TP2 : Les boucles for

Guillaume Aubian

Ce document a été grandement inspiré par – et largement copié sur – le document correspondant du responsable précédent de ce cours, Juliusz Chroboczek, avec son accord.

1 Premières boucles

1. Écrivez un programme `vertical.c` qui demande à l'utilisateur un entier n et affiche une colonne de « * » de longueur n.

2. Écrivez un programme `horizontal.c` qui demande à l'utilisateur un entier n et affiche une ligne de « * » de longueur n.

3. Écrivez un programme `carres.c` qui affiche les carrés des 10 premiers nombres naturels, c'est-à-dire la suite d'entiers $1, 4, 9, \ldots, 100$.

4. Écrivez un programme `multiples.c` qui demande à l'utilisateur un entier n puis affiche les 10 premiers multiples de n. Par exemple, si l'utilisateur entre 7, votre programme devra afficher la suite $7, 14, 21, \ldots, 70$.

2 Boucles avec accumulation

1. Écrivez un programme `somme-cubes.c` qui demande à l'utilisateur un entier n puis qui affiche la somme des cubes des n premiers nombres entiers. Par exemple, si l'utilisateur entre 5, votre programme devra afficher 225, car

$$\sum_{k=1}^{5} k^3 = 1^3 + 2^3 + 3^3 + 4^3 + 5^3 = 1 + 8 + 27 + 64 + 125 = 225$$

2. Écrivez un programme `somme.c` qui demande à l'utilisateur un entier n, puis qui lit n entiers et affiche leur somme.

3 Boucles avec flags

– Écrivez un programme `sept.c` qui demande à l'utilisateur un entier n puis qui lit n entiers et indique à l'utilisateur si le nombre 7 se trouvait parmi ces n entiers.

– Écrivez un programme `premier.c` qui demande à l'utilisateur un entier n puis qui indique à l'utilisateur si cet entier est premier.

4 Boucles imbriquées.

– Écrivez un programme `carre1.c` qui demande à l'utilisateur un entier n et affiche un carré plein de « * » de côté n.

```
*****
*****
*****
*****
*****
```

– Écrivez un programme `carre2.c` qui demande à l'utilisateur un entier n et affiche un carré creux de « * » de côté n.

```
*****
*   *
*   *
*   *
*****
```

– Écrivez un programme `triangle1.c` qui demande à l'utilisa- teur un entier n et affiche un triangle plein de « * » de largeur n avec la pointe en haut à gauche.

```
*
**
***
****
*****
```

– Écrivez un programme `triangle2.c` qui demande à l'utilisa- teur un entier n et affiche un triangle plein de « * » de largeur n avec la pointe en bas à droite.

```
*****
 ****
  ***
   **
    *
```