

On a fait des test avec des fichiers plus volumineux (candide.txt, gargantua.txt, huff_encode.c, huff_decode.c, tout les fichiers fournis).

```
[10:13:10]tagui$ ./huff_encode gargantua.txt out.txt
Occuerences du caractere
(code 10) : 2457
Occuerences du caractere (code 32) : 44947
Occuerences du caractere ! (code 33) : 260
Occuerences du caractere # (code 35) : 1
Occuerences du caractere ' (code 39) : 1775
Occuerences du caractere ( (code 40) : 211
Occuerences du caractere ) (code 41) : 211
Occuerences du caractere , (code 44) : 5054
Occuerences du caractere - (code 45) : 372
Occuerences du caractere . (code 46) : 1506
Occuerences du caractere 1 (code 49) : 1
Occuerences du caractere 2 (code 50) : 1
Occuerences du caractere : (code 58) : 199
Occuerences du caractere ; (code 59) : 330
Occuerences du caractere < (code 60) : 1998
Occuerences du caractere = (code 61) : 1
Occuerences du caractere > (code 62) : 1998
Occuerences du caractere ? (code 63) : 166
Occuerences du caractere A (code 65) : 393
Occuerences du caractere B (code 66) : 121
Occuerences du caractere C (code 67) : 434
Occuerences du caractere D (code 68) : 258
Occuerences du caractere E (code 69) : 348
Occuerences du caractere F (code 70) : 96
Occuerences du caractere G (code 71) : 358
Occuerences du caractere H (code 72) : 164
Occuerences du caractere I (code 73) : 237
Occuerences du caractere J (code 74) : 182
Occuerences du caractere L (code 76) : 327
Occuerences du caractere M (code 77) : 275
Occuerences du caractere N (code 78) : 91
Occuerences du caractere O (code 79) : 99
Occuerences du caractere P (code 80) : 489
Occuerences du caractere Q (code 81) : 101
Occuerences du caractere R (code 82) : 138
Occuerences du caractere S (code 83) : 248
Occuerences du caractere T (code 84) : 220
Occuerences du caractere U (code 85) : 39
Occuerences du caractere V (code 86) : 155
Occuerences du caractere X (code 88) : 78
Occuerences du caractere Y (code 89) : 1
Occuerences du caractere Z (code 90) : 1
Occuerences du caractere [ (code 91) : 1
Occuerences du caractere ] (code 93) : 1
Occuerences du caractere _ (code 95) : 4
Occuerences du caractere a (code 97) : 13702
Occuerences du caractere b (code 98) : 2346
Occuerences du caractere c (code 99) : 6736
Occuerences du caractere d (code 100) : 6383
Occuerences du caractere e (code 101) : 33669
Occuerences du caractere f (code 102) : 2233
Occuerences du caractere g (code 103) : 2316
Occuerences du caractere h (code 104) : 1886
Occuerences du caractere i (code 105) : 12575
Occuerences du caractere j (code 106) : 764
Occuerences du caractere k (code 107) : 4
Occuerences du caractere l (code 108) : 11173
Occuerences du caractere m (code 109) : 5297
Occuerences du caractere n (code 110) : 14814
Occuerences du caractere o (code 111) : 12679
Occuerences du caractere p (code 112) : 4966
Occuerences du caractere q (code 113) : 2607
Occuerences du caractere r (code 114) : 13501
Occuerences du caractere s (code 115) : 17212
Occuerences du caractere t (code 116) : 15101
Occuerences du caractere u (code 117) : 13600
Occuerences du caractere v (code 118) : 2933
Occuerences du caractere w (code 119) : 3
Occuerences du caractere x (code 120) : 867
Occuerences du caractere y (code 121) : 2659
Occuerences du caractere z (code 122) : 1508
Occuerences du caractere   (code 167) : 1
Occuerences du caractere   (code 171) : 180
Occuerences du caractere   (code 187) : 124
Occuerences du caractere   (code 192) : 1
Occuerences du caractere   (code 198) : 1
Occuerences du caractere   (code 199) : 2
Occuerences du caractere   (code 224) : 1038
Occuerences du caractere   (code 226) : 3
Occuerences du caractere   (code 231) : 126
Occuerences du caractere   (code 232) : 140
Occuerences du caractere   (code 233) : 1041
Occuerences du caractere   (code 234) : 1
Occuerences du caractere   (code 235) : 1
Occuerences du caractere   (code 239) : 3
Occuerences du caractere   (code 244) : 4
Occuerences du caractere   (code 249) : 32
#
```



L'utilisation des modules :

Pour encoder nos textes, on utilise les structures d'arbres, file-à-priorité et pour décoder on utilise les arbres et les bfiles car l'algorithme d'huffmann encode chaque lettre avec des 0 et des 1, donc pour le décoder il fallait le lire bit-à-bit.

Conclusion :

Au début, on encodait nos fichiers de la sorte
«fprintf(fichier_encode,"%i",HuffmanCode[(int)c].code[i])»,
et ça faisait le travail demander avec des fichiers de petites tailles et dès qu'on testait avec des fichiers plus volumineux, on obtenait une segmentation fault. Après le débogage, on s'est aperçu qu'il fallait caster le caractère lu par le type Element et non pas par int «fprintf(fichier_encode,"%i",HuffmanCode[(Element)c].code[i])».