Unité 8 : Module Turtle avec Python

Application : Assemblons nos connaissances

Dans cette leçon, vous allez assembler les connaissances vues dans les leçons précédentes afin de réaliser un script permettant de concevoir vous-même la réalisation d'un projet.

Objectifs :

- Appliquer les notions vues dans les leçons précédentes.
- Réaliser une figure complexe nécessitant de la programmation fonctionnelle.

Dans les leçons précédentes, nous avons vu que l'utilisation de la librairie Turtle dans un script Python afin de construire une figure géométrique nécessitait fréquemment la mise en place de calculs.

La clarté du script impose que celui-ci soit commenté, mais également que chaque tâche complexe soit décomposée en un certain nombre de tâches élémentaires.

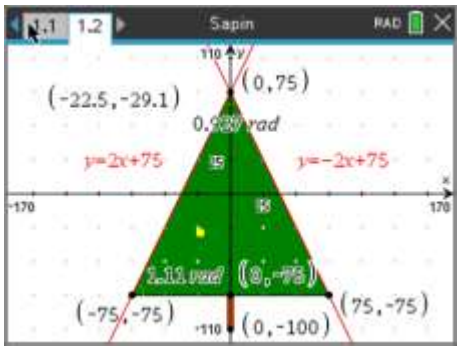
Dans cette leçon, nous allons réaliser une décoration de Noël, le programme principal appelant des fonctions pour :

- Dessiner et colorier un triangle isocèle.
- Dessiner le pied du sapin.
- Tracer et colorier une étoile.
- Décorer et animer le sapin.

Conseil à l'enseignant : La programmation fonctionnelle favorise auprès des étudiants réunis en groupe, la réalisation de projets complexes. Chaque groupe travaillant à la réalisation d'un élément particulier de celui-ci.

Mise en place du projet

Tout projet comporte une sorte de cahier des charges. Pour celui-ci utilisant le module Turtle et affichant un objet graphique, les contraintes sont aussi celles de la taille de l'écran (grille d'unité 25 pixels).



- La décoration est constituée de points de couleurs **t.dot(diamètre)**, la taille et la position étant aléatoires.
- Les points de couleurs doivent être contenus dans le triangle isocèle.
- La représentation préalable à l'aide d'un logiciel de géométrie dynamique sur lequel le repère orthonormal est identique à la grille de la librairie Turtle facilite le travail de recherche.
- Ainsi les points de coordonnées (-75 , 75) ; (75 , -75) et (0 , 75) seront les sommets du triangle.
- Les droites d'équation réduites $y = 2x + 75$, $y = -2x + 75$ et $y = -75$ délimitent le plan de représentation des points de couleur.

Conseil à l'enseignant : Selon les souhaits, il est possible de réaliser un sapin dont la largeur de la base est variable $(-a, b)$ étant les coordonnées du premier point, les équations des droites, les mesures des angles peuvent être calculés dans chaque fonction. On peut aussi envisager de faire varier la position de l'origine du tracé et représenter plusieurs sapins sur le même graphique.

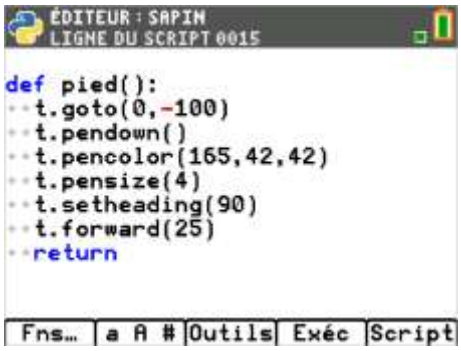
Création du script

- Commencer un nouveau script et le nommer SAPIN.
- Appuyer sur **Fns…** puis dans les modules complémentaires (**F4**), sélectionner le module Turtle.

- Importer également le module **time** dont nous aurons besoin pour réaliser une petite animation.
- Importer le module **random** pour le calcul aléatoire des positions et des nombres des objets colorés.

Dessin du pied du sapin

- Le dessin du pied commence au point de coordonnées (0,-100) **t.goto(0,-100)**.
- Il a pour longueur 25 pixels **t.forward(25)**.
- Afin de le tracer le plus simplement possible, choisir la taille la plus épaisse du crayon **t.pensize(4).**
- Orienter la tortue vers les $y > 0$ **t.setheading(90)**.
- La couleur du crayon est un marron correspondant au code RVB suivant (165, 42, 42) et l'instruction pour l'utiliser est **t.pencolor( )**.



Toutes les instructions sont disponibles en appuyant sur **Fns…**, puis **Modul** et en choisissant **8 : Turtle** (Voir Compétence 1).

Dessin de l'étoile

Pour la fonction **etoile( )**, nous aurions pu rappeler le script de la **compétence 1** au cours de laquelle nous avons dessiné des polygones réguliers. Le rappel d'un autre script s'effectue à l'aide de l'instruction
from SCRIPT import* disponible dans le catalogue.

- L'étoile est fixée au sommet du sapin au point de coordonnées (0, 75).
- Les branches sont remplies **t.fillcolor( )** d'une teinte orangée de code RVB (255,127,0).
- L'instruction **t.fillcolor( )** termine la coloration des branches de l'étoile.
- À chaque itération, déplacer la tortue de 30 pixels vers l'avant et la déplacer à droite de 144 degrés.
- Cela constituera un angle de **36** degrés à l'intérieur d'une étoile**.**
- **5** itérations constitueront parfaitement une étoile.

Conseil à l'enseignant : Comment réaliser une étoile pleine ?
Il s'agit en fait d'un polygone pour lequel on doit tracer 2 lignes pour chaque côté. Les angles sont alors incrémentés de 72 (360/5).

Le script proposé ci-contre réalise le tracé d'une étoile pleine à l'aide d'une fonction.
Selon le temps accordé à votre projet, vous pourrez choisir le type d'étoile à placer au sommet de votre sapin.

Dessin du sapin

L'enveloppe de celui-ci est constituée d'un simple triangle isocèle comme nous l'avons vu dans la mise en place du projet.

L'ensemble des mesures (valeurs des angles, longueur des côtés..) sont issues de ce travail préparatoire.

Les valeurs des angles mesurées en radian lors de l'utilisation du logiciel de géométrie dynamique sont converties en degrés.

Le triangle isocèle est rempli de couleur « vert sapin », dont le code RVB est : (1, 121, 111).

Conseil à l'enseignant : la non utilisation de l'instruction **degrees( )** de la librairie **math…** est volontaire. Les jeunes élèves doivent être capables de convertir une mesure d'angle d'une unité vers une autre.

Dessin des boules de Noël

- On dispose de k boules de couleur R, V, B à placer à l'intérieur du triangle isocèle. Chaque composante de couleur est un nombre entier aléatoire compris entre 0 et 255 (Le codage d'une composante étant sur 8 bits, soit 2^8 possibilités).
- La taille d'une boule de couleur est aléatoirement choisie entre 5 et 10 pixels
- L'instruction conditionnelle permet de limiter l'emplacement de chaque décoration à l'intérieur du triangle isocèle. (voir **mise en place du projet**).
- Si on souhaite mettre des guirlandes à la place de boules de Noël, supprimer ou commenter l'instruction **t.penup( )** de la fonctions **boules( )**.

Le programme principal

- Chaque fonction élémentaire est exécutée, il n'y a pas d'ordre particulier.
- La boucle permet de réaliser une petite animation de 6 secondes
- L'instruction **t.hideturtle( )** cache la tortue et permet d'obtenir une belle image.
- L'instruction **t.done( )** laisse l'image affichée. L'appui sur la touche **annul** réaffichera le **prompt** >>>.