Polytech Nice Sophia
SI3 / 2021-2022
Erick Gallesio

Programmation Procédurale

Feuille 2

Encore des exercices simples

> *Les exercices de cette feuille sont simples. Leur but est de vous familiariser avec le langage C et le compilateur. La plupart de ces exercices prennent leurs données sur le fichier standard d'entrée et placent leurs résultats sur le fichier standard de sortie. Bien sûr, vous pouvez utiliser les redirections Unix si vous voulez travailler sur des vrais fichiers.*

1 Comptage

Ecrire un programme qui affiche le nombre d'occurences pour chacun des chiffres et des lettres qu'il a lu sur le fichier standard d'entrée.

```
$ echo "abc 123 cba xy 42" | ./comptage
1:  1 fois
2:  2 fois
3:  1 fois
4:  1 fois
a:  2 fois
b:  2 fois
c:  2 fois
x:  1 fois
y:  1 fois
```

2 Une version simplifiée de wc

Ecrire un programme qui affiche le nombre de caractères, lignes et mots lus sur le fichier standard d'entrée. On considère ici, comme dans la commande standard `wc` , qu'un mot est une suite de caractères délimitée par les caractères SPACE, TAB, ou NEWLINE.

```
$ cat in_wc.txt
7       mots
2   lignes et 45 caractères
$ ./wc < in_wc.txt
lines: 2
words: 7
chars: 45
```

3 Insertion dans un tableau

Écrire une fonction `insertion` qui insère une valeur dans un tableau d'entiers triés. Cette fonction prend en paramètres:

- un tableau,
- le nombre d'éléments présents dans ce tableau et
- la valeur `v` à insérer.

On suppose ici que les éléments du tableau sont triés par ordre croissant. La fonction `insertion` insère alors la valeur `v` à sa place.

Votre programme de test saisira une suite d'entiers sur le fichier standard d'entrée et les insérera successivement dans un tableau. A la fin de la saisie, le contenu du tableau sera affiché. Si tout se passe bien, les valeurs affichées seront triées, même si elle ont été saisies *dans le désordre*.

4 Dump hexadécimal

Ecrire un programme permettant d'afficher le contenu du fichier standard d'entrée sur le fichier standard de sortie en hexadécimal et en caractères. Ce programme travaillera par lignes de 16 caractères. Pour chacune de ces lignes, on aura à gauche le code hexadécimal des caractères et à droite leur équivalent sous forme caractère (les caractères non imprimables étant représentés par le caractère '.').

Note: Vos ordinateurs utilisent le jeu ASCII pour coder les caractères. Les caractères imprimables dans ce jeu sont ceux compris entre `' '` et `'~'`.

Exemple:

Soit le fichier d'entrée input.txt:

```
Un exemple avec des accents: éèàê
Les 6 derniers caractères sont: '\0', '0', '\t', '9', 'A' et enfin '\n':
0  9A
```

Le résultat de la commande `dump_hexa < input.txt` :

```
$ dump_hexa < input.txt
55 6e 20 65 78 65 6d 70 6c 65 20 61 76 65 63 20  | Un exemple avec
64 65 73 20 61 63 63 65 6e 74 73 3a 20 c3 a9 c3  | des accents: ...
a8 c3 a0 c3 aa 0a 4c 65 73 20 36 20 64 65 72 6e  | ......Les 6 dern
69 65 72 73 20 63 61 72 61 63 74 c3 a8 72 65 73  | iers caract..res
20 73 6f 6e 74 3a 20 27 5c 30 27 2c 20 27 30 27  |  sont: '\0', '0'
2c 20 27 5c 74 27 2c 20 27 39 27 2c 20 27 41 27  | , '\t', '9', 'A'
20 65 74 20 65 6e 66 69 6e 20 27 5c 6e 27 3a 0a  |  et enfin '\n':.
00 30 09 39 41 0a                                | .0.9A.
```

