Software Engineering 265
Software Development Methods
Spring 2013

Assignment 3

Due: Wednesday, March 13th, 9:00 am by submission via Subversion
(no late submissions accepted)

Programming environment

For this assignment you must ensure your work executes correctly on the Linux machines in ELW B215 (i.e., these have Python 2.6 installed). You are free to do some of your programming on your own computers; if you do this, give yourself a few days before the due date to iron out any bugs in the Python script you have uploaded to the BSEng machines.

Individual work

This assignment is to be completed by each individual student (i.e., no group work). Naturally you will want to discuss aspects of the problem with fellow students, and such discussion is encouraged. **However, sharing of code fragments is strictly forbidden without the express written permission of the course instructor (Zastre).** If you are still unsure regarding what is permitted or have other questions about what constitutes appropriate collaboration, please contact me as soon as possible. (Code-similarity analysis tools may be used to examine submitted programs.)

Objectives of this assignment

- Use some of the object-oriented features of Python.
- Use some of the error-handling features of Python.
- Use Subversion to manage changes in your source code and annotate the evolution of your solution with "messages" provided during commits.
- Test your code against the twenty provided test cases.
- Be prepared to justify your script's handling of errors during the assignment demonstration.

This assignment: "s265fmt2.py" & "formatting.py"

For this assignment you will be wrapping a solution to the formatting problem into a class. The class constructor will be given either a filename or a list as a parameter. The formatted output will be available from the class instance via the "get_lines()" method.

- If a non-blank filename is provided, then that file will be opened and its contents formatted.
- If a blank filename is provided, then the contents of stdin will be formatted.
- If no filename is provided (i.e., None is passed as filename argument), then the lines in the list provided as a parameter will be formatted. (Have a look at the code in /home/zastre/seng265/assign3/driver.py to find out how this would look.)
- The specifications from the second assignment are to be used for this assignment. However, the interpretation to be used for "?mrgn" commands must be that of the files in /home/zastre/seng265/assign3.
- The class must be named "seng265_formatter" and appear in a file named "formatting.py".
- The script that uses this class will be in a file named "s265fmt2.py".
- You must now provide some error handling. Your task is to enumerate the things that could go wrong when faced with such a formatting task, and to provide the code handling these.

With your completed "s265fmt2.py" script, the input would be transformed into the output (here redirected to a file) via one of the two following UNIX commands:

```
% ./s265fmt2.py /home/zastre/seng265/assign3/in11.txt > ./myout11.txt

% cat ~zastre/seng265/assign3/in11.txt |./s265fmt2.py > ./myout11.txt
```

where myout11.txt would be written in your current directory. As in the previous two assignments, our Teaching Assistants will use the "diff" UNIX command to compare your submission's output with what is expected.

Exercises for this assignment

1. Within your Subversion project create an "assign3" subdirectory. Use the test files in /home/zastre/seng265/assign3. (Files in01.txt through to in20.txt are the same as those used for the second assignment.) Your "s265fmt2.py" and "formatting.py" script files must appear here. Ensure the subdirectory and files are added to Subversion control.

2. Reasonable run-time performance of your script is expected. None of the test cases should take longer than 15 seconds to complete on a machine in ELW B215.

What you must submit

- Python script files named "s265fmt2.py" and "formatting.py" within your subversion repository containing a solution to Assignment #3.

- A text file named "error_handling.txt" which enumerates the errors that are handled by your submission.

Evaluation

Students will demonstrate their work to a member of the course's teaching team. Sign-up sheets for demos will be provided a few days before the due-date; each demo will require from 10 to 15 minutes.

Our grading scheme is relatively simple.

- "A" grade: An exceptional submission demonstrating creativity and initiative. "s265fmt.py" and "formatting.py" run without any problems. Errors have been enumerated and are suitably handled. The program is clearly written and uses functions appropriately (i.e., is well structured).
- "B" grade: A submission completing the requirements of the assignment. "s265fmt.py" and "formatting.py" run without any problems. Errors have been enumerated and are suitably handled. The program is clearly written.
- "C" grade: A submission completing most of the requirements of the assignment. "s265fmt.py" and "formatting.py" run with some problems. Some important error cases have been overlooked or are not handled.
- "D" grade: A serious attempt at completing requirements for the assignment. "s265fmt.py" and "formatting.py" run with quite a few problems.
- "F" grade: Either no submission given, or submission represents very little work.