Programmation scientifique en Python

v2.6

Table des matières

TP – Utilisation de fonctions mathématiques

Exercice 1

Calculer en Python :

- $\sin\pi-\cos\pi$
- $\sin(\pi/2)-\cos(\pi/2)$
- $\cos(\pi)$
- $\cos(\pi+2\pi)$
- $\cos 5$
- $\cos\sqrt{5}$

Exercice 2

Implanter la fonction : $f(x,y)=\sin x-\cos y$.

Mathématiquement, $f(\pi,\pi)=1$ et $f(\pi/2,\pi)=2$.

Exercice 3

Calculer en Python :

- $2\sqrt{2}+2$
- $2\sqrt{2+2}$
- $\frac{3^3+3^5}{2}+\sqrt{3}$

Exercice 4

Implanter la fonction :

$$f(x,y,z)=\frac{x^2+\sqrt{y+2}+\cos z}{3}$$

Mathématiquement, $f(2,-2,\pi)=1$

Exercice 5

Calculer en Python :

- $2\ln 2+3$
- $2\log 2+3$

- $2\log_2 2+3$

Exercice 6

Implémenter la fonction :

$$f(x)=\frac{2\log x+3}{5}$$

Vérifiez que $f(10)=1$.

Exercice 7

Implémenter la fonction :

$$g(x,y)=\frac{5\log x}{5}+y$$

Vérifiez que :

- $g(10,7)=8$
- $g(10,8)=9$
- $g(10,501)=502$

Exercice 8

Calculer en Python :

- e^3
- $e^{\frac{3}{2}}$
- $e^{\ln 301}$
- $10^{\log 501}$

Rappel : mathématiquement on a :

- $e^{\ln 301}=301$
- $10^{\log 501}=501$

Exercice 9

Soit l'équation $y=3\ln\frac{x+2}{5}$ (pour $x>-2$).

Nous avons les équivalences :

$$y=3\ln\frac{x+2}{5}\Leftrightarrow\frac{y}{3}=\ln\frac{x+2}{5}\Leftrightarrow e^{\frac{y}{3}}=e^{\ln\frac{x+2}{5}}\Leftrightarrow e^{\frac{y}{3}}=\frac{x+2}{5}\Leftrightarrow 5.e^{\frac{y}{3}}=x+2\Leftrightarrow 5.e^{\frac{y}{3}}-2=x$$

1. Implémenter la fonction : $toto(x)=3\ln\frac{x+2}{5}$
2. Implémenter la fonction : $tata(y)=5.e^{\frac{y}{3}}-2$

   Mathématiquement, pour tout k réel positif, $toto(tata(k))=tata(toto(k))=k$.

3. Via l'interpréteur de code, l'égalité est elle vérifiée pour k=5 ? Idem pour k=101 ?

Sujet d'étude 1 : opérations binaires

Nous nous intéressons dans un premier temps aux opérations binaires ; puis étendons ces opérations pour gérer les opérations bits à bits sur les octets.

Opérations binaires

- La négation binaire NON est définie par : NON(0)=1 et NON(1)=0.
- La disjonction binaire OU est définie par : 1 OU 1 = 1, 1 OU 0 = 1, 0 OU 1 = 1 et 0 OU 0 = 0.
- La conjonction binaire ET est définie par : 1 ET 1 = 1, 1 ET 0 = 0, 0 ET 1 = 0 et 0 ET 0 = 0.

Exemple d'opérations bits à bits sur octets

- NON(00110000) = 11001111
- 00110000 OU 11000011 = 11110011
- 00110011 ET 11000011 = 00000011

Partie A : Algorithmique

1 – Opérations binaires

Fonctions

1- Réaliser la fonction ***non*** qui prend en argument un bit et retourne sa négation.

2- Réaliser la fonction ***ou*** qui prend en argument deux bits et retourne leur disjonction.

3- Réaliser la fonction ***et*** qui prend en argument deux bits et retourne leur conjonction.

2 – Opérations bits à bits sur les octets

Structure de données

Déterminer un type de donnée nécessaire pour stocker un octet (8 bits).

Fonctions

4- Réaliser la fonction ***non_oct*** qui prend en argument un octet et retourne sa négation bit à bit.

