Arithmétique en OCaml

Florian Bourse

Question 1. Écrire une fonction `divise : int -> int -> bool = <fun>` qui prend en argument deux entiers d et n et qui détermine si d divise n.

Question*. Écrire une fonction `premier : int -> bool = <fun>` qui prend en argument un entier n et qui détermine si n est premier.

1 Nombres rationnels

Pour représenter un nombre rationnel $\frac{a}{b}$, nous allons utiliser un couple d'entiers `(a,b)` dont le premier élément est le numérateur et le second le dénominateur.

Question 2. Écrire une fonction `pgcd : int -> int -> int = <fun>` qui prend en argument deux entiers a et b et calcule leur plus grand diviseur commun.

Question 3. Écrire une fonction `simplify : int * int -> int * int = <fun>` qui prend en argument un nombre rationnel $\frac{a}{b}$ et retourne la fraction irréductible qui lui est égal.

Question 4. Écrire les fonctions

```
add_frac : int * int -> int * int -> int * int = <fun>
mult_frac : int * int -> int * int -> int * int = <fun>
div_frac : int * int -> int * int -> int * int = <fun>
```

qui prennent en argument deux nombres rationnels $\frac{a}{b}$ et $\frac{c}{d}$ et calculent respectivement leur somme, leur produit et leur quotient.

Question 5. On considère la fonction u définie par son premier élément u_0 et la relation de récurrence suivante :

$$u_{n+1} = \frac{u_n^2}{1 + 2u_n}$$

Écrire une fonction `u : int * int -> int -> int * int = <fun>` qui prend en argument un nombre rationnel $\frac{a}{b}$ et un entier n et qui renvoie u_n sachant que $u_0 = \frac{a}{b}$. Tester cette fonction avec plusieurs valeurs et émettre une conjecture quand à la convergence de la suite u_n. Essayez de la prouver.

Question*. Écrire une fonction `approx : int * int -> int * int = <fun>` qui prend en argument un nombre rationnel $\frac{a}{b}$ et qui renvoie le couple (n, d) vérifiant les propriétés suivantes :

— n est l'arrondi à l'unité de $\frac{a}{b}$;

— si $\frac{a}{b}$ est un entier, $d = 0$;

— sinon, d est le plus grand entier tel que

$$n - \frac{1}{d} \leq \frac{a}{b} \leq n + \frac{1}{d}$$

2 Limites de la machine et efficacité du code

Question 6. Écrire une fonction `factorielle : int -> int = <fun>` qui calcule la factorielle d'un entier n. Quelles sont ses limites ? Quelles erreurs peuvent apparaître lors de l'utilisation de cette fonction. Tester votre fonction pour n valant 0, 1, -1, 50, 1 000 000.

Question 7. Voici une fonction mystère :

```ocaml
(* n doit être positif *)
let mystere n =
  let rec aux n acc =
    if n = 0 || n = 1 then
      acc
    else
      aux (n-1) (acc*n)
  in
  aux n 1;;
```

Que calcule-t-elle ? Quelles erreurs peuvent apparaître lors de l'utilisation de cette fonction ?

Question 8. Écrire une fonction `puissance_puissance_8 : int -> int = <fun>` qui prend en argument un entier n et qui calcule n^8 en utilisant seulement 3 multiplications.

Question 9. Écrire une fonction `fast_exp : int -> int -> int = <fun>` qui prend en argument deux entiers a et b et qui calcule a^b en utilisant la propriété suivante :

$$a^{2p} = (a^p)^2, \qquad a^{2p+1} = a\,(a^p)^2$$

3 Suites de Syracuse

On appelle suite de Syracuse une suite d'entiers naturels définie de la manière suivante : on part d'un nombre entier strictement positif ; s'il est pair, on le divise par 2 ; s'il est impair, on le multiplie par 3 et l'on ajoute 1.
Par exemple, à partir de 14, on construit la suite des nombres : 14, 7, 22, 11, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1, 4, 2... C'est ce qu'on appelle la suite de Syracuse du nombre 14.
La conjecture de Syracuse, ou conjecture de Collatz, est l'hypothèse selon laquelle la suite de Syracuse de n'importe quel entier strictement positif atteint 1.
On définit alors :

le temps de vol : *c'est le plus petit indice n tel que $u_n = 1$. Il est de 17 pour la suite de Syracuse 15 et de 46 pour la suite de Syracuse 127 ;*

l'altitude maximale : *c'est la valeur maximale de la suite. Elle est de 160 pour la suite de Syracuse 15 et de 4 372 pour la suite de Syracuse 127.*

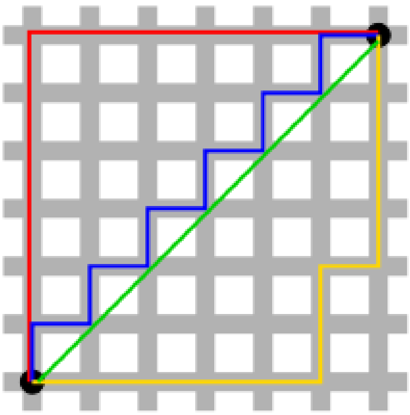
Wikipédia (Conjecture de Syracuse)

Question 10. Écrire une fonction `vol_syracuse : int -> int = <fun>` qui prend en argument un entier n et calcule le temps de vol de la suite de Syracuse de n.

Question 11. Écrire une fonction `pic_syracuse : int -> int = <fun>` qui prend en argument un entier n et calcule l'altitude maximale de la suite de Syracuse de n.

Question*. Écrire une fonction `syracuse_atteint : int -> int = <fun>` qui prend en argument un entier n et renvoie le plus petit nombre i tel que l'altitude maximale de la suite de Syracuse de i dépasse n.

4 Distance de Manhattan



La distance de Manhattan, appelée aussi taxi-distance, est la distance entre deux points parcourue par un taxi lorsqu'il se déplace dans une ville où les rues sont agencées selon un réseau ou quadrillage, à l'image de Manhattan.

Wikipédia (Distance de Manhattan)

Question 12. Écrire une fonction

```
distance : int * int -> int * int -> int = <fun>
```

qui prend en argument deux couples de coordonnées entières (x_1, y_1) et (x_2, y_2) et renvoie la distance de manhattan entre ces deux points.

Question 13. Écrire une fonction

```
chemins : int * int -> int * int -> int = <fun>
```

qui prend en argument deux couples de coordonnées entières (x_1, y_1) et (x_2, y_2) et renvoie le nombre de chemins de plus courte distance entre ces deux points.