Présentation du projet N°3 du parcours de développeur d'applications en Python :

lien du projet : Direction Github !

I°) L'environnement.

1°) Introduction.

C'est avec un certain enthousiasme que j'ai attaqué ce projet, fin Novembre. Je ne savais pas à quoi m'attendre mais plus que jamais motivé, j'ai sauté dedans à pieds joints. L'avantage de ce projet c'est qu'on a de quoi s'initier facilement avec un cours sur pygame, disponible sur le site.

2°) Une histoire de librairie.

Dans un premier temps, afin m'initier à la programmation d'un jeu vidéo en python, je me suis très fortement appuyé sur le cours concernant pygame. Grâce à celui ci, j'ai pu me familiariser très rapidement à cette librairie qui est l'un des prérequis du projet.

Après une progression très rapide, je me suis retrouvé dans une impasse concernant l'une des parties du projet et non des moindres : Le placement aléatoire des objets, nécessaires à la victoire du personnage du jeu. Car effectivement si la partie « labyrinthe » était très simple, la partie citée ci-dessus l'était déjà moins.

II°) Les problèmes.

1°) Les objets.

Dans un premier temps, j'avais en tête de me servir de la méthode de génération du niveau pour y apposer mes objets, mais curieusement, ça ne fonctionnait pas. Ce qui m'a conduit à étudier de manière plus approfondie les classes et les méthodes. Malgré tout, le problème persistait.
Finalement, le problème provenait simplement de mon array mal déclaré :

```
self.structure = 0
```

❌

```
self.structure = []
```

✔

2°) L’algorithme de placement aléatoire.

```
# A loop to check of the position picked by random is a freespace
while count < count_max:
    self.case_x = random.randint(0, 14)
    self.case_y = random.randint(0, 14)

    # If the sprite is a freespace
    if self.level.structure[self.case_y][self.case_x] == '0':

        # then change it to a mark where the program gonna pop the item.
        self.level.structure[self.case_y][self.case_x] = "o1"
```

En ce qui concerne l’algorithme du placement aléatoire des objets, j’ai fais en sorte de respecter les dimensions de mon niveau, afin qu’aucun objet n’apparaissent en dehors.

Le « programme » va choisir une case aléatoire sur l’abscisse (case_x) et l’ordonnée (case_y). Si celle ci est représentée par le sprite « 0 » (qui est une case libre), alors le programme marque la case dans l’array du niveau, par o1-2-3, suivant l’objet.

De plus, afin qu’aucun objet ne se chevauche, chacun se verra attribuer un marquage sur la carte pour ne pas qu’une position soit réutilisé deux fois.

Une fois ceci fait, le programme quitte la boucle :

```
        # And quit the loop
        count += 1
```

III°) La finalisation du projet.

1°) PEP8 et l’Anglais.
Une fois le problème du placement des objets résolu, j’ai pu finalisé mon projet en prenant soin de traduire la totalité mon code, ainsi que mes commentaires. Puis j’ai ensuite formater tout le code pour répondre correctement à la norme PEP8.

J’ai ensuite terminé par rédiger cette présentation, étant le dernier livrable que je devais mettre à disposition.