5- Réaliser la fonction ***ou_oct*** qui prend en argument deux octets et retourne leur disjonction bit à bit.

6- Réaliser la fonction ***et_oct*** qui prend en argument deux octets et retourne leur conjonction bit à bit.

Intéressez-vous à l'extension de vos résultats, pour gérer des opérations bit à bit, sur des données binaires composées de 4 octets.

Partie B : Implémentation

Implanter en Python les fonctions de la Partie A.

Définir un jeu de test qui vous servira à tester la bonne implantation des fonctions demandées.

Sujet d'étude 2 : audiences

L'audience d'un média définit l'ensemble des individus exposés à ce média ; elle fait l'objet de mesures : l'audiométrie qui relève de sondages.

La part d'audience (ou part de marché)

C'est l'indicateur le plus utilisé. Il s'agit du pourcentage des personnes ayant suivi un support (chaîne de télévision, station de radio, site Web) par rapport à l'audience globale du média (télévision, radio, Web).

Cet indicateur peut être calculé, pour une émission ou une tranche horaire, et par cible.

Ci-dessous, le top 5 des audiences de la soirée du jeudi 31 décembre 2015 (source : www.ozap.com)[1].

Chaîne	Programme	Part de marché	Téléspectateurs
france 2	LE PLUS GRAND CABARET DU MONDE (VARIETES)	28.2 %	4 015 000
TF1	LE 31 TOUT EST PERMIS AVEC Arthur (HUMOUR)	25.5 %	3 630 585
france 3	HUGO CABRET (FILM)	10.6 %	1 509 184
6	SCENES DE MENAGES (SERIE)	7.7 %	1 096 294
W9	LES 30 ANS DU TOP 50 (VARIETES)	4.1 %	583 741

Ci-dessous, un extrait issu d'un article du site www.ozap.com.

Audiences : France 2 remporte d'un rien le match du réveillon, France 3 en forme avec « Hugo Cabret »

Comme en 2010, 2011, 2012, 2013 et en 2014, France 2 est arrivée hier soir en tête des audiences de la dernière soirée de l'année 2015. Selon Médiamétrie, plus de 4 millions de téléspectateurs ont regardé hier, entre 20h55 et 1h00, "Le Grand cabaret sur son 31", la soirée de réveillon présentée par Patrick Sébastien [...]. La part de marché sur le public de quatre ans et plus s'élève à 28,2% (et 13% sur les ménagères de moins de cinquante ans). [...]

En face, près de 3,6 millions de téléspectateurs ont choisi de passer en 2016 en regardant Arthur sur TF1, entre 21h05 et 0h35. (...)

France 3 est troisième avec la première diffusion d' « Hugo Cabret ». Le long-métrage américain de Martin Scorsese (...) a attiré plus de 1,5 million de cinéphiles, soit 10,6% du public.

Dans ce qui suit, nous nous plaçons dans le contexte de la soirée du jeudi 31 décembre 2015.

Sachant que :

- 1 % de part de marché correspond à environ 142375.89 téléspectateurs ;

On trouve que :

- 1 téléspectateur correspond à environ 7.02×10^{-6}% de part de marché.

1 Les nombres des téléspectateurs sont « normalisés ».

En effet,

$$\begin{gathered}D'une\ part,\\ 1\%\ \ de\ part\ de\ marché \rightarrow 142375.89\ téléspectateurs\\ D'autre\ part,\\ x\%\ \ de\ part\ de\ marché \rightarrow 1\ téléspectateur\\ D'où,\\ 1\% \times 1 = x\% \times 142375.89\\ D'où,\\ 1 = x \times 142375.89\\ Soit,\\ x = \frac{1}{142375.89} \approx 7.02 \times 10^{-6}\end{gathered}$$

Nous obtenons les formules :

- $NbTelespectateurs(X) = 42375.89 \times X \times 100$ (X est un pourcentage de part de marché) ;

- $PartDeMarché(Y) = 7.02 \times 10^{-6} \times Y$ (Y est nombre de téléspectateurs).
  $PartDeMarché(Y)$ est en pourcentage.

