Projet logo Turtle

Le Cahier des Charges :

L’objectif est de créer un logo numérique à l’aide de python et Turtle. Turtle est une bibliothèque Python préinstallée qui permet aux utilisateurs de créer des images et des formes en leur fournissant une toile virtuelle. Le stylet à l'écran que vous utilisez pour dessiner s'appelle la tortue et c'est ce qui donne son nom à la bibliothèque. En bref, le module Python Turtle aide les programmeurs à se faire une idée de ce qu'est la programmation graphique avec Python.

Le logo numérique original contient :

- Plusieurs colonnes de rectangles ou de cercles colorés.
- Un texte au choix en dessous de ces colonnes.
- Des couleurs variés.
- Le logo doit être composé de cercles
- Si 2 cercles se superpose l’intersection de ses deux cercles doit être colorisé

Réalisation de l’algorithme :

L’ensemble des actions est effectué par l’ordinateur, il n’y a aucune interaction manuel.

- paramètre la tortue
- la tortue se positionne a l’emplacement voulu
- choisit la couleur de remplissage
- la tortue se met en mode écriture
- la tortue se met en mode remplissage de zone
- répéter 4 fois
    avancer de la longueur voulu
    tourne a gauche de 90°
- termine le remplissage
- lève le crayon
- la tortue génère 2 cercles horizontales ainsi que l’intersection entre les deux cercles
- la tortue se positionne a l’emplacement voulu
- choisit la couleur d’écriture
- la tortue écrit le texte voulu avec la bonne taille et le bon style
- répète 7 fois
    la tortue génère une colonne de cercles qui diminue a chaque boucle et trace l'intersection

Choix des constantes :

```
POSITION_CERCLE_DEPART_HAUT_X = -270 # definit la position x du premier cercle du haut
POSITION_CERCLE_DEPART_HAUT_Y = -25 # definit la position y du premier cercle du haut
RAYON_CERCLES_HAUT = 22 # definit le rayon des cercles du haut
DISTANCE_ENTRE_CERCLE_HAUT= 22 # definit la distance séparant les cercles du haut
NOMBRE_DE_CERCLE_HAUT_MAXIMUM = 7 # definit le nombre de cercles du haut
ALIGNEMENT_HAUT = "vertical" # definit l'alignement du haut
AUGMENTATION_DISTANCE_ENTRE_LES_CERCLES_HAUT = 5 # augmentation de la distance du haut
R_COULEUR_CERCLE_ORIGNE_HAUT = 0.0 # couleur r (pourcentage de rouge) des cercles du haut
G_COULEUR_CERCLE_ORIGNE_HAUT = 1.0 # couleur g (pourcentage de vert) des cercles du haut
B_COULEUR_CERCLE_ORIGNE_HAUT = 0.0 # couleur b (pourcentage de bleu) des cercles du haut
ECARTEMENT_ENTRE_COLONNE_DE_CERCLE_HAUT = 70 # ecartement entre 2 colonnes du haut


POSITION_CERCLE_DEPART_BAS_X = -50 # definit la position x du premier cercle bas
POSITION_CERCLE_DEPART_BAS_Y = -240 # definit la position y du premier cercle bas
RAYON_CERCLES_BAS = 100 # definit le rayon des cercles bas
DISTANCE_ENTRE_CERCLE_BAS = 100 # definit la distance separant les cercles bas
NOMBRE_DE_CERCLE_BAS_MAXIMUM = 2 # definit le nombre de cercles bas
ALIGNEMENT_BAS = "horizontal" # definit l'alignement bas
AUGMENTATION_DISTANCE_ENTRE_LES_CERCLES_BAS = 0 # augmentation de la distance bas
R_COULEUR_CERCLE_ORIGNE_BAS = 1.0 # couleur r (pourcentage de rouge) des cercles du bas
G_COULEUR_CERCLE_ORIGNE_BAS = 0.5 # couleur g (pourcentage de vert) des cercles du bas
B_COULEUR_CERCLE_ORIGNE_BAS = 0.0 # couleur b (pourcentage de bleu) des cercles du bas


NOM_DU_LOGO = "Antoine" # nom du logo
TAILLE_DU_TEXTE = 36 # taille du logo
POSITION_DEPART_NOM_LOGO_X = -75 # definit la position x du logo
POSITION_DEPART_NOM_LOGO_Y = -175 # definit la position y du logo
COULEUR_PRENOM = '#00FFC5' # couleur du logo


POSITION_DEPART_CARRE_X = -250 # definit la position x du carre
POSITION_DEPART_CARRE_Y = -250 # definit la position y du carre
LONGUEUR_COTE_CARRE = 500 # longueur du cote du carre
COULEUR_DU_CARRE = '#0000FF' # couleur du carre
```

