582332: Programming in Python, Fall 2009

Exercise sheet 5
5-9.10.2009

1. Give pieces of code that trigger the following exceptions. You may not use `raise` directly. `AttributeError`, `FloatingPointError`, `IOError`, `ImportError`, `IndentationError`, `IndexError`, `KeyError`, `KeyboardInterrupt`, `MemoryError`, `SyntaxError`, `TypeError`, `UnboundLocalError`, `ValueError`, `ZeroDivisionError`.

2. Create a program that prints out the relative frequencies of each character in an input file. The relative frequencies should always sum to one. The name of the input file is given as a command line parameter. Remember to handle possible exceptions.

   For example, when the program is given a file that contains only the string "Jarkko", the output should look something like the following:

   ```
   The char a has relative frequency 0.167
   The char J has relative frequency 0.167
   The char r has relative frequency 0.167
   The char k has relative frequency 0.333
   The char o has relative frequency 0.167
   ```

   Try the program with the following test files: http://www.cs.helsinki.fi/u/jttoivon/python-09/alice.txt http://www.cs.helsinki.fi/u/jttoivon/python-09/ecoli.txt

   Which characters would seem to be most common in these files?

3. Create a module named `file_utils` that contains two functions:

   - `count_lines(`*filename*`)`
   - `count_characters(`*filename*`)`

   These functions return the number of lines/characters in a given file. Add docstrings for the module and for the functions.

4. Create a main program that uses the above module `file_utils`. The main program gets two command line parameters: *filename1* and *filename2*. It should then print which of the files contains more lines and which contains more characters. If the amount of lines/characters are equal, it should report this as well. Remember to handle possible exceptions: give an appropriate error message and exit the program with exit value 1.

   An example run:

   ```
   $ python main.py file1 file2
   The files file1 and file2 have equal amount of lines.
   The file file2 contains more characters.
   ```

5. Create a program called `summary` that gets filenames as a command line parameters. The input files should contain a floating point number on each line of the file. Make your program read these numbers and then print the average, sum, and standard deviation of these numbers for each file. If some line doesn't represent a number, you can just ignore that line.

6. Present the design of your project in a design document. Explain in detail what you are going to implement. How will the interface look? What modules and classes, etc you are going to have? What external libraries are needed?

   Return this document (at most two pages) by email to the teacher of your group beforehand, and also take a printed copy of your document with you to the session.

   If your description of the design of your project is quite inadequate, you will get only one point. Otherwise three points are given for this task.