ET_1_Corrigé (Piles)

Exercice 1 : Notion de Pile et programmation Python

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

- creer_pile_vide : ∅ ⟶ Pile
  creer_pile_vide() : renvoie une pile vide
- est_vide : Pile ⟶ Booléen
  est_vide(pile) : renvoie True si pile est vide, False sinon
- empiler : Pile, Élément ⟶ Rien
  empiler(pile, element) : ajoute element au sommet de la pile
- depiler : Pile ⟶ 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). Quel sera le contenu de la pile Q après exécution de la suite d'instructions suivante

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

On dépile la pile P et on empile la pile Q.

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

```
def hauteur_pile(P):
    Q = creer_pile_vide ()
    n = 0
    while  not(est_vide(P)):
        n=n+1 # c'est l'incrémentation du compteur
        x = depiler(P)
        empiler(Q,x)
    while not(est_vide(Q)):
        x=depiler(Q) # On récupère l'élément au sommet de la pile Q
        empiler(P, x)
    return n
```

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.**

```
def max_pile(P,i):
    '''In : P pile et i entier <= hauteur_Pile
      Out : renvoie la position j de l'élément maximum parmi les i
      derniers éléments empilés de la pile P'''
    Q=creer_pile_vide()
    n=1
    x=depiler(P)
    empiler(Q,x)
    maximum=x
    rang=1
    while  not(est_vide(P)) and n<i:
        n=n+1
        x = depiler(P)
        empiler(Q,x)
        if x>maximum:
            maximum=x
            rang=n
    while not(est_vide(Q)):
            x=depiler(Q)
            empiler(P, x)
    return rang
```

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.

```
def retourner(P,j):
    '''In : P pile et j entier <= hauteur_Pile
      Out : None.
      Cette fonction inverse l'ordre des j derniers éléments empilés
      et ne renvoie rien'''
    Q = creer_pile_vide ()
    K = creer_pile_vide ()

    # on dépile les j derniers éléments de P que l'on empile dans Q
    # (ordre inversé)
    n=0
    while  n<j and not(est_vide(P)):
        empiler(Q,depiler(P)) # comme dans la question 1
        n=n+1

    # on dépile les j derniers éléments de Q que l'on empile dans K
    #  (ordre inversé) donc l'ordre redevient celui initial
    n=0
    while  n<j and not(est_vide(Q)):
        empiler(K,depiler(Q))
        n=n+1

    # on dépile les j derniers éléments de K que l'on empile dans P
    # (ordre inversé)
    n=0
    while  n<j and not(est_vide(K)):
        empiler(P,depiler(K))
        n=n+1
```

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.*

```
def tri_crepe(Pile):
    '''In : Pile
       out : None. Cette fonction va trier la pile'''
    n=hauteur_pile(Pile)
    while n!=0:
        k=max_pile(Pile,n)
        retourner(Pile,k)
        retourner(Pile,n)
        n=n-1
```