Math93.com

# TD - NSI
# Minis Projets Python 2

*Les mini projets sont des épreuves de qualification pour le concours PROLOGIN adaptées à pyhton*

# GPS – Niveau 2

## 1 Énoncé et objectifs

### 1.1 Énoncé

On vous donne une liste de coordonnées de type $(x_i, y_i)$ , nombres entiers relatifs, représentant les coordonnées cartésiennes sur une carte de France des différents centres d'examen pour les demi-finales d'une compétition. Vous vous situez en $(x, y)$.
L'objectif est d'écrire une fonction qui renvoie le centre le plus proche de vous ainsi que divers fonctions permettant de faire des tests. On va aussi proposer un affichage des points avec Matplotlib.
Vous utiliserez la distance euclidienne dans vos calculs.

**Distance euclidienne**

On se place dans un repère orthonormée. Soit les points $A\left(x_A\,;\,y_A\right)$ et $B\left(x_B\,;\,y_B\right)$.
La distance euclidienne $AB$ est donnée, en unité de longueur par :

$$AB = \sqrt{(x_B - x_A)^2 + (y_B - y_A)^2}$$

### 1.2 Exemples d'entrée/sortie (ne commencer pas à programmer, lisez bien les consignes)

```
# Dans la console Python
>>> mesCoords=(38,5)
>>> centreExam=[(54, 82),(75, 21),( 6, 21),( 61, 21),( 60, 68)]
>>> GPS(mesCoords,centreExam)
(61,21)
```

## 2 Des fonction tests

1. Écrire une fonction **alea(*a*,*b*)** qui génère un couple d'entiers aléatoires de $a$ à $b$.

```
from random import randint

def alea(a,b):
    '''in : a,b des entiers relatifs avec a < b
    out : un couple d'entiers compris entre a et b'''
    return ...
```

```
# On obtient dans la console
> alea(-50,50)
(-21, -7)
```

2. Écrire une fonction **centreExam($a,b,n$)** qui génère une liste de $n$ couples distincts, d'entiers aléatoires compris entre $a$ et $b$, avec $a < b$.
   Cela va donc nous donner une liste des coordonnées des centres d'examen.
   ***Remarque*** *: On peut dans un premier temps créer une liste quelconque puis améliorer la fonction afin d'obtenir n couples tous distincts.*

```
from random import randint

def centreExam(a,b,n):
    '''in : a,b des entiers relatifs (a<b) et n entier naturel >1
      out : une liste de n couples distincts d'entiers
      compris entre a et b'''
      ...
    return ...
```

```
# On obtient dans la console
> centreExam(-10,10,5)
[(8, -6), (-4, -9), (2, 9), (-7, 1), (8, 3)]
```

## 3 La fonction distance

1. Écrire la fonction **distance($A,B$)** qui renvoie la distance (euclidienne) AB si A et B sont deux couples de coordonnées dans un repère orthonormé.

```
from math import sqrt
def distance(A,B):
    '''in : A et B des couples de coordonnées
      out : la distance AB dans un r.o.n.'''
    ....
```

```
# On obtient dans la console
>>> A=(1,2)
>>> B=(4,6)
>>> distance(A,B)
5.0
```

**Remarque**

On peut vérifier par le calcul que dans un R.O.N on a bien, en unité de longueur :

$$\begin{cases} A(1\,;\,2) \\ B(4\,;\,6) \end{cases} \Longrightarrow AB = \sqrt{(4-1)^2+(6-2)^2} = \sqrt{3^2+4^2} = \sqrt{25} = 5$$

2. Tester votre fonction **distance** avec deux points générés avec votre fonction **alea**.

## 4 La fonction GPS

Écrire une fonction **GPS(*mesCoord*,*listeCentres*)** qui renvoie un couple contenant deux entiers, représentant les coordonnées du centre le plus proche de vous. Dans le cas où plusieurs centres sont à la même distance de vous, renvoyez le premier apparaissant dans l'entrée.

```
def GPS(mesCoords,listeCentres):
    '''in : mesCoords un couple de coordonnée et
            listeCentres une liste des coordonnées des centres
      out : un couple, les coordonnées du centre le plus proche de vous'''
    ....
```

```
# On obtient dans la console
>>> mesCoords=(38,5)
>>> listeCentres=[(54, 82),(75, 21),( 6, 21),( 61, 21),( 60, 68)]
>>> GPS(mesCoords,listeCentres)
(61,21)
```

## 5 La phase de tests

Tester votre fonction GPS en utilisant des listes aléatoires et donc vos fonctions **alea** et **centreExam**.

## 6 Des graphiques

1. Avec Matplotlib, placer les centres d'examen en bleu et visualiser votre position en rouge.
   ***Aide** : il faudra créer une liste d'abscisses et une des ordonnées à partir de la liste de centres d'examen.*

**Aide**

Consultez la page dédiée à **matplotlib** :
https://www.math93.com

2. Tracer en vert le segment reliant vous et le centre d'examen le plus proche puis le cercle centré sur votre position et de rayon ce segment.

**Aide**

Consultez la page dédiée à **matplotlib** pour tracer un cercle :
https://www.math93.com

3. Afficher aussi le distance minimale sur le graphique.

## 7 Une nouvelle fonction de tri

Adapter un des algorithmes de tri étudiés afin de renvoyer la liste des coordonnées des centres par ordre croissant des distances par rapport à votre position.
Imaginez une visualisation sur Matplotlib.

```
def tri(mesCoord,listeCentres):
    '''in : mesCoord un couple de coordonnée et
          listeCentres une liste des coordonnées des centres
      out : la liste des coordonnées des centres par ordre croissant
            des distances par rapport à votre position mesCoord'''
    ....
```