#include <iostream>

#include <fstream>

#include <string>

Using namespace std;

// Function to compress the content using Run-Length Encoding

String compress(const string &input) {

String compressed = “”;

Int n = input.length();

For (int I = 0; I < n; i++) {

Int count = 1;

While (I < n – 1 && input[i] == input[I + 1]) {

Count++;

I++;

}

Compressed += input[i];

Compressed += to\_string(count);

}

Return compressed;

}

// Function to decompress the content encoded with Run-Length Encoding

String decompress(const string &input) {

String decompressed = “”;

Int n = input.length();

For (int I = 0; I < n; i++) {

Char ch = input[i];

I++;

String countStr = “”;

While (I < n && isdigit(input[i])) {

countStr += input[i];

i++;

}

i--; // Adjust index after the loop

int count = stoi(countStr);

decompressed.append(count, ch);

}

Return decompressed;

}

// Function to read a file’s content

String readFile(const string &fileName) {

Ifstream file(fileName, ios::in | ios::binary);

If (!file) {

Cerr << “Error: File could not be opened.” << endl;

Return “”;

}

String content((istreambuf\_iterator<char>(file)), istreambuf\_iterator<char>());

File.close();

Return content;

}

// Function to write content to a file

Void writeFile(const string &fileName, const string &content) {

Ofstream file(fileName, ios::out | ios::binary);

If (!file) {

Cerr << “Error: File could not be created.” << endl;

Return;

}

File.write(content.c\_str(), content.size());

File.close();

}

Int main() {

String inputFileName = “input.txt”;

String compressedFileName = “compressed.txt”;

String decompressedFileName = “decompressed.txt”;

// Read input file

String fileContent = readFile(inputFileName);

// Handle edge case: empty file

If (fileContent.empty()) {

Cerr << “Error: Input file is empty or could not be read.” << endl;

Return 1;

}

// Compress the file content

String compressedContent = compress(fileContent);

writeFile(compressedFileName, compressedContent);

cout << “File compressed and saved to “ << compressedFileName << endl;

// Decompress the file content

String decompressedContent = decompress(compressedContent);

writeFile(decompressedFileName, decompressedContent);

cout << “File decompressed and saved to “ << decompressedFileName << endl;

return 0;

}