TP2. Types entiers, affichage et boucles

Dimitri Galayko

1 Types et affichage formatté

Exo 1.1. Écrivez un programme qui affiche les valeurs maximales *en format décimal*des types non-signés char, short, long. Modifiez le programme pour que soient affichées les limites supérieures des types signés.

Ecrivez deux versions du programme:

– l'une, utilisant la connaissance *a priori* des valeurs extrêmes des types. Par exemple, la valeur extrême du type *unsigned long* est 0xFFFFFFFFFFFFFFFF (8 octets remplis de 1).

– l'autre, utilisant les constantes prédéfinies dans le fichier limits.h (se faire aider par l'encadrant). Avec cette version, affichez également les limites pour les types *int* and *unsigned int*.

Exo 1.2. Soit le début du code suivant:

```
#include <math.h> // bibliothèque incluant les fonctions mathématiques

int main(){

double fa, fb;
int b;
long c;

fa=sqrt(1000); // racine carrée de 1000
fb=exp(-10); // exponentielle de -10
c=296483247; // un nombre entier
...
...
```

Compléter le code afin d'afficher :
fa=31.6228 (la variable fa avec 4 chiffres après la virugle)
fa=3.1623e+01 (la même variable en format exponentielle, avec 4 chiffres après la virgule,
fa=3.1623E+01 (La même chose, mais le caractère "e" est en majuscule)
fb=4.539992976248485417314557 1e-05 (La variable fb avec 25 chiffres après la virgule)
c=11abf9af (La variable c en hexadécimale)
c=11ABF9AF (La variable c en hexadécimale avec lettres majuscules)
c= 296483247 (La variable c sur 15 positions)
c= +296483247 (La même chose mais avec le signe explicite)

2 Boucles et algorithmes élémentaires

Exo 2.1. Ecrivez un programme qui affiche une table de 15 valeurs de sinus, pour l'argumant valant de -$\pi/2$ à +$\pi/2$, à intervalles régulières. Vous devez obtenir une sortie suivante:

```
Argument   Sinus
-1.571     -1.0000000000
-1.257     -0.9510565163
-0.942     -0.8090169944
-0.628     -0.5877852523
```

```
-0.314     -0.3090169944
 0.000      0.0000000000
 0.314      0.3090169944
 0.628      0.5877852523
 0.942      0.8090169944
 1.257      0.9510565163
 1.571      1.0000000000
```

Pretez attention à la mise en forme: les colonnes alignées avec les légendes, la position des signes, etc.

Exo 2.2. Rectangle. Ecrivez un programme qui affiche un rectangle à l'aide des étoiles "*", de taille que l'utilisateur saisit au clavier (par ex., 10x5). Vous utiliserez les boucles *for* imbriquées.

Exo 2.3. Losange. Dessinez une losange de taille L que l'utilisateur entre au clavier.

L peut uniquement être impaire (pourquoi ?). Par conséquent, il faut faire une vérification de la parité et demander à l'utilisateur de refaire la saisie si L est paire.

3 Nombres de Fibonacci

Exo 3.1. Ecrivez un algorithme et un programme C calculant N premiers nombres de Fibonacci. N est spécifié par l'utilisateur. Les nombres calculés sont affichés en colonne, sur le terminal.

Pour mémoire, les nombres de Fibonacci sont donnés par la formule récursive :

$$F_n = \begin{cases} 1, & si \;\; n = 0, 1, \\ F_{n-1} + F_{n-2}, & si \;\; n \geq 2. \end{cases} \tag{1}$$

Exo 3.2. Modifiez ce programme pour calculer les rapports entre les nombres voisins, et afficher les rapports.

Testez pour le programme pour 10, 20, 30 et 50 nombres. Quelle tendance observez vous sur le rapport ?

4 Le plus grand diviseur commun (algorithme d'Euclide)

Ecrivez le programme calculant le plus grand diviseur commun (PGCD) de deux nombres entiers saisis au clavier par l'utilisateur. En voici le résumé d'algorithme:

– si un des nombres est nul, l'autre est le PGCD

– sinon il faut soustraire le plus petit du plus grand et laisser le plus petit inchangé.

– recommencer ainsi avec la nouvelle paire jusqu'à ce que un des deux nombres soit nul. Dans ce cas, l'autre nombre est le PGCD.

```
Exemple: si a vaut 32 et b vaut 14, on obtient successivement
 32              14
 32-14=18        14
 14             4=18-14
 14-4=10           4
 10-4=6            4
 6-4=2           4
 4-2=2           2
 2              2-2=0
```