Essayer ensuite votre programme avec le fichier d'entrée all.txt qui contient les 255 caractères possibles (2 fois). Vérifiez que vous obtenez bien le résultat suivant (et non pas la première moitié du fichier seulement):

```
$ dump_hexa < all.txt
00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f  | ................
10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f  | ................
20 21 22 23 24 25 26 27 28 29 2a 2b 2c 2d 2e 2f  |  !"#$%&'()*+,-./
30 31 32 33 34 35 36 37 38 39 3a 3b 3c 3d 3e 3f  | 0123456789:;<=>?
40 41 42 43 44 45 46 47 48 49 4a 4b 4c 4d 4e 4f  | @ABCDEFGHIJKLMNO
50 51 52 53 54 55 56 57 58 59 5a 5b 5c 5d 5e 5f  | PQRSTUVWXYZ[\]^_
60 61 62 63 64 65 66 67 68 69 6a 6b 6c 6d 6e 6f  | `abcdefghijklmno
70 71 72 73 74 75 76 77 78 79 7a 7b 7c 7d 7e 7f  | pqrstuvwxyz{|}~.
80 81 82 83 84 85 86 87 88 89 8a 8b 8c 8d 8e 8f  | ................
90 91 92 93 94 95 96 97 98 99 9a 9b 9c 9d 9e 9f  | ................
a0 a1 a2 a3 a4 a5 a6 a7 a8 a9 aa ab ac ad ae af  | ................
b0 b1 b2 b3 b4 b5 b6 b7 b8 b9 ba bb bc bd be bf  | ................
c0 c1 c2 c3 c4 c5 c6 c7 c8 c9 ca cb cc cd ce cf  | ................
d0 d1 d2 d3 d4 d5 d6 d7 d8 d9 da db dc dd de df  | ................
e0 e1 e2 e3 e4 e5 e6 e7 e8 e9 ea eb ec ed ee ef  | ................
f0 f1 f2 f3 f4 f5 f6 f7 f8 f9 fa fb fc fd fe ff  | ................
fe fd fc fb fa f9 f8 f7 f6 f5 f4 f3 f2 f1 f0 ef  | ................
ee ed ec eb ea e9 e8 e7 e6 e5 e4 e3 e2 e1 e0 df  | ................
de dd dc db da d9 d8 d7 d6 d5 d4 d3 d2 d1 d0 cf  | ................
ce cd cc cb ca c9 c8 c7 c6 c5 c4 c3 c2 c1 c0 bf  | ................
be bd bc bb ba b9 b8 b7 b6 b5 b4 b3 b2 b1 b0 af  | ................
ae ad ac ab aa a9 a8 a7 a6 a5 a4 a3 a2 a1 a0 9f  | ................
9e 9d 9c 9b 9a 99 98 97 96 95 94 93 92 91 90 8f  | ................
8e 8d 8c 8b 8a 89 88 87 86 85 84 83 82 81 80 7f  | ................
7e 7d 7c 7b 7a 79 78 77 76 75 74 73 72 71 70 6f  | ~}|{zyxwvutsrqpo
6e 6d 6c 6b 6a 69 68 67 66 65 64 63 62 61 60 5f  | nmlkjihgfedcba`_
5e 5d 5c 5b 5a 59 58 57 56 55 54 53 52 51 50 4f  | ^]\[ZYXWVUTSRQPO
4e 4d 4c 4b 4a 49 48 47 46 45 44 43 42 41 40 3f  | NMLKJIHGFEDCBA@?
3e 3d 3c 3b 3a 39 38 37 36 35 34 33 32 31 30 2f  | >=<;:9876543210/
2e 2d 2c 2b 2a 29 28 27 26 25 24 23 22 21 20 1f  | .-,+*)('&%$#"! .
1e 1d 1c 1b 1a 19 18 17 16 15 14 13 12 11 10 0f  | ................
0e 0d 0c 0b 0a 09 08 07 06 05 04 03 02 01 00     | ...............
```

5 Calculatrice (facultatif)

Construire une petite calculatrice qui exécute la boucle (infinie) suivante:

1. lecture ligne contenant un nombre, un opérateur (parmi `+`, `-`, `*`, `/`) suivi d'un nombre (par exemple `2 + 2` ou `3.5 * 8`.)
2. calcul du résultat 3 affichage du résultat.

Exemple d'utilisation du programme:

```
$ ./calc
? 3+2
= 5
? 10   / 3
= 3.33333
? 7.5 * 42e6
= 3.15e+08
? ^D
$
```

6 Suppression des commentaires C (facultatif)

Ecrire le filtre `uncomment` qui remplace chaque commentaire C `/* ...*/` par un espace. On ne tiendra pas compte des chaînes de caractères C. De plus, les commentaires ne peuvent pas être imbriqués.

Soit le fichier suivant:

```
/* foo.c */
int ident/* comm */ificateur;
char *s = "/* chaine */";
1/*/2 3/*/4
/*1 comm /*2 mentaire 3*/ 4*/
```

L'exécution de `uncomment sur ce fichier doit produire:

```
int ident ificateur;
char *s = " ";
1 4
  4*/
```

7 Manipulation des tabulations (facultatif)

Les terminaux Unix possèdent des taquets de tabulation, placés tous les 8 caractères à partir du début de la ligne. A l'affichage, une tabulation (caractère `'\t'`) positionne le curseur au prochain taquet.

- Ecrire le filtre `detab` qui remplace chaque tabulation par des espaces, de telle sorte que l'entrée et la sortie aient la même apparence à l'affichage.
- Ecrire le programme `entab` qui remplace des séquences d'espaces par des tabulations. Par souci de simplification on ne remplacera que la plus longue séquence d'espaces commençant au début d'une ligne.

On peut généraliser ce mécanisme à des taquets de tabulation placés tous les `n` caractères. Intégrer cette généralisation dans `entab` et `detab` : par exemple on tapera

```
$ detab -3 < foo > bar
```

pour supprimer les tabulations (avec ici `n` = 3).