[] a)` that returns the sum of the elements of array `a`.

15 2. Write a method `double mean(double[] a)` that returns the mean (or average) of the elements of array `a`. You may assume that the array is not empty.

15 3. Write a method `int[] prio(int[] a)` that returns an array that contains the same elements as `a` in a possibly different order. The largest value of `a` should be first and the originally first element should replace a largest value.

   Examples:
   `prio(new int[]{1, 2, 3})` returns `{3, 2, 1}` .
   `prio(new int[]{1, 3, 3})` returns `{3, 1, 3}` or `{3, 3, 1}` .

2 Humor

Submit your code either in a single file with the name Conference.java or in one file for each class, named after the class.

In the following exercises you are asked to add methods and instance variables to some classes. Feel free to add more methods and variables if you think this makes sense.

A comedian wants to keep track of his jokes using Java.

10 1. Write a class `Joke` with instance variables for the name of the joke, the text of the joke, the time it takes to tell the joke (in seconds), and a fun factor (a double between 0.0 and 10.0 that represents how funny the joke is). Add a constructor and a method `void print()` that prints the data of the joke.

10 2. Add a class `Conference` (Dutch word, sorry) that contains an ArrayList of the jokes that he wants to tell during a performance. Add a method `add(Joke j)` to `Conference` that adds a joke to the conference.

5 3. Add a method `int totalDuration()` to `Conference` that calculates the total amount of time needed to tell all the jokes of the conference.

10 4. The humorist wants to make subcategories for his jokes. He distinguishes the insulting joke from the normal joke (already existing). Apart from the normal data, an insulting joke has an instance variable of type `String` with the name of the social group that is primarily insulted by the joke.

   Add a class `InsultingJoke` that represents insulting jokes. Use inheritance.

10 5. After years of experience, the comedian has developed a calculation method to determine the *funniness* of a joke. It works as follows:

   - The funniness of a normal joke (i.e., not insulting) is the fun factor of the joke divided by the duration.
   - The funniness of an Insulting Joke is the fun factor multiplied by 1.5 and divided by the duration.

   Add methods `double calculateFunniness()` that calculate the funniness of a joke.

To help your intuition, a piece of code that uses these classes could look like this. You could, for example, include this in a main or demo method and use it for testing, but this is not required.

```
Conference conference = new Conference();
Joke j = new Joke("Afgeleide",
       "Twee functies komen elkaar tegen in de Kalverstraat...",
       30, 5.0);
conference.add(j);
Joke k = new InsultingJoke("Pastoorke",
        "Guus en Theo komen elkaar tegen op het Stratums Eind...",
        120,
        10.0,
        "Brabanders");
conference.add(k);
```

3 Reversal

Submit your code in a file named Reversal.java

20 Write a *recursive* method `String reverse(String s)` that returns the reversal of the String `s`. No loops are allowed.

Examples:
`reverse("papetrog")` returns `"gortepap"`;
`reverse("")` returns `""` (the empty String);
`reverse("parterretrap")` returns `"parterretrap"`

Some of the following operations on strings may be helpful (you probably don't need them all). In all the examples below, assume that `alfa` is the String `"abcd"`.

`s.substring(start)` returns the tail of `s`, starting with the character at position `start`; e.g., `alfa.substring(1)` returns `"bcd"`.

`s.substring(start, end)` returns the substring of `s` that starts at position `start` and ends at `end-1`; e.g., `alfa.substring(1,3)` returns `"bc"`.

`s.charAt(n)` returns the character at position `n` in String `s`; e.g., `alfa.charAt(3)` returns `'d'`.

`s.length()` (not `s.length` !) gives the number of characters in String `s`, e.g., `alfa.length()` returns 4.

Good luck!