Jeu du Simon

L'activité va permettre d'implémenter le jeu du Simon en utilisant la programmation orientée objet et les structures de données files.

1. Jeu du Simon

Simon est un jeu de société électronique de forme circulaire comportant quatre grosses touches de couleurs différentes : rouge, vert, bleu et jaune. Le jeu joue une séquence de couleurs que le joueur doit mémoriser et répéter ensuite. S'il réussit, une couleur parmi les 4 est ajoutée à la fin de la séquence. La nouvelle séquence est jouée depuis le début et le jeu continue. Dès que le joueur se trompe, la séquence est vidée et réinitialisée avec une couleur et une nouvelle partie commence.

Jeu du Simon

Exemple de séquence jouée : rouge, bleu, rouge, jaune, bleu.

1.1. Modes et niveaux de jeu

Le jeu Simon est composé de trois modes :

- Le mode 1 consiste à reproduire une suite créée aléatoirement par le jeu ;
- Le mode 2 consiste à créer votre propre suite sans vous tromper ;
- Le mode 3 sert à jouer de 2 à 4 joueurs.

Il y a 4 niveaux jouables sur Simon, correspondant à une plus ou moins grande rapidité d'exécution.
Le jeu est gagné lorsqu'une séquence de 32 couleurs est trouvée.

2. Implémentation

- Une **structure de file** doit permettre de **stocker la séquence** à afficher.
- La **programmation objet** sera utilisé pour gérer la **structure file**.
- La **classe File** sera dans un **fichier à part**.
    - Le contenu de la classe est décrit ci-après, elle ne doit pas être modifié.
    - Les actions sur la structure de file ne peuvent être que celles qui figure dans la classe File.
- Des assertions de test doivent être présentes pour chaque fonction.

2.1. Interface

Une interface doit être créer (Tkinter, Pygame ou en utilisant la librairie pyttsx4).

3. Bonus

Réaliser l'implémentation des 3 modes du jeu.

4. Classe File

```
class File:
    def __init__(self): # constructeur
        self.file = []

    def file_vide(self): # accesseur
        """ teste si la file est vide"""
        return self.file == []

    def ajout(self, element): # mutateur
        """ajoute un élément à la file"""
        self.file.append(element)

    def retire(self): # mutateur
        """renvoie le premier élément de la file et le supprime"""
        if self.file == []:
            return None
        return self.file.pop(0)

    def get_file(self): # accesseur
        """retourne le contenu de la pile"""
        return self.file
```