

1 Felhasználói dokumentáció

1.1 Felhasználói dokumentáció

A Huffman kódoló program arra való, hogy szöveg fájlokat tömörítsünk, illetve tömörített fájlokat visszaállítsuk. A Huffman algoritmus arra alapszik, hogy az egyes karaktereket kevesebb bittel és ezáltal kevesebb byte-al írja ki egy output fájlba. A tömörített, illetve kódolt szöveget nem akármilyen dekódoló táblával lehet visszaállítani. Emiatt a program egy dekódoló táblát is létrehoz. Ezekkel a fájlokkal kell szolgáltatni a programot amikor tömörítés vagy visszaállítás műveleteket végzünk. Parancssori hívással lehet elindítani a programot.

```
leveute@leveute-OMEN-Laptop-15-en0xxx:~/all_projects/c_projects/progi/nagy_hazi$ gcc main.c HuffmanDecompression.c HuffmanCompression.c HuffmanTree.c HashMap.c Queue.c Input.c -o Source
leveute@leveute-OMEN-Laptop-15-en0xxx:~/all_projects/c_projects/progi/nagy_hazi$ ./Source
```

A program kiírja az egyszerű menüt. Egyet kell gépelni, ha a kívánt művelet tömörítés, illetve kettőt, ha visszaállítás. Egyéb karakterek hibás gepelesként vannak tekintve, és a program kilép.

```
leveute@leveute-OMEN-Laptop-15-en0xxx:~/all_projects/c_projects/progi/nagy_hazi$ ./Source
Kodolas (1), dekodolas (2), kilepes (egyeb karakter):
-----
1
```

Tömörítés esetén, a program három fájlt kér: tömörítendő input fájl, üres output fájl és üres dekódoló tábla fájl. Az output fájl és dekódoló tábla fájlba írja a program az eredményeket. Az egyes fájlokat külön szekvenciálisan igényli a program. Hibás fájlok esetén kilép a program.

```
leveute@leveute-OMEN-Laptop-15-en0xxx:~/all_projects/c_projects/progi/nagy_hazi$ ./Source
leveute@leveute-OMEN-Laptop-15-en0xxx:~/all_projects/c_projects/progi/nagy_hazi$ ./Source
Kodolas (1), dekodolas (2), kilepes (egyeb karakter):
-----
1
error:--memory allocation failed--
return NULL;
}
Gepelje be a kodolando .txt fajlt:
pelda_input.txt
-----
Gepelje be az output .txt fajlt:
pelda_output.txt
-----
Gepelje be az ures kod tabla .txt fajlt:
pelda_decode_table.txt
-----
output[0] = '\0'; // new output string
int count = 0;
unsigned char codedChar = 0x00;
int index = 0;
```

Visszaállítás esetén, a program szintén három fájlt kér: visszaállítandó input fájl, üres output fájl és tömörítés által létrehozott dekódoló tábla fájl. Egyéb dekódoló táblák nem működnek visszaállításakor! Az output fájlba írja a program a visszaállított eredeti szöveget. Az egyes fájlokat tömörítés eljárásához hasonlóan külön szekvenciálisan fogja igényelni a program. Hibás fájlok esetén szintén kilép a program.

```
leveute@leveute-OMEN-Laptop-15-en0xxx:~/all_projects/c_projects/progi/nagy_hazi$ ./Source
leveute@leveute-OMEN-Laptop-15-en0xxx:~/all_projects/c_projects/progi/nagy_hazi$ ./Source
Kodolas (1), dekodolas (2), kilepes (egyeb karakter):
-----
2
error:--memory allocation failed--
return NULL;
}
Gepelje be a kodolt .txt fajlt nevet:
pelda_output.txt
-----
Gepelje be az ures output szoveg fajlt nevet:
pelda_decode_output.txt
-----
Gepelje be a decode tabla fajlt nevet:
pelda_decode_table.txt
-----
output[0] = '\0'; // new output string
int count = 0;
unsigned char codedChar = 0x00;
int index = 0;
```

A sikeres tömörítés és visszaállítás esetén kilép a program, és jelzi a felhasználónak, hogy a kívánt műveletet elvégezte.

```
leveute@leveute-OMEN-Laptop-15-en0xxx:~/all_projects/c_projects/progi/nagy_hazi$ ./Source
leveute@leveute-OMEN-Laptop-15-en0xxx:~/all_projects/c_projects/progi/nagy_hazi$ ./Source
Kodolas (1), dekodolas (2), kilepes (egyeb karakter):
-----
1
Gepelje be a kodolando .txt fajlt:
pelda_input.txt
-----
Gepelje be az output .txt fajlt:
pelda_output.txt
-----
Gepelje be az ures kod tabla .txt fajlt:
pelda_decode_table.txt
*
*
*
A kodolas megvan! Az eredmenyek megtekinthetok a fajokban. Vizsontlatasra!
*****
* Debugmallo: nincs memoriaszivargas a programban.
* Osszes foglalas: 226 blokk, 10067 bajt.
*****
leveute@leveute-OMEN-Laptop-15-en0xxx:~/all_projects/c_projects/progi/nagy_hazi$
leveute@leveute-OMEN-Laptop-15-en0xxx:~/all_projects/c_projects/progi/nagy_hazi$
leveute@leveute-OMEN-Laptop-15-en0xxx:~/all_projects/c_projects/progi/nagy_hazi$
leveute@leveute-OMEN-Laptop-15-en0xxx:~/all_projects/c_projects/progi/nagy_hazi$
leveute@leveute-OMEN-Laptop-15-en0xxx:~/all_projects/c_projects/progi/nagy_hazi$
```

Sikertelen memóriafoglalás vagy megnyújtás esetén az alábbi hibüzenetek közül az egyiket kiírja, és a program kilép.

memory allocation failed
failed to build Huffman Tree
failed to reallocate memory for buffer

1.2 Példa

```
pelda_input.txt
1 Az írott szöveg az emberiség történelmében hatalmas előrelépés,
2 hiszen így a történelme folyamán egyedüli módon lehetővé vált az
3 információ személytől térben és időben független tárolása szemben
4 a szójhagyománnyal, amely mind térben mind időben adott személyhez
5 vagy személyekhez kötött. A történelemről ránk maradt információk
6 legnagyobb része a XX. századig írásos szövegemlékekből áll. Azok
7 a szövegek, amelyek olyan kultúráktól származnak, ahol az írásos
8 információrögzítés létezik, a szövegek felépítése
```

Figure 1: Példa input szöveg fájl

```
pelda_decode_table.txt
1 y: 00000
2 k: 00001
3
4 : 000100
5 h: 000101
6 v: 000110
7 0: 000111
8 0: 001000
9 0: 00100100
10 X: 00100101
11 p: 00100110
12 0: 00100111
13 g: 00101
14 i: 00110
15 o: 00111
16 0: 010
17 : 011
18 t: 1000
19 0: 10010
20 d: 100110
21 b: 100111
22 0: 10100
23 0: 101010
24 ,: 1010110
25 0: 1010111
26 r: 10110
27 m: 10111
28 e: 1100
29 j: 110100000
30 u: 110100001
31 .: 11010001
32 f: 1101001
33 0: 1101010
34 A: 11010110
35 c: 11010111
36 n: 11011
37 s: 11100
38 z: 11101
39 a: 11110
40 t: 11111
41 4
```

Figure 2: Példa dekódoló tábla



| | | |
|--|-----------|-------|
|  pelda_input.txt | 582 bytes | 09:48 |
|  pelda_output.txt | 344 bytes | 10:11 |

Figure 3: Tömörítés eredménye: 582 byte → 344 byte