TP5 - Programmation avancée

Exercice 1 - Calcul de combinaisons

On se propose de calculer de façon optimale les combinaisons. On rappelle que :

$$C_n^p = \frac{!n}{!p*!(n-p)} = \frac{n*(n-1)*(n-2)*\cdots*(n-p+1)}{p*(p-1)*(p-2)*\cdots*2} \quad \text{pour } 0 \leq p \leq n$$

Pour éviter de faire des calculs qui dépassent la capacité de la machine, on peut imaginer de simplifier encore cette fraction en utilisant un PGCD.

1. Ecrire une fonction renvoyant le PGCD de deux nombres entiers passés en paramètres. On utilisera l'algorithme itératif suivant pour calculer le pgcd de deux nombres a et b :

```
{
  x <- a ;
  y <- b ;
  tant que (y>0) faire
   {
    aux <- y ;
    y <- x modulo y ;
    x <- aux ;
   }
}
```

2. Ecrire programme de calcul de C_n^p en utilisant la fonction précédente. Pour cela on simplifiera la fraction au fur et à mesure de la génération des termes. Il suffit de diviser le numérateur et le dénominateur par leur pgcd.

Exercice 2 - Tri d'un tableau

On veut trier un tableau d'entier de taille n par valeurs croissantes. On va utiliser pour cela la méthode du "tri à bulle" :
En démarrant successivement au début du tableau, on compare deux cases successives du tableau et on les échange si elles ne sont pas déjà dans l'ordre. Après chaque passage on est donc sûr que la plus grande valeur est mise en bout de tableau.
Exemple :

tableau initial

12	6	3	1	9	10	15	4

au premier passage, par comparaisons successives, on doit obtenir le tableau suivant :

6	3	1	9	10	12	4	15

et on recommence en s'arrêtant à l'avant-dernière case (puisque le plus grand est maintenant à la fin) et ainsi de suite.

1. Écrire le programme qui demande à l'utilisateur la dimension (≤ 50) et les valeurs d'un tableau puis qui réalise le tri à bulle. Afficher ensuite le tableau trié.
2. On peut améliorer les performances de cet algorithme en arrêtant l'algorithme dès que le tableau est trié (avant d'arriver à la fin du processus). Comment peut-on tester que le tableau est trié au cours de l'algorithme ?
   Ecrire cette nouvelle version.
3. Sur quel genre de tableau cet algorithme va-t-il être le moins rapide ?

Exercice 3 - Recherche d'une valeur par dichotomie

On considère un tableau T de taille n, contenant des **entiers ordonnés** :

$$\forall i, j, (1 \leq i \leq j \leq n) \quad T[i] \leq T[j]$$

On veut retrouver un entier p contenu dans le tableau, en appliquant la méthode de recherche par dichotomie :

on coupe l'intervalle de recherche $[a, b]$ en deux et à l'indice k correspondant on regarde :
- si $T[k] = p$ on retourne k
- si $T[k] < p$ on recommence en prenant comme intervalle de recherche [k,b]
- si $p < T[k]$ on recommence en prenant comme intervalle de recherche [a,k]

1. Ecrire un programme qui effectue une recherche par dichotomie d'un entier appartenant à un tableau, en affichant l'indice k_p pour lequel $T[k_p] = p$ (on utilisera le programme précédent pour la saisie et le tri préalable du tableau).
   Remarque : on suppose que la valeur recherchée est dans le tableau (pas de vérification).
2. Ecrire un programme qui effectue une recherche par dichotomie d'un entier dans un tableau, et qui, si l'entier n'appartient pas au tableau, trouve l'indice i du tableau où on devrait insérer l'entier pour le laisser ordonné ($T[i-1] < p \wedge T[i] > p$), et l'insère.

Exercice 4 - Calcul d'un plateau

Un plateau est une suite d'éléments d'un tableau qui ont la même valeur.
On va chercher le plateau de taille maximum dans un tableau donné, donné par son indice de départ et sa longueur.

Par exemple, avec le tableau suivant :

0	3	4	4	2	14	5	5	5	5	3	3	3	0

le plateau maximal a une longueur 4, et il commence à l'indice 7.

1. Écrire un programme qui calcule le plateau maximal (et l'indice de départ de ce plateau) dans un tableau (exercice vu en TD).
2. Améliorer cet algorihme pour diminuer le nombre d'opérations effectuées.

Exercice 5 - Séquence de valeurs

Une séquence d'un tableau est une suite d'éléments consécutifs.
On va réaliser un programme qui, étant donné un entier N, va déterminer la position d'une séquence dont la somme des éléments vaut N. Cette position sera donnée par le couple des indices de début et de fin de la séquence.

Soit (d,f) ce couple et T le tableau considéré. On a :

$$\sum_{i=d}^{i=f} T[i] = N$$

On supposera que cette séquence existe et est unique.

Par exemple, dans le tableau suivant :

0	3	4	4	2	14	5	5	5	5	3	3	3	0

Il y a une et une seule séquence de somme 19, de départ 6 et de fin 7.