But

Le but est de proposer une classe NbComplexe qui donne des méthodes pour calculer avec des nombres complexes.

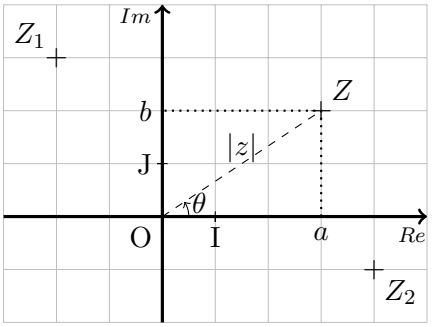
La problématique

Dans un plan $\mathscr{P}$ muni d'un repère orthonormé, un point $Z(a,b)$ est défini par son abcisse et son ordonnée.

On décide de s'autoriser des calculs algébriques sur Z en écrivant Z comme le nombre $z = a + bi$.

Un tel ensemble de nombres se nomme l'ensemble des nombres complexes (noté $\mathbb{C}$).
Rq : lorsque $b = 0$, on retrouve les nombres réels.

Pour tous les exemples on prendra $z_1 = -2 + 3i$ et $z_2 = 4 - i$.



◇ Considérons $Z(a,b)$ et $z = a+bi$. L'abscisse du point Z est la partie réelle du nombre z tandis que l'ordonnée du point Z est la partie imaginaire du nombre z.
expl : z_1 est un nombre complexe de partie réelle -2 et de partie imaginaire 3.

◇ Deux nombres complexes sont égaux s'ils ont même partie réelle et même partie imaginaire.
expl : $z_1 \neq z_2$ car ils n'ont pas leur partie réelle égale (ni même leur partie imaginaire)

◇ le conjugué d'un nombre complexe $z = a + bi$ est $\bar{z} = a - bi$.
expl : $\bar{z_1} = -2 - 3i$

◇ Deux nombres complexes s'additionnent (resp. se soustraient) en additionnant (resp. soustrayant) les parties réelles entre elles et les parties complexes entre elles.
expl : $z_1 + z_2 = (-2 + 3i) + (4 - i) = (-2 + 4) + (3 - 1)i = 2 + 2i$
Rq : si les deux nombres sont réels, on retrouve l'addition (resp. soustraction) réelle.

◇ Deux nombres complexes se multiplient ainsi $(a + ib)(c + id) = ac - bd + (ad + bc)i$
expl : $z_1 z_2 = (-2 + 3i)(4 - i) = (-2 \times 4 - (3 \times -1)) + (-2 \times (-1) + 3 \times 4)i = -5 + 14i$
Rq : si les deux nombres sont réels, on retrouve la multiplication réelle. On remarque également que $i^2 = -1$.

◇ le module de $z = a + bi$ est le réel $|z| = \sqrt{a^2 + b^2}$ (dans un repère orthonormé, il s'agit de la distance OZ).
expl : $|z_1| = \sqrt{(-2)^2 + (3)^2} = \sqrt{13}$

◇ l'argument de z est l'angle radian θ que fait $\overrightarrow{OZ}$ avec $\overrightarrow{OI}$.

◇ enfin on proposera une relation d'ordre compatible pour l'addition et la multiplication en posant que $z = a + bi < z' = a' + b'i$ si $a < a'$ ou si $a = a'$ et $b < d'$
expl : $z_1 < z_2$ car $-2 < 4$

Exemple / aide

◇ Pour déterminer l'argument de z, on adaptera à la classe `NbComplexe` la fonction suivante :

```
def argument(a : float, b : float) -> float :
    from math import atan, sqrt, pi
    m = sqrt( a**2 + b**2 )
    assert m != 0, "argument : zéro ne possède pas d'argument"
    if a < 0 and b == 0 :
        return pi
    else :
        return 2*atan( b / ( m + a ) )
```

◇ Afin d'afficher un nombre avec 3 chiffres après la virgule, on peut faire

```
val = 90.12345
n = math.round ( val * 1000 )
print( n / 1000 )
```

◇ On peut importer le module `projet02_pygame.py` via la ligne `import projet02_pygame as ppg` .

