Ma foi il est pas mal ce TP !

Table des matières


```
from math import pi
from math import cos,sin
import matplotlib.pyplot as plt
```

I Représentation des nombres complexes

Choix de la structure de données : on décide de *représenter* en machine le nombre complexe $z = a + ib$ (a, b réels) par le tuple de flottants `(a,b)`.

[Q1.] Rappeler rapidement la principale différence en termes d'opérations entre liste et tuple sous python.

[Q2.] Écrire une fonction :

1. `rpart(z)` qui prend en entrée un [tuple qui représente un] complexe z et qui renvoie en sortie sa partie réelle.
2. `ipart(z)` qui en fait autant avec la partie imaginaire.

II Opérations algébriques sur les complexes

[Q3.]

1. Écrire une fonction `somme(z1,z2)` qui prend en entrée deux nombres complexes z_1, z_2 et qui renvoie en sortie le tuple représentant le complexe $z = z_1 + z_2$
2. Écrire une fonction `produit(z1,z2)` qui prend en entrée deux nombres complexes z_1, z_2 et qui renvoie en sortie le tuple représentant le complexe $z = z_1 \times z_2$.
3. Écrire une fonction `bar(z)` qui prend en entrée un nombre complexe z et qui renvoie en sortie le tuple représentant le complexe $\bar{z}$.

[Q4.]

1. Écrire une fonction `module2(z)` qui prend en entrée un complexe z et qui renvoie en sortie le *flottant* $|z|^2$.

en termes de structures de données

2. Quel est le «problème» en programmant cette fonction par le renvoi de `produit(z,bar(z))` ? Modifier la fonction en conséquence

[Q5.] Écrire une fonction `inverse(z)` qui prend en entrée un nombre complexe z (non nul) et qui renvoie en sortie le tuple représentant le complexe $1/z$.

III Représentation géométrique des nombres complexes

[Q6.] Recopier les scirpts ci-dessous, qui serviront à tracer, pour un nombre complexe z donné, le point du plan d'affixe M et le vecteur $\overrightarrow{OM}$ du plan.

```python
def texte(z):
    """ Entrée : z (tuple de floats). Un nombre complexe.
        Sortie : (str) la chaine de caractères représentant z sous
        forme algébrique.
    """
    a,b = z
    s = b>=0
    signe = '+'*s + '-'*(1-s)
    return str(a)+signe+str(abs(b))+'i'

def dessine(z,text=''):
    """ Entrée : z (tuple de floats). Un nombre complexe.
        Sortie : aucune. Trace le  point M d'affixe z,  et le vecteur OM.
    """
    a,b = z
    plt.plot([0,a],[0,b])
    plt.annotate(text,z)
    plt.arrow(0,0,a,b,width=0.05,length_includes_head=True)
```

[Q7.] Tester les commandes ci-dessous et comparer :

```python
z = (1,-3)
dessine(z)
plt.grid('on')
plt.gca().set_aspect('equal') # orthonormé
plt.show()
```

```python
z=(1,-3)
dessine(z,texte(z))
plt.grid('on')
plt.gca().set_aspect('equal')
plt.show()
```

[Q8.] Choisissez deux nombres complexes de votre choix z_1, z_2 (ni réels, ni imaginaires purs !), calculez à la main les nombres $S = z_1 + z_2$. Tracez graphiquement les points d'affixe z_1, z_2, S. Répétez la question avec d'autres valeurs de z_1 et z_2 jusqu'à ce que vous ayez compris graphiquement l'effet de la somme de complexes.

IV Nombres complexes de module 1

[Q9.] Écrire une fonction `expi(theta)` qui prend en entrée un flottant θ et qui renvoie en sortie le [tuple représentant le nombre] complexe $e^{i\theta}$.

[Q10.] Choisir des valeurs de θ de votre choix, et tracer pour ces valeurs de θ, à l'aide de la fonction `dessine`, les points d'affixe z et $1/z$, où $z = e^{i\theta}$. Que constatez-vous ?

On fixe un nombre complexe C de votre choix, ainsi qu'un nombre réel θ, et on considère la suite de nombres complexes $(u_n)_{n\in\mathbf{N}}$ définie par récurrence de la manière suivante :

$$\begin{cases} & u_0 & = & C \\ \forall n \in \mathbf{N} & u_{n+1} & = & e^{i\theta} \times u_n \end{cases}$$

[Q11.] Écrire une fonction `u(C,theta,n)` qui prend en entrée un complexe C, un flottant θ, un entier n et qui renvoie en sortie la liste $[u_0, \dots, u_n]$ des $(n+1)$ premiers termes de la suite (u_n). Attention : chaque terme de la suite est, du point de vue informatique, un tuple.

[Q12.] Donner la forme explicite de la suite (u_n).

[Q13.] Tracer les 15 premiers termes de la suite ainsi programmée. Interpréter géométriquement la multiplication d'un nombre complexe par $e^{i\theta}$. Pour y voir clair, un jeu de constantes pertinent est par exemple :

```
C=(1,0)
tour = 2*pi
N = 5
theta =  tour/N
n = N-1
```

V Équation $z^n = 1$

Soit $n \geq 2$ un entier fixé. On s'intéresse à l'équation d'inconnue $z \in \mathbf{C}$:

$$z^n = 1 \qquad (E)$$

[Q14.] Écrire une fonction `puissance(C,n)` qui prend en entrée un complexe C, un entier n, et qui renvoie en sortie le complexe C^n.

[Q15.] Tracer la suite des 10 premières puissances d'un nombre complexe C de votre choix. On choisira des exemples couvrant les deux cas suivants : $|C| > 1$ et $|C| < 1$. Que conjecturer sur le module d'une solution de (E) ?

[Q16.] À l'aide des questions précédentes, trouver une solution particulière (distincte de 1 !) à l'équation (E).

[Q17.] Montrer (mathématiquement) que les solutions de (E) sont les nombres $1 = \omega^0$, $\omega, \omega^2, \ldots, \omega^{n-1}$ où $\omega = \exp(2i\pi/n)$. *Indication : on pourra prouver que toute solution est non nulle, puis chercher ces dernières sous forme exponentielle.*

[Q18.] Créer en python la liste $L_n = [\omega^0, \ldots, \omega^{n-1}]$:

1. À l'aide d'une boucle.
2. En compréhension.

On souhaite calculer les nombres $S_n = \sum_{k=0}^{n-1} \omega^k$ et $P_n = \prod_{k=0}^{n-1} \omega^k$

[Q19.]

1. Créer une fonction en python qui prend en entrée un entier n et qui renvoie en sortie la liste des deux nombres complexes S_n et P_n.
2. La tester pour différentes valeurs de $n \geq 2$ et formuler une conjecture.
3. Calculer mathématiquement S_n et P_n.

Lycée Chateaubriand, Rennes
Classe de Biospé 2022-2023
MYP