**EXERCICE 5 (4 points)**

*Cet exercice porte sur les structures de données linéaires*

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) ?
   * 1. liste
     2. dictionnaire
     3. pile
     4. file

Réponse : d) file car c’est le fonctionnement même d’une file.

1. 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.

Réponse :

def ajouter(lst,proc) :

lst.append(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 |
| enfiler(file,elt) | p1 p2 | |  |  |  | | --- | --- | --- | | elt |  |  | |  |  |   p1 p2 |
| defiler(file)    cas où p2 n’est pas vide | |  |  |  | | --- | --- | --- | |  |  |  | |  | **X** | |  |  |   p1 p2 | p1 p2 |
| defiler(file) cas où p2 est vide | |  |  |  | | --- | --- | --- | |  |  |  | |  |  | | **X** |  |   p1 p2 | p1 p2 |

Illustration du fonctionnement des fonctions enfiler et défiler.

1. On considère la situation représentée ci-dessous.

|  |  |  |
| --- | --- | --- |
| ps5 |  |  |
| ps4 | ps1 |
| ps3 | ps2 |

p1 p2

On exécute la séquence d'instructions suivante

enfiler(file,ps6)

defiler(file)

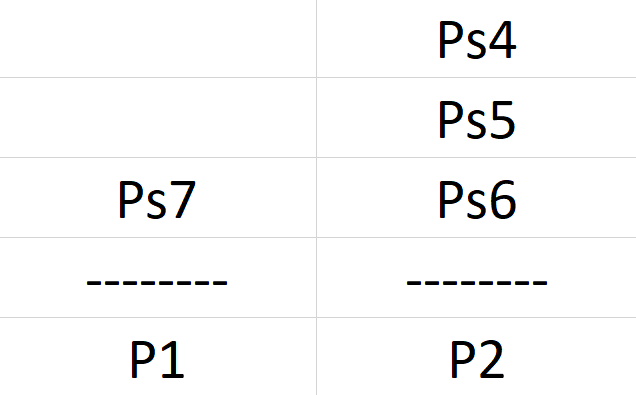
defiler(file)

defiler(file)

enfiler(file,ps7)

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

Réponse :



1. 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.

* + 1. 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.

Réponse :

def est\_vide(f) :

return pile\_vide(f[0]) and pile\_vide(f[1])

* + 1. 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.

Réponse :

def enfiler(f,elt) :

empiler(f[0],elt)

* + 1. Écrire en Python une fonction defiler(f) qui prend en argument un couple de piles f et qui renvoie l'élement en tête de la file représentée par f en le retirant.

Réponse :

def defiler(f) :

if est\_vide(f[1]) :

while not(pile\_vide(f[0])) :

empiler(f[1],depiler(f[0]))

return depiler(f[1])