1 Modélisation de la grille de jeu.

1. Ouvrir Thonny et créer un fichier **NomSokoban.py**. Les réponses aux questions ci-dessous seront à mettre en commentaire dans le fichier python.

2. Définir la variable :

```
Grille= [[0,0,0,0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0],
         [0,0,0,0,1,2,2,2,1,0,0,0,0,0,0,0,0,0,0],
         [0,0,0,0,1,3,2,2,1,0,0,0,0,0,0,0,0,0,0],
         [0,0,1,1,1,2,2,3,1,1,0,0,0,0,0,0,0,0,0],
         [0,0,1,2,2,3,2,3,2,1,0,0,0,0,0,0,0,0,0],
         [1,1,1,2,1,2,1,1,2,1,0,0,0,1,1,1,1,1,1],
         [1,2,2,2,1,2,1,1,2,1,1,1,1,1,2,2,2,4,1],
         [1,2,3,2,2,3,2,2,2,2,2,2,2,2,2,2,2,2,1],
         [1,1,1,1,1,2,1,1,1,2,1,5,1,1,2,2,2,2,1],
         [0,0,0,0,1,2,2,2,2,2,1,1,1,1,1,1,1,1,1],
         [0,0,0,0,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0]]
```

Cette grille modélise le plateau de jeu, elle contient :

— des 0 qui représente : rien
— des 1 qui représente : un mur
— des 2 qui représente : un emplacement libre
— des 3 qui représente : une caisse
— des 4 qui représente : une case objectif

3. Définir les variables globales :

```
nb_caisse=6
ligneSokoban = 2
colonneSokoban= 5
```

4. Avec Thonny, afficher résultat de **Grille[ligneSokoban][colonneSokoban]** ?

5. Localiser le 5 dans la variable Grille (Quelle ligne et quelle colonne) et modifier les variables **ligneSokoban** et **colonneSokoban** pour que **Grille[ligneSokoban][colonneSokoban]** renvoi 5 (qui désigne le Sokoban).

6. On souhaite avoir un visuel plus "sympathique" du jeu (voir Question 8) :
A partir de la variable **Grille**, les 0 seront remplacés par " ", les 1 par "M", les 2 par " ", les 3 par 'C', les 4 par "O" et le 5 par "S".

Compléter la procédure ci-dessous qui permet d'afficher une ligne du plateau de jeu.
Par exemple :

```
>>>representer([1,1,1,1,1,2,1,1,1,2,1,5,1,1,2,2,4,4,1])
MMMMM  MMM  MSMM    OOM
```

```
def representerLigne(ligne) :
    affichage=""
    ..................# A compléter
        if case==0 :
            symbole=" "#rien
        elif case==1 :
            symbole="M"#mur
        elif case==2 :
            symbole=" "#libre
        elif case==3 :
            symbole="C"#caisse
        elif case==4 :
            symbole="O"#objectif
        elif case==5 :
            symbole="S"#sokoban
        affichage=affichage+symbole
    print(affichage)
```

7. Si vous avez bien travaillez la procédure ci-dessous permet d'avoir un visuel de toutes les lignes de la variable **Grille** :

```
def representer(jeu) :
    for ligne in jeu :
        representerLigne(ligne)
```

```
>>> representer(Grille)

          M M M M M
          M       M
          M C     M
      M M M     C M M
      M     C   C   M
  M M M   M   M M   M         M M M M M M M
  M       M   M M   M M M M M M       O M
  M   C     C                         M
  M M M M M   M M M   M S M M           M
          M             M M M M M M M M M
          M M M M M M M
```

2 Déplacement du Sokoban

Maintenant notre objectif est de déplacer le Sokoban.

8. Définir la procédure suivante :

```
def deplacer(direction) :
    """
    La variable locale direction est une chaîne de caractère :
    "z" : aller en haut
    "q" : aller à gauche
    "d" : aller à droite
    "s" : aller en bas
    """
    global Grille
    global ligneSokoban
    global colonneSokoban
    global nb_caisse
```

2.1 "z" : monter le Sokoban

9. Compléter la procédure **deplacer** avec :

```
if direction=="Z" or direction=="z" :
```

2.1.1 Case libre

10. Au départ, la case au dessus du Sokoban est-elle une case libre (2 dans la **Grille**) ?

11. Compléter la procédure **deplacer** avec le code :

a. **if Grille[ligneSokoban-1][colonneSokoban]==2**. Que permet de faire ce test ?

