Text File Compression using Huffman Coding

DAA PBL – Group – 5

Content overview

- Introduction
- Libraries Used
- Algorithm
- File compression algorithm
- File decompression algorithm
- Result
- Analysis
- Applications
- References

Team Members

GANESH BHAVAR

Prn: 22010331
Roll No: 371018

HARSHIT SINGH

Prn: 22011072
Roll No: 371021

ISHWAR ZATKE

Prn: 22010600
Roll No: 371022

VIDYAPATI SHELKE

Prn: 22120037
Roll No: 371069

Introduction

Huffman Coding

- Huffman coding is a lossless way to compress and encode text based on the frequency of the characters in the text.
- Huffman code is a special type of optimal prefix code that is often used for lossless data compression.
- The idea is to assign variable-length codes to input characters and the most frequent character gets the smallest code and the least frequent character gets the largest code.

Libraries Used

OS

This module provides the facility to establish the interaction between the user and the operating system. It offers many useful OS functions that are used to perform OS-based tasks.

HEAPQ

This module provides an implementation of the heap queue algorithm, also known as the priority queue algorithm.

TKINTER

Tkinter is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter is the fastest and easiest way to create the GUI applications

Algorithm

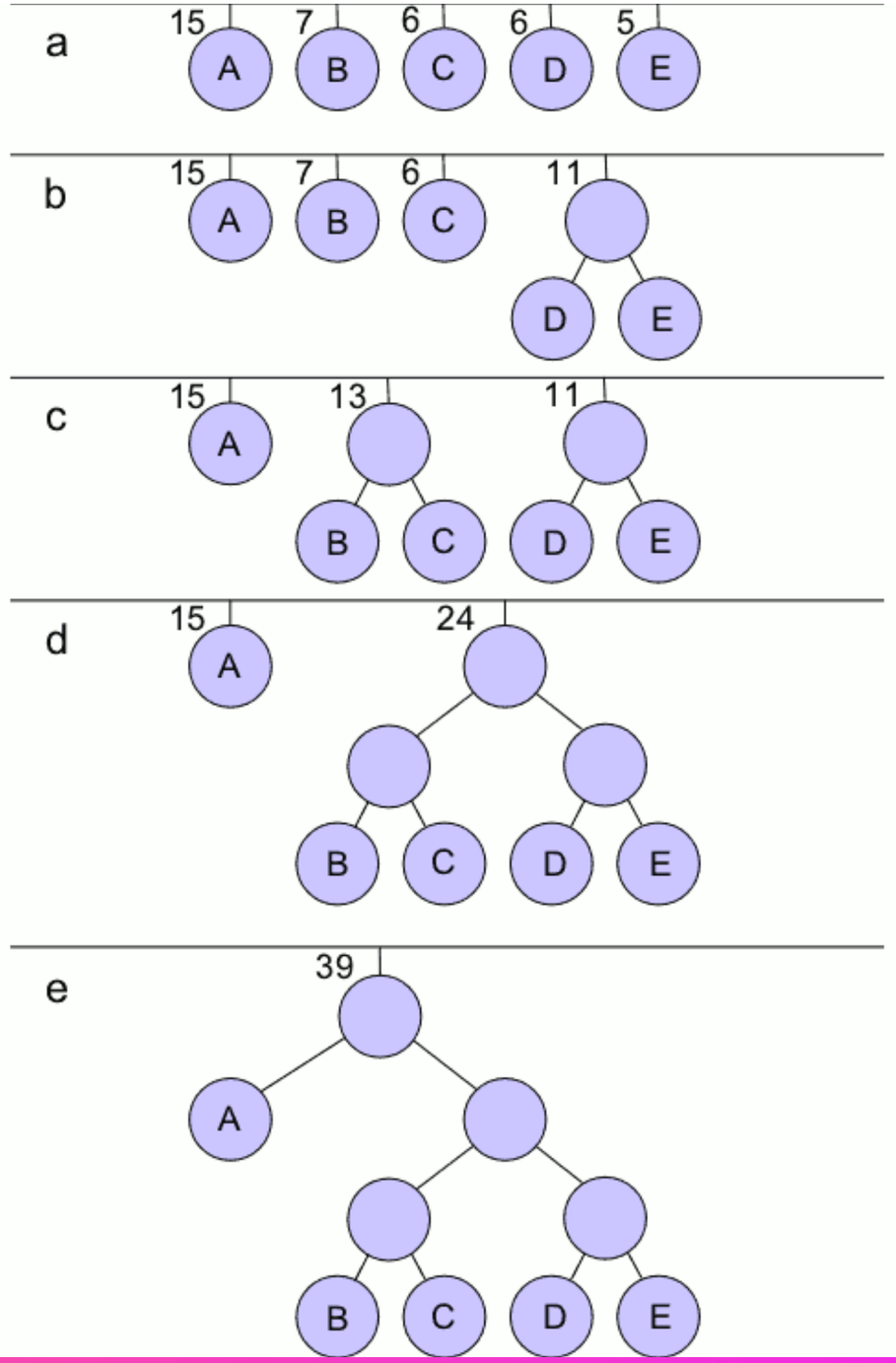
- Build a min heap that contains N nodes where each node represents the root of a tree with a single node. All nodes have a weight equal to the weight of the character in the node.

