Prototypes

11 janvier 2010

1. Une routine permet de mettre en majuscules une chaîne de caractères (l'original ne doit pas être conservé).

   ```
   Prototype procédure prMajuscules(adresse chaine sTexte)
   ```

   La procédure modifie directement une chaîne (donc passée par adresse). Il n'y a pas de valeur de retour.

2. Une routine détermine le montant des allocations familiales pour un enfant, connaissant son âge, sa place dans la fratrie (premoier, deuxième, ...) et son taux d'invalidité.

   ```
   Prototype fonction flottante fnAllocation(valeur entière iAge,
                                             valeur entière iPosition,
                                             valeur flottante fHandicap)
   ```

   Le but est de calculer un montant. L'âge et la position sont simplement utilisés (ce sont des entiers). Le handicap lui aussi simplement lu, sera proposé comme flottant pour éviter tout problème ultérieur.

3. Une routine détermine si un tableau de 10 entiers contient trois fois une valeur donnée.

   ```
   Prototype fonction logique flContenir(tableau entiers iTableau,
                                         valeur entière iValeur)
   ```

   La routine renvoie une valeur vrai ou faux. Elle utilise un tableau et la valeur à rechercher. La taille du tableau étant fixée, il n'est pas nécessaire de la passer en paramètre.

4. Une routine effectue une lecture directe du clavier et renvoie la valeur du caractère tapé. Dans certains cas, la valeur lue ne correspond pas un caractère mais à un code spécifique (caractère normal, *escape*, touche de fonction, ou caractère d'édition).

   ```
   Prototype fonction caractère fnLecture(adresse entière iCodRetour)
   ```

   La routine renvoie en principe un caractère, mais une variable, passée par adresse, recueille le code de retour.

5. Une routine détermine le montant de l'impôt des personnes physiques (calcul final) sur base du montant des revenus après déduction des avantages fiscaux et d'un tableau reprenant pour chaque catégorie le pivot, la somme fixe pour le montant égal au pivot et le taux pour la dernière tranche.

   ```
   Prototype fonction flottante fnCalculIPP(valeur flottante fRevenus,
                                            tableau flottants fCategorie :3)
   ```

   La routine effectue un cacul. Elle utilise une valeur non modifiée et un tableau comprenant une ligne de trois données par catégorie.

6. Il faut compter le nombre de voyelles dans une chaine de caractères.

```
Prototype fonction entière fnVoyelle(valeur chaine sChaine)
```

La routine compte les voyelles et renverra donc un entier, c'est une fonction. Elle doit connaitre la chaine de caractères, mais ne va pas la modifier (passage idéalement par valeur, bien que ce soit impossible en C).

7. Il faut remplacer tous les nombres négatifs d'un tableau de 10 entiers par leur équivalent positif.

```
Prototype procédure prPositiver(tableau entiers iTableau)
```

La routine ne renvoie rien (procédure). Elle doit connaitre le tableau et pouvoir le modifier (mais les tableaux sont toujours passés par adresse en C). Notons que la taille est une constante (on pourrait la définir comme telle au début du programme).

8. Une routine affiche les nombres supérieurs à 50 contenus dans un tableau de flottants. Il est important qu'on sache combien de nombres ont été ignorés.

```
Prototype fonction entière fnGrandNombre(tableau flottant iTableau,
                                         valeur entière iTaille)
```

Pour transmettre le nombre de nombres ignorés, il faut une fonction qui renvoie un entier. On doit utiliser un tableau (qu'on devrait pouvoir passer par valeur si c'était possible). La taille du tableau peut varier selon les besoins des utilisateurs, on doit donc passer cette taille (par valeur, il ne s'agit pas de la modifier).

9. copier un tableau d'entiers dans un autre, la taille des tableaux pouvant varier.

```
prototype procédure prCopier(tableau entiers iTableau1,
                             tableau entiers iTableau2,
                             valeur entière iTaille)
```

10. compter le nombre de consonnes dans une chaine de caractères.

```
Prototype fonction entière fnConsonnes(valeur chaine sChaine)
```

La routine compte les consonnes et renverra donc un entier, c'est une fonction. Elle doit connaitre la chaine de caractères, mais ne va pas la modifier (passage idéalement par valeur, bien que ce soit impossible en C).

11. déterminer si deux nombres sont premiers entre eux (admettent un comme seul diviseur commun).

```
Prototype fonction logique fnPremiers(valeur entière iNombre1,
                                      valeur entière iNombre2)
```

La routine renvoie vrai ou faux (fonction). Elle a besoin de la valeur des deux nombres à examiner.

12. remplir un tableau avec un maximum de 20 nombres flottants. Il est important de savoir combien de nombres ont été saisis.

```
Prototype procédure prRemplir(tableau flottants fTableau,
                              adresse entière iNbrNombres)
```

La routine va simplement modifier le tableau (procédure). Elle reçoit un tableau de flottants (toujours passé par adresse) et l'adresse d'une variable entière pour y placer le nombre d'éléments saisis. On peut aussi envisager une version fonctionnelle.

```
Prototype fonction entière  fnRemplir(tableau flottants fTableau)
```

13. afficher les nombres compris entre deux limites.

```
Prototype procédure prAfficherNombres(valeur entière iMinimum,
                                      valeur entière iMaximum)
```

La routine ne renvoie rien (procédure). Elle doit utiliser la valeur des deux limites.