Algorithmes et Structures de Données
Mardi 14 Janvier 2014

Durée 3h – Cours et TD **NON** autorisés

1. Récursivité – 5 pts

L'algorithme de Lucas permet de calculer la fonction x^n pour x réel positif ($x \neq 0$) et n entier positif ou nul en utilisant la définition suivante :

$x^n = 1$ pour n=0

$x^n = (x^{n/2})*(x^{n/2})$ pour n pair

$x^n = x^{n-1}*x$ pour n impair

1.1. Ecrire cette fonction `puiss(x,n)` en pseudo-langage. Ne pas oublier de numéroter les appels récursifs.

1.2. Simuler la pile sur l'appel `{@0} puiss(3,4)` dans le programme principal.

2. Liste chainée - 5 pts

Soit `t` un tableau de `n` entiers trié dans l'ordre croissant dont les valeurs peuvent être répétées. On veut écrire une fonction qui transforme ce tableau en une liste chainée `l` triée dans l'ordre croissant dont les valeurs sont uniques (on ne garde qu'une seule occurrence de chaque valeur de t).

2.1. Expliquer en français le principe de la fonction `transforme(t,n)`.

2.2. Ecrire en pseudo-langage la fonction `transforme(t,n)`.

3. Fichier texte et tableau de listes – 5 pts

Soit un fichier texte. On veut répartir les mots de ce fichier dans des listes, chaque liste contenant les mots de même longueur. La taille des mots varie de 2 à 20. Un mot peut apparaitre plusieurs fois dans la liste voulue. Les listes ne sont pas triées. Pour cela, on définit les types `liste` et `tabliste` de la façon suivante :

```
Type liste = ^cellule
     cellule = Enregistrement
                  mot : chaine
                  suiv : liste
               FinEnregistrement
     tabliste = tableau [2..20] de liste
```

3.1. Expliquer en français la méthode utilisée. Faire un dessin avec le texte suivant.

```
Ceci est un court exemple de texte
qui permet de comprendre la structure de données.
```

3.2. Ecrire en pseudo-langage une procédure qui utilise le fichier texte en entrée pour remplir cette structure de données. Vous pourrez utiliser la procédure `litmot(E c:chaine, E/S i:entier, S m:chaine)` vue en TD.

4. Tri des mécanographes – 5 pts

On veut trier des mots d'exactement N lettres dans l'ordre lexicographique suivant l'algorithme des mécanographes. Le principe est le suivant :

Si on sait trier des mots suivant leur $k^{ième}$ lettre ($1 \leq k \leq N$), en laissant les mots dans leur ordre d'origine en cas d'égalité, on peut effectuer le tri complet des mots. Il suffit en effet de faire varier k de N à 1 en utilisant des files de mots. Pour effectuer le tri suivant la $k^{ième}$ lettre, on éclate les mots sur 26 files de mots (contenues dans un tableau de files) selon cette lettre puis on concatène les files en une seule en partant de la file 1 à la file 26.

Ecrire en pseudo-langage l'algorithme complet de tri, appelé tri des mécanographes. Vous utiliserez le TAD file avec ses opérations usuelles plus l'opération `concaténer(f1,f2)`, qui renvoie une file formée de f1 puis f2.

Remarque : la place d'une lettre `let` dans l'alphabet est donnée par (`ord(let)-ord('a')+1)`.