Fil Rouge 2022

Codage de Huffman

Organisation

- Ce Fil Rouge doit être réalisé en équipe. Chaque équipe doit être constituée au maximum de 4 étudiants, appartenant au même groupe TD. Il ne doit pas y avoir plus de 8 équipes par groupe TD. Les équipes doivent être les mêmes que celles du TEA2 et du TEA3.
- Chaque équipe rendra une seule archive sur moodle, contenant :
    - Les fichiers sources du programme produit ;
    - Un fichier makefile permettant de compiler le programme
    - Un compte-rendu CR rédigé (avec introduction, développement, conclusion, perspectives et bibliographie) présentant l'organisation du programme et celle du groupe (qui a fait quoi, avec quel planning, quelles ont été les difficultés, etc.)
- Des soutenances seront organisées à la rentrée des congés de Noël. Elles auront lieu en groupe TD, avec un jury composé de deux enseignants. Chaque soutenance durera 30 minutes : 20 minutes de présentation, 10 minutes de questions.
- L'évaluation du fil rouge portera sur la qualité du code livré et de la livraison, la qualité du CR et la soutenance.

Programme 1 : heapsort.exe

Produire un programme heapsort permettant de trier des T_elt et répondant aux contraintes suivantes :

- Une constante symbolique permet de sélectionner la nature du tas (minimier ou maximier) et donc l'ordre de tri
- La fonction de tri est adaptée aux outils d'évaluation expérimentale mis en œuvre dans la séance 3 (complexité)
    - Son prototype est donc T_data heapSort(T_data d, int n)

Comparer expérimentalement les vitesses des tris de votre TEA3 (tri fusion, tri rapide) avec celle du tri par tas

Programme 2 : codage.exe

Produire un programme codage.exe prenant sur son entrée standard les caractères d'un texte à compresser

- Il pourra donc être appelé de plusieurs manières :
    - ./codage.exe (suivi d'une saisie au clavier des caractères à entrer)
    - ./codage.exe < fichier (pour envoyer les caractères d'un fichier au programme à l'aide d'une redirection d'entrée)

- cat fichier | ./codage.exe (pour envoyer les caractères d'un fichier au programme à l'aide d'une redirection de commandes)

Le programme devra réaliser un codage de huffman et afficher sur sa sortie standard la table de codage générée, le texte compressé ainsi qu'un bilan affichant le ratio de compression.

Par exemple, lors de l'encodage de la chaîne "algorithme de huffman pour la compression de chaines", l'arbre de codage devra être affiché sous la forme suivante :

```
car : occ | long | bits
----+-----+------+-------
' ' :   7 |    3 | 010
'a' :   4 |    4 | 0011
'c' :   2 |    5 | 00000
'd' :   2 |    5 | 01110
'e' :   5 |    3 | 111
'f' :   2 |    5 | 00001
'g' :   1 |    5 | 11000
'h' :   3 |    4 | 1010
'i' :   3 |    4 | 1001
'l' :   2 |    5 | 00011
'm' :   3 |    4 | 1000
'n' :   3 |    4 | 1011
'o' :   4 |    4 | 0010
'p' :   2 |    5 | 00010
'r' :   3 |    4 | 1101
's' :   3 |    4 | 0110
't' :   1 |    5 | 11001
'u' :   2 |    5 | 01111
```

Le code généré devra être affiché sous cette forme :

```
00110001111000001011011001110011010100011101001110111010101001111000010000110000011101101000010001001111111010100001100110100000000101000000101101111011001101001001010110100111011101000001010001110011011111110110
```