En effet,

$$\begin{gathered}D'une\ part,\\ 1\% \rightarrow 42375.89\\ D'autre\ part,\ \ pour\ un\ pourcentage\ X \neq 0,\\ X \rightarrow NbTelespectateurs(X)\\ D'où,\\ 1\% \times NbTelespectateurs(X) = X \times 42375.89\\ \frac{1}{100} \times NbTelespectateurs(X) = X \times 42375.89\\ Soit,\\ NbTelespectateurs(X) = 100 \times X \times 42375.89\end{gathered}$$

En outre,

$$\begin{gathered}D'une\ part,\\ 1\ téléspectateur \rightarrow 7.02 \times 10^{-6}\%\\ D'autre\ part,\\ Y\ téléspectateur(s) \rightarrow PartDeMarché(Y)\%\\ D'où,\\ 1 \times PartDeMarché(Y)\% = Y \times 7.02 \times 10^{-6}\%\\ Soit,\\ PartDeMarché(Y) = Y \times 7.02 \times 10^{-6}\end{gathered}$$

Nous nous intéressons à l'évaluation du nombre de téléspectateurs pour une suite de parts de marché. Cette suite de parts de marché permet de représenter les parts de marché d'un ensemble de chaînes de télévision.

Exemple :

$$\langle\ \ PartDeMarché_{TF1},\ \ PartDeMarché_{France\,2},\ \ PartDeMarché_{France\,3}\ \ \rangle$$

Partie A : Algorithmique

Dans ce qui suit, nous nous plaçons dans le contexte de la soirée du 31-12-2015.

Type de donnée

Déterminer un type de donnée permettant de stocker une suite de parts de marché.

Fonctions et procédures

- Réaliser une fonction ***nb_telespectateurs*** prenant en paramètre un pourcentage de part de marché, et retournant le nombre de téléspectateurs correspondant.
- Réaliser une fonction ***part_de_marché*** prenant en paramètre un nombre de téléspectateurs, et retournant le pourcentage de part de marché correspondant.

Procédures sur la structure de donnée demandée

- Réaliser une procédure ***afficher*** prenant en argument la structure de donnée permettant de stocker une suite de parts de marché, et affichant cette structure de donnée.
- Réaliser une procédure, prenant en argument la structure de donnée permettant de stocker une suite de parts de marché, et affichant le nombre de téléspectateurs correspondant à chaque part de marché.
- Réaliser une procédure, prenant en argument la structure de donnée permettant de stocker une suite de parts de marché, et affichant la part de marché la plus élevée.
- Réaliser une procédure prenant en argument la structure de donnée permettant de stocker une suite de parts de marché, et affichant le nombre de téléspectateurs correspondant à l'ensemble de ces parts de marché.

Partie B : Implémentation

- Implanter en Python, les fonctions/procédures de la Partie A.
- Définir un jeu de test qui vous servira à tester la bonne implantation des fonctions/procédures demandées.

Sujet d’étude 3 : tournoi de jeu vidéo

Contexte d’un tournoi de jeu vidéo.

1. Comment stocker les scores des joueurs d’une équipe ?

2. Fonction prenant en argument les scores d’une équipe, et retournant le nombre de joueurs d’une équipe ?

3. Fonction prenant en argument les scores d’une équipe, et retournant Vrai si l’équipe n’a pas de joueurs ; et Faux sinon ?

4. Procédure prenant en argument les scores d’une équipe, et affichant les scores qui sont strictement positifs.

5. Procédure prenant en argument les scores d’une équipe, ainsi qu’un certain score x, et affichant les scores supérieurs ou égaux au score x.

6. Fonction prenant en argument les scores d’une équipe, et retournant la moyenne des scores.

$$\mu = \frac{1}{n} \sum_{i=0}^{n-1} scores[i]$$

(Le nombre d’équipes *n* est supposé non nul).

7. Fonction prenant en argument les scores d’une équipe, et retournant la variance des scores.

$$V = \frac{1}{n} \sum_{i=0}^{n-1} (scores[i] - \mu)^2$$

(Le nombre d’équipes *n* est supposé non nul).

