Université Paris XII
IUT de Sénart-Fontainebleau
Département Informatique
Algorithmique
1999/2000
Première année

PARTIEL 3

2 heures

Seuls les notes manuscrites et les listings portant le nom de l'étudiant (dans le programme et sur chaque page) sont permis à titre de documentation.

Exercice 1.- Écrire un programme C qui demande le nom d'un fichier texte, celui-ci étant constitué d'un certain nombre de lignes, chacune étant de la forme :

`Prénom Nom`

(avec une seule espace par ligne, celle qui sépare le premier composant du second), et qui affiche le contenu du fichier à l'écran sous la forme :

`Nom Initiale.`

où `Initiale` est, évidemment, la première lettre du prénom.

Exercice 2.- Considérons les déclarations suivantes :

```
typedef struct employe
        {
        int age;
        float salaire;
        } * PEMP;

PEMP compagnie[100];
```

- 1^o) Écrire une fonction `update` ayant quatre paramètres – un tableau de PEMP, un entier d'index, un entier indiquant l'âge, et un réel indiquant le salaire – et qui crée un nouvel élément à la position de l'index.

Par exemple l'appel :

```
update(compagnie, 20, 39, 15000.0f)
```

crée un nouvel employé à la position 20 de 39 ans et ayant un salaire de 15 000 F par mois.

- 2^o) Écrire une fonction `total` ayant deux paramètres – un tableau de PEMP, indiquant la compagnie et un entier, indiquant le nombre d'employé – et qui renvoie un réel indiquant la somme des salaires des employés de plus de 50 ans.

[On suppose que la mise à jour a été réalisée pour tous les éléments.]

- 3^o) Écrire un programme C complet permettant de tester ces deux fonctions.

Exercice 3.- Considérons un fichier, appelé TEST, dont chaque ligne est de la forme suivante :

```
entier espace suite-de-caractères
```

où l'entier, inférieur à 75, représente le nombre de caractères de la suite-de-caractères (fin de ligne non comprise). On a par exemple :

```
12 Bonjour vous
6 Ah non
0
27 La ligne ci-dessus est vide
```

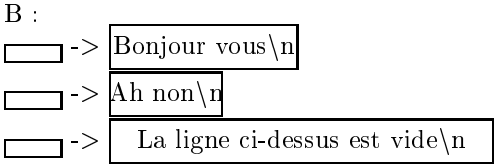
Écrire un programme C :

- qui demande le nom d'un tel fichier, possédant moins de 100 lignes non vides,
- qui utilise un tableau B de pointeurs sur des chaînes de caractères pour stocker les N lignes non vides,
- qui lit le fichier et qui, pour chaque ligne non vide :
  + alloue de la mémoire pour les caractères de la ligne, y compris le caractère de fin de ligne,
  + stocke l'adresse de ce bloc à la position adéquate de B,
  + stocke les caractères de la ligne dans le bloc de mémoire,
- qui affiche les lignes dans l'ordre inverse du fichier.

[Dans le cas du fichier ci-dessus, on aura :

N = 3

B :



L'affichage sera :

La ligne ci-dessus est vide
Ah non
Bonjour vous
]