PROGRAMMATION 1

Contrôle Finale + Rattrapage
2010-2011

WWW.FACEBOOK.COM/SMIA.STUDIES

SMIA
STUDIES

Université Mohammed V-Agdal — Année Universitaire 2010/2011
Faculté des Sciences de Rabat — Semestre 3 (SMI)
Département d'Informatique

Matière Langage C

[Contrôle Final– Janvier 2011 – (Durée : 1h30mn)]

N.B. Les réponses doivent être claires et précises. Les commentaires sont appréciés.

Exercice 1 (15 points)

Dans cet exercice, on considère les nombres entiers X constitués de 4 chiffres. On exclut les nombres formés d'un même chiffre (1111, 2222,...).

L'algorithme de Kaprekar consiste à associer à chaque nombre X un autre nombre K(X) généré de la façon suivante :

- On considère les **chiffres** de X. On **forme** le nombre **X1** en arrangeant **ces chiffres** dans **l'ordre croissant** et le nombre **X2** en les arrangeant dans **l'ordre décroissant**.
- On pose **K(X) = X2-X1**
- On **répète** ensuite le **processus** avec **K(X)**

L'algorithme de **Kaprekar** produit au final un **nombre constant** c.-à-d. K(X) = X. Ce nombre est atteint au maximum avec **8** itérations.

Exemple : En partant du nombre **X = 5294**, on obtient :

Itération 1 : X=5294 ⟶ (X1=2459 et X2=9542) ⟶ K(X) = X2-X1 = 7083
Itération 2 : X=7083 ⟶ (X1=378 et X2=8730) ⟶ K(X) = X2-X1 = 8352
Itération 3 : X=8352 ⟶ (X1=2358 et X2=8532) ⟶ K(X) = X2-X1 = 6174
Itération 4 : X=6174 ⟶ (X1=1467 et X2=7641) ⟶ K(X) = X2-X1 = 6174

1) Ecrire une fonction, nommée **Decomposition**, qui mémorise, dans l'ordre décroissant, les 4 chiffres de X dans un tableau T1.

Exemple : si X = 5294 alors T1 = (9 , 5 , 4 , 2) ;

Les paramètres de la fonction sont le tableau T1 et l'entier X (Le tableau T1 est par défaut passé par adresse ; On prévoit un passage par valeur de l'entier X).

2) Ecrire une fonction, nommée **Calcul_X1_X2**, qui calcule les nombres X1 et X2 à partir de X (Utiliser la fonction **Decomposition**).

Exemple : si **X = 5294** alors **X1 = 2459** et **X2 = 9542** ;

Les paramètres de la fonction sont les entiers X, X1 et X2 (On prévoit un passage par valeur de l'entier X et par adresse pour les entiers X1 et X2).

(Indication : déclarer un tableau local T1 et utiliser le pour calculer X1 et X2).

3) Ecrire un programme en C qui :

a. Saisie un entier X au clavier

b. Puis remplit un tableau T2 par les nombres générés par **l'algorithme de Kaprekar** en partant de X (Utiliser la fonction **Calcul_X1_X2**). Le dernier nombre à ajouter dans T2 sera le **nombre constant**. L'espace mémoire du tableau T2 sera alloué **dynamiquement**.

Exemple : si **X = 5294** alors **T2 = (5294 , 7083 , 8352 , 6174)**

c. Enfin Affiche le Tableau T2.

Exercice 2 (5 points)

Une chaine de caractère w est un carré s'il existe une chaine u telle que w = uu (par exemple "chercher" et "bonbon" sont des carrés).

Ecrire une fonction, nommée **Estcarre,** qui retourne 1 si la chaine passé en paramètre est un carré, 0 sinon.

(je rappelle que la fonction **int strlen(char *ch)** retourne le nombre de caractère d'une chaine ch sans compter le caractère de fin de chaine)

Université Mohammed V-Agdal — Année Universitaire 2010/2011

Faculté des Sciences de Rabat — Semestre 3 (SMI)

Département d'Informatique

Matière Langage C

[Contrôle du Rattrapage– Janvier 2011 - (Durée : 1h30mn)]

N.B. Les réponses doivent être claires et précises. Les commentaires sont appréciés.

Exercice 1 (8 points)

Question 1 (4 points):

Donner la suite des affichages produits par ce programme (Expliquer).

```
#include <stdio.h>
#include <stdlib.h>
int z = 0 ;
void affiche(int x , int y)
{
  printf("\nx = %d\ty = %d\tz = %d", x , y , z ) ;
}
void transforme(int *x, int y)
{
     z =  (*x>y) ? ++y : (*x)++ ;
}
void main()
{
  int  x = 2 , y = 2 ;

          transforme( &x , y ) ;
          affiche( x , y ) ;
          transforme( &x , y ) ;
          affiche( x , y ) ;
}
```


Question 2 (4 points):

On définit le nombre de combinaison de **P** éléments parmi **N** par : $C_N^P = \frac{N!}{P!(N-P)!}$

On démontre que $C_N^P = C_{N-1}^{P-1} + C_{N-1}^P$ et on remarque que $C_N^0 = 1$ et $C_N^N = 1$

Ecrire une fonction récursive qui retourne le nombre de combinaison de P éléments parmi N. Les deux entiers P et N sont les paramètres de la fonction.

Exercice 2 (12 points)

Soit A une matrice d'**entiers courts** de M lignes et N colonnes (au maximum **20** lignes et **30** colonnes). Les éléments de A qui sont à la fois un **maximum sur leur ligne** et un **minimum sur leur colonne**, sont appelés des **points-cols**.

Exemple: **Pour** la matrice $A=\begin{pmatrix}1&2&3\\4&5&6\\7&8&9\end{pmatrix}$, l'élément A[0@[2] est un **point-col** (**maximum** sur la 1ère ligne et Minimum sur la 3ème colonne).

Le problème consiste à déterminer les **points-cols** d'une matrice A.

Méthode : **Construire** deux matrices d'aide **Max** et **Min** de même dimensions que A, telles que:

$$Max[i][j] = \begin{cases} 1 & \text{si } A[i][j] \text{ est un maximum sur la ligne } i \\ 0 & \text{sinon} \end{cases}$$

$$Min[i][j] = \begin{cases} 1 & \text{si } A[i][j] \text{ est un minimum sur la colonne } j \\ 0 & \text{sinon} \end{cases}$$

1) Ecrire une fonction, nommée ConstructionMatricesMaxMin, qui construit les matrices Max et Min à partir de la matrice A. Les paramètres de la fonction sont la matrice A, le nombre de ligne M de A, le nombre de colonne N de A, la matrice Max et la matrice Min. L'entête de la fonction est :

void ConstructionMatricesMaxMin(short A[][30], short M , short N, short Max[][30], short Min[][30])

2) Ecrire un programme en C qui :

- Saisie la matrice A au clavier,
- Puis construit les matrices Max et Min à partir de la matrice A. (utiliser la fonction ConstructionMatricesMaxMin),
- Enfin affiche les positions (ligne ; colonne) de tous les **points-cols** trouvés (Utiliser les matrices **Max** et **Min**).