TD 1 - Piles et Files

Exercice 1: Implémentation d'une pile et d'une file

Proposez une implémentation de la pile et de la file en écrivant les fonctions suivantes :

```
def creer_pile():
    # Renvoie une nouvelle pile vide.
def empiler( P, e ):
    # Empile l'élément 'e' dans la pile 'P'.
def depiler( P ):
    # Dépile un élément de la pile 'P' et renvoie cet élément.
def creer_file():
    # Renvoie une nouvelle file vide.
def enfiler( P, e ):
    # Enfile l'élément 'e' dans la file 'F'.
def defiler( P ):
    # Défile un élément de la file 'F' et renvoie cet élément.
```

Dessinez le contenu de la pile et de la file pendant l'éxécution des programmes suivants. Vous donnerez aussi le contenu du terminal.

```
pile = creer_pile()
empiler( pile, 1 )
empiler( pile, 2 )
empiler( pile, 3 )
print( depiler( pile ) )
print( depiler( pile ) )
print( depiler( pile ) )
```

```
file = creer_file()
enfiler( file, 1 )
enfiler( file, 2 )
enfiler( file, 3 )
print( defiler( file ) )
print( defiler( file ) )
print( defiler( file ) )
```

Exercice 2

Supposons que vous ne pouvez plus créer ou manipuler de listes, de dictionnaires ou de tuples. Supposons que l'on vous donne les primitives suivantes pour manipuler les piles :

```
def creer_pile()
def empiler( P, e )
def depiler( P )
```

1) Proposez une implémentation des files en utilisant uniquement les piles, c'est à dire les primitives précédentes. Pour cela implémentez les primitives suivantes :

```
def creer_file()
def emfiler( F, e )
def defiler( F )
```

Evaluez la complexité de chacune de vos fonctions précédentes.

2) Proposez une implémentation des tableaux en utilisant uniquement les primitives précédentes. Pour cela, vous implémenterez les primitives suivantes :

```
def creer_tableau():
    # Créé et renvoie un tableau vide.
def inserer_element( T, id, e ):
    # Insère un nouvel élément 'e' à la position 'id' du tableau
        # 'T'. La taille du tableau augmente de 1.
def supprimer_element( T, id ):
    # Supprime l'élement situé à la position 'id' du tableau 'T'.
    # La taille du tableau diminue de 1.
def remplacer_element( T, id, e ):
    # Remplace l'élément à la position 'id' de 'T' par 'e'.
    # La taille du tableau reste inchangé.
def obtenir_element( T, id ):
    # Renvoie l'élément situé à la position 'id' du tableau.
```

Evaluez la complexité de chacune de vos fonctions.

3) Proposez une implémentation des dictionnaires en utilisant uniquement les primitives précédentes. Pour cela, vous implémenterez les primitives suivantes :

```
def creer_dictionnaire():
    # Créé et renvoie un nouveau dictionnaire
def inserer_association( D, cle, valeur ):
    # Ajoute une nouvelle association ('cle','valeur') dans le
    # dictionnaire D.
    # S'il existait déjà une association ('cle', valeur1) dans D,
    # alors cette association est remplacée par ('cle','valeur').
def supprimer_association( D, cle ):
    # Supprime l'assocation de D qui à pour cle 'cle'.
def appartient_au_dictionnaire( D, cle ):
    # Renvoie Vraie si 'cle' est une clé du dictionnaire D.
def obtenir_valeur( D, cle ):
    # Renvoie la valeur associée à la clé dans le dictionnaire D.
```

Évaluez la complexité de vos fonctions.