Sujet d’étude 4 : fibre optique

Une fibre optique est un fil, en verre ou en plastique, très fin, qui a la propriété d’être un conducteur de la lumière ; elle sert dans la transmission de données, offrant un débit d’information nettement supérieur à celui des câbles coaxiaux ; et peut servir de support à un réseau « large bande » par lequel transitent aussi bien la télévision, le téléphone, la visioconférence ou les données informatiques.

Ci-dessous, l’extrait d’un article du Figaro datant du 17/07/2015.

> 2016, année de l’accélération pour la fibre en France
>
> En France, la fibre optique va se généraliser à partir de 2016 par la mise en place de « réseaux d’initiative publique ».
>
> La fibre optique s’apprête à sortir des plus grandes villes de France. D’ici à la fin de l’année, plusieurs dizaines de milliers
>
> de lignes auront été activées dans de nouveaux « réseaux d’initiative publique » détenus par les collectivités territoriales.
>
> « Nous atteindrons notre rythme de croisière fin 2016, avec un million de nouvelles prises raccordables chaque année. Ce sera une lame de fond », explique Antoine Darodes, qui pilote le plan "France Très Haut Débit".

L’atténuation caractérise l’affaiblissement du signal au cours de la propagation. Le principal atout des fibres optiques est une atténuation extrêmement faible.

Le coefficient d’atténuation A , exprimée en dB/km, est donné par la relation :

$$A = \frac{1}{L}.10.\log\frac{P_e}{P_s}$$

Où :

- L est la longueur de la fibre optique (en km) ;
- P_e (en milliwatt ou mW) est la puissance lumineuse d’entrée ;
- P_s (en milliwatt ou mW) est la puissance lumineuse de sortie.

Nous souhaitons connaître les coefficients d’atténuation pour des fibres optiques dont les références sont FO1, FO2 et FO3.

Un technicien mesure une puissance lumineuse d’entrée sur chaque fibre (FO1, FO2 et FO3) de 5 mW.

En outre, chaque fibre a une longueur de 5 km.

Des techniciens mesurent les puissances lumineuses de sortie sur chaque fibre, et notent leurs résultats dans le tableau ci-dessous.

Référence de la fibre optique	**FO1**	**FO2**	**FO3**
L (Longueur de la fibre) (en km)	5	5	5
P_e (puissance lumineuse d'entrée) (en mW)	5	5	5
P_s (puissance lumineuse de sortie) (en mW)	0.8891	1.7741	3.7495

Ainsi, dans ce qui suit, nous considérerons que :

- Toute fibre est de longueur $L=5$ km ;
- Toute puissance lumineuse d'entrée P_e est fixée à 5 mW.

Partie A : Algorithmique

Type de donnée

1- Déterminer un type de donnée permettant de stocker les puissances de sortie de l'ensemble des fibres FO1, FO2 et FO3.

2- Déterminer un type de données permettant de stocker les puissances de sortie d'un ensemble de fibres optiques.

Fonctions

3- Réaliser la fonction ***coefficient_attenuation*** prenant en paramètre une puissance de sortie mesurée sur une fibre optique, et retournant le coefficient d'atténuation correspondant.

Procédures traitantes des fibres FO1, FO2 et FO3

4- Réaliser la procédure ***afficher*** prenant en paramètre les puissances de sortie des fibres FO1, FO2 et FO3, et affichant ces puissances de sortie (on affichera une puissance par ligne).

5- Réaliser une procédure prenant en paramètre les puissances de sortie des fibres FO1, FO2 et FO3, et affichant les coefficients d'atténuations correspondants à ces puissances de sortie.

Procédures traitantes d'un ensemble de fibres

6- Réaliser la procédure ***afficher2*** prenant en paramètre les puissances de sortie d'un ensemble de fibres, et affichant ces puissances de sortie (on affichera une puissance par ligne).

7- Réaliser une procédure prenant en paramètre les puissances de sortie d'un ensemble de fibres, et affichant les coefficients d'atténuations correspondants à ces puissances de sortie.

Partie B : Implémentation

Implanter en Python les fonctions/procédures de la Partie A.

Définir un jeu de test qui vous servira à tester la bonne implantation des fonctions/procédures demandées.

