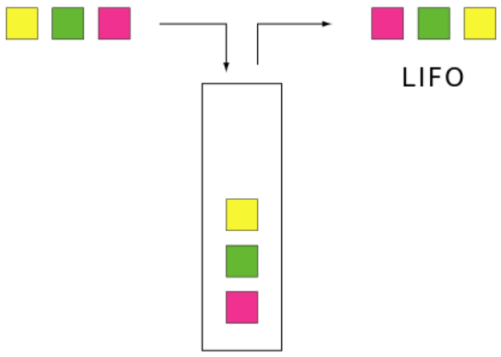
Les piles

1 Définfition

Dans les piles, il est uniquement possible de manipuler le dernier élément introduit dans la pile. On prend souvent l'analogie avec une pile d'assiettes : dans une pile d'assiettes la seule assiette directement accessible et la dernière assiette qui a été déposée sur la pile.

Les piles sont basées sur le principe LIFO (Last In First Out : le dernier rentré sera le premier à sortir). On retrouve souvent ce principe LIFO en informatique.



2 Interface d'une pile

On note Pile[T] le type des piles contenant des éléments de type T.

fonction	description
creer_pile() -> Pile[T]	On peut créer une pile vide. (creer_pile)
est_vide(p :Pile[T])-> bool	on peut savoir si une pile est vide (pile_vide)
empiler(p :Pile[T], e :T) -> None	on peut empiler un nouvel élément sur la pile.
depiler(p :Pile[T]) -> T	on peut récupérer l'élément au sommet de la pile tout en le supprimant.

3 Utilisation des piles ?

Les piles sont extrêmement utiles en informatique et vous les utilisez quotidiennement, parfois même sans vous en rendre compte :

- La fonction annuler (Ctrl-Z) de votre traitement de textes par exemple est une pile : Quand vous tapez Ctrl-Z, vous annulez la dernière opération effectuée. Quand vous faites une nouvelle opération, celle-ci est mémorisée au sommet de la pile. Vous ne pouvez pas annuler l'avant dernière opération sauf à annuler la dernière.
- Le bouton retour de votre navigateur internet fonctionne également à l'aide d'une pile. Les pages web consultées lors de votre navigation sur une page sont empilées et le bouton retour permet d'accéder à la dernière page présente sur la pile.
- Certaines calculatrices fonctionnent à l'aide d'une pile pour stocker les arguments des opérations : c'est le cas de beaucoup de calculatrices de la marque HP, dont la première calculatrice scientifique ayant jamais été produite : la HP 35 de 1972.

A vous de jouer ! .1

UTILISATION DES PILES POUR CRÉER UNE GESTION DYNAMIQUE DU PARENTHÉSAGE

Dans un logiciel de calcul formel, ou plus généralement dans un éditeur de texte qui permet d'écrire un programme informatique, il y a une gestion dynamique du parenthésage.
Pour exemple, les expressions A et B sont mal parenthésées, l'exprissions C bien parenthésée :

$$A = (2 + \sqrt{3}))^2$$
$$B = ((4 + n)^2$$
$$C = (2 + 3n)^2 \times (4 + 5n)$$

On souhaite proposer une fonction qui reçoit comme argument une chaîne de caractères composée uniquement de parenthèses ouvrantes et fermantes (pour simplifier l'exercice, on n'entre pas les autres caractères) et qui renvoie True si le parenthésage est bon, False sinon.
On parcourt pour cela le "mot" de gauche à droite et on utilisera une pile, afin de stocker les parenthèses ouvrantes non encore fermées.

1. Proposer une fonction booléenne, en langage naturel, qui permet de répondre au problème du parenthèsage.

```
Fonction Verif_parentheses(chaine):
    ''' On pourra utiliser les fonctions :
    creer_pile(), est_vide(P),
    empiler(P,e), depiler(P)
    '''
    pass
```

2. Implémenter cette fonction en Python. On utilisera une implémentation de pile au choix.

3. Proposer en plus, dans la cas d'un bon parenthésage, une indication des parenthèses ouvrantes et fermantes. La fonction ne retournera plus True, mais la position des parenthèses. Par exemple, verif_parentheses('(())()') donnera (0,3),(1,2) et (4,5).