Exercice 1. Classe Rectangle :

1. Ecrire une classe Rectangle en langage Python, permettant de construire un rectangle doté d'attributs **longueur**(0 par défaut) et **largeur**(0 par défaut) **.**
2. Créer une méthode Perimetre() permettant de calculer le périmètre du rectangle et une méthode Surface() permettant de calculer la surface du rectangle
3. Créer la méthode qui permet d'afficher à l'écran les caractéristiques de ce rectangle grâce à un print()
4. Créer deux instances de cette classe : une avec les attributs par défaut l'autre avec 1 et 1
5. Appliquer la méthode Perimetre() et utiliser la puis utiliser un print() pour afficher les caractéristiques des rectangles

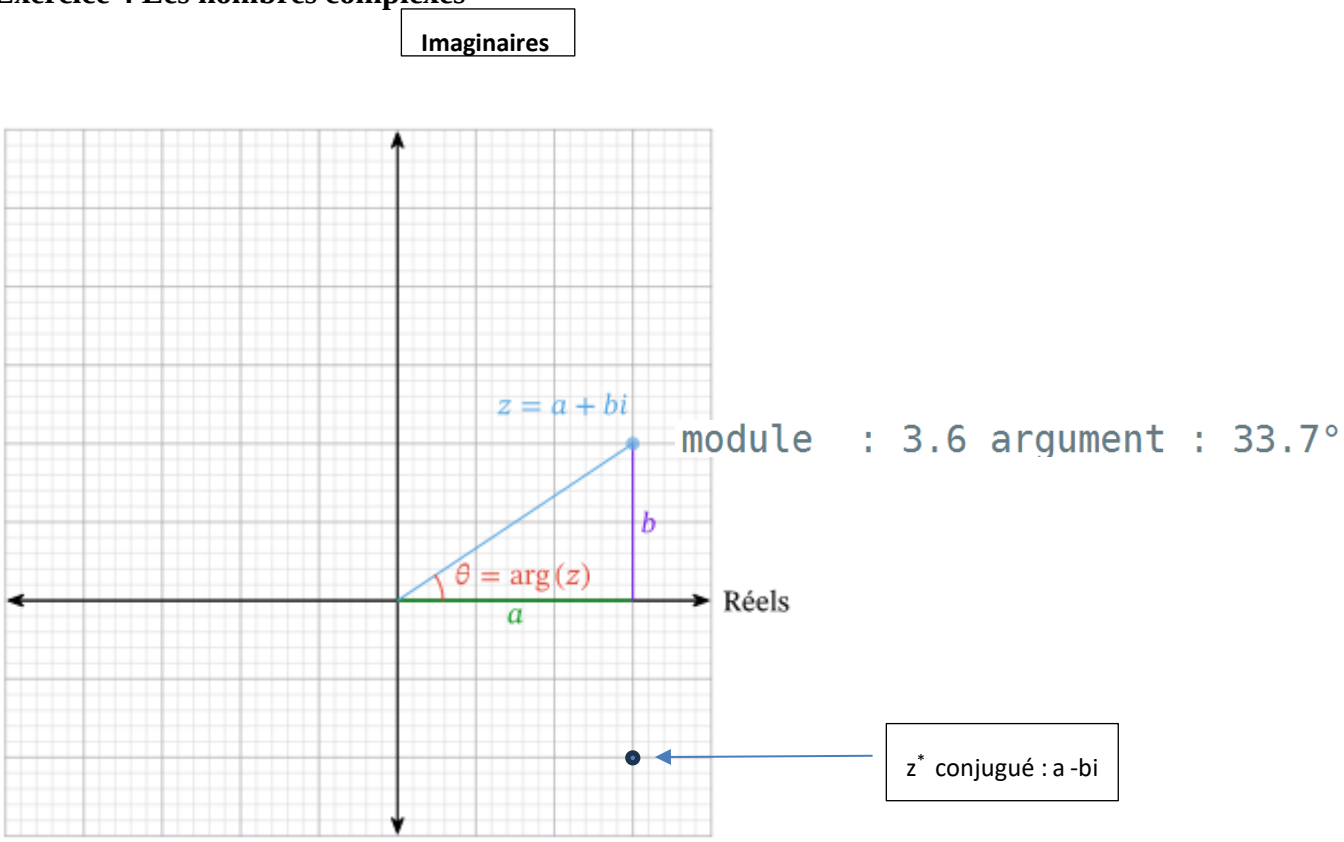
Exercice 2. Classe Cercle :

1. Définir une classe Cercle permettant de créer un cercle C(O,r) de centre O(a,b) et de rayon r à l'aide d'un constructeur paramétré.
2. Définir une méthode Surface() de la classe qui permet de calculer la surface du cercle
3. Définir une méthode Perimetre() de la classe qui permet de calculer le périmètre du cercle
4. Créer deux instances de cette classe : une avec les attributs par défaut l'autre avec 1 et 1
5. Appliquer les méthodesPerimetre()et Surface() et utiliser les puis utiliser un print() pour afficher les caractéristiques des cercles

Exercice 3. Classe Compte bancaire :

1. Créer une classe Python nommée CompteBancaire qui représente un compte bancaire, ayant pour attributs : numeroCompte (type numérique) , nom (nom du propriétaire du compte du type chaine), solde.
2. Créer un constructeur ayant comme paramètres : numeroCompte, nom, solde.
3. Créer une méthode Versement() qui gère les versements.
4. Créer une méthode Retrait() qui gère les retraits.
5. Créer une méthode Agios() permettant d'appliquer les agios à un pourcentage de 5 % du solde
6. Créer une méthode afficher() permettant d'afficher les détails sur le compte
7. Donner le code complet de la classe CompteBancaire.

Exercice 4 Les nombres complexes



from math import sqrt , atan,pi

1 Définir un objet Complex qui possède une partie réelle" re" et une partie imaginaire " im" que l'on peut choisir mais qui sont à 0 par défaut.

2 Coder la méthode **module** qui renvoie le module du nombre complexe.

On rappel l'argument se calcule par :

$$\theta = \begin{cases} \arctan\left(\frac{b}{a}\right) & \text{si } a > 0 \\ \arctan\left(\frac{b}{a}\right) + \pi & \text{si } a < 0. \end{cases}$$

3 Coder la méthode **conjugue** qui renvoie le conjugué du nombre complexe

Le complexe conjugué du nombre complexe z = a + ib est a − ib. Il est noté z ou z*.

4 Prévoyez une méthode **argument** qui affiche le module et l'argument en ° quand on print un objet Complex

```
C1 = Complex(0,-1)
print(C1)
```

affiche

```
module  : 1.0 argument : -90.0°
```

```
C1 = Complex(1,0)
print(C1)
```

```
module  : 1.0 argument : 0.0°
```

Exercice 5 Mini jeux de Rôle

1 Définir un objet "Joueur" qui permet de créer un joueur dont on peut choisir le nom qui possède 100 points_de_vies et 100 points_de_mana

2 Définir une méthode **est_attaque** qui prend comme argument un autre Joueur et qui enlève 5 point de vie .

3 Définir une méthode **recupere qui** permet au joueur de récupérer 10 points de vie

4 Définir un Objet Guerrier qui hérite de Joueur.

```
class Guerrier(Joueur):
    def __init__(self,nom):
        Joueur.__init__(self,nom)
```

On modifie les points de vie à 120 et les points de mana à 20

Redéfinir la méthode est_attaque pour enlever 15 points de vie et recupere + 20

5 Même chose avec un nouvel Objet Magicien attaque -30 et recupère +5

6 Faites battre tout ce petit monde