CS 201, Fall 2022

Homework 5

DUE: January 9, Monday @23:59

Please check the submission rules towards the end of the document. Points will be deducted in case of a violation of these rules!

Description: In this assignment, you will write a C++ program that can **compress a text file and reconstruct (decompress) a compressed text**. The goal of the assignment is to gain experience in implementation of efficient algorithms and data structures in C++ for developing a practical tool.

1) The Usage of the Program

The program will be executed from the command prompt.

Assume that the executable is named as "lzw.exe", and there exists a text file, called "data.txt", in the same directory as the program executable.

Then, the following command should parse the text in "data.txt", generate a new file (or overwrite an existing one) called "data.lzw", which includes the corresponding compressed text.

```
lzw c data.txt data.lzw
```

Then, the following command should decompress "data.lzw" and put the reconstructed text in "data2.txt"

```
lzw d data.lzw data2.txt
```

The contents of the files "data.txt" and "data2.txt" should be the same, and the size of each should be larger than "data.lzw" (assuming that there exist a lot of repeated character sequences in the original text file).

Notice that we use the same executable ("lzw.exe") for two functions (compresss/decompress) with command line arguments (c/d). The name of the input and output files are also obtained as command line arguments.

The program should be **robust** against usage errors. If the provided arguments are wrong/missing, or if the input file cannot be found, the program should print out an informative error message.

Short tutorials on command line arguments and file I/O in C++ can be found in the following links (many more resources can be found on the Internet):

- *Parsing command line arguments:*

http://www.site.uottawa.ca/~lucia/courses/2131-05/labs/Lab3/CommandLineArguments.html

- *Reading from / writing to files in C++:*

http://www.cs.hmc.edu/~geoff/classes/hmc.cs070.200109/notes/io.html

http://courses.cs.vt.edu/~cs2604/fall02/binio.html

2) The Implementation

LZW method will be used for the implementation of the text compression algorithm. In this method, character sequences in the original text are replaced by codes. If there are frequently repeating and long character sequences, replacing them with (shorter) codes reduces the size of the file.

The mapping between character strings and their codes is stored in a **hash.** To make the implementation easier, you will need to set hash size to 65536 (In practice, shorter sizes that require usage of bitwise operations are used). In this case, the length of each code is 2 bytes = 16 bits (2^16 = 65536).

Initially, the dictionary contains 256 characters ("standard" ASCII character set). During compression/decompression, the file is parsed and the dictionary is extended according to the observed character sequences/codes.

Although the length of each code (16 bits) is larger than the size of a character (8 bits), if a character sequence (e.g., a frequently used word) occurs again, 16-bit code is used instead of the whole sequence. This makes the output file smaller.

A comprehensive description of the **LZW method** can be found in the following link:

http://www.cs.duke.edu/csed/curious/compression/lzw.html

In this assignment, you cannot use the existing hash table implementations in C++ libraries such as std::map, std::unordered_map, etc. You need to implement your own hash table class from scratch.

WARNING: You can benefit from sources on the Internet to learn more about the LZW method, implementation variations and practical issues (e.g., how to read command line arguments in C++). However, the submitted implementation must be of your own.

3) Submission

You will submit this homework via the LMS system. You should follow the file-naming conventions and guidelines below.

- You should submit your source files as a **ZIP** archive file (**NOT** RAR or other formats). The name of the file should be in format "**<USER-ID>_hw5.zip**". For example, if your username is vy1043, then the name of the submitted file should be "vy1043_hw5.zip". Pay attention that all the letters are in lower-case. ZIP archive is supposed to contain **just the source files**, no folders are allowed by any means.

- The contents of the ZIP file should be **lzw.cpp** (includes the *main* function) and optionally any other class definition and implementation files depending on the data structures you utilize.

- Late submissions and C++ files that do not compile are **not** accepted.

- You can resubmit your homework (until the deadline) if you need to.

- Make sure that your program does **not** include commands specific to a development environment, e.g., `system("pause")` or `#pragma once` in Visual Studio.