- Characters that occur most frequently have the highest weights. Characters that occur least frequently have the smallest weights. Repeat this step until there is only one tree.

- Choose two trees with the smallest weights, and call these trees T1 and T2. Create a new tree whose root has a weight equal to the sum of the weights T1 + T2, whose left subtree is T1, and whose right subtree is T2.

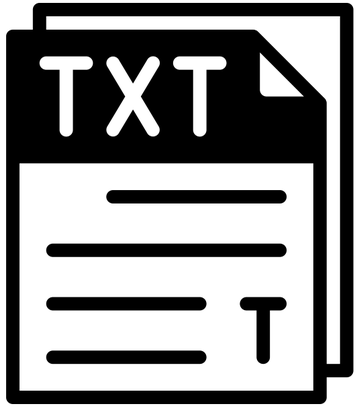
Algorithm

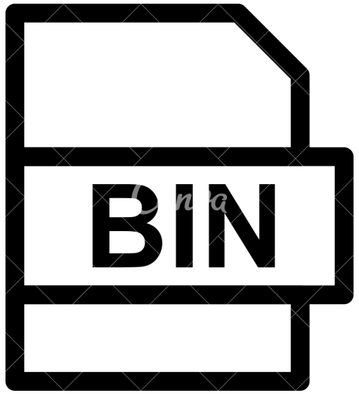
- A single tree left after the previous step is an optimal encoding tree.



File Compression

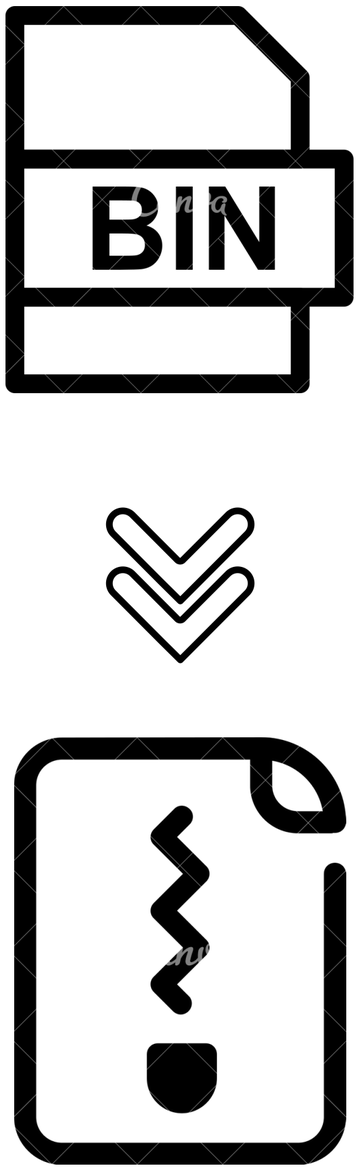
1. Build frequency dictionary
2. Build priority queue (used MinHeap)
3. Build Huffman Tree by selecting 2 min nodes and merging them
4. Assign codes to characters (by traversing the tree from root)
5. Encode the input text (by replacing a character with its code)
6. If the overall length of the final encoded bit streams is not multiple of 8, add some padding to the text
7. Store this padding information (in 8 bits) at the start of the overall encoded bit stream
8. Write the result to an output binary file