Définition de chaque fonction :

```
def parametrage_tortue():

    """
    Optimise les règlages de la tortue pour être plus rapide
    Args :
    aucun
    """

def tracer_un_cercle(cercle_pos_dep_x,cercle_pos_dep_y,cercle_rayon):

    """
    trace un cercle, de départ cercle_pos_dep_x et cercle_pos_dep_y et de rayon cercle_rayon
    Args :
    cercle_pos_dep_x (int) : position du départ du cercle x
    cercle_pos_dep_y (int): position du départ du cercle y
    cercle_rayon (int) : rayon du cercle
    """

def trace_intersection_complete_Cos(cercle_pos_dep_x,cercle_pos_dep_y,
                                    cercle_rayon,distance_entre_les_cercles,
                                    alignement_des_cercles):

    """
    trace l'intersection de 2 cercles qui se coupe soit horizontalement soit verticalement avec comme position
    de départ cercle_pos_dep_x et cercle_pos_dep_y, le rayon des 2 cercles cercle_rayon, la distance entre les
deux cercles
    distance_entre_les_cercles et l alignement des cercles(vertical ou horizontal).

    Args :
    cercle_pos_dep_x (int) : position du départ du cercle x
    cercle_pos_dep_y (int) : position du départ du cercle y
    cercle_rayon (int) : rayon du cercle
    distance_entre_les_cercles (int) : la distance séparant les deux cercles
    alignement_des_cercles (str) : alignement soit vertical soit horizontal
    """

def groupement_cercle(cercle_pos_dep_x,cercle_pos_dep_y,
                      cercle_rayon,distance_entre_les_cercles,nombre_de_cercle
                      ,alignement_des_cercles,
                      augmentation_distance_entre_cercle,
                      r_couleur_cercle,g_couleur_cercle,b_couleur_cercle):

    """
    trace  une colonne ou une ligne (alignement_des_cercles) de 1 à nombre_de_cercle cercles à partir de la
position cercle_pos_dep_x et cercle_pos_dep_y de rayon cercle_rayon,
    chaque cercle et espacé d'une distance_entre_les_cercles plus augmentation_distance_entre_cercle
multiplié par (1 à nombre_de_cercle) cercles
```

```
    et sont de couleur r_couleur_cercle , g_couleur_cercle , b_couleur_cercle avec une augmentation de 0.15 si celui-ci est inférieur à 1
    ainsi que les intersections lorsque qu'il se croise leur couleur est aussi en fonction de r_couleur_cercle, g_couleur_cercle, b_couleur_cercle
    Args :
    cercle_pos_dep_x (int) : position du départ du cercle x
    cercle_pos_dep_y (int) : position du départ du cercle y
    cercle_rayon (int) : rayon du cercle
    distance_entre_les_cercles (int) : la distance initiale séparant les cercles
    nombre_de_cercle (int) : nombre de cercles composant la série
    alignement_des_cercles (str) : alignement soit vertical soit horizontal
    augmentation_distance_entre_cercle (int) : facteur d augmentation de la distance entre les cercles
    r_couleur_cercle (float) : couleur r (pourcentage de rouge) initial
    g_couleur_cercle (float) : couleur g (pourcentage de vert) initial
    b_couleur_cercle (float) : couleur b (pourcentage de bleu) initial
"""
```

Test retenu :

