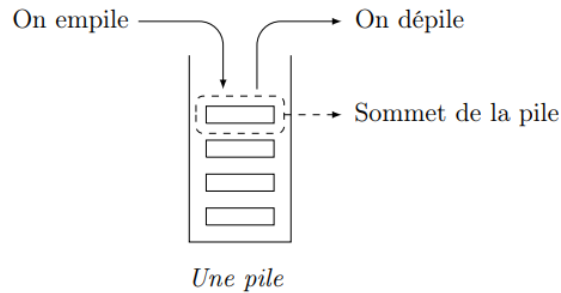


Piles Files Exercices

Sujets bac

Exercice 1 : Piles et langage Python

On rappelle qu'une pile est une structure de données abstraite fondée sur le principe « dernier arrivé, premier sorti » :



On munit la structure de données Pile de quatre fonctions primitives définies dans le tableau ci-dessous. :

Structure de données abstraite : Pile

Utilise : Éléments, Booléen

Opérations :

- `creer_pile_vide` : $\emptyset \rightarrow \text{Pile}$
`creer_pile_vide()` renvoie une pile vide
- `est_vide` : $\text{Pile} \rightarrow \text{Booléen}$
`est_vide(pile)` renvoie True si `pile` est vide, False sinon
- `empiler` : $\text{Pile}, \text{Élément} \rightarrow \text{Rien}$
`empiler(pile, element)` ajoute `element` au sommet de la `pile`
- `depiler` : $\text{Pile} \rightarrow \text{Élément}$
`depiler(pile)` renvoie l'élément au sommet de la `pile` en le retirant de la `pile`

Question 1 On suppose dans cette question que le contenu de la pile P est le suivant (les éléments étant empilés par le haut) :

4
2
5
8

Quel sera le contenu de la pile Q après exécution de la suite d'instructions suivante ?

```
1   Q = creer_pile_vide()
2   while not est_vide(P):
3       empiler(Q, depiler(P))
```

Question 2

1. On appelle *hauteur* d'une pile le nombre d'éléments qu'elle contient. La fonction `hauteur_pile` prend en paramètre une pile `P` et renvoie sa hauteur. Après appel de cette fonction, la pile `P` doit avoir retrouvé son état d'origine.

Exemple : si `P` est la pile de la question 1 : `hauteur_pile(P) = 4`.

Recopier et compléter sur votre copie le programme Python suivant implémentant la fonction `hauteur_pile` en remplaçant les `???` par les bonnes instructions.

```
1      def hauteur_pile(P):
2          Q = creer_pile_vide()
3          n = 0
4          while not(est_vide(P)):
5              ???
6              x = depiler(P)
7              empiler(Q,x)
8          while not(est_vide(Q)):
9              ???
10             empiler(P, x)
11         return ???
```

2. Créer une fonction `max_pile` ayant pour paramètres une pile `P` et un entier `i`. Cette fonction renvoie la position `j` de l'élément maximum parmi les `i` derniers éléments empilés de la pile `P`. Après appel de cette fonction, la pile `P` devra avoir retrouvé son état d'origine. La position du sommet de la pile est 1.

Exemple : si `P` est la pile de la question 1 : `max_pile(P, 2) = 1`

Question 3 Créer une fonction `retourner` ayant pour paramètres une pile `P` et un entier `j`. Cette fonction inverse l'ordre des `j` derniers éléments empilés et ne renvoie rien. On pourra utiliser deux piles auxiliaires.

Exemple : si `P` est la pile de la question 1(a), après l'appel de `retourner(P, 3)`, l'état de la pile `P` sera :

5
2
4
8

Question 4 L'objectif de cette question est de trier une pile de crêpes.

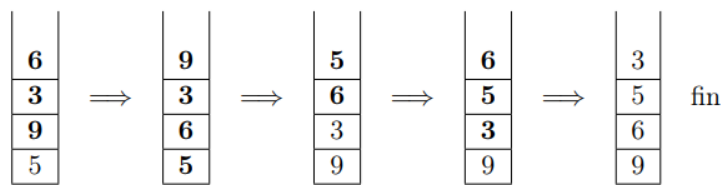
On modélise une pile de crêpes par une pile d'entiers représentant le diamètre de chaque crêpe. On souhaite réordonner les crêpes de la plus grande (placée en bas de la pile) à la plus petite (placée en haut de la pile).

On dispose uniquement d'une spatule que l'on peut insérer dans la pile de crêpes de façon à retourner l'ensemble des crêpes qui lui sont au-dessus.

