Jane Jeong

15-112: Fundamentals of Programming and Computer Science

TP2 Updated Project Proposal

4/17/2018

Project Description: The name of my term project is DIY Data Compression. The goal of this project is to visualize a data (text) compression method to teach the user how the data compression algorithm works. The method that I will be using for my project is Huffman’s compression. The user will have a text box in which they can type in while a Huffman tree is drawn in response to the addition or deletion of characters in the textbox.

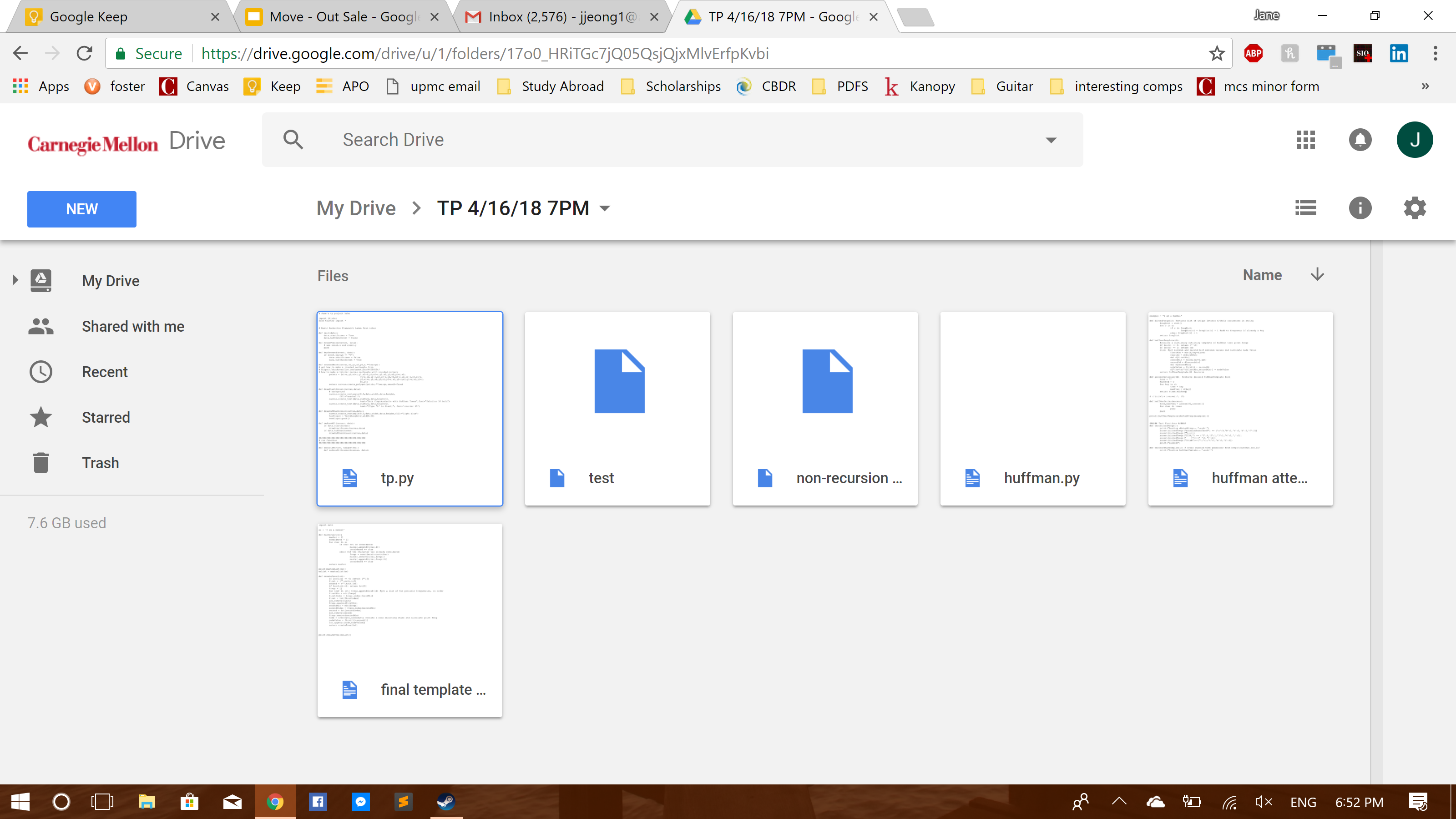
Competitive Analysis: There are other programs online that I’ve found that also visualize the Huffman tree using customized text, so in that way my project is like such programs. However, mine project will be different in that it also has an educational component. My program will walk the user through the algorithm prior to allowing the user to utilize the visualization tool.

Structural Plan: The finalized project will be organized into different files. There will be a main file that uses tkinter and provides the information needed for the UI and the visualization of the Huffman tree (which will use the algorithm data from a different file). There also may be a couple to a few image files. If needed, the main UI file may be separated from the visualization data; however, I believe this is unlikely.

Algorithmic Plan: I believe the trickiest part of the project is in the way that I will be visualizing the Huffman compression algorithm. The visualization will probably have to access different parts of the main algorithm file, which provides the structure of the Huffman tree. My algorithm so far gets the frequencies of every unique character in a list with a tuple for every unique character. It also creates a large nested tuple of the “order” or template of the tree. The next part of the algorithm will be to go access the elements of the tuple in the order of the nesting to begin assigning bits to each character. This will take a little time to understand how to convert the Huffman code to bits (the bit conversion may only be done if there is enough time to figure out how to allow the user to save the text that they have just written into the textbox in a compressed format).

Timeline Plan: I intend to complete most of the UI that is not related to the algorithm visualization by TP1. I also hope to complete the algorithm by at least halfway between TP1 and TP2. I will leave the rest of the time for configuring the visualization of the algorithm, and if there is enough time, I will attempt a different text compression algorithm, or, as mentioned above, I will attempt to allow the user to save the text they have just written into the textbox.

Version Control Plan: I plan to backup my code into Google Drive into folder with a date label every time I make significant progress.



Module List: n/a, will only be using tkinter

TP2 Update:

I no longer have a text box, instead the user can type the characters in and the program will still register each character. Another change that I have made is that the modification of the user input can only be made in the screen prior to the screen that produces the tree, instead of live changes.

TP3 Update:

The program has three main features: a slideshow for learning the Huffman algorithm, a visualization that implements the Huffman algorithm based on live user input, and a utility option that also implements the Huffman algorithm in a useful way. The utility option is completely new since the last TP2. This option allows the user to submit a text file and obtain a Huffman reading of the text contained in the text file. There is then a save option, through which the user is allowed access to another two features – the first feature is to create a new text file with the encoding information, and the second is to create a new text file translated into the Huffman rendition of the original text. The save option also shows the user how much data they saved through circles representing the size of the files before and after the compression.