Devoir Non Surveillé 1

À rendre le 4 novembre 2024

INFORMATIQUE MP2I

Tournez la page S.V.P.

Analyse d'une fonction

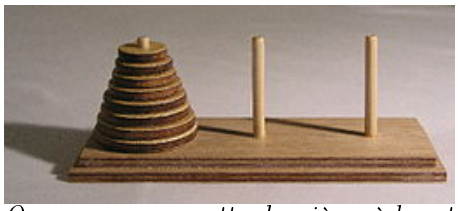
☐ 1 – On donne la fonction suivante :

```
let rec fast_fib n =
  if n = 0 || n = 1 then
    1
  else
    fast_fib (n-2) + fast_fib (n-3);;
```

Que fait cette fonction ? étudier sa terminaison et sa correction.
On pourra dessiner un arbre d'appels, ou une partie d'arbre d'appels pour des valeurs bien choisies.

Tours de Hanoi

Les tours de Hanoï (originellement, la tour d'Hanoï) sont un jeu de réflexion imaginé par le mathématicien français Édouard Lucas, et consistant à déplacer des disques de diamètres différents d'une tour de « départ » à une tour d'« arrivée » en passant par une tour « intermédiaire », et ceci en un minimum de coups, tout en respectant les règles suivantes :



— *on ne peut déplacer plus d'un disque à la fois ;*
— *on ne peut placer un disque que sur un autre disque plus grand que lui ou sur un emplacement vide.*

On suppose que cette dernière règle est également respectée dans la configuration de départ. Wikipédia (Tours de Hanoï)
On dispose de 3 tours que l'on numérotera 0, 1, et 2.

☐ 2 – En remarquant que la somme des numéros des tours fait 3, étant donné 2 tours de numéros a et b, quelle expression désigne la 3e tour ?

Pour désigner le déplacement d'un disque depuis la tour a vers la tour b, on écrit `a->b`.

☐ 3 – On suppose que la position de départ contient 4 disques dans la tour 0 et que les tours 1 et 2 sont vides. Donner la position après la séquence de coups suivantes :

0–>1
0–>2
1–>0

Si on veut déplacer une pyramide depuis la tour 0 vers la tour 1, on peut remarquer que le plus grand disque (disons, le n-ième) doit être déplacé de la tour 0 vers la tour 2. Pour effectuer cette opération, il faut d'abord que les $n-1$ disques précédents se retrouvent sur la tour 1. On peut alors se ramener au problème de déplacer une pyramide de taille $n-1$, mais en échangeant les rôles des tours.

☐ 4 – Écrire une fonction `hanoi : int -> unit` qui prend en entrée un entier n, et affiche les coups nécessaires pour déplacer une pyramide de taille n de la tour 0 vers la tour 2, les tours 1 et 2 étant initialement vide.

En s'inspirant de ce code, nous allons déterminer le nombre de coups nécessaires (notons le x_n) pour déplacer une tour de n disques.

☐ 5 – Trouver une relation de récurrence liant x_{n+1} à x_n.

☐ 6 – Calculer x_n.

Rot13

on définit les opérateurs `^+` et `+^` permettant de concaténer un caractère à la fin et au début d'une chaîne de caractères :

Concaténation d'un char à la fin d'une string

```
# let (^+) str c = str ^ String.make 1 c;;
val ( ^+ ) : string -> char -> string = <fun>
# let (+^) c str = String.make 1 c ^ str;;
val ( +^ ) : char -> string -> string = <fun>
# ("Bonjour"^+'n')^+'o';;
- : string = "Bonjourno"
# 'a'+^('b'+^"cd");;
- : string = "abcd"
```

☐ 7 – Écrire une fonction `pos_of_letter : char -> int` qui prend en entrée une lettre minuscule et qui renvoie sa position dans l'alphabet. 0 pour `'a'`, et 25 pour `'z'`. On vérifiera que la précondition (le caractère donné représente une lettre minuscule) est bien vérifiée.

☐ 8 – Écrire une fonction `letter_of_pos : int -> char` qui effectue la transformation inverse. On vérifiera aussi la précondition nécessaire.

