NSI · TERMINALE / COURS

Listes, piles et files

Astrolab Cours · astrolabcours.fr

01 Le problème

Le bouton « Annuler » d'un traitement de texte doit défaire la dernière action, puis l'avant-dernière. Une imprimante partagée doit imprimer les documents dans l'ordre où ils sont arrivés. Il faut deux façons différentes de ranger des éléments.

01 L'idée : la pile d'assiettes et la file du self

Sur une pile d'assiettes, on pose et on prend toujours sur le dessus : la dernière posée est la première reprise. Dans la file du self, on entre par l'arrière et on sort par l'avant : le premier arrivé est le premier servi.

01 La pile (LIFO)

Une **pile** (*Last In, First Out*) offre :

- `empiler(x)` : poser `x` au sommet ;
- `depiler()` : retirer et renvoyer le sommet ;
- `est_vide()`.

En Python, une liste fait une bonne pile : `append` pour empiler, `pop()` pour dépiler. Les deux se font en temps constant.

```
pile = []
pile.append("taper")
pile.append("gras")
pile.pop()     # "gras" : on annule la dernière action
```

La file (FIFO)

Une **file** (*First In, First Out*) offre `enfiler(x)` à l'arrière, `defiler()` à l'avant, et `est_vide()`.

`liste.pop(0)` fonctionne, mais décale tous les éléments : c'est lent sur une grande file. On utilise `collections.deque` (`append` et `popleft`), ou deux piles.

Interface et implémentation

L'interface dit ce qu'on peut faire, l'implémentation dit comment c'est codé. Une même interface de file peut

© Astrolab Cours · astrolabcours.fr/nsi/terminale/listes-piles-files/ · Tous droits réservés : usage personnel uniquement, republication et usage commercial interdits

être codée avec une liste Python, une `deque`, deux piles ou une liste chaînée. Le programme qui l'utilise n'a pas à changer.

La liste chaînée

Chaque élément (un **maillon**) contient une valeur et un lien vers le maillon suivant. Ajouter en tête est immédiat ; accéder au 1000e élément oblige à suivre 1000 liens.

Exemple guidé

Vérifier qu'une expression est bien parenthésée, comme `(a[b]c)`, avec une pile.

```
def bien_parenthesee(s):
    pile = []
    paire = {")": "(", "]": "["}
    for c in s:
        if c in "([":
            pile.append(c)
        elif c in ")]":
            if not pile:
                return False
            if pile.pop() != paire[c]:
                return False
    return pile == []
```

1. Chaque ouvrante est **empilée**.
2. Chaque fermante doit correspondre au **sommet** de la pile, qu'on dépile.
3. À la fin, la pile doit être vide. Pour `"(]"`, le sommet `(` ne correspond pas à `]` : on renvoie `False`.

Retour au problème

« Annuler » utilise une pile : la dernière action enregistrée est la première défaite. L'imprimante utilise une **file** : le premier document arrivé est le premier imprimé.

Les pièges

- Confondre LIFO et FIFO.
- Dépiler une pile vide : il faut tester `est_vide()` avant.
- Croire qu'une liste Python est une liste chaînée : c'est un tableau dynamique.

© Astrolab Cours · astrolabcours.fr/nsi/terminale/listes-piles-files/ · Tous droits réservés : usage personnel uniquement, republication et usage commercial interdits