Midterm
CPE 357
Systems Programming

CAL POLY
SAN LUIS OBISPO

Midterm

(100 points + 10 EC)

Remember bitmap files?
Remember Huffman Encoding?
You guessed it!

It's time to compress a bitmap file using Huffman Encoding (by the way: lossless JPEG compression uses the Huffman algorithm in its pure form.)

Your program should compress a bitmap file and store the compressed file as **.cbmp**

For simplicity reasons, turn your image gray and compress a gray image.

How I will run your code:

mycompress [flags] [filename]

flags can be chosen to be
-g … gray
-c … color (+10 EC)

Please also submit a **readme.txt** file with answers the following questions:

- How much smaller (in %) is the compressed image?
- How long does it take to compress and decompress it? Any ideas how to parallelize it? Where? What is the time you win?
  This actually indicates to use fork() in your code and measure the time!

All good until here?
Go for colored images!

Examples how I will run your code:

The filename ending in .bmp is indicating that it should be compressed, the filename ending in .cbmp indicates, to decompress it.

```
./mycompress -g jar.bmp
```
This should create a file named jar.cbmp which will be the compressed version of jar.bmp

```
./mycompress -g jar.cbmp
```
This should create a file named jar.bmp which will be the gray version of the original jar.bmp

If you go for the EC, the commands will be
```
./mycompress -c jar.bmp
./mycompress -c jar.cbmp
```

I will not invalid input or invalid input arguments.

Hints:

- Go for smaller images first! (and test your Huffman tree!)
- Use the function you wrote in the class activity in week 3
- It's probably easier to do the compression and decompression on char level first before going to bits (get "1"'s and "0"'s into the cbmp file first)

Submission:

You will get 10% EC if you turn in your code by Thursday midnight!

```
handin ihumer 357-midterm-early *c *h readme.txt
```

OPENS ON THURSDAY MIDNIGHT:

```
handin ihumer 357-midterm *c *h readme.txt
```