# I. DESKRIPSI TUGAS

Dalam komunikasi data, pesan yang dikirim seringkali ukurannya sangat besar sehingga waktu pengirimannya lama. Begitu juga dengan penyimpanan data, arsip yang berukuran besar membutuhkan ruang penyimpanan yang besar. Kedua masalah ini dapat diatasi dengan mengkodekan pesan atau isi arsip sesingkat mungkin, sehingga waktu pengiriman pesan relatif cepat dan ruang penyimpanan yang dibutuhkan juga sedikit. Cara pengkodean seperti ini disebut kompresi (pemampatan) data dan pemulihan data tersebut kembali seperti aslinya disebut dekompresi (penirmampatan). Salah satu cara pemampatan dan nirmampat data ini adalah dengan menggunakan kode Huffman.

Pada tugas pertama Strategi Algoritma ini, anda diminta membuat aplikasi “*mini WinZip*” yang mengimplementasikan algoritma Huffman untuk memampatkan dan menirmampatkan data. Algoritma Huffman menggunakan prinsip *greedy* dalam pembentukan kode Huffman. Program anda harus dapat memampatkan semua jenis data baik berupa teks, gambar, suara, dan video dan Anda harus mampu mengembalikan data yang sudah dikompres tersebut ke bentuk asalnya (dekompresi). Sebagai contoh, jika *executable file* dimampatkan (misalnya notepad.exe), maka program notepade.exe tersebut harus dapat dijalankan kembali.

Program yang Anda buat selain mampu memampatkan dan menirmampatkan data juga harus dapat menunjukkan perubahan hasil kompresi tersebut melalui nisbah pemampatannya. Nisbah (*ratio*) pemampatan dhitung dengan rumus:

*P* = Ukuran *file* hasil pemampatan/Ukuran *file* sebelum dimampatkan × 100%

Yang berarti ukuran *file* menjadi *P* (dalam persentase) dari ukuran semula.

Waktu yang dibutuhkan untuk memampatkan dan menirmampatkan data juga dicatat. Selain itu, program harus dapat memampatkan banyak *file* sekaligus, dan ketika penirmampatan harus dapat mengembalikannya menjadi *file-file* semula dengan nama sama. Hitung juga entropi pesan dan rata-rata panjang bit setiap simbol di dalam pesan (file). Lihat Lampiran untuk menghitung entropi.

**Extension**: Algoritma Huffman standard memampatkan data agak lambat karena file harus dibaca dua kali. Pertama, file dibaca untuk menghitung frekuensi kemunculan karakter di dalam file. Kedua, file dibaca kembali untuk mengkodekan setiap karakter di dalam file dengan kode Huffman. Varian dari algoritma Huffman adalah algoritma *adaptive Huffman coding*. Pada algoritma adaptif ini, file dibaca hanya satu kali saja. Setiap kali karakter dibaca, pohon Huffman dibangun dan kode Huffman untuk karakter itu ditentukan, sekaligus karakter dikodekan langsung.

Anda boleh menggunakan algoritma adaptive *Huffman coding* di dalam Tubes ini, dengan bonus nilai = 15. Silakan mencari referensinya di Interrnet.

**Spesifikasi program :**

1. Program mampu memampatkan berkas (*file*) berjenis apapun secara tepat dan benar.
2. Program mampu memampatkan banyak file sekaligus menjadi satu berkas mampat.
3. Program mampu menirmampatkan data yang sebelumnya telah dikompres dengan tepat dan benar.
4. Program harus mampu menampilkan proses pemampatan dan nirmampat tersebut (misal dalam *progressive bar*).
5. Program juga harus manampilkan statistik berupa: lama waktu pemampatan (dan penirmampatan), ukuran file semula, ukuran file setelah dimampatkan, nisbah pemampatan, entropi, rata-rata bit/simbol, dll.
6. Antarmuka program sebaiknya kompatibel seperti *Winzip*.

# II. DASAR TEORI

Dalam membuat aplikasi ini kami menggunakan beberapa algoritma dan teori sebagai dasar bagi kami untuk membangun programnya. Algoritma dan teori-teori yang kami gunakan adalah:

## Algoritma Greedy

Algoritma greedy adalah suatu algoritma yang biasanya digunakan untuk menyelesaikan masalah optimisasi dengan cara mencari optimisasi lokal dalam setiap langkah. Algoritma greedy tidak dapat dipastikan akan menghasilkan hasil paling optimal dalam setiap masalah, tetapi algoritma greedy ini dapat menghasilkan suatu perkiraan nilai optimal dalam waktu yang relatif singkat.

## Pohon Biner

Pohon biner adalah suatu pohon dimana setiap simpul memiliki paling banyak 2 buah anak.

## Algoritma Huffman

Algoritma Huffman adalah suatu algoritma untuk mempersingkat jumlah bit yang digunakan untuk menyampaikan suatu pesan atau file. Algoritma ini menggunakan algoritma greedy dan pohon biner untuk melakukannya.

* *Adaptive Huffman Coding*

Algoritma Huffman biasa perlu untuk membaca file dua kali. Pertama untuk menghitung frekuensi tiap karakter yang muncul dan membangun pohonnya, lalu kedua untuk mengkodekan setiap karakter dalam file. Maka ada varian dari algoritma Huffman yaitu *adaptive Huffman coding. Adaptive Huffman coding* membangun pohon dan mengkodekan suatu karakter langsung setiap pembacaan karakter, jadi cukup dilakukan satu kali pembacaan file. Akibat dari metode ini adalah jumlah bit yang digunakan untuk memampatkan suatu file bisa sedikit lebih banyak dari menggunakan algoritma Huffman biasa, tetapi waktu eksekusi program jauh lebih cepat.