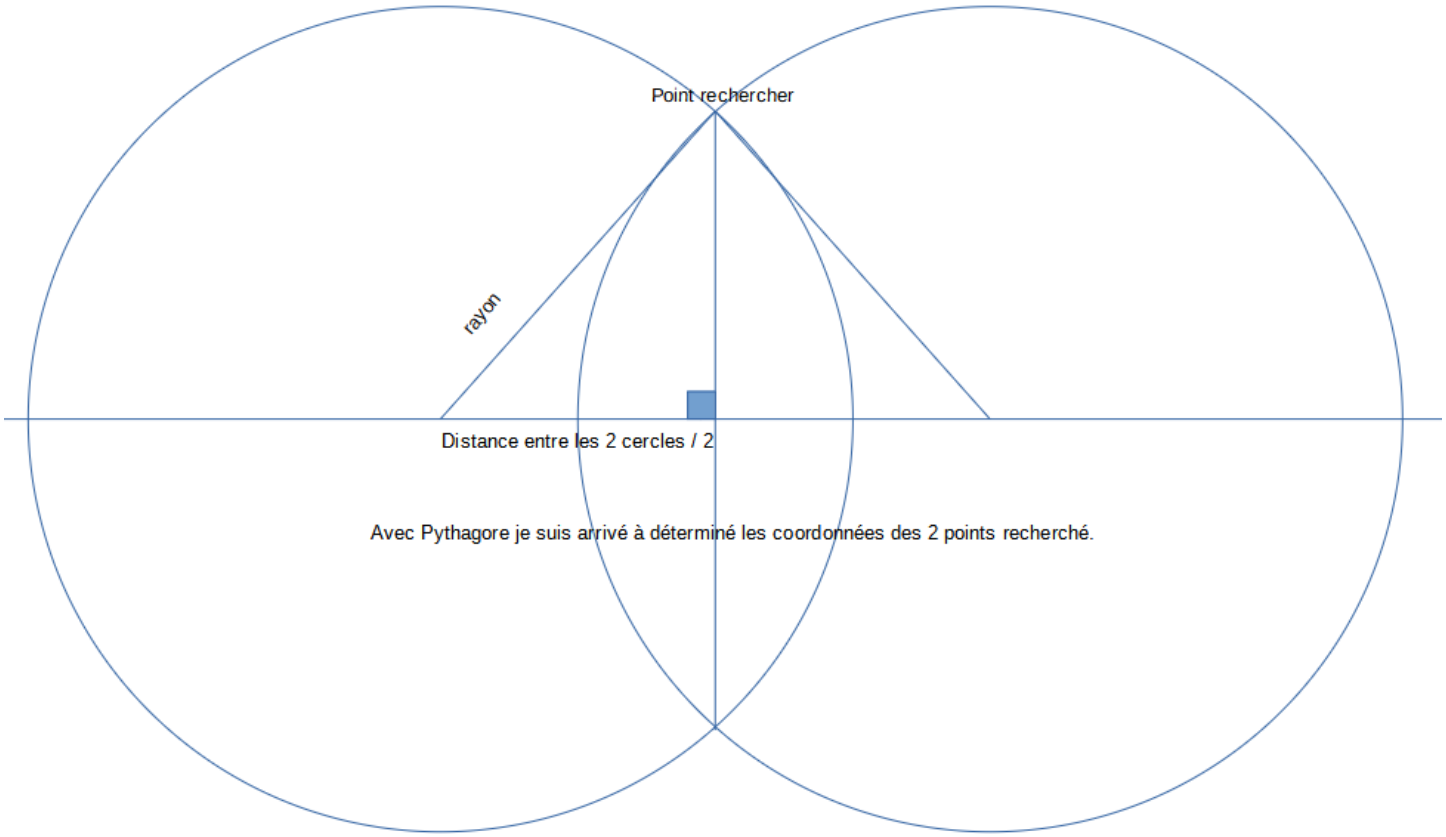
Le programme n'ayant besoin d'aucun paramètre et d'aucune interaction manuel, il suffit de l éxécuter. Résultat obtenu :

Problème rencontré :

Comment colorier l'intersection des deux disques ?

La première solution était d'utiliser turtle.setfillopacity mais les résultats ne correspondais pas à mes attentes.

La deuxième solution était de calculer les points d intersections et parcourir le cercle jusqu'au premier point, de lancer le remplissage et de parcourir jusqu'au deuxième point. La solution fonctionné mais rencontré 2 problèmes :
- la précision des points qui obligé a parcourir le cercle degrès par degrès.
- le temps d'éxécution qui était très long



La troisième solution qui fut le bonne a été de determiner les angles des deux points de la zone.