Étapes

NbComplexe
re (float)
im (float)

a) Télécharger le fichier `projet02_pygame.py` sur le site [forhan.maths.free.fr].

b) Proposer une classe `NbComplexe` dont les objets auront deux attributs flottant : `re` (la partie réelle) et `im` (la partie imaginaire).

c) Définir la méthode `get_re() -> float` (resp. `get_im() -> float`) qui renvoie la partie réelle (resp. imaginaire).
nb : à partir de là, vous pouvez afficher le point Z en faisant l'appel `ppg.affichage(z)`.
Pour afficher plusieurs points : `ppg.affichage(z1, z2, z3)` ou `ppg.affichage( listeZ )`.

d) Définir une méthode `__repr__() -> str` qui précise comment récréer le nombre complexe.
expl : si $z_1 = -2 + 3i$, la méthode renverra la chaîne de caractères `"NbComplexe(-2,3)"`

e) Définir une méthode `__str__() -> str` qui donne la valeur du nombre complexe.
expl : si $z_1 = -2 + 3i$, la méthode renverra la chaîne de caractères `"-2 +3i"`
si $z_2 = 4 - i$, la méthode renverra la chaîne de caractères `"4 -1i"`
nb : à partir de là, vous pouvez afficher les coordonnées des points en ajoutant en dernier argument `True` *:* `ppg.affichage(z1, z2, True)`.

f) Définir la méthode `conjugue() -> 'NbComplexe'` qui renvoie le nombre conjugué.

g) Définir la méthode privée `_module() -> float` qui renvoie le module du nombre.

h) Définir la méthode privée `_argument() -> float` qui renvoie l'argument d'un nombre complexe non-nul.

i) Définir la méthode privée `_argPi() -> float` qui renvoie l'argument d'un nombre complexe non-nul en nombre de π. Si le nombre complexe est nul, elle lèvera une exception **ValueError** expliquant le problème.

j) Définir la méthode `formExp(p : int) -> str` qui renvoie une chaîne de caractères contenant le module et l'argument avec p digits de précision. Si le nombre complexe est nul, elle lèvera une exception **ValueError** expliquant le problème.
expl : pour $z_1 = -2 + 3i$, $|z_1| = \sqrt{13} \approx 3{,}605\,55$ et $\theta \approx 2{,}158\,80 \approx 0{,}687\,17\pi$
ainsi `formExp( NbComplexe(-2,3), 3 )` renverra `"3.606 exp(i 0.687 pi)"`

k) Définir la méthode `__eq__(z : 'NbComplexe') -> bool` qui définit l'égalité $==$.

l) Définir la méthode `__lt__(z : 'NbComplexe') -> bool` qui définit l'inégalité stricte $<$.

m) Définir les méthodes `__add__(z : 'NbComplexe') -> 'NbComplexe'` et `__sub__(z : 'NbComplexe') -> 'NbComplexe'` qui définissent l'addition et la soustraction complexe.

n) Définir la méthode `__mul__(z : 'NbComplexe') -> 'NbComplexe'` qui définit la multiplication complexe. Vérifier que pour des réels, on retrouve la multiplication classique et que $i^2 = -1$.

o) Définir la méthode `pow(n : int) -> str` qui renvoie le nombre complexe à la puissance n. Si $n = 0$, elle renvoie 1 et si $n < 0$ elle lèvera une exception **ValueError** expliquant le problème.
Faire un test qui montre que si $|z| = 1$, $\forall n \in \mathbb{N}$ `z.pow(n)` reste sur le cercle de centre O et de rayon 1.

p) Faire une fonction de test (tester notamment les opérations avec des réels, lever et rattraper les exceptions).

q) Approfondissement :
◇ la division d'un nombre complexe dérive d'une multiplication lorsque l'on passe par le conjugué.
expl $\dfrac{z_1}{z_2} = \dfrac{-2+3i}{4-i} = \dfrac{(-2+3i)\times(4+i)}{(4-i)\times(4+i)} = \dfrac{(-2+3i)(4+i)}{17} = \dfrac{z_1\bar{z_2}}{|z_2|^2}$
Définir ainsi la méthode `__truediv__(z : 'NbComplexe') -> 'NbComplexe'`.
Si z est nul, la méthode lèvera une erreur **ZeroDivisionError**.
◇ se renseigner sur la racine carrée et l'implémenter.

Rendu

À la fin du temps imparti, chaque groupe rendra le fichier contenant le code `python` qui contiendra également une fonction `main() -> None` avec un exemple d'utilisation et passera à l'oral.