Université de Perpignan
Licence 2e année Informatique

Année 2003-2004

S. Graillat [1]

Programmation impérative

TD n° 1 : Introduction à l'algorithmique

Le but de ce TP est d'étudier les différentes structures de boucles et structures conditionnelles. Nous n'allons pas utiliser un langage de programmation réel, mais plutôt un "pseudo-langage" contenant les instructions suivantes :

- :=, l'instruction d'affectation ;
- *si ... alors ... sinon* ;
- *tant que ... faire ...* ;
- *répéter ... jusqu'à ...* ;
- *pour ... allant de ... à ... faire ....*

1 Calcul du milieu de plusieurs entiers

1.1 Calcul avec trois entiers

La fonction `milieu` est un fonction retournant, parmi un ensemble ordonné de $2n+1$ valeurs, celle qui est l'élément milieu. Par exemple, pour $n = 1$, `milieu3(8,3,5)` vaut `5`.

1. Programmer la fonction `milieu3` en utilisant la structure *si ... alors ... sinon*.

2. Programmer maintenant la fonction `milieu3` en utilisant les fonctions `min2` (minimun de deux entiers) et `max3` (maximum de trois entiers).

3. Quelle est, selon vous, la plus facile à programmer et la plus simple à comprendre. Justifier votre réponse.

1.2 Calcul avec cinq entiers

On va essayer de la même façon de programmer la fonction `milieu5`.

1. Essayer de programmer la fonction `milieu5` en utilisant la structure *si ... alors ... sinon*.

2. Programmer maintenant la fonction `milieu5` en utilisant les fonctions `min3`, `max3` et `milieu3`.

3. Pensez-vous pouvoir généraliser facilement à `milieu7`. Si oui, comment.

Les exemples précédents ont montré la difficulté de programmer des algorithmes sans structures sur les données. Nous allons maintenant écrire des algorithmes avec comme structure un tableau d'entiers.

[1] graillat@univ-perp.fr,http://gala.univ-perp.fr/~graillat

2 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 algorithme de recherche qui parcourt la séquence à la recherche de v.

3 Le tri par sélection

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. Écrire l'algorithme dans notre pseudo-langage.

4 Le 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. Écrire l'algorithme dans notre pseudo-langage.