b.
```
Grille[ligneSokoban][colonneSokoban]=2
ligneSokoban=ligneSokoban+1
Grille[ligneSokoban][colonneSokoban]=5
```

(Mettre un commentaire sur le rôle des trois lignes ci-dessus)

12. Tester le bon fonctionnement de votre code avec :

```
>>> representer(Grille)
>>> deplacer("z")
>>> representer(Grille)
>>> deplacer("z")
>>> representer(Grille)
```

2.1.2 Case cadeau

13. Mettre en place un test qui permet de vérifier que la case au dessus du Sokoban est une caisse

14. Compléter les deux cas qui vont alors se présenter :

a. Si la case après le cadeau est libre alors on peut monter la caisse puis le Sokoban. Cette condition ce traduit en python par :

```
if Grille[ligneSokoban-2][colonneSokoban]==..... :
    Grille[ligneSokoban][colonneSokoban]=2 # On remplace le 5 par un 2
    ligneSokoban=..... # On monte le Sokoban
    Grille[ligneSokoban][colonneSokoban]=..... # On met le Sokoban
    Grille[.....][colonneSokoban]=3 #on met la caisse au dessus du Sokoban
```

b. Si la case après le cadeau est un objectif alors la caisse disparait et le Sokoban monte. Cette condition ce traduit en python par :

```
if Grille[ligneSokoban-2][colonneSokoban]==.............................:
    Grille[ligneSokoban][colonneSokoban]=2  # On remplace le 5 par un 2
    ligneSokoban=.....................# On monte le Sokoban
    Grille[ligneSokoban][colonneSokoban]=.........# On met le 5 de Sokoban
    nb_caisse=................# On  diminue le nb de caisse de 1
```

15. Avec le code ci-dessous, tester la procédure **deplacer** pour vérifier que Sokoban est capable de "monter" une caisse.)

```
>>> ligneSokoban = 8
>>> colonneSokoban= 5
>>> Grille[ligneSokoban][colonneSokoban]="S"
>>> representer(Grille)
>>> deplacer("z")
>>> representer(Grille)
```

2.2 "s" :descendre le Sokoban

16. A l'aide des questions 9 à 15, compléter la procédure pour gérer la descente du Sokoban.

```
if direction=="S" or direction=="s":
```

2.3 "q" : décaler à gauche le Sokoban

C'est à vous ....

2.4 "d" : décaler à droite le Sokoban

C'est encore à vous ....

3 Jouer

Dans la procédure **jouer_Sokoban()** ci-dessous, on a mis une variable **reponse** qui est une chaîne de caractère saisie par l'utilisateur.

17. D'après le code ci-dessous, quelles sont les valeurs possibles pour la variable **reponse** ?

18. Dans le code ci-dessous, pour chacune des variables dire si elle est locale ou globale et en donner le type.

19. Dans quel**S** cas une partie de Sokoban s'arrête ?

20. Compléter alors le **while** de la procédure ci-dessous.

```
1  def jouer_Sokoban():
2      global Grille
3      global ligneSokoban
4      global colonneSokoban
5      global nb_caisse
6      representer(Grille)
7      reponse="nImporteQuoi"
8      while ..................... :
9          reponse=input("Votre déplacement (z/q/s/d) ou r pour recommencer ?")
10         deplacer(reponse)
11         representer(Grille)
12     if reponse=="r":
13         print("ce n'est pas grave, tu feras mieux la prochaine fois...")
14      else :
15         print("Bravo !!!")
16     try_again=input("Voulez vous refaire une partie (y/n - y par défaut)")
17     if try_again!="n":
18         jouer_Sokoban()
19     else :
20         print("A bientôt")
```

4 Améliorons le graphisme ...

21. Dans le même dossier que **NomSokoban.py**, télécharger l'image **mario.gif**.

22. Récupérer et copier au début de votre fichier le code python du fichier : fichier **python sokoban_graphisme.py**.

23. Dans la procedure **jouer_Sokoban()**, remplacer les trois lignes de la boucle **while** (C'est à dire les lignes 9,10 et 11 du code ci-dessus) par :

```
print("Votre déplacement (z/q/s/d) ou r pour recommencer ? ")
afficher_jeu(Grille)
deplacer(reponse)
```

5 Vous pouvez enfin jouer !

24. Trouver une solution à ce jeu.

25. Construire une autre grille de jeu pour Sokoban.