<u>Digital Empowerment Pakistan</u>

Name: Yashfa Mir

C++ Programming Internship

Task 3

Implementing a Simple File Compression Algorithm

Code:

```
#include <iostream>

#include <fstream>

#include <string>

#include <cctype>

 #include <sstream>

 std::string to_string(int num)
{     std::stringstream ss;
```

```
 ss << num;

  return ss.str();

}

int stoi(const std::string &str)
{     std::stringstream ss(str);

int num;

ss >> num;

return num;

} void readFile(const std::string &filename,

 std::string &content)

 {     std::ifstream file(filename.c_str());

 if (file.is_open()) {

std::string line;

 while (std::getline(file, line))

{  content += line;

 if (!file.eof())

 {content += "\n";

}}  file.close();     }

 else { std::cerr << "Unable to open file " << filename << std::endl;     } }
```

```
void writeFile(const std::string &filename, const std::string
&content) { std::ofstream file(filename.c_str());

if (file.is_open())

{file << content;

file.close();  }

 else {std::cerr << "Unable to open file " << filename << std::endl;

} } std::string compress(const std::string &data)

{std::string compressed;

int n = data.size();

for (int i = 0; i < n; i++) { int count = 1;         while (i < n - 1
&& data[i] == data[i + 1]) {             count++;

i++; }

compressed += data[i];

compressed += to_string(count);

} return compressed;

} std::string decompress(const std::string &data)
{  std::string decompressed;
int n = data.size();

 for (int i = 0; i < n; i++)

 { char c = data[i];
 i++;

 std::string countStr;

while (i < n && std::isdigit(data[i])) {            countStr
+= data[i];

 i++;         }

 i--;

 int count = stoi(countStr);
decompressed.append(count, c);

}  return decompressed;

} int main()

 { std::string content;

readFile("input.txt", content);

if (content.empty()) {

 std::cerr << "The file is empty." << std::endl;         return 1;

}

std::string compressedData = compress(content);
writeFile("compressed.txt", compressedData);

std::string decompressedData = decompress(compressedData);
writeFile("decompressed.txt", decompressedData);

std::cout << "Compression and decompression complete." << std::endl;
```

```
  return 0;

}
```

Documentation for File Compression and Decompression Program

Overview

This C++ program implements a basic file compression and decompression tool using RunLength Encoding (RLE). The program reads a text file, compresses its contents, writes the compressed data to a new file, then decompresses the compressed data and writes it back to another file.

Functions

1. std::string to_string(int num)

- **Purpose:** Converts an integer to a string.
- **Parameters:**
  - num: The integer to be converted.
- **Returns:** A std::string representing the integer.
- **Description:** Uses a std::stringstream to convert the integer to a string.

2. int stoi(const std::string &str)

- **Purpose:** Converts a string to an integer.
- **Parameters:** o str: The string to be converted.
- **Returns:** An integer parsed from the string.
- **Description:** Uses a std::stringstream to parse the integer from the string.

3. void readFile(const std::string &filename, std::string &content)

- **Purpose:** Reads the content of a file into a string.
- **Parameters:**
  - filename: The name of the file to read.

- content: A reference to a string where the file content will be stored.

- **Returns:** None.
- **Description:** Opens the file specified by filename, reads its content line by line, and appends it to the content string. If the file cannot be opened, an error message is printed.

4. void writeFile(const std::string &filename, const std::string &content)

- **Purpose:** Writes a string to a file.
- **Parameters:**
  - filename: The name of the file to write. o content: The string to be written to the file.
- **Returns:** None.
- **Description:** Opens the file specified by filename and writes the content string to it. If the file cannot be opened, an error message is printed.

5. std::string compress(const std::string &data)

- **Purpose:** Compresses a string using Run-Length Encoding (RLE).
- **Parameters:**
  - data: The string to be compressed.
- **Returns:** A compressed string where consecutive characters are replaced with the character followed by the number of occurrences.
- **Description:** Iterates through the input string, counts consecutive identical characters, and constructs the compressed string.

6. std::string decompress(const std::string &data)

- **Purpose:** Decompresses a string encoded with Run-Length Encoding (RLE).
- **Parameters:**
  - data: The compressed string to be decompressed.
- **Returns:** The original string before compression.
- **Description:** Iterates through the compressed string, extracts each character and its count, and reconstructs the original string by repeating each character according to its count.

Main Function

- **Purpose:** Executes the compression and decompression process.
- **Steps:**
  1. Reads the content of input.txt into a string.

2. Checks if the content is empty and exits with an error message if true.
3. Compresses the content using the compress function.
4. Writes the compressed content to compressed.txt.
5. Decompresses the compressed content using the decompress function.
6. Writes the decompressed content to decompressed.txt.
7. Prints a completion message to the console.

Usage

1. **Prepare Input File:**
   - Create a text file named input.txt and place it in the same directory as the executable.
   - Add the text content you want to compress.
2. **Compile and Run:**
   - Compile the program using a C++ compiler.
   - Run the executable.
3. **Check Output Files:**
   - After execution, check the directory for: ▪ compressed.txt: Contains the compressed data.
     ▪ decompressed.txt: Contains the decompressed data, which should match the original content of input.txt.

Error Handling

- **File Errors:** If the program cannot open a file, it prints an error message indicating the failure.
- **Empty File:** If the input file is empty, the program prints an error message and exits