Rishab Verma

0026681707 [verma24@purdue.edu](mailto:verma24@purdue.edu)

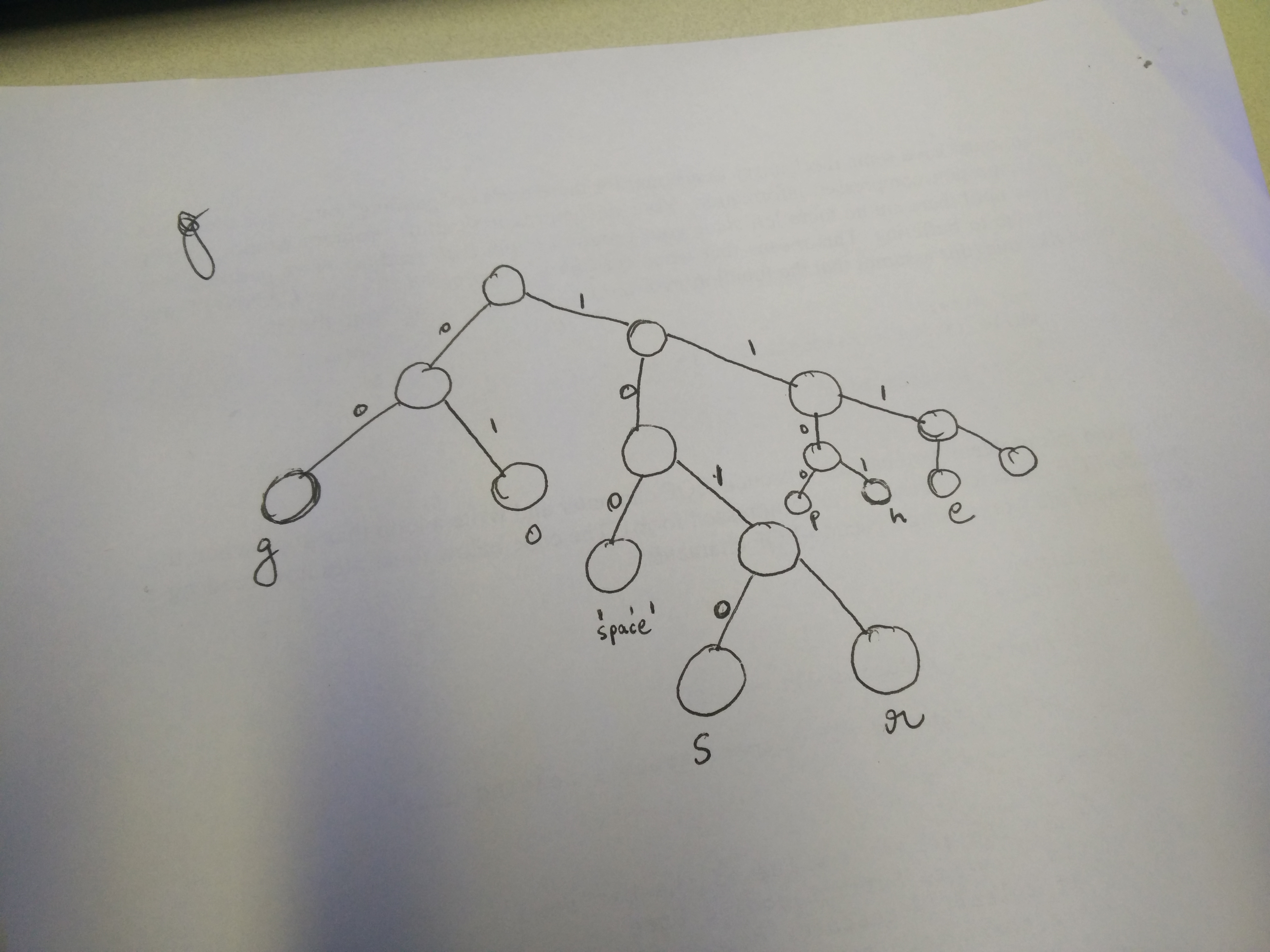
**Project 2-ECE368**

For Huff program , I started out by getting the frequencies of the characters in an array and then converted it to a linked list. Once the linked list is made , I sorted them using insertion sort and converted them into single list nodes. After that, once I received the sorted nodes , I merged them with the similar frequencies and generated a binary tree In the same process. Once I was done with my tree, I made an array called 3d array where I stored the height of the tree as cols and the #of leafs as rows .

Since I used this method my Forrest generated was a little different than that of a normal tree made in project2 sheet.

This is what my tree looks like for go go gophers, which is the example provided in the p2 sheet.

You can even expand it and see it .



The following table gives the time and the size of the compressed files.

HUFF

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| FILE size |  | time(s) | Comp\_size | Compress ratio |  |  |
| 8 bytes |  | 0.0000 | 14 bytes | 8/14 =0.571 |  |  |
| 6mb |  | 0.76 | 4.2 MB | 6/4.2=1.42 |  |  |
| 9 mb |  | 1.23 | 6.1MB | 9/6.1=1.47 |  |  |
|  |  |  |  |  |  |  |

For the unhuff program, I used my old code from Ece 264, I did not have to make a lot of changes to it except for the header reading part. The header function in my huff part, printed the binary code for the header first on to the outpufile.

For this I made a readbits function which I used as a helper for the unhuff function. The readbits function was responsible for taking care of that.

Th ratio for my compression has been provided in the table. The time also seems fine , however, there are some print statements in the unhuff function which take some time to print when unhuffing large files.

My compression ratio get better as we got to bigger file . Also the time increases.

In my code, I have a lot of garbage code which I commented out . I tried to give an explanation of the steps I took in the code by comments.

Lastly, I got some structure code from the website <http://cslibrary.stanford.edu/110/BinaryTrees.html>.

This source was where I got some basic structure definition of my .h file for trees and list nodes .

Also I have created 4 files, unfuff.c unhuff.h and huff.c and huff.h repectively .