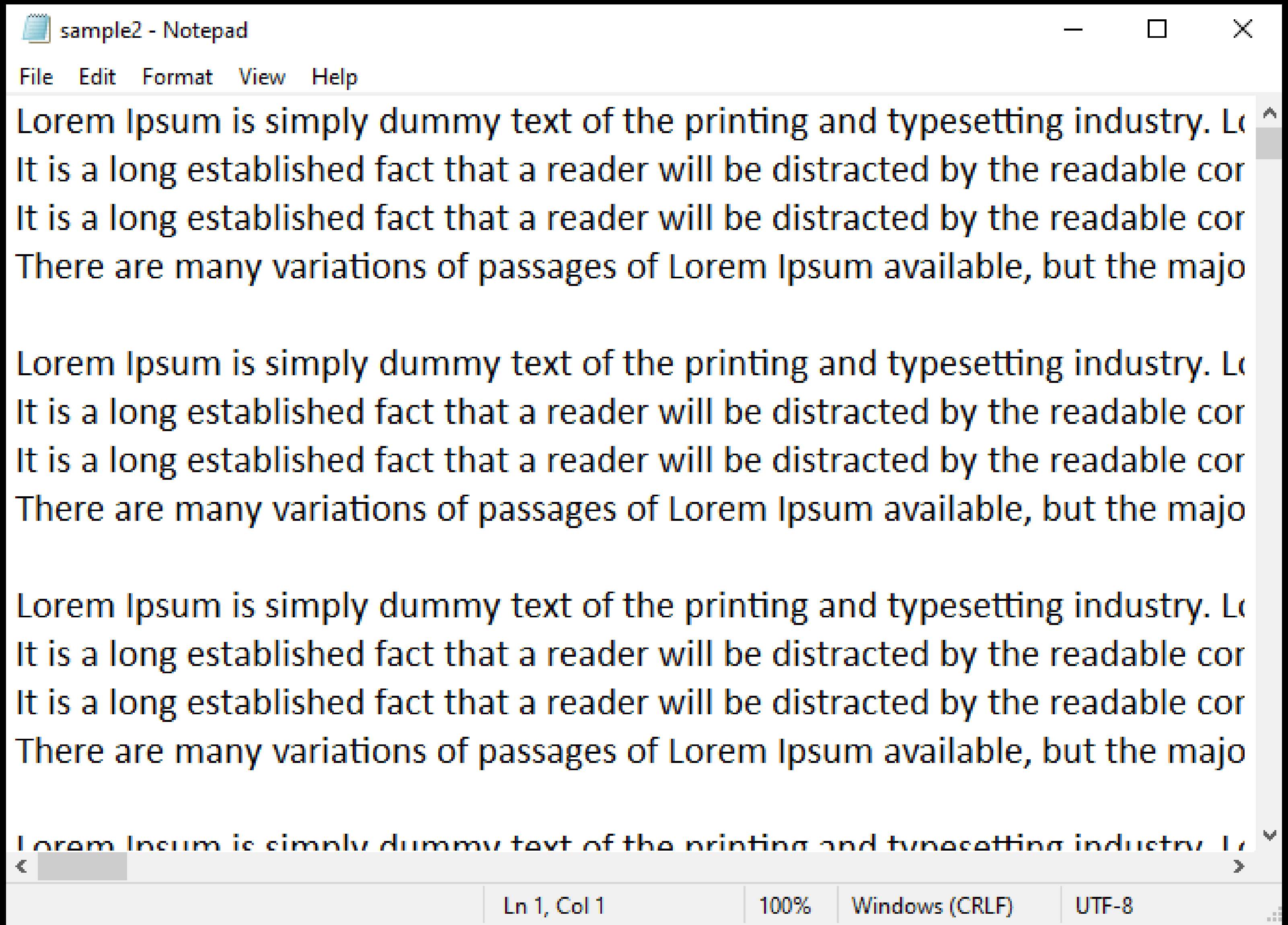
File Decompression

1. Read binary file
2. Read padding information. Remove the padded bits
3. Decode the bits - read the bits and replace the valid Huffman Code bits with the character values (Need to store the Huffman codes mapping while compression)
4. Save the decoded data into the output file (Gets original data back)

