TP - Affichage de formes géométriques

Préparation

Le code fournit est une base permettant de dessiner les formes géométriques sur un canvas de pixels.

La classe `Paint` contient les données de chaque pixel, avec les méthodes `set_pixel` pour modifier un pixel, et `show` pour afficher le résultat.

À partir de cela, votre tâche est d'implémenter des *classes* représentant des formes géométriques, en vous aidant des classes prédéfinies:

- `Point` est une classe représentant un point à partir de ses coordonnées
- `Ligne` est une classe représentant la ligne rejoignant 2 points, permettant de les dessiner

La liste des points décrivant les sommets d'un polygone seront donnés dans l'ordre suivant:

- **A**: En bas à gauche
- **B**: En haut à gauche
- **C**: En haut à droite
- **D**: En bas à droite

1 - Polygone

1. Créer une classe `Polygone`, son constructeur prends *4 points* en paramètres

Associer à cette classe 4 objets `Ligne` correspondant aux lignes entre chacuns de ces points.

2. Créer une méthode `draw` qui appelle la méthode `draw` de chacune des lignes

3. Tester à partir des points:

- `A = (50,  300)`
- `B = (100,  50)`
- `C = (200, 150)`
- `D = (150, 250)`

2 - Parallélogramme

1. Créer une classe `Parallelogramme`, qui hérite de la classe `Polygone`.

Son constructeur prends *3 points* en paramètres

À partir des coordonnées des 3 premiers points, calculer les coordonnées du 4e point:
- Sa coordonnée D_x est définie par: $D_x = C_x + (A_x - B_x)$
- Sa coordonnée D_y est définie par: $D_y = C_y + (A_y - B_y)$

2. Tester et valider le fonctionnement à partir des points

- `A = (50,  300)`
- `B = (100,  50)`
- `C = (200, 150)`

`D` devra avoir pour coordonnées `(150, 400)`

3 - Rectangle

1. Créer une classe `Rectangle` qui hérite de la classe `Parallelogramme`, son constructeur prends 3 paramètres:

- 2 points A et B
- La longueur séparant B et C, notée BC ici

2. Dans la fonction `__init__`, calculer les coordonnées du 3e point C:

Calculer la distance AB séparant le point A au point B

$$C_x = B_x + \mathrm{BC} \times \frac{A_y - B_y}{\mathrm{AB}}$$
$$C_y = B_y - \mathrm{BC} \times \frac{A_x - B_x}{\mathrm{AB}}$$

3. Enfin, initialiser la classe parente `Parallelogramme` en lui passant en paramètre les 3 points A, B et C.

4. Tester et valider le fonctionnement à partir des valeurs

- `A = (50, 350)`
- `B = (80,  25)`
- `BC = 800`

4 - Carré

1. En suivant le même principe, créer une classe `Carre`

Note: Un carré est un rectangle dont chaque côté est égal

2. Tester et valider le fonctionnement à partir des points

- `A = (50, 350)`
- `B = (80, 25)`

`D` devra avoir pour coordonnées `(375, 380)`

5 - Triangle quelconque

1. Créer une classe `Triangle`, dont le constructeur attends 3 paramètres de type `Point`,

et créé des `Ligne` entre ces points.

2. Créer sa fonction `draw` affichant chacune de ces lignes

3. Tester et valider le fonctionnement

6 - Triangle rectangle

1. Créer une classe `TriangleRectangle` héritant de la classe `Triangle`

Son constructeur prends en paramètre 2 points A et B, et un angle θ en degrés

Dans le constructeur, nous allons calculer les coordonnées du point C tel que ABC forme un triangle rectangle

2. Convertir l'angle θ en radians

Calculer la distance entre le point B et C en utilisant la formule $\mathrm{BC} = \frac{\mathrm{AB}}{\cos(\theta)}$

Calculer la distance entre le point A et le point C en utilisant la formule $\mathrm{AC} = \mathrm{BC} \times \sin(\theta)$

On peut alors calculer les coordonnées du point C:

$C_x = A_x + \mathrm{AC} \times \frac{B_y - A_y}{\mathrm{AB}}$

$C_y = A_y - \mathrm{AC} \times \frac{B_x - A_x}{\mathrm{AB}}$

3. Initialiser alors la classe parente `Triangle` à partir des 3 points.

4. Tester et valider le fonctionnement de cette classe

7 - Triangle isocèle

1. Créer une classe `TriangleIsocele` héritant de la classe `Triangle`

Son constructeur prends en paramètre 2 points A et B formant la base, et un angle θ en degrés correspondant aux angles aux extrêmités

2. Calculer les coordonnées du point au milieu du segment AB, que l'on appelera M

Créer un `TriangleRectangle` sur les points M, B, et avec l'angle θ

Ce triangle rectangle correspond à "la moitié" de notre triangle isocèle

3. Initialiser la classe parente `Triangle` avec les points A, B et le point C

qui a été calculé lors de l'initialisation du triangle rectangle

8 - Triangle rectangle isocèle

En réutilisant les composants précédents, créer une classe `TriangleRectangleIsocele`

Un triangle rectangle isocèle est un triangle isocèle dont les angles aux extrêmités ont une valeur de 45°

9 - Triangle équilatéral

De même, créer une classe `TriangleEquilateral`, qui est un triangle isocèle dont les angles aux extrêmités ont une valeur de 60°

10 - Calcul de périmètre

Pour chacune des classes parentes, créer une méthode `perimeter` qui retourne la somme de la longueur de chaque ligne composant le polygône

Valider le fonctionnement avec des cas simples (carré, triangle équilatéral, etc...)

11 - Calcul d'aire

Pour chacune des classes parentes, créer une méthode `area` qui retourne l'aire de la forme créée

Note: On ignorera le cas du polygône quelconque, son calcul est trop complexe pour le moment

Aire d'un triangle

Calculer un nombre $S = \frac{\mathrm{AB} + \mathrm{BC} + \mathrm{AC}}{2}$

L'aire du triangle ABC est alors $\sqrt{S \times (S - \mathrm{AB}) \times (S - \mathrm{BC}) \times (S - \mathrm{AC})}$

Note Cette méthode alternative est appelée **formule de Héron**

Aire d'un parallèlogramme

Un parallèlogramme ABDC peut être divisé en deux triangles ABC et BDC égaux.

L'aire du parallèlogramme est alors 2 fois l'aire du triangle ABC