

Assignment “COMPRESS/DECOMPRESS”

EXERCISE :

Write two programs: A BMP compressing and a decompressing program. Use Huffman Code.

COMPRESSING BMP:

Syntax:

`./compress file.bmp [QUALITY]`

File.bmp input bitmap file in BMP file format, 24 Bit colors.

[QUALITY] A number from 1 to 10, with 10 being lossless and 1 being maximal compression under losses.

The output file should be the name of the input file name with an extension of your choosing:

e.g.: File.bmp -> File.zzz

or

File.bmp -> File.abc

DECOMPRESSING:

Syntax:

`./decompress file.zzz output.bmp`

File.zzz The compressed file with your file format.

output.bmp output file in 24 Bit bitmap format.

ENVIRONMENT:

You can develop this assignment in any OS of your choice, but generate binary files for submission in UNIX, see “what to submit”.

ALGORITHM:

The program should compress a bitmap file and store the compressed data in a file format of your choice. This file format should contain the compressed data and also all necessary header information.

1. For the compression, load the whole BMP file.
2. Make an integer division of the color bytes for loss compression. E.g.
If [QUALITY] is 10, integer divide by 1.
If [QUALITY] is 5, integer divide by 10.
And so forth.
Choose yourself how much losses you want for the [QUALITY] range.
3. Make a table of all pixel color values for red, green and blue separately. This table should contain the BUTE value and how often it appears.
4. Sort this table to least appearance first. Use any sorting available (online source) but I suggest downloading a quicksort algo in C or better use the sort function in C++.
5. Build the Hoffman tree based on the tables. You need a Hoffman tree for each color. Use a structure/class for linking the nodes. Don't forget to clean up.
6. The data part of the compressed file will consist of the bit pattern for each pixel from the Hoffman tree. Write this data. How you arrange that (red first, mixed, etc) is totally up to you.
E.g. (in case only one color): The first six pixels have bit patterns:
10010,10,0101110,10,10010,11110
Then you need to store this pattern DENSE in the array like this:
[10010 10 0][101110 10] [10010 111][10] and so on. Spaces just indicate the separate pixels; the brackets represent each byte. That means one byte can eventually hold more than one pixel value and even a fraction of it. You need to use bit operators!
7. Save the file with necessary header information such as width/height, compression factor, hoffman table, etc
8. Reverse the process in the DECOMPRESS program.

How to start

It is probably a good idea to follow the order above but leave quality losses out for now.
It is also recommended firstly focus on one color (e.g. Red).
It is VERY MUCH encouraged to start on this assignment immediately.

What to submit?

A ZIP file with your name (first and last) containing the source file for both programs, both programs as binary executable file in UNIX and some pictures we can try.