Devoir Maison 3

M2 AIGEME, année 2008-2009

A rendre pour le 12/02

Exercice 1 : Un peu de géographie (mais vraiment un peu)

1. Définir un type `Pays` qui contient les informations suivantes : le nom du pays, la liste des pays limitrophes et si le pays a un accès à la mer ou non. Pour cela, choisir pour chacun des champs suivants un type et justifier ce choix :

```
Type Pays={  
    nom : ?;  
    pays_voisins : ?;  
    acces_mer : ?;  
}
```

2. Déclarer une variable de ce type et l'initialiser avec des valeurs de votre choix.
3. Ecrire une fonction qui dit si deux pays sont limitrophes ou non :

```
Fonction voisins(p1,p2 : Pays) : Booleen
```

4. Etant donné un pays, on se pose les questions suivantes :
 - a-t-il accès à la mer ? (accès direct)
 - a-t-il un voisin qui a accès à la mer ? (accès indirect)

Ecrire deux fonctions qui répondent à ces questions :

```
Fonction acces_direct(p : Pays) : Booleen  
Fonction acces_indirect(p : Pays) : Booleen
```

Exercice 2 : File d'attente

On souhaite modéliser une file d'attente au rayon poissonnerie d'un supermarché. Le principe est le suivant : chaque client se voit attribuer un numéro d'ordre de passage à son arrivée au stand. Les clients sont servis par ordre de numéro croissant (le client avec le plus petit numéro est servi en premier). Les numéros sont attribués par ordre croissant : le premier client de la journée reçoit le numéro 1, le second le 2, etc ... de sorte que le principe "premier arrivé, premier servi" est respecté.

1. Définir un type récursif `Client` permettant de construire une file d'attente. On veut qu'une variable de type `Client` possède (entre autres) un champ appelé `numero`. On définit ensuite le type `FileAttente` comme pointeur vers une variable de type `Client` :

```
Type FileAttente : *Client;
```

2. Dessiner une file d'attente avec trois clients ayant comme numéros respectifs 32, 33 et 35.
3. Définir une fonction qui supprime de la file d'attente le client avec le plus petit numéro (le client avec le plus petit numéro a été servi et quitte la file d'attente) :

```
Fonction client_servi(f : FileAttente)1 : FileAttente
```

Illustrer sur un dessin ce que fait cette fonction (en reprenant l'exemple de la question 2).

4. Définir une fonction qui ajoute un client à une file d'attente (un client prend sa place dans la file d'attente). Attention ce nouveau client n'a pas forcément un numéro plus grand que les autres clients de la file d'attente (on peut imaginer un client quittant la file d'attente pour aller faire d'autres courses et revenant dans la file) :

```
Fonction ajout_client(c : Client; f : FileAttente) : FileAttente
```

Illustrer sur un dessin l'ajout d'un nouveau client de numéro 36 et l'ajout d'un client de numéro 34.

5. Définir une fonction qui supprime un client de la liste d'attente (pas forcément celui avec le plus petit numéro) :

```
Fonction depart_client(c : Client; f : FileAttente) : FileAttente
```

Illustrer sur un dessin le départ du client de numéro 35 de l'exemple de la question 2.

6. On imagine le scénario suivant :

- trois clients (Ulrich, Marc et Franck) arrivent successivement dans la file d'attente
- Ulrich est servi pendant que deux autres clients (Mathieu et Benoît) arrivent
- Benoît voit qu'il y a du monde devant lui et va faire un tour
- Marc est servi pendant qu'un autre client (Yoann) arrive
- Benoît revient dans la file d'attente et Yoann part se promener pendant que Franck est servi
- Mathieu est servi et Yoann revient dans la file
- Benoît puis Yoann sont servis

Compléter cette suite d'instructions pour qu'elle corresponde au scénario :

```
Var ulrich, marc, franck, mathieu, benoit, yoann : Client;  
    f : FileAttente;  
    compteur : Entier;
```

Debut

```
//au début la file d'attente est vide, le compteur est à 1
```

```
f ← NULL;
```

```
compteur ← 1;
```

```
//Ulrich arrive, prends le numéro 1 et s'installe dans la file d'attente
```

```
ulrich.numero ← compteur;
```

```
compteur ← compteur+1;
```

```
f ← ajout_client(f,ulrich);
```

...

Fin