University of Toronto
CSC228S, Winter 2001

Assignment 1 Advice

Advice on tackling the program

This program will be much easier to tackle if you break it down into small, incremental steps, and if you test and debug along the way. And if you do so, you will be sure to have something coherent to hand in even if you haven't completed the entire assignment. Here is a suggested way to tackle the assignment:

1. Decide on what classes you'll need for this program. Design the interface for each one, including function headers and detailed external comments.

   We suggest you have one class which handles the translation between characters and codes. This class could hold the table relating the character/ code pairs. We suggest that in this class you use a string comprised of the characters '0' and '1' to represent the codes. Then you will need another class with methods to take these strings and pack them into a buffer. A third class will handle unpacking the buffer during uncompression. Pay particular attention in your design to how you will handle the end of the file on uncompression.

2. Write an algorithm for the main program, at a high level. This may lead you to redesign the classes somewhat.

3. Decide on the main data structure for each class, and define the relevant instance variables.

4. Write the part of the main program that will deal with the command-line arguments. Call the right functions (according to what the command line tells you), but give those functions "stub" bodies, e.g., simply print "Inside function blah." Get this program running properly.

5. Write code (wherever it belongs in your program) to read the encoding from a file and store it in memory (in whatever class is appropriate). Add code for looking up the encoding of a character. Test this code.

6. Write code to read a file to be compressed, one character at a time until end-of-file is reached. Just print out each character as you read it. Now print this output to the appropriate file instead. (You're not doing any compression yet.) Test this.

7. Now modify the program to print the encoding of the character instead of the character itself, but print the encoding as a string for now. You are not doing compression yet; in fact the output file should be much larger than the input file, since each bit of the encoding will take `sizeofchar()` bytes in the output file. Test this.

8. Now write your buffer class and modify your program to write bits (rather than strings) to the file. Test this. You will need to use `od` to look at your output file.

9. Now add the code for doing uncompression.

This is just one suggestion for how to proceed. The important thing is that you (a) don't rush into coding before you have a good design in place and (b) have some strategy for breaking down the debugging so that you don't get overwhelmed.