Spécialité NSI Terminale
Correction du contrôle n° 4
Lundi 12 décembre 2022

I Dictionnaires

Questions de cours

1. On écrit :

```
personnage = {}
```

ou bien

```
personnage = dict()
```

2. On écrit :

```
personnage["Prénom"] = "Jean"
personnage["Nom"] = "Valjean"
```

3. a. On écrit :

```
tabCles = list(monDico.keys())
```

3. b. On écrit :

```
tabValeurs = list(monDico.values())
```

4. On obtient l'affichage suivant :

```
Danemark Copenhague
Norvège Oslo
Suède Stockholm
Finlande Helsinki
```

Exercice 1

1. a. La valeur associée à la clé `"D"` est `"C"`.
On l'obtient en écrivant `alpha["D"]`.

b. Le chiffrement du mot `"BAGAGE"` donne `"DBEBEF"`.

2. On peut écrire le code suivant :

```
def chiffrer(mot, alpha) :
    res = ""
    for c in mot :
        res = res + alpha[c]
    return res
```

3. a. Le dictionnaire permettant le déchiffrement est le suivant :

```
{"A":"C", "B":"A", "C":"D", "D":"B", "E":"G", "F":"E",
"G":"F"}
```

3. b. On peut compléter le code de la fonction de la façon suivante :

```
def dico_dechiffrement(dico):
    nouveau = {}
    for lettre in dico :
        code = dico[lettre]
        nouveau[code] = lettre
    return nouveau
```

3. c. On peut écrire le code suivant :

```
def dechiffre(mot, dico):
    dico_dechif = dico_dechiffrement(dico)
    return chiffrer(mot, dico_dechif)
```

4. On peut écrire le code suivant :

```
from random import shuffle

def dico_chiffrement(alphabet):
    alpha_mel = list(alpha) # on recopie l'alphabet
    shuffle(alpha_mel) # on le mélange en place
    dico = {}
    lg = len(alphabet)
    for i in range(lg) :
        dico[alphabet[i]] = alpha_mel[i]
    return dico
```

II Piles et files

Exercice 2

1 a. Après exécution, la pile `P` a le contenu suivant :

et la file `F` est vide.

1. b. On peut compléter le code de la façon suivante :

```
def taille_file(F) :
    F2 = creer_file_vide()
    cpt = 0
    while not est_vide(F) :
        enfiler(F2, defiler(F))
        cpt = cpt + 1
    while not est_vide(F2) :
        enfiler(F, defiler(F2))
    return cpt
```

2. On peut compléter le code de la façon suivante :

```
def former_pile(F) :
    P1 = creer_pile_vide()
    P2 = creer_pile_vide()
    while not est_vide(F) :
        empiler(P1, defiler(F))
    while not est_vide(P1) :
        elt = depiler(P1)
        enfiler(F, elt)
        empiler(P2, elt)
    return P2
```

3. On peut écrire le code suivant :

```
def nb_elements(F, elt) :
    F2 = creer_file_vide()
    cpt = 0
    while not est_vide(F) :
        e = defiler(F))
        if e == elt :
            cpt = cpt + 1
        enfiler(F2, e)
    while not est_vide(F2) :
        enfiler(F, defiler(F2))
    return cpt
```

4. On peut écrire le code suivant :

```
def verifier_contenu(F, nb_rouge, nb_vert, nb_jaune) :
    if nb_elements(F, "rouge") <= nb_rouge and \
       nb_elements(F, "vert") <= nb_vert and \
       nb_elements(F, "jaune") <= nb_jaune :
        return True
    else :
        return False
```