☐ 9 – Écrire une fonction `rot13_char : char -> char` qui prend en entrée une lettre minuscule et qui renvoie la lettre décalée de 13 positions dans l'alphabet, en revenant au début si nécessaire. `rot13 'a'` vaut `'k'` par exemple, et `rot13 'u'` vaut `'h'`.

☐ 10 – Écrire une fonction `rot13_string : string -> string` qui décale toutes les lettres minuscules apparaissant dans une chaîne de caractères de 13 positions.

On écrit le code suivant dans le fichier `rot13.ml` que l'on compile avec la commande

```
ocamlopt rot13.ml -o rot13
```

rot13.ml

```
(* définition de rot13_string de la question précédente *)

let rec rot13_input () =
  try
    print_endline (rot13_string (read_line ()));
    rot13_input ()
  with
    End_of_file -> ();;

let () = rot13_input ();;
```

☐ 11 – On exécute la commande

```
./rot13 | ./rot13 > montexte.txt
```

Et on saisi au clavier "helloworld".
Que se passe-t-il ?

Numérotation Shadok

Les shadoks ne possèdent que 4 mots : "GA", "BU", "ZO" et "MEU".
Quand il n'y a pas de Shadoks, on dit "GA". Quand il y a un Shadok de plus, on dit "BU". Quand il y a encore un Shadok on dit "ZO". Et quand il y a encore un autre, on dit "MEU".
Si je met un Shadok en plus, évidemment je n'ai plus assez de mots pour les compter, alors c'est très simple : on les jette dans une poubelle et je dis que j'ai "BU" poubelles, et pour ne pas confondre avec le "BU", je dis qu'il n'y a pas de Shadok à côté de la poubelle et j'écris "BU" "GA".
Arrivé à "MEU" "MEU", si je mets un Shadok en plus il me faut une autre poubelle. Et comme je n'ai plus de mots pour compter les poubelles, je m'en débarasse en les jetant dans une grande poubelle j'écris "BU" grandes poubelles avec pas de poubelles et pas de Shadoks à côté : "BU" "GA" "GA". et on continue.

☐ 12 – Convertir le nombre "BU" "ZO" "GA" "MEU" en décimal.

☐ 13 – Convertir le nombre 255 en numérotation Shadok.

☐ 14 – Écrire une fonction `shadok_of_int : int -> string` qui prend en entrée un entier et qui le converti en numérotation Shadok.

Egg throwing

On cherche à savoir à partir de quel étage N d'un certain immeuble les œufs se cassent. On suppose que tous les œufs se cassent à la même hauteur, et qu'un œuf non cassé peut être réutiliser.
Pour l'instant, on suppose que l'immeuble possède 100 étages.

☐ 15 – On dispose d'un seul œuf, quelle est la meilleure stratégie pour déterminer la valeur de N, et combien de lancers doit-on effectuer ?

☐ 16 – On dispose d'une infinité d'œufs, quelle est la meilleure stratégie pour déterminer la valeur de N et combien de lancers doit-on effectuer ?

☐ 17 – On dispose de 2 œufs, quelle est la meilleure stratégie pour déterminer la valeur de N et combien de lancers doit-on effectuer ?

☐ 18 – Généraliser à tout immeuble et tout nombre d'œufs, et proposer un algorithme qui résoud le problème. On pourra s'intéresser au nombre maximum d'étages qui peuvent être testés en k lancers avec n œufs.

Petit problème

☐ 19 – Une petite fille compte de 1 à 1 000 en utilisant les doigts de sa main gauche de la manière suivante : 1 pour le pouce, 2 pour l'index, 3 pour le majeur, 4 pour l'annulaire, 5 pour l'auriculaire. Puis elle change de direction : 6 pour l'annulaire, 7 pour le majeur, 8 pour l'index, et 9 pour le pouce. Puis 10 pour l'index, et ainsi de suite. Si elle continue ainsi, sur quel doigt va-t-elle s'arrêter ?

Fin de l'épreuve