DM 1 pour le 22 mars 2022 - 2 heures

Les programmes non commentés ou dont le fonctionnement n'est pas expliqué ne seront pas relus. Une présentation raisonnablement aérée (avec une identation convenable des structures de choix ou de répétition), ainsi que des résultats soulignés ou encadrés, sont des éléments de l'évaluation... C'est vous qui voyez...

Exercice 1 [**Démonstration d'un résultat informatiquement**] : on se propose de méontrer que : Les entiers égaux à la somme des cubes de leurs chiffres en écriture décimale, sont $0, 1, 153, 370, 371$ et 407

1. Montrer que si x est un tel entier alors $x \leqslant 10^4$
2. Écrire une fonction, `somme_cubes`, qui prend en paramètre un entier naturel et renvoie la somme des cubes des chiffres de ce nombre (en écriture décimale).
3. Écrire une fonction qui démontre le résultat énoncé ci-dessus.

Exercice 2 [**Votre entrée chez M$**] **:** Lors des entretiens d'embauche d'une célèbre firme de Redmond (près de Seattle), les recruteurs posaient souvent aux candidats la question suivante : on vous donne un entier naturel n non nul et une liste de n entiers distincts compris entre 0 et n ; Écrivez un programme qui renvoie le nombre manquant.

1. a. On considère la fonction suivante :

```
let rec mystere x = function
          [] -> false
        | h::t -> h = x || mystere x t;;
```

   Quel est le type de cette fonction ? A quoi sert-elle ?

   b. Déterminer le nombre de comparaisons effectuées par la fonction quand on l'applique à une liste de longueur n. On distinguera le meilleur et le pire des cas (en précisant quels sont ces cas).

   c. On considère la fonction suivante :

```
let nb_manquant_1 l =
        let i = ref 0 in while mystere !i l do i := !i + 1 done;
        !i;;
```

   Que renvoie `nb_manquant_1 [4; 2; 1; 5; 0]` ? Expliquer rapidement pourquoi cette fonction répond au problème posé.

   d. Déterminer le nombre de comparaisons effectuées par la fonction en fonction de n. On distinguera le meilleur et le pire des cas.

2. a. Écrire une fonction `somme` de type `int list -> int` qui, recevant une liste d'entiers, renvoie sa somme.

   b. On considère la fonction suivante :

```
let nb_manquant_2 l =
         let n = List.length l in n*(n+1)/2 - somme l;;
```

   Expliquer pourquoi cette fonction répond au problème posé.

   c. Déterminer le nombre d'additions effectuées par la fonction `nb_manquant_2` en fonction de n.

Exercice 3 Compression RLE : La compression RLE (**R**un-**L**ength **E**ncoding) est une méthode de compression de données très simple. On l'utilise lorsque l'objet à compresser contient de longues séries de répétitions d'un même élément (image en noir et blanc par exemple).
L'idée est de remplacer chaque série d'un même élément par le couple formé de la longueur de la série et de la valeur de cet élément. On travaillera ici avec des listes d'entiers. Par exemple, la liste `[1; 1; 1; 1; 2; 2; 2; 2; 2; 1; 1; 1; 4]` est form´ee successivement de quatre `1`, de cinq `2`, de trois `1` et d'un `4`. La liste compressée sera alors `[4; 1; 5; 2; 3; 1; 1; 4]`.

1. Ecrire une fonction `compresser` de type `int list -> int list` qui prend comme argument une liste d'entiers et renvoie la liste obtenue par la méthode précédente.

```
# compresser [1; 1; 1; 1; 2; 2; 2; 2; 2; 1; 1; 1; 4];;
- : int list = [4; 1; 5; 2; 3; 1; 1; 4]
```

2. Ecrire une fonction `decompresser` de type `int list -> int list` qui prend comme argument la liste compressée et renvoie la liste d'origine.

```
# decompresser [4; 1; 5; 2; 3; 1; 1; 4];;
- : int list = [1; 1; 1; 1; 2; 2; 2; 2; 2; 1; 1; 1; 4]
```

Exercice 4 [Un escalier 1/2] : Vous êtes au pied d'un escalier. Sur chacune des marches de celui-ci est inscrit un entier relatif. Vous pouvez grimper d'une ou de deux marches à chaque fois. On vous demande d'atteindre la dernière marche en maximisant la somme des nombres inscrits sur les marches que vous avez empruntées. Par exemple, si les nombres inscrits sur les marches sont 5, -3, -6, 4, 3 et -2, la somme maximale possible est 5 + (-3) + 4 + 3 + (-2) = 7.

1. Quelle est la somme maximale possible lorsque
   a. tous les nombres sont positifs ?
   b. les nombres sont 5, -3, -6, -10, 3 et -2 ?
2. Écrire une fonction escalier qui permet d'obtenir la somme maximale. Ainsi :

```
# escalier [5; -3; -6; 4; 3; -2];;
- : int = 7
```