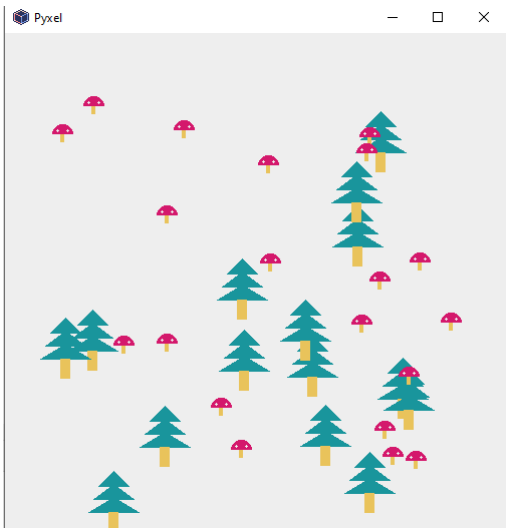
DM - Une forêt de sapins



Le but est d'afficher une forêt de sapins avec des champignons.

Pour cela, on utilise un module `pyxel` qui permet de dessiner et de programmer des petits jeux.

La programmation est en langage `Python`. Une documentation succinte se trouve à l'adresse : `https://github.com/kitao/pyxel/blob/main/docs/README.fr.md`

Mode d'emploi

1. Aller sur le site `https://www.pyxelstudio.net/`
2. Cliquer sur `create`
3. Ecrire le code dans la fenêtre d'édition.

```python
#Importations des modules
import pyxel
from random import randint
#Les fonctions (Zone 1)


def update():
    pass

def draw():
    pass #ligne à enlever si on rajoute du code dans la fonction draw()
    # (Zone 2)

pyxel.init(500, 500) #crée une fenêtre de 500 pixels par 500 pixels

#(Zone 3)

pyxel.run(update, draw)
```

4. Cliquer sur `Run` pour lancer le programme.
5. Cliquer sur `download` pour télécharger le programme, une fois celui-ci terminé.

Lorsqu'on utilise une fonction du module `pyxel`, on doit écrire `pyxel.` puis le nom de la fonction. Ainsi, on voit que `init` est une fonction du module `pyxel`. Pour voir des informations sur les fonctions de ce module, on regarde dans la documentation indiquée plus haut.

L'instruction `pyxel.init(500, 500)` crée une fenêtre graphique de 500 pixels par 500 pixels. Le pixel tout en haut à gauche a pour position (0, 0), celui tout en haut à droite a pour position (499, 0), celui tout en bas à gauche a pour position (0, 499).

L'instruction `pyxel.run(update, draw)` appelle la fonction `update`, puis la fonction `draw` 30 fois par seconde.

Voici les grandes étapes du DM.

1. Ecrire (dans la zone 1) une fonction `sapin(x, y)` qui a pour paramètre deux entiers `x` et `y` et qui affiche un sapin en position (x, y). Cette fonction ne renvoie rien.
   On pourra utiliser la fonction `tri` du module `pyxel` en regardant la documentation. Il faudra bien-sûr utiliser d'autres fonctions pour dessiner un sapin.
2. Appeler la fonction `sapin` dans la zone 2 pour la tester.

```
def draw():
  sapin(200, 100) #dessine un sapin en position (200, 100)
```

   On veut dessiner plusieurs sapins à des positions aléatoires.

   La position du sapin est modélisée par une liste de deux éléments dont le premier élément correspond à l'abscisse et le deuxième élément à l'ordonnée.

   Les positions de tous les sapins sont stockées dans une liste `les_sapins`.

   Par exemple : `les_sapins = [ [20, 400], [154, 210], [325, 52] ]` correspond aux positions de trois sapins : le 1er sapin a pour position (20, 400), le 2e (154, 210) et le 3e (325, 52).
3. Ecrire (dans la zone 1) une fonction `remplir_positions(n)` qui a pour paramètre un entier `n` et qui renvoie une liste contenant n positions aléatoires entre 50 et 450.
   Par exemple, `remplir_positions(4)` peut renvoyer `[ [55, 210], [441, 58], [210, 211], [154, 321] ]`
   On peut tester cette fonction dans `Spyder` pour plus de commodités
4. Ecrire (dans la zone 3) une ligne permettant de stocker, dans une variable `les_sapins`, les positions aléatoires de 15 sapins.
5. Ecrire (dans la zone 2) le code permettant d'afficher les 15 sapins.
   Démarche :
   — On parcourt la liste `les_sapins`.
   Pour chaque élément de la liste, c'est-à-dire pour chaque position :
   - on stocke dans une variable `x` l'abscisse de cette position
   - et dans une variable `y` l'ordonnée de cette position.
   - on affiche un sapin à cette position.
6. On souhaite dessiner 20 champignons à des positions aléatoires.
   Suivre la même démarche ou bien rajouter des améliorations comme :
   - Il peut y avoir des champignons de forme et/ou de couleur différentes.
   - Les sapins peuvent être plus ou moins grands.