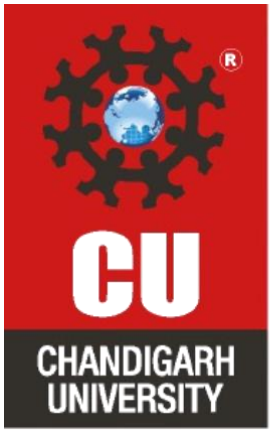

CHANDIGARH UNIVERSITY

Discover. Learn. Empower.

UNIVERSITY INSTITUTE OF COMPUTING

GROUP 11

STUDENT NAME: Kritika Chauhan(24BCU10005)

Kritika Sharma(24BCU10008)

Gungun Vijayvargiya(24BCU10024)

BRANCH: BACHELOR OF COMPUTER APLLICATION (UI/UX)

SEMESTER: 2nd

SECTION: 24BCU-1A

SUBJECT: Data Structure Lab

SUBJECT CODE: 24CAP-152

TOPIC: Building a File Zipper

SUBMITTED TO-

Ms. SHILPI GARG

Project: Building a File Zipper

1. Abstract

The File Zipper is a tool that compresses and decompresses files and folders to reduce file size, making storage and transmission more efficient. It uses lossless compression algorithms like Huffman Encoding and LZ77, and efficient data structures such as arrays, stacks, hash tables, and binary trees.

---

2. Introduction

With the increasing size of digital data, managing storage and ensuring faster file transfers has become important. The File Zipper compresses files without losing data, reducing file size and improving speed. It provides a simple interface and is built using well-known algorithms and data structures for performance and accuracy.

---

3. Data Structures Used

- **Arrays:**
  Used to hold file content and compressed data in memory. They allow fast, sequential access and make reading/writing data efficient during compression and decompression.
- **Stacks:**
  Mainly used during Huffman Tree construction. They temporarily store nodes when combining characters based on frequency, making tree generation easier.

- **Hash Tables:**
  Useful in LZ77 to quickly find repeated sequences. By mapping strings to positions, they allow faster pattern matching and efficient replacement during compression.
- **Binary Trees:**
  Central to Huffman Encoding. Each character becomes a node in a binary tree, where frequently used characters are closer to the root. This helps assign shorter binary codes to frequent characters.

---

4. Algorithms Implemented

Huffman Encoding

Lossless compression that assigns shorter codes to more frequent characters.

Steps:

1. **Count how often each character appears.**
2. **Build a min-heap or priority queue.**
3. **Create a binary tree (Huffman Tree) from frequencies.**
4. **Assign binary codes to characters.**
5. **Encode the file using these codes.**

LZ77 Compression

Replaces repeated data with references to earlier data in a sliding window.

Steps:

1. **Use a sliding window over input data.**

2. **Look for matching substrings.**
3. **Replace matches with (distance, length) pairs.**
4. **Continue until the whole file is encoded.**

---

5. Applications

- **File Storage:**
  Large files can be compressed to save space on devices or cloud storage platforms.
- **Data Transfer:**
  Smaller files upload/download faster, saving time and internet bandwidth.
- **Backups:**
  Zipped files take less space, making backup processes more efficient and quicker.
- **Archiving:**
  Multiple files and folders can be combined into a single compressed file, making it easy to store or share.
- **Software Distribution:**
  Developers package software into compressed files, reducing the size of downloads and improving install times.

---

6. Conclusion

The File Zipper project provides a simple yet powerful tool to compress files using efficient algorithms and data structures. With Huffman Encoding and LZ77, it achieves high

compression ratios. The use of arrays, stacks, hash tables, and binary trees ensures fast and accurate processing. Future upgrades can include support for more compression methods and an enhanced interface for better user experience.

Github Link- https://github.com/Gungun766/Minor-Project.git

- THANK YOU -