Dessin à l'aide de turtle en mode itératif

- **turtle** est un module de Python permettant de faire avancer une « pointe de crayon » portée par une « tortue » sur un « feuille ».
- Cette feuille est munie d'un repère dont l'origine a pour coordonnées (0, 0) situé au centre.
- Il est possible de dire à la tortue d'aller à un point de coordonnées (-30, 150) par exemple.
- Il est possible de lui dire si elle doit lever la pointe de crayon (donc elle avancera sans tracer de trait) ou de baisser la pointe de crayon (donc elle avancera en traçant).
- Il est possible de lui dire d'avancer de 25 pixels avec `forward(25)` ou de reculer de 10 pixels avec `backward(10)`
- Il est possible de lui dire de tourner à gauche d'un angle de 45° avec `left(45)` ou de tourner à droite d'un angle de 30° avec `right(30)` etc.

Principales fonctions de la bibliothèque turtle

- Importation : `from turtle import *`
- La fenêtre à dessin est munie d'un repère orthonormé, centré au milieu de la fenêtre.

Le point de coordonnées (0, 0) est le centre de la fenêtre, c'est le point de départ, par défaut, de tout tracé.

`reset()`	On efface tout et on recommence
`up()`	Relever le crayon (pour pouvoir le déplacer sans dessiner)
`goto(x, y)`	Déplacer le crayon au point de coordonnées (x, y)
`down()`	Abaisser le crayon (pour recommencer à dessiner)
`forward(a)` ou `fd(a)`	Avancer de a pixels
`backward(a)` ou `bk(a)`	Reculer de a pixels
`color('couleur')`	'couleur' peut être une chaîne prédéfinie ('red', 'blue', 'green', etc.)
`left(angle)` ou `lt(angle)`	Tourner à gauche d'un angle donné (exprimé en degrés)
`right(angle)` ou `rt(angle)`	Tourner à droite
`width(épaisseur)`	Choisir l'épaisseur du trace

`fillcolor('red')`
`begin_fill()`
figure fermée
`end_fill()`
	Remplir un contour fermé à l'aide de la couleur sélectionnée,
	ici rouge.

`write("texte")`	<texte> doit être une chaîne de caractères délimitée par des " ou des '.
`exitonclick()`	Fin du programme, fermeture de la fenêtre graphique d'un clic à l'intérieur

1 Apprentissage

1.1 Aperçu avec le yin et le yang

1. Sur votre Raspberry, dans votre dossier *jupyter_notebook_raspberry*, créez un dossier *turtle*.

2. Enregistrez dedans le fichier **apprentissage_turtle_yin_yang.ipynb**

- Exécutez le code.
- Observez le dessin réalisé par la tortue.
- Pour terminer l'exécution du programme, cliquez dans la fenêtre feuille de dessin.

3. Notez bien l'organisation générale du code :

```
# Importation du module turtle
from turtle import *


#  Code boilerplate
try:
    reset()
except Terminator:
    pass

# ----------------- Code Turtle ICI ---------
...
# -------------------------------------------

exitonclick()
```

1.2 Observation du dessin d'un polygone

Enregistrez dans le dossier *turtle* le fichier **apprentissage_polygone_regulier_mode_iteratif.ipynb**

- Exécutez le code.
- Observez le dessin réalisé par la tortue.
- Pour terminer l'exécution du programme, cliquez dans la fenêtre feuille de dessin.
- Faites varier quelques instructions et exécutez à nouveau le code pour voir les effets sur le dessin.

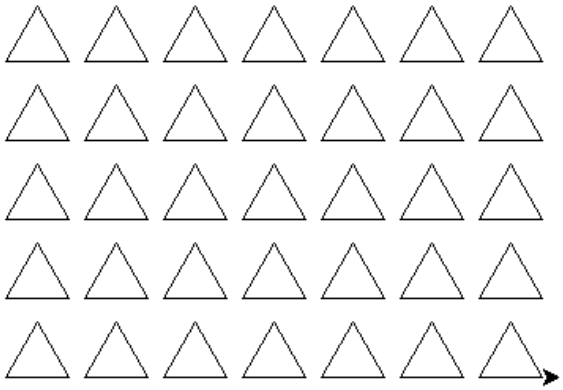
2 Entrainement

- Créez un nouveau jupyter notebook et nommez-le **entrainement_rangees_triangles_mode_iteratif.ipynb**
- Importez la bibliothèque turtle

1. Créez une fonction « triangle_equilateral(c) » qui dessine un triangle équilatéral de côté c.
2. En partant du point (-200, 0), dessinez une ligne de 5 triangles équilatéraux de côté 50 pixels, espacés de 10 pixels, comme ci-dessous :

3. Créez une fonction « ligne_triangles(n, c) » qui dessine une ligne de n triangles équilatéraux de côtés c. Cette fonction devra impérativement faire appel à la fonction « triangle_equilateral(c)».
4. Créez une fonction « p_lignes_triangles(p, n, c) » qui dessine p lignes de n triangles équilatéraux de côtés c. Cette fonction devra impérativement faire appel à la fonction « ligne_triangle(n, c)».

Exemple : p_lignes_triangles(5, 7, 40) dessinera :



L'interligne sera choisi à votre convenance pour éviter le chevauchement.