Assignment 2 - evaluation

Submission and grading details

Submission Structure

Assignment 2 must be submitted in the `A2` branch in the GitLab repo.

The directory structure for A2 will be the same as A1.

Reminders:
- Use the `SVG` prefix for all `.c` and `.h` files that you wan to include in your graded A2
- **Do not** put the `main()` function into any of the files with the `SVG` prefix. I recommend you use a descriptive name for your `.c` file with main, e.g. `mainTester.c`
- A1 will use the same Makefile as A1

Evaluation

You must commit and push your code into the `A2` branch of your `CIS2750W22_Project` project on gitlab.socs.uoguelph.ca by the specified date. **No other submissions will be accepted.**

Remember to commit and push your assignment code on a regular basis. The SoCS GitLab server will be your backup, and the version control will be helpful - for example, you can roll back to an earlier commit if you mess up your code and cannot fix it.

If no code has been submitted to the `A2` branch of your `CIS2750W22_Project` repo prior to the end of the grace period, your assignment will be considered **not submitted** and will receive the grade of zero.

Late penalty calculation:
- If the last commit to the `A2` branch was prior to the assignment deadline, no late penalty will be applied
- Otherwise, the late penalty will be calculated based on the timestamp of the last commit to the `A2` branch during the grace period at 2% per hour, rounded up to the nearest hour.
- If there are no commits to the `A2` branch prior to the end of the grace period, your assignment will be considered **not submitted** and will receive the grade of zero.

Your code will be tested by an automated harness. The Makefile provided for A1 will be used to produce a shared library. A2 uses an identical Makefile, since all the filenames remain the same. However, your A2 will be compiled using the updated `SVGParser.h`, which is provided for you in the `A2` folder.

This library will then be tested on the SoCS Linux servers by a precompiled executable file containing the test harness, as well as another executable file with simple memory leak tests. Your library must implement the assignment API exactly as specified, or you will get run-time errors because the executable files will not find functions in the library that they expect.

Your code must compile, run, and have all of the specified functionality implemented. Any compiler errors will result in the automatic grade of **zero** for the assignment. Infinite loops may also result in a grade of **zero**.

Marks will be deducted for:
- Incorrect and missing functionality
- Deviations from the assignment requirements

- Run-time errors, including infinite loops
- Compiler warnings
- Memory leaks reported by valgrind
- Memory errors reported by valgrind
- Failure to follow submission instructions

As with A1, the test harness will primarily use your functions. For example, for writing tests, a valid file will be opened with your `createSVG`, then the resulting `SVG` will be saved with your writeSVG. The new file will then be opened using your `createValidSVG`, and the contents of the two `SVG` structs will be compared.

Similarly, validation will be done by using your functions to create an `SVG` struct from a valid file (the `SVG` will be expected to be valid), manually placing the `SVG` struct into a known invalid state (e.g. one of the Attributes would get a NULL name), and then verifying that your `validateSVG` correctly identifies this invalid state.

Therefore make sure all your A1 and A2 your functions work correctly, since the tests will fail if any of your functions do not work as required.

Grading scheme

Functions (graded using an automated test harness): 100%
- A large number of individual tests, worth 1-4% each
- The preliminary A2 harness contain a subset of these test cases worth 53%
- As stated in Assignment 2 description, all functions listed in `SVGParser.h` **must** be implemented - even if the implementation is incomplete or buggy. Failure to do so may crash the test harness, and result in an automatic grade of zero.

You will lose marks for run-time errors and incorrect functionality. Additional deductions include:
- Any compiler warnings: -15%
- Any memory leaks: -15%
- Any memory errors other than leaks, e.g. under-allocating memory, using uninitialized variables, etc.: -15%
- Incorrect directory structure: -10%
- Incorrect output filenames created by makefile: -5%
- Submitting to the wrong branch: -5%
- Any additional failures to follow submission instructions: up to -10%

- Any compiler errors: automatic grade of **zero** (0) on the assignment.
- Failure to run the shared library with the test harness: automatic grade of **zero** (0) on the assignment.

Your assignment will be graded on `linux.socs.uoguelph.ca`. The makefile that will be used to compile your assignment has been provided to you - it is the same Makefile that was used in A1.

Late submissions: see course outline for late submission policies.

This assignment is individual work and is subject to the University Academic Misconduct Policy. See course outline for details.