COMP9319 2021T2 Assignment 1: Entropy and Size Determination of Huffman & LZW Encoding

This is a warm-up assignment for you to practise the knowledge that you learned from the first 2 weeks and to get familiar with the programming tools (e.g., makefile). Your task in this assignment is to implement a C or C++ program that determines the average number of bits to encode a given file using static Huffman encoding and LZW as well as its entropy. The given file can be a text or binary file (e.g., images, videos or executable programs). When you determine the average number of bits, you assume that the size of the encoded file should be the minimum size (i.e., do not need to include any overheads such as the size of any statistic information) and should not consider adding any other encoding / further optimisation into it.

Your program should be called `csize`. Your program should accept: 1) an optional commandline argument `-s` and a number between 9 and 18 (inclusively) to specify the fixed width (in bits) of the codes for LZW (where the width determines the number of dictionary entries available). 2) a path to the given file to be encoded (e.g., ~/folder1/subfolder2/file2.jpg). If `-s` is omitted, the default width of 12 bits is assumed.

Your program `csize` will output 3 numbers, one each line, and nothing else (e.g., do not output any spaces). These 3 numbers are the Shannon's entropy, the average of bits per symbol for static Huffman, and the average of bits per symbol for LZW, respectively. Each number must be round to two decimal places (even if the number is an integer, output 00 for the two decimal places). For example:

```
%wagner> csize ~/Desktop/test.jpg
2.76
3.00
3.40
%wagner>
```

In the above example, 2.76 is the entropy; 3.00 and 3.40 are for Huffman and LZW, respectively.

Marks will be deducted if you do not follow the output format described above. Your solution will be compiled and run on a typical CSE Linux machine e.g. grieg. Your solution should read the input file as read-only (because you might not have write permission to the file) and should **not** write out any external files. Any solution that fails to compile on a CSE Linux machine, fails to read a read-only file, or writes out external files, will receive **zero** points for the entire assignment.

Compilation and Usage Example

Since your submission may need to be compiled with specific C or C++ compilation options, it's part of the assignment that you need to provide a valid makefile to compile your submitted code. If you have never used make before, you can easily google it (e.g., https://opensource.com/article/18/8/what-how-makefile) and find lots of tutorials and examples. You can also see us in the consultations if there are still problems with your makefile. Your submitted file will be compiled using the `make` command:

```
cs9319@grieg:~/a1$ make
```

and it will then successfully produce a program called `csize` for testing.

An example showing the usage and output of csize is as follows:

```
cs9319@grieg:~/a1$ ls -s test.txt
-rw------- 1 cs9319 cs9319 19 Jun 13 17:09 test.txt
cs9319@grieg:~/a1$ more test.txt
^WED^WE^WEE^WEB^WET
cs9319@grieg:~/a1$ cat test.txt
```

```
^WED^WE^WEE^WEB^WETcs9319@grieg:~/a1$ wc test.txt
 0  1 19 test.txt
cs9319@grieg:~/a1$ csize test.txt
2.21
2.26
7.58
cs9319@grieg:~/a1$ csize -s 10 test.txt
2.21
2.26
6.32
cs9319@grieg:~/a1$
```

Performance

Your solution will be marked based on memory and runtime performance. Your soluton will not be tested against any files that are larger than **1MB**. Any solution that takes more than **3 seconds** (sum of the sys and usr time using the time command, `/usr/bin/time`) to run on a given test file on grieg will be killed and you will receive zero points for that auto test.

Runtime memory is assumed to be always less than **16MB**. Runtime memory consumption will be measured by `valgrind massif` with the option `--pages-as-heap=yes`, i.e., all the memory used by your program will be measured. You may find `ms_print` useful to interpret the valgrind's output (https://www.valgrind.org/docs/manual/ms-manual.html). Any solution that violates this memory requirement will receive zero points for that test. Again, if you have questions using `valgrind` or `time` to measure your program, please see your tutor in the consultation sessions.

Documentation and Code Readability

Although the assignment is based on auto marking, your source code may be inspected and marks may be deducted if your code has poor readability and is very difficult to understand.

Remarks

1. It is part of the assignment that you are responsible for the correctness of your program output. For example, you can test your program by starting with some small files and manually verify the entropy and average bit calculations.
2. When your program opens the encoded file, make sure it is opened as read-only mode because you do not need and may not have the write permission to the given file.
3. If you haven't set your PATH to include your current folder, you may need to specify `./csize` instead of just `csize` to run your program. However, this will not affect how your program is tested in the testing account (as the current folder will be included in PATH).
4. Marks will be deducted for output of any extra text, other than the required, correct answer (i.e., the number of matches as shown in the examples above). This extra information includes (but not limited to) debugging messages, line numbers and so on.
5. **You may post to the WebCMS forum for clarification questions. For more specific questions, please see us in the consultation sessions.**

Submission

Deadline: Monday 28th June 17:00. The assignment is worth a total of 15 marks and is auto marked. Late submissions will have marks deducted from the maximum achievable mark at the rate of 1% of the total mark per hour that they are late (i.e., 24% per day), and no submissions will be accepted after 3 days late.

Use the give command below to submit the assignment:

```
give cs9319 a1 *.h *.c makefile
```

or:

```
give cs9319 a1 *.h *.cpp makefile
```

Note that the give command is available on any CSE linux machines (such as wagner) but not on grieg. You can check your submission status by:

```
9319 classrun -check a1
```

Plagiarism

The work you submit must be your own work. Submission of work partially or completely derived from any other person or jointly written with any other person is not permitted. The penalties for such an offence may include negative marks, automatic failure of the course and possibly other academic discipline. Assignment submissions will be examined both automatically and manually for such submissions.

Relevant scholarship authorities will be informed if students holding scholarships are involved in an incident of plagiarism or other misconduct.

Do not provide or show your assignment work to any other person - apart from the teaching staff of this subject. If you knowingly provide or show your assignment work to another person for any reason, and work derived from it is submitted you may be penalized, even if the work was submitted without your knowledge or consent. This may apply even if your work is submitted by a third party unknown to you.