Quick - Éléments de correction.

Exercice 1 Écrire un programme complet qui demande un entier x à l'utilisateur et imprime tous les entiers entre x et $x+42$ (compris).

CORRECTION : Sans surprise, on utilise une variable pour x et une boucle pour :

```
Programme Main()
    L: i,x :Entiers
    Demander(x)
    Pour i de x à x+42 Faire
        Imprimer(i)
    Fpour
    Retourner 0
FProgramme
```

Bien noter que i et x sont des variables locales (leurs valeurs ne sont pas données en paramètres du main, et l'usage du i et x est limité au main). On pouvait aussi faire varier i de 0 à 42 en imprimant $x+i$. ■

Exercice 2 Écrire un programme complet (**sans utiliser de tableau**) qui demande des entiers strictement positifs à l'utilisateur, s'arrête lorsque l'utilisateur rentre un entier négatif ou nul, et imprime le maximum des entiers positifs entrés.

CORRECTION : Ici on ne connaît pas le nombre d'entiers que va entrer l'utilisateur avant d'entrer 0 ou un nombre négatif. On utilise donc une boucle Tant que (cas terminal : l'utilisateur rentre un x négatif ou nul). On utilise une variable pour stocker le maximum (max) que l'on initialise à 0 :

```
Programme Main()
    L: x,max :Entiers
    max ← 0
    Demander(x)
    Tq x > 0 faire
        Si x>max alors
            max ← x
            Demander(x)
        Fsi
    Ftq
    Imprimer(max)
    Retourner 0
FProgramme
```

Idem, noter que i et x sont locales. On pouvait utiliser une boucle Faire ... Tant que, ce qui est assez élégant. Remarque : il n'est pas nécessaire de stocker la valeur précédente de x. dans la boucle. ■

Exercice 3 On considère la suite récurrente suivante :

$$u_n = \begin{cases} 1 & \text{si n=0} \\ 3 * u_{n-1} + 42 & \text{si n>0} \end{cases}$$

Écrire un algorithme (fonction ou action ? justifier !) qui calcule et stocke dans un vecteur de taille N>1 (tableau passé en paramètre modifiable, N constante fixée supposée supérieure à 1), l'entier u_i à la case numéro i du tableau.

CORRECTION : On écrit une action car on ne va effectuer que des modifications au tableau passé en paramètre R (ou D/R), sinon les modifications apportées au tableau ne sont pas enregistrées. L'algorithme consiste à parcourir le tableau et à remplir en s'apercevant que $t[0] = 1$, puis $t[i] = 3 * t[i-1] + 42$:

Action *facttab(t)*
| **R**: t :Vecteur[N] d'entiers
| **L**: i :Entier
| t[0] ← 0
| **Pour** *i* **de** *1 à N-1* **Faire**
| | t[i] ← 3*t[i-1]+42
| **Fpour**
FAction

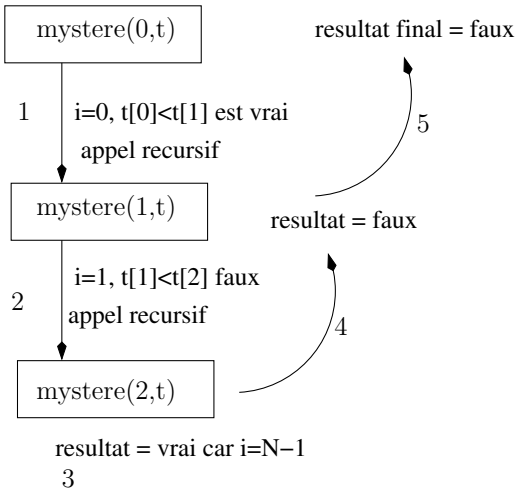
On peut (pas obligatoire en pseudo code) aussi passer la taille N en paramètre donnée. Bien noter l'initialisation de la première case, qui n'est pas dans la boucle Pour (faire un test $i = 0$ à l'intérieur de la boucle est de la mauvaise programmation).
On peut aussi créer une fonction qui retourne un tableau t, le tableau est alors une variable locale (et non un paramètre), et la signature de la fonction est `Action facttab():Vecteur[N]d'entiers`. Noter que retourner un tableau est autorisé en pseudo-code mais pas en C.

Cet algorithme réalise une multiplication par tour de boucle, donc en tout $O(N)$ multiplications (au passage la complexité ne dépend que des données du programme, ici N est l'unique donnée). ■

Exercice 4 Soit la fonction mystère suivante : ($N = 3$)

Fonction *mystere(i,t) :Booleen*
| **D**: t :Vecteur[N] d'entiers
| **Si** *i<0* **ou** *i=N-1* **alors**
| | **Retourner** *Vrai*
| **Sinon**
| | **Retourner** *((t[i]<t[i+1])* **et** *mystere(i+1,t))*
| **Fsi**
FFonction

Quels calcule réalise l'appel `mystere(0,t)` pour $t = [10, 22, 1]$? (avec les étapes) puis pour $t = [12, 23, 1515]$ (donner uniquement le résultat) ? Dans le cas général ?
CORRECTION : Bien numéroter les appels récursifs, et ne pas oublier les retour des fonctions. Attention, lorsqu'un appel récursif retourne, il redonne la main à son appelant. Déroulage pour le premier cas :



Dans le premier cas, `Faux` est retourné. Dans le deuxième cas `Vrai` est retourné.

Dans le cas général, si $i < 0$, l'appel `mystere(i,t)` retourne vrai, et si $i \geq 0$ l'appel retourne vrai si et seulement si le sous tableau $t[i..N-1]$ est trié dans l'ordre strictement croissant.

Attention aux formulations logiquement fausses (ou pas complètes). ■