Lycée Loges

Chapitre 4

Manipulations de fichiers Exercices

Simon Dauguet
simon.dauguet@gmail.com

28 novembre 2024

1 Piles

Exercice 1 :
Avant de traiter toute demande informatique, on souhaite souvent vérifier si ce qui est écrit respecte les règles d'écriture. On effectue donc une vérification syntaxique : le *parsage*.

Lors de calculs mathématiques ou de l'écriture de langage à balises (par exemple `html`), le bon parenthésage des balises constitue une étape de ce parsage.

Ainsi

- ◇ avec un type de balise : $(1+4)$ est correct mais $)1+4($, $((1+4)$ sont incorrects
- ◇ avec deux types de balises : $[1+(4+5)]$ est correct mais $[1+(4+5])$ est incorrect

1. proposer une fonction `verif_balise(exp: str) -> bool`, qui vérifie le bon parenthésage avec un type de balises (une simple variable suffit).

   Par exemple, `verif_balise( "(1+4)-(3+5)")` renverra `True`.
2. proposer une fonction `verif_balises(exp: str) -> bool`, qui vérifie le bon parenthésage avec plusieurs types de balises (une pile contenant les symboles '(' ou '[' pourra être utile).

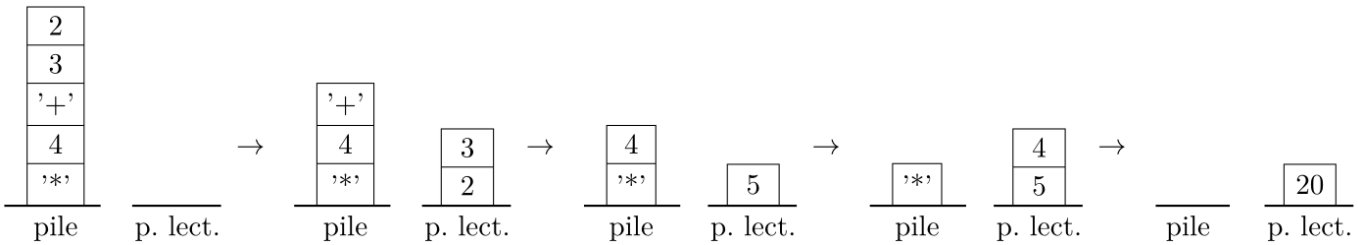
   Par exemple, `verif_balise( "[1-(3+5])")` renverra `False`.
3. proposer une fonction `position_balise(exp : str) -> list`, qui renvoie un tableau de couples contenant la position dans l'expression des parenthèses ouvrantes et fermantes.

   Par exemple, `verif_balise( "3*((1+4)-(3+5))")` renverra `[(2,14), (3,7), (9,13)]`.

Exercice 2 :
La notation polonaise inverse (du Polonais Jan Lukasiewicz 1878-1956) (ou notation post-fixée) consiste à écrire chaque opérande puis l'opérateur. Ainsi $3-5$ s'écrit `['-',5,3]`, $1-2\times3$ s'écrit `['-','*',3,2,1]` et $(1-2)\times3$ s'écrit sans besoin de parenthèses `['*',3,'-',2,1]`.

Si on considère l'entrée de l'expression mathématique comme un pile (de sommet à droite), on peut alors l'évaluer comme suit. On dépile les nombres lus dans une deuxième pile, puis dès qu'on atteint un opérateur, on dépile les deux derniers éléments, on effectue l'opération et on empile le résultat. On arrête une fois la pile vide.

exemple : $(2+3)*4$



Ici on ne se focalisera que sur les opérateurs $+$ et $\times$.

1. Évaluer les expressions `A = ['*','-',2,5,3]` et `B = ['-',5,'*','-',2,4,7]`.
2. Écrire en notation polonaise inverse les expressions $C = 3 \times (5 - 2)$ et $D = -3 \times (4 \times 5 - 1)$
3. Proposer une fonction `eval_polo(p: list) -> float` qui, à partir d'une pile `p` correspondant aux entrées d'une expression en notation polonaise inverse, évalue la-dite expression.

2 Fichiers

Exercice 3 :

On considère un fichier `mesMots.txt` contenant des lignes composé d'un mot

```
alpha
arithmetique
beta
...
```

.

Proposer une fonction `longMot() -> list` qui lit le fichier `mesMots.txt` (supposé dans le répertoire courant) et renvoie tous les mots les plus longs.