Université Mohamed Premier **Filière SMI**
Faculté des Sciences, Oujda **Semestre 4**
Département d'Informatique **Année 2016/2017**

Examen de Structures de données
Session Ordinaire – Juillet 2017

Exercice 1

1. Donner quatre exemples de structures de données linéaires.
2. Expliquer pourquoi on ne peut pas avoir dans une structure C nommée « X », un champ de type « X ». Comment résoudre ce problème en C ?
3. Donner la définition d'une pile et celle d'une file.
4. Etant donné une liste L doublement chaînée circulaire et triée. Donner la complexité dans le pire des cas de chacun des algorithmes suivants :
    a. Le calcul du plus petit élément de L.
    b. Le calcul du plus grand élément de L.
    c. La suppression d'un élément de L d'adresse donnée.
    d. La recherche d'un élément x dans L.

Exercice 2

Soit les déclarations C suivantes :

```
typedef struct Node
{
  int data;
  struct Node* next;
} Node;
typedef Node* List;
List head = NULL;
```

1. Écrire un programme principal qui permet d'insérer les éléments 0, 1, 4, 9, 16 et 25 (dans cet ordre), dans la liste **head**.
2. Écrire une procédure qui affiche dans l'ordre, les éléments impairs de la liste **head**.
3. Écrire une procédure récursive qui supprime tous les éléments pairs de la liste **head**.
4. Écrire une procédure qui teste si la liste **head** contient un élément donné « x ».
5. Écrire une procédure qui permet de transformer la liste **head** en une liste circulaire.
6. Sachant maintenant que la liste **head** est circulaire, écrire une procédure qui affiche sa longueur (c'est-à-dire le nombre de ses éléments).

Exercice 3

1. Supposons qu'avec une pile *S* vide au départ, on a effectué au total : 25 opérations « empiler », 12 opérations « sommet », et 10 opérations « dépiler », dont 3 ont généré une erreur « Pile Vide ». Quelle est la taille actuelle de la pile *S* ?

2. Écrire une fonction qui prend deux piles ordonnées d'entiers A et B (minimum en sommet) et qui crée une seule pile ordonnée (minimum en sommet) en utilisant uniquement les opérations sur une pile (**pop**, **push**, **isEmpty** et **top**). Aucune autre structure de données n'est permise.

Exercice 4

Etant donné une file Q, écrire une fonction qui retourne le plus petit élément de Q (le minimum). Seules les opérations suivantes sur une file sont permises : **eneque**, **dequeue** et **size**. Aucune autre structure de données n'est permise. La file Q doit rester intact après la recherche du minimum.