Calculatrice en notation polonaise

Florian Bourse

1 Calculer une expression numérique

Le but de cette section est de construire une mini calculatrice. Pour simplifier le problème et éviter les parenthèses, nous allons utiliser la *notation polonaise*, ou *notation préfixée*, proposée en 1924 par Jan Łukasiewicz.
Dans la notation polonaise, les opérateurs sont écrits avant les opérandes. Par exemple $8 + 3$ s'écrit `+ 8 3`. On différencie donc $3 + 4 \times 2$ qui devient `+ 3 * 4 2` de $(3 + 4) \times 2$ qui devient `* + 3 4 2`.
Pour mieux comprendre cette notation, il faut visualiser une expression numérique comme un arbre :

Cette notation rappelle la langue française :

— la somme de 3 et du produit de 4 par 2 ;
— le produit de la somme de 3 et de 4 par 2 ;

On veut faire un programme qui lit sur l'entrée standard des expressions en notation polonaise, qui calcule leurs résultats et les affiche sur la sortie standard. Le programme peut renvoyer une exception si l'expression est mal formée.

Propositions de checkpoints

1. `prochain_espace : string -> int -> int` qui renvoie la position du prochain espace à partir de l'indice donné, ou la taille de la chaîne si il n'y a pas d'espaces à partir de cette position.
2. `calcule_aux : string -> int -> int * int` qui calcule la sous expression située à la position donnée et renvoie le résultat ainsi que la position à laquelle se termine cette expression. On pourra commencer par n'implémenter que les expressions sans opérateurs, puis ajouter les opérateurs petit à petit.
3. `listen : unit -> unit` qui appelle la fonction de calcule en boucle pour écouter l'utilisateur et lui répondre.
4. Gestion des exceptions.

2 Pour aller plus loin

2.1 Générer des expressions donnant un certain résultat

En utilisant la fonction `Random.int`, proposer des fonctions telles que

```
addition_valant : int -> string
```

qui génère une addition aléatoire dont le résultat vaut l'entier donné, donné sous forme d'une chaîne de caractère en notation polonaise.