Sujet d'étude 5 : triplet pythagoricien

En arithmétique, un triplet pythagoricien est un triplet (x,y,z) d'entiers naturels non nuls vérifiant la relation de Pythagore : $x^2+y^2=z^2$. (z est appelée hypoténuse).

Les triplets pythagoriciens peuvent intervenir en cryptographie.

Pgcd

Le plus grand commun diviseur, abrégé en général *pgcd* (ou *gcd* en anglais), de deux nombres entiers naturels non nuls, est le plus grand entier qui divise simultanément ces deux entiers.

Exemple :

$14=7x2$

$70=7x5x2$. On a $pgcd(14,70)=2x7=14$.

Utilisation du pgcd dans Python sur le même exemple :

```
>>> import fractions
>>> fractions.gcd(14,70)
14
```

Entiers premiers entre eux

On dit que des entiers a et b sont premiers entre eux, si leur plus grand commun diviseur est égal à 1.

Exemple :

$pgcd(5,14)=1$.

Pgcd de trois entiers

Le pgcd de trois entiers x , y et z , noté $pgcd(x,y,z)$, est défini par la relation :

$pgcd(x,y,z)=pgcd(pgcd(x,y),z)$.

Triplet pythagoricien primitif

Un triplet d'entiers (x,y,z) est *pythagoricien* si $x^2+y^2=z^2$.

Triplet pythagoricien primitif

Un triplet pythagoricien (x,y,z) est dit *primitif* si $pgcd(x,y,z)=1$.

Partie A : Algorithmique

Type de donnée

1- Déterminer un type de donnée permettant de stocker un triplet d'entiers.

Fonctions

2- Réaliser une fonction ***pgcd*** prenant en paramètre deux entiers et retournant leur pgcd.

3- Réaliser une fonction ***pgcd3*** prenant en paramètre trois entiers et retournant leur pgcd.

Procédure et Fonction sur triplets d'entiers

4- Réaliser une fonction ***est_pythagoricien*** prenant en paramètre la structure donnée permettant de stocker un triplet d'entiers, et retournant Vrai si le triplet est pythagoricien et Faux sinon.

5- Réaliser une fonction ***est_primitif*** prenant en paramètre la structure de donnée permettant de stocker un triplet d'entiers, et retournant Vrai si le triplet est pythagoricien primitif et Faux sinon.

Procédures générant des triplets pythagoriciens primitifs

6- Réaliser une procédure ***generer1***, prenant en argument un entier *n*, et affichant tous les triplets pythagoriciens de la forme :

$$(i, i, \sqrt{(2.i^2)})$$

$$\text{Où} \quad 1 \leq i \leq n-1 \quad .$$

7- Réaliser une procédure ***generer2***, prenant en argument un entier *n*, et affichant tous les triplets pythagoriciens de la forme :

$$(i, j, \sqrt{(i^2+j^2)})$$

$$\text{Où} \quad 1 \leq i, j \leq n-1 \quad .$$

8- Réaliser une procédure ***generer3***, prenant en argument un entier *n*, et affichant tous les triplets pythagoriciens primitifs de la forme :

$$(i, j, \sqrt{(i^2+j^2)})$$

$$\text{Où} \quad 1 \leq i, j \leq n-1 \quad .$$

Partie B : Implémentation

Implanter en Python les fonctions/procédures de la Partie A.

Définir un jeu de test qui vous servira à tester la bonne implantation des fonctions/procédures demandées.

Remarques

Rappel 1 : $\mathbb{N} \subset \mathbb{Z} \subset \mathbb{Q} \subset \mathbb{R}$

Rappel 2 : $\mathbb{Q} = \{ \frac{p}{q} \quad , \quad p, q \in \mathbb{Z} \quad et \quad q \neq 0 \}$

Rappel 3 : $\sqrt{2} \notin \mathbb{Q}$ (on donne plus-bas la démonstration).

Théorème : $p = 0 \, (mod \, 2) \Leftrightarrow p^2 = 0 \, (mod \, 2)$

Démonstration.