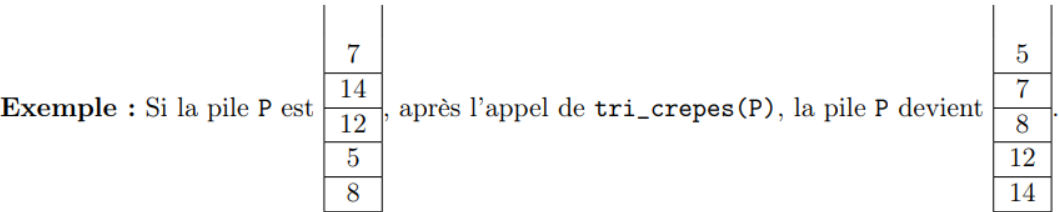
Le principe est le suivant :

- On recherche la plus grande crêpe.
- On retourne la pile à partir de cette crêpe de façon à mettre cette plus grande crêpe tout en haut de la pile.
- On retourne l'ensemble de la pile de façon à ce que cette plus grande crêpe se retrouve tout en bas.
- La plus grande crêpe étant à sa place, on recommence le principe avec le reste de la pile.

Exemple :



Cr  er la fonction `tri_crepes` ayant pour param  tre une pile P. Cette fonction trie la pile P selon la m  thode du tri cr  pes et ne renvoie rien. On utilisera les fonctions cr    es dans les questions pr  c  dentes.



Exercice 2 : Piles et files

Une méthode simple pour gérer l'ordonnancement des processus est d'exécuter les processus en une seule fois et dans leur ordre d'arrivée.

1. Parmi les propositions suivantes, quelle est la structure de données la plus appropriée pour mettre en œuvre le mode FIFO (First In First Out) ?

- a) liste
- b) dictionnaire
- c) pile
- d) file

2. On choisit de stocker les données des processus en attente à l'aide d'une liste Python `lst`.

On dispose déjà d'une fonction `retirer(lst)` qui renvoie l'élément `lst[0]` puis le supprime de la liste `lst`. Écrire en Python le code d'une fonction `ajouter(lst, proc)` qui ajoute à la fin de la liste `lst` le nouveau processus en attente `proc`.

On choisit maintenant d'implémenter une file `file` à l'aide d'un couple $(p1, p2)$ où `p1` et `p2` sont des piles. Ainsi `file[0]` et `file[1]` sont respectivement les piles `p1` et `p2`.

Pour enfiler un nouvel élément `elt` dans `file`, on l'empile dans `p1`.

Pour défiler `file`, deux cas se présentent.

- La pile `p2` n'est pas vide : on dépile `p2`.
- La pile `p2` est vide : on dépile les éléments de `p1` en les empilant dans `p2` jusqu'à ce que `p1` soit vide, puis on dépile `p2`.

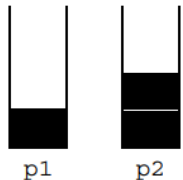
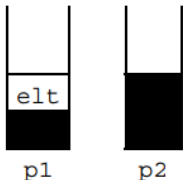
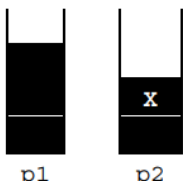
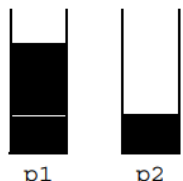
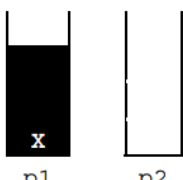
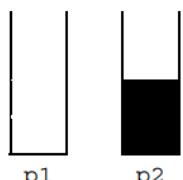
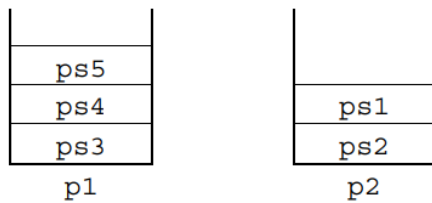
	État de la file avant	État de la file après
<code>enfiler(file, elt)</code>	 p1 p2	 p1 p2
<code>defiler(file)</code> cas où <code>p2</code> n'est pas vide	 p1 p2	 p1 p2
<code>defiler(file)</code> cas où <code>p2</code> est vide	 p1 p2	 p1 p2

Illustration du fonctionnement des fonctions `enfiler` et `defiler`.

3. On considère la situation représentée ci-dessous.



On exécute la séquence d'instructions suivante :

```
enfiler(file,ps6)
defiler(file)
defiler(file)
defiler(file)
enfiler(file,ps7)
```

Représenter le contenu final des deux piles à la suite de ces instructions.

4. On dispose des fonctions :

- `empiler(p,elt)` qui empile l'élément `elt` dans la pile `p`,
- `depiler(p)` qui renvoie le sommet de la pile `p` si `p` n'est pas vide et le supprime,
- `pile_vide(p)` qui renvoie `True` si la pile `p` est vide, `False` si la pile `p` n'est pas vide.

- a. Écrire en Python une fonction `est_vide(f)` qui prend en argument un couple de piles `f` et qui renvoie `True` si la file représentée par `f` est vide, `False` sinon.
- b. Écrire en Python une fonction `enfiler(f,elt)` qui prend en arguments un couple de piles `f` et un élément `elt` et qui ajoute `elt` en queue de la file représentée par `f`.
- c. Écrire en Python une fonction `defiler(f)` qui prend en argument un couple de piles `f` et qui renvoie l'élément en tête de la file représentée par `f` en le retirant.