Université de Perpignan — Année 2003-2004
Licence 2e année Informatique — S. Graillat

Programmation impérative

TD n° 4 : Gestion des tableaux et tris

1 Encore un peu d'algorithmique

Exercice 1. Écrire une procédure `AIGUILLAGE`, qui selon le nom de ville qui lui est proposé, appelle l'une des procédures `GARE_DE_L_EST`, `GARE_DU_NORD`, `GARE_ST_LAZARE`, `GARE_MONTPARNASSE`, `GARE_DE_LYON` ou `GARE_D_AUSTERLITZ` ; le type VILLE est ainsi défini :

```
type VILLE is (AMIENS, BASTIA, BEAUVAIS, BESANCON, BORDEAUX,
BREST, CAEN, DREUX, DUNKERQUE, GRENOBLE, LILLE, LYON, MARSEILLE,
METZ, MONTPELLIER, NANCY, NANTES, NICE, PARIS, PERPIGNAN, POITIER,
RENNES, ROUEN, STRASBOURG, TOULON, TOULOUSE);
```

Est-il possible de réorganiser le type `VILLE` pour simplifier le programme ?

Exercice 2. Écrire un programme pour calculer la somme des N ($N \geq 0$) premiers entiers : une procédure puis une fonction. De même pour le produit.

Exercice 3. Écrire un programme calculant le $PPCM$ de deux entiers strictement positifs.

Exercice 4. Proposer deux fonctions pour calculer X^N ($N \geq 0$) sans utiliser l'opérateur "**".

Exercice 5. Écrire une fonction `LOG2` calculant le logarithme entier de l'entier $N > 0$ en base 2.

Exercice 6. Écrire un programme calculant le n-ième terme de la suite de Fibonacci définie par

$$\begin{cases} u_0 = 0, \quad u_1 = 1, \\ u_{n+1} = u_n + u_{n-1}. \end{cases}$$

2 Les tableaux

Exercice 7 (Un problème de recherche). On considère le problème de recherche suivant :

Entrée : Une séquence de n nombres $T = [a_1, a_2, \ldots, a_n]$ et une valeur v.

Sortie : Un indice i tel que $v = T[i]$ et la valeur spéciale NIL si v n'appartient pas à T.

Écrire un programme qui parcourt la séquence à la recherche de v.

Exercice 8 (Tri par selection). L'un des algorithmes de tri les plus simples procède de la manière suivante : on commence par rechercher l'élément de plus petit valeur du tableau pour l'échanger avec celui en première position, puis on recherche l'élément ayant la deuxième plus petite valeur pour l'échanger avec celui en deuxième position et l'on continue ainsi jusqu'à ce que le tableau soit entièrement trié. Cette méthode porte le nom de *tri par selection* car elle procède à la sélection successive de l'élément parmi ceux restant.

La procédure TRI-SÉLECTION que nous voulons écrire prend comme paramètre un tableau $T[1, \ldots, n]$ qui contient une séquence d'entiers, de longueur n.

1. Écrire le fonctionnement de l'algorithme sur l'exemple $T = [5, 2, 4, 6, 1, 3]$.

2. Progammer l'algorithme.

Exercice 9 (Tri par insertion). Le tri par insertion s'inspire de la manière dont la plupart des gens trient une poignée de cartes, au bridge ou au tarot. On commence avec une main gauche vide et les cartes face contre table. On retire ensuite du paquet une carte à la fois, pour l'insérer à sa bonne place dans la main gauche. Pour trouver cette bonne place, on la compare avec chacune des cartes déjà présentes dans la main gauche, de droite à gauche.

La procédure TRI-INSERTION que nous voulons écrire prend comme paramètre un tableau $T[1, \ldots, n]$ qui contient une séquence d'entiers, de longueur n. Les nombres de l'entrée seront triés sur place : ils sont réorganisés à l'intérieur du tableau.

1. Écrire le fonctionnement de l'algorithme sur l'exemple $T = [5, 2, 4, 6, 1, 3]$.

2. Programmer l'algorithme.