Par double implication,

$$(\Rightarrow)$$
$$p = 0 \, (mod \, 2)$$
$$Donc, \ \exists k \in \mathbb{Z}, \ p = 2.k$$
$$Donc, \ p^2 = (2.k)^2 = 4.k^2 = 2.(2.k^2)$$
$$D'où, \ p^2 \, est \, pair.$$

$$(\Leftarrow)$$
$$p^2 = 0 \, (mod \, 2)$$

$$Deux \, cas :$$

$$Cas \, 1 : p = 1 \, (mod \, 2),$$
$$Alors \ \exists k \in \mathbb{Z}, \ p = 2.k + 1$$
$$Donc, \ p^2 = (2.k + 1)^2 = 4.k^2 + 1 + 4.k$$
$$C'est - à - dire, \ p^2 = 4k(k+1) + 1$$
$$Soit, \ p^2 = 2(2k(k+1)) + 1$$
$$D'où, \ p^2 \, est \, impair.$$
$$Cas \, 1 \quad faux.$$

$$Cas \, 2 : p = 0 \, (mod \, 2),$$
$$Alors \ \exists k \in \mathbb{Z}, \ p = 2.k$$
$$D'où, \ p^2 = 4.k^2 = 2(2.k^2)$$
$$p^2 \, est \, pair.$$
$$Cas \, 2 \quad valable.$$

CQFD.

Théorème : $\sqrt{2}\notin\mathbb{Q}$

Démonstration.

Par descente infinie (absurde),

$$\begin{gathered}
On\ suppose : \sqrt{2}\in\mathbb{Q}.\\
Donc, \exists p, q\in\mathbb{Z}, q\neq 0,\\
\sqrt{2}=\frac{p}{q}\\
On\ se\ place\ \ sous\ l'hypothèse : p, q\in\mathbb{N}-\{0\}\\
D'où,\\
\sqrt{2}>1\Leftrightarrow\frac{p}{q}>1\Leftrightarrow p>q\\
En\ outre,\\
2=\frac{p^2}{q^2}\Leftrightarrow 2.q^2=p^2\\
Donc, p\ est\ pair.\\
Donc,\ \exists k\in\mathbb{N}, p=2k.\\
Par\ suite, 2q^2=4k^2\Leftrightarrow q^2=2k^2\\
D'où,\ (q, k\in\mathbb{N}-\{0\}),\ q=\sqrt{2}.k\\
Ainsi,\\
\sqrt{2}=\frac{p}{q}=\frac{q}{k}\\
Avec,\ \ p>q
\end{gathered}$$

q et k verifient nos hypothèses de départ. On peut ainsi recommencer le procédé ; et cela sans fin.

Nous obtenons alors, une suite infinie, strictement décroissante, d'entiers strictement positifs. D'où la fausseté de l'hypothèse de départ : $\sqrt{2}\in\mathbb{Q}$

CQFD.

Théorème : $\forall k\in\mathbb{N}-\{0\},\ k.\sqrt{2}\notin\mathbb{Q}$

Démonstration.

Par l'absurde,

$$\begin{gathered}
k.\sqrt{2}\in\mathbb{Q}\\
Donc,\ \exists p, q\in\mathbb{Z},\ q\neq 0,\\
k.\sqrt{2}=\frac{p}{q}\Leftrightarrow\sqrt{2}=\frac{p}{q.k}\\
D'où,\ \sqrt{2}\in\mathbb{Q}.
\end{gathered}$$

CQFD.

Théorème : $\forall k \in \mathbb{N} - \{0\},\ k.\sqrt{2} \notin \mathbb{N}$

Démonstration.

$\forall k \in \mathbb{N} - \{0\},\ k.\sqrt{2} \notin \mathbb{Q}$

$\mathbb{N} \subset \mathbb{Q}$

CQFD.

Théorème : Il n'existe pas de triplet pythagoricien primitif de la forme : $(k,\ k,\ k.\sqrt{2})$ pour k entier positif non nul.

Démonstration.

$\forall k \in \mathbb{N} - \{0\},\ k.\sqrt{2} \notin \mathbb{N}$

CQFD.

TP – Opérations sur points

Nous utilisons la Programmation Orientée Objet (POO) pour manipuler les points (2D).

