CSC 485B/578B - SUMMER 2023
DATA COMPRESSION
ASSIGNMENT 3
UNIVERSITY OF VICTORIA

Due: Sunday, July 23rd, 2023 at 11:59pm. **Late assignments will not be accepted.**

This assignment will be submitted electronically through Brightspace (as described in 'Submission Instructions' below). All code submissions must be your own work. Sending code to other students, receiving code from other students, or exchanging assignment code with anyone else via any medium is plagiarism and is prohibited.

1 Overview

This assignment covers lossy encoding and quantization of still images with the discrete cosine transform (DCT), as well as the mechanics of encoding a DCT result into a bitstream. You will likely want to reuse some or all of your code from this assignment in your Assignment 4 submission, since Assignment 4 will cover lossy video compression.

For this assignment, we will work with the uncompressed `.bmp` (bitmap) format, which stores each pixel as an RGB triple with 8 bits per sample. You will write a compressor that takes a `.bmp` image and produces a compressed representation via the following pipeline.

- Transforms the image to the YCbCr colour system.
- Downscales the Cb and Cr colour planes by a factor of 2 (both horizontally and vertically).
- Applies a block-based DCT to all three colour planes, followed by quantization. The compressor will support three quality modes: '`low`', '`medium`' and '`high`'.
- Stores the quantized DCT coefficients (and any other metadata needed, such as details on the quality setting) into an output file.

You will also write a decompressor which takes a compressed image file (produced by your compressor) and reconstructs the input `.bmp` image. The decompressor does not take any parameters besides the compressed data file (so any information needed by the decompressor about the quality setting must be encoded into the compressed bitstream).

Once your programs are complete and correct, you can compress and decompress files with the following commands. Note that for this assignment, due to the formats used, the compressor and decompressor work with files on disk instead of standard input/output.

```
$ ./uvg_compress low input_image.bmp compressed_image.uvg
$ ./uvg_decompress compressed_image.uvg decompressed_image.bmp
```

(After these commands, if your compressor worked correctly, the `decompressed_image.bmp` file will be a valid image, hopefully very similar to the original `input_image.bmp`.

Starter packages have been provided (in both C and C++) which handle reading the input bitmap image, transforming the pixels into YCbCr and separating/downscaling the colour planes. Therefore, your objective is to implement the DCT, quantization and encoding steps.

It is your decision how to implement the three quality settings '`low`', '`medium`' and '`high`', except that '`low`' should have lower visual quality and smaller file size than '`medium`', which in turn should have lower quality and smaller file size than '`high`'.

This assignment has been designed as an incremental step toward Assignment 4. Therefore, some aspects, particularly the compression ratio achieved and the quality level of the decompressed result compared to the original input, will be less important than the successful implementation of a lossy compressor with variable quality settings. On the other hand, the marking for Assignment 4 will take compression ratio and image quality into account.

2 Submission Format

We will use automation to validate your submission before human inspection. To ease this process, we expect all submissions to be in a standard format.

Your submission must be a `.tar.gz` archive called `uvg.tar.gz` containing a directory called `uvg`. It must be possible to extract and run your submission using the **exact** following commands (starting in the directory containing the `uvg.tar.gz` archive).

```
$ tar -zxf uvg.tar.gz
$ cd uvg
$ make
$ ./uvg_compress medium my_input.bmp compressed.uvg
```

(where the input file is some local file not contained in your archive). To make the process easier, some starter archives have been provided, all of which contain a basic example of the expected Makefile.

To create an archive to submit, run the following command (in the parent directory of your `uvg` directory):

```
$ tar -zcvf uvg.tar.gz uvg/
```

3 Evaluation

Your submission must be an archive named '`uvg.tar.gz`' containing a directory `uvg` (which will contain a Makefile and your code, as described above). Your code must compile and run correctly on `csc485b.csc.uvic.ca`. If your code does not compile as submitted, you will receive a mark of zero.

This assignment is worth 5% of your final grade. All submissions will be marked out of 10.

Marks	Component
5	The compressor and decompressor use a block-based, quantized DCT to represent the image data.
3	The encoding scheme used for DCT coefficients achieves compression. For full marks, the compressed representation of an image must be smaller than its PNG representation (since PNG is a lossless format).
2	The three quality settings 'low', 'medium' and 'high' are supported by the compressor, and the file size is appreciably different between the three settings (with 'low' quality yielding the smallest file size and 'high' quality yielding the largest file size).

Submission Instructions

All submissions for this assignment will be accepted electronically. You are permitted to delete and resubmit your assignment as many times as you want before the due date, but no submissions will be accepted after the due date has passed.

Ensure that each file you submit contains a comment with your name and student number, and that the files for each question are named correctly (as described in the question). If you do not name your files correctly, or if you do not submit them electronically, it will not be possible to mark your submission and you will receive a mark of zero.

If you have problems with the submission process, send an email to the instructor **before** the due date.

It is your responsibility to ensure that we receive the correct file. To verify that you submitted the correct file, you can download your submission from Brightspace after submitting and test that it works correctly. **We will assume that all submissions have been tested this way**, and therefore that there is no reasonable chance that an incorrect file was submitted accidentally, so no exceptions will be made to the due date as a result of apparently incorrect versions being submitted.