

### Exercice 1

La notation postfixe des expressions arithmétique (appelée également notation polonaise inverse ou notation de Łukasiewicz) consiste à écrire les opérandes d'une expression avant les opérateurs. Par exemple, l'expression  $(3 \times (5 + 4) - 2)$  s'écrira `3 5 4 + × 2 -`. Aucune parenthèse n'est requise pour écrire une telle expression.

1. Écrire une fonction `str_to_lst` de signature `string -> char list` qui transforme une chaîne de caractères en une liste de caractères, en supprimant les espaces. On utilisera une référence de liste.
2. En déduire une fonction `postfixe` de signature `string -> int` telle que `postfixe s` renvoie l'évaluation de l'expression `s` écrite en notation postfixe. On supposera que `s` ne contient que des chiffres et des symboles d'opérateurs (`+`, `-`, `*`, `/`) séparés par des espaces. On utilisera une structure de pile codée par une simple liste, ainsi que les fonctions suivantes permettant d'associer un caractère d'opérateur à l'opération associée ou un caractère de chiffre au chiffre associé :

```
let ops = function
  | '+' -> ( + )
  | '-' -> ( - )
  | '*' -> ( * )
  | '/' -> ( / );;

# ops '*' 2 3;;
- : int = 6

let chiffre x = int_of_char x - 48;;

# chiffre '5';;
- : int = 5
```

### Exercice 2

On considère un message formé des lettres de l'alphabet encodé par un entier selon le principe suivant : une lettre est transformé en son rang dans l'alphabet ( $A \rightarrow 1, \dots, Z \rightarrow 26$ ). Ainsi, un message sera codé par une `int list` ne contenant que des chiffres (donc des entiers entre 0 et 9).

1. Écrire une fonction `compte` qui compte le nombre de façons différentes de décoder le message. Par exemple, `[2;6;1;0;3;7;1;1]` peut se décoder de quatre façons différentes : `ZJCGK`, `ZJCGAA`, `BFJCGK` et `BFJCGAA`.
2. Lors de la transmission du message codé, certains chiffres sont transformés en `*`. On s'intéresse à nouveau au nombre de façons de décoder le message. Par exemple, `"2*"` peut se décoder comme `"BA"`, `"BB"`,  $\dots$ , `"BI"`, `"T"`, `"U"`, `"V"`,  $\dots$ , `"Z"`, soit 16 possibilités. Pour éviter les problèmes de types, on considérera que les `*` sont codées par des `-1`. Écrire une fonction `comptebis` qui compte le nombre de façons différentes de décoder le message.