Ci-après, une première implémentation.

```
import math
sqrt = math.sqrt

class Point():

    def __init__(self, x, y):
        self.x = x
        self.y = y

    def norme(self):
        return sqrt(self.x**2 + self.y**2)

    def produit_scalaire(self, p):
        return True #A FAIRE

    def addition(self, p):
        return Point(self.x + p.x, \
                     self.y + p.y)

    def distance(self, p):
        return True #A FAIRE


    # pour l'impression
    def __str__(self):
        return "("+\
               str(self.x)+", "+\
               str(self.y)+\
               ")"


if __name__ == "__main__":

    p1 = Point(5, 2)
    print(p1)
```

Exercice 1

Tester le code ci-avant.

Exercice 2

Créer (*instancier*) le point $p1=(3,4)$ et afficher-le.

Exercice 3

Afficher la norme du point $p1$ par la commande :

```
print(p1.norme())
```

Exercice 4

Afficher la norme du point $p2$.

Exercice 5

Afficher la somme des points $p1$ et $p2$ par la commande :

```
print(p1.addition(p2))
```

Exercice 6

Tester la commande :

```
print(p2.addition(p1))
```

Exercice 7

Implémenter la méthode ***produit_scalaire***, prenant en argument un point, et retournant le produit scalaire du « point objet » (*self*) et du « point argument ».

Rappel : pour deux points de $\mathbb{R}^2$: $p=(x,y)$ et $p'=(x',y')$, le produit scalaire de p et p' est défini par :

$$p.p'=x.x'+y.y'$$

Tester les commandes :

```
print(p1.produit_scalaire(p2))
```

et

```
print(p2.produit_scalaire(p1))
```

Exercice 8

Implémenter la méthode ***distance***, prenant en argument un point, et retournant la distance euclidienne du « point objet » (*self*) et du « point argument ».

Rappel : pour deux points de $\mathbb{R}^2$: $p=(x,y)$ et $p'=(x',y')$, la distance euclidienne de p et p' est définie par :

$$dist(p,p')=\sqrt{(x-x')^2+(y-y')^2}$$

Tester les commandes :

```
print(p1.distance(p2))
```

et

```
print(p2.distance(p1))
```

TP – Nombres complexes

Un nombre complexe *z* se présente, en général, sous **forme algébrique**, comme une somme :

$$a+i.b$$

Où :

– *a* et *b* sont des nombres réels quelconques ;

– *i* (l'unité imaginaire) est un nombre particulier tel que $i^2=-1$.

– Le réel *a*, est appelé **partie réelle** de *z*, et se note $\Re(z)$;

– Le réel *b*, est appelé **partie imaginaire** de *z*, et se note $\Im(z)$.

Deux nombres complexes sont égaux si et seulement si : ils ont la même partie réelle et la même partie imaginaire :

$\forall z, z' \in \mathbb{C}$,

$$z=z' \quad \textit{iff} \quad [\ \Re(z)=\Re(z')\ \textit{et}\ \Im(z)=\Im(z')\ ]$$

Un nombre complexe *z* est dit **imaginaire pur** si sa partie réelle est nulle ; dans ce cas il s'écrit sous la forme :

$$z=i.b$$

Un nombre complexe, dont la partie imaginaire est nulle, est dit **réel**.

Exercices

1. Définir une classe **Complexe**, qui nous permettra de manipuler les nombres complexes.

   Implémenter :

2. Une méthode d'affichage d'un nombre complexe ;
3. Une méthode effectuant la somme de deux nombres complexes ;
4. Une méthode calculant l'opposé d'un nombre complexe (l'opposé de $a+b.i$ est $-a-b.i$) ;
5. Une méthode retournant Vrai si le nombre complexe considéré est un imaginaire pur (i.e. : $z=0+b.i$) ; et Faux sinon.

6. Une méthode retournant le conjugué du nombre complexe considéré (le conjugué de $a+b.i$ est $a-b.i$) ;
7. Une méthode retournant le module du nombre complexe considéré (le module de $z=a+b.i$ (noté $|z|$) est défini par : $|z|=\sqrt{a^2+b^2}$).
8. Une méthode retournant la multiplication de deux nombres complexes ($(a+i.b)\times(a'+i.b')=(aa'-bb')+(ab'+ba').i$).