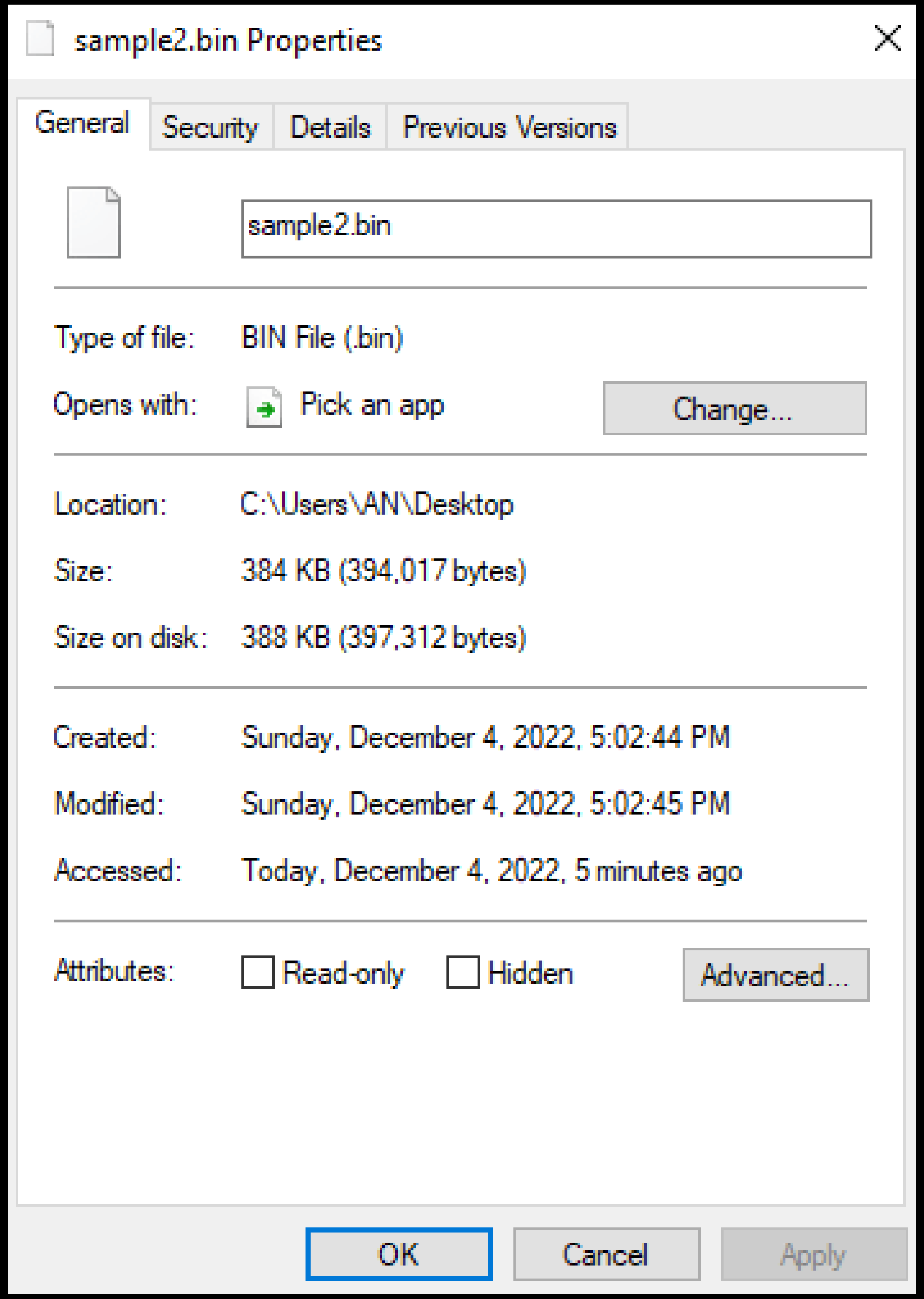


Result

.txt file before compression



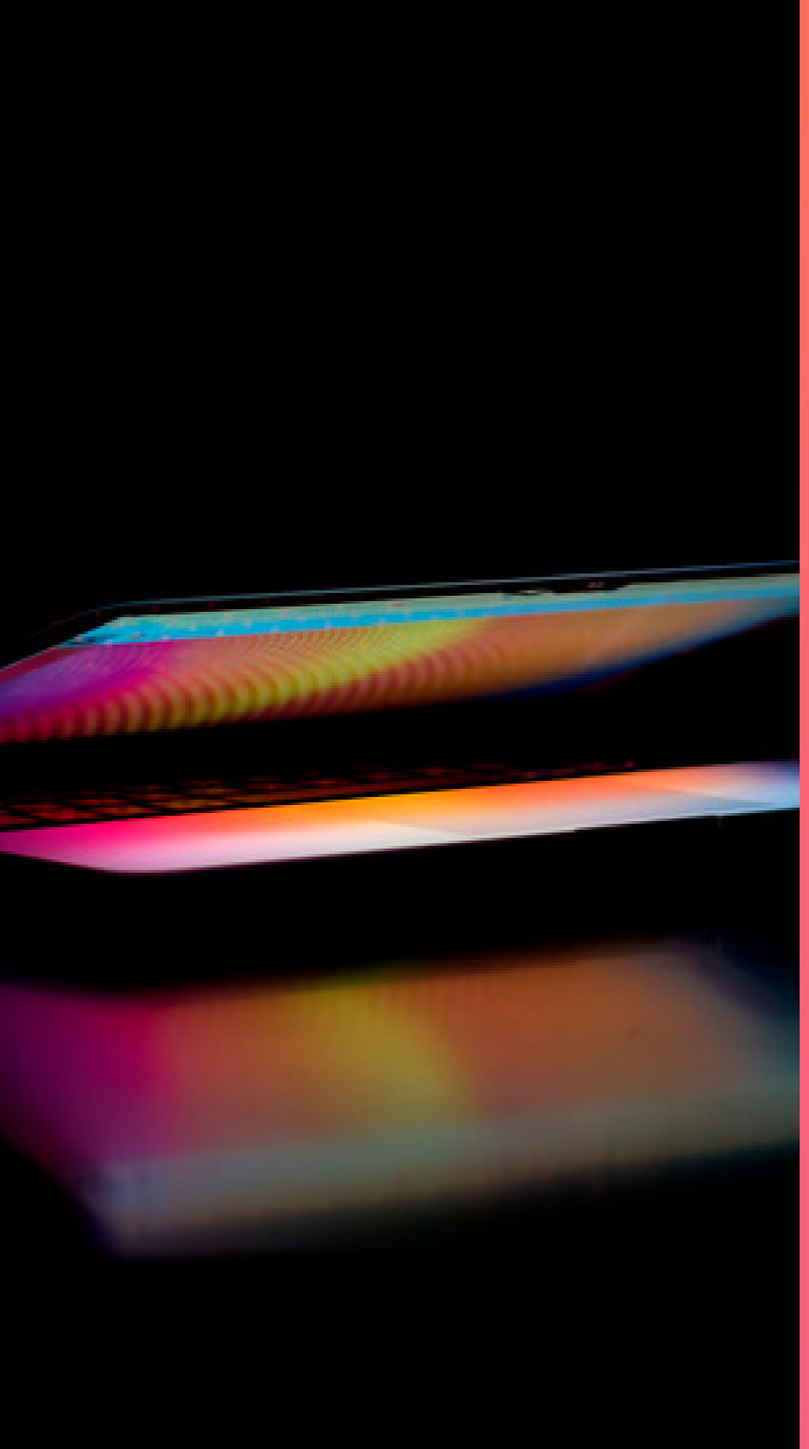
.bin file generated after compression



Result

```
TERMINAL   PROBLEMS 1   OUTPUT   DEBUG CONSOLE
[Running] python -u "c:\Users\AN\Desktop\Study\DAA PBL\huffman-coding\useHuffman.py"
Original size:716664
Compressed
compressed size:394017
Compressed file path: C:/Users/AN/Desktop/sample2.bin
Decompressed
Decompressed size:716664
Decompressed file path: C:/Users/AN/Desktop/sample2_decompressed.txt

[Done] exited with code=0 in 11.681 seconds
```



Analysis

The performance of the Huffman Code depends upon 2 factors

1. Average Length: It is defined as the average number of bits required to represent a character in the string.
2. Efficiency: Efficiency is a measure of the number of bits wasted.

Time complexity: O(N log N)
where n is the number of unique characters.

Applications of Huffman Coding

- They are used for transmitting fax and text.
- They are used by conventional compression formats like PKZIP, GZIP, etc.
- Multimedia codecs like JPEG, PNG, and MP3 use Huffman encoding(to be more precise the prefix codes).

References

HTTPS://WWW.GEEKSFORGEEKS.ORG/HUFFMAN-CODING-GREEDY-ALGO-3/

HTTP://EN.WIKIPEDIA.ORG/WIKI/HUFFMAN_CODING

HTTPS://WWW.SECTION.IO/ENGINEERING-EDUCATION/HUFFMAN-CODING-PYTHON/

THANK YOU!