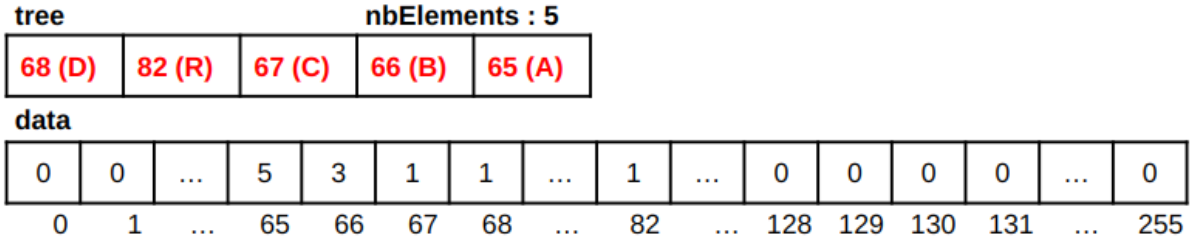
La conclusion devra être affichée sous cette forme :

```
Longueur du code binaire : 416 bits
Longueur du code de huffman :210 bits
Ratio de compression : 50.48%
```

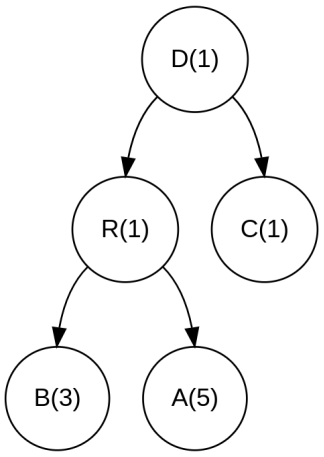
Le programme devra respecter les contraintes suivantes :

- Au fur et à mesure de son exécution, le programme génère des fichiers permettant de visualiser l'arbre partiellement ordonné et l'arbre de codage de manière graphique, à l'aide de l'outil graphviz.

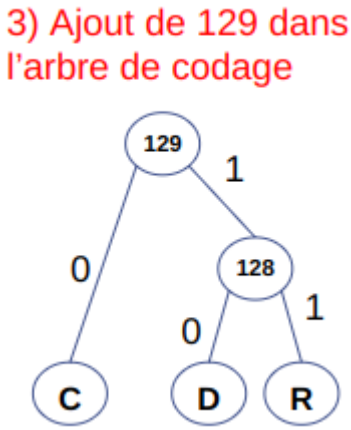
Par exemple, le minimier indirect présenté à la page 133 du poly :



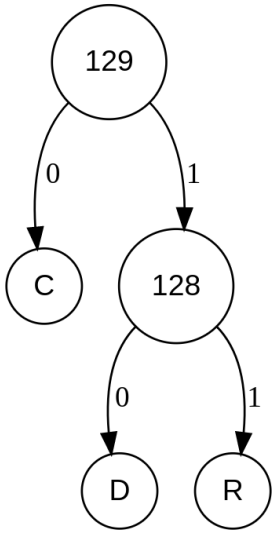
devra être présenté sous la forme suivante :



L'arbre de codage présenté page 135 :



devra être présenté sous la forme :



ce qui peut être obtenu par le code graphviz ci-dessous :

```
digraph POT_test {
        node [fontname="Arial", shape="circle", width=0.5];
        129 [label = "129"];
        129:sw -> 67 [label = " 0"];
        129:se -> 128 [label = " 1"];
        128 [label = "128"];
        128:sw -> 68 [label = " 0"];
        128:se -> 82 [label = " 1"];
        67 [label = "C"];
        68 [label = "D"];
        82 [label = "R"];
}
```

- Le code devra être canonique en respectant l'ordre lexicographique des caractères lors de leur extraction du minimier indirect (et donc lors de sa création), et plus du nombre d'occurrences des caractères dans le texte.

Programme 3 : huffman.exe

Produire un programme huffman.exe prenant un ou deux paramètres :

- Lorsque le programme prend deux paramètres, il doit compresser un texte et produire un fichier contenant le texte compressé précédé de l'entête de huffman permettant de le décompresser
    - paramètre 1 : chemin du fichier à compresser
    - paramètre 2 : chemin du fichier à produire

Exemple d'appel : `huffman.exe ./fichierACompresser ./fichierResultat`

- Lorsque le programme prend un seul paramètre, il s'agit du chemin d'un fichier à décompresser. Le contenu du document décompressé doit s'afficher sur la sortie standard.

Exemple d'appel : `huffman.exe ./fichierADecompresser`

Bien entendu, votre programme doit pouvoir décompresser les fichiers qu'il a lui-même compressé.