DS n° 01 – Initiation à l'algorithmie

- Faire tous les exercices dans un fichier NomPrenom.py à sauvegarder,
- mettre en commentaire l'exercice et la question traités (ex : # Exercice 1),
- ne pas oublier pas de commenter ce qui est fait dans votre code (ex : # Je crée une fonction pour calculer la racine d'un nombre),
- il est possible de demander un déblocage pour une question, mais celle-ci sera notée 0,
- il faut vérifier avant de partir que le code peut s'exécuter et qu'il affiche les résultats que vous attendez. Les lignes de code qui doivent s'exécuter sont décommentées.

1 Gradient de couleur en spirale

En géométrie plane, les spirales forment une famille de courbes d'allure similaire : une partie de la courbe semble s'approcher d'un point fixe tout en tournant autour de lui, tandis que l'autre extrémité semble s'en éloigner.

Le premier but de cet exercice est de tracer le gradient de couleur (du rouge vers le vert) en spirale, comme sur la figure 1.

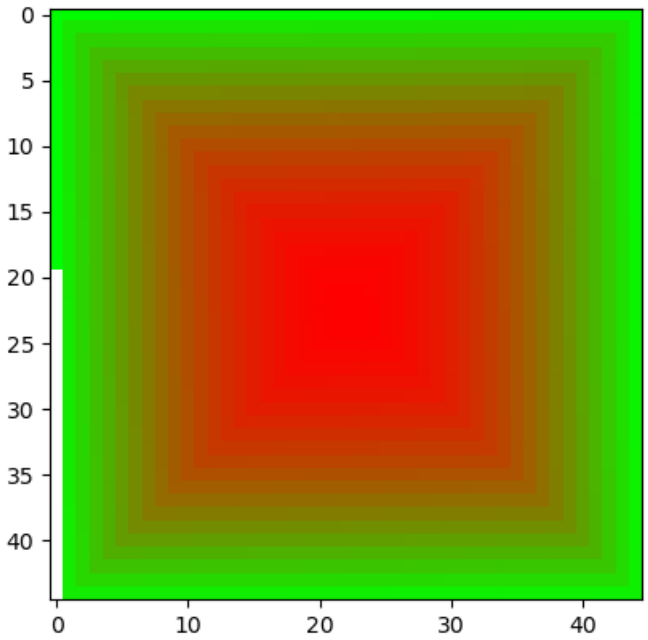


FIGURE 1 – Gradient de couleur en spirale

Le code suivant permet de tracer cette spirale colorée (les commentaires dont les lignes qui commencent par # ne sont pas obligatoires, sauf la première ligne `# -*- coding: utf-8 -*-`).

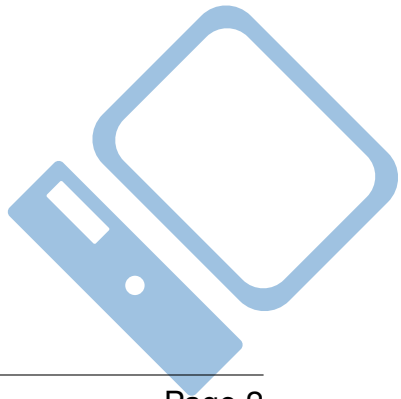
```python
# -*- coding: utf-8 -*-

# Import des bibliotheques
import numpy as np
import matplotlib.pyplot as plt
import matplotlib as mpl

# Fonction qui détermine la couleur d'un pixel
# en fonction de sa position
def colorFader(mix=0):
    c1=np.array([1,0,0])
    c2=np.array([0,1,0])
    return (1-mix)*c1 + mix*c2

# Parcours de la spirale a partir du centre
# en fonction de la valeur de entier
# a recopier pour la question 1 et à copier et
# modifier pour la question 3
max=2000
imagebase = np.ones([int(max**0.5)+1,int(max**0.5)+1,3])
coords=[0,0]
i=0
entier=0
while entier<max:
    entier+=1
    # le pixel de imagebase qui a pour coordonnées
    # [coords[0]+(int(max**0.5))//2,coords[1]+(int(max**0.5))//2]
    # est coloré avec une couleur déterminée par la fonction colorFader
    dec=(int(max**0.5))//2
    imagebase[coords[0]+dec,coords[1]+dec]=colorFader(entier/max)
    # on passe ensuite au pixel suivant pour parcourir toute la spirale
    if coords==[i,-i]:
        coords[0]+=1
        i+=1
    elif coords[0]==i and coords[1]<i:
        coords[1]+=1
    elif coords[0]>-i and coords[1]==i:
        coords[0]-=1
    elif coords[0]==-i and coords[1]>-i:
        coords[1]-=1
    elif coords[0]<i and coords[1]==-i:
        coords[0]+=1

# Tracé de la figure de la réponse
plt.imshow(imagebase)
plt.show()
```



Question 1 Recopier le code précédent et exécuter le script. La figure 1 doit alors apparaître (Si ce n'est pas le cas, la console python vous indique la ligne où vous avez fait une erreur) ?

2 Parité des nombres

On donne le code suivant :

```
def est_pair(n):
    return n%2==1

print(est_pair(3))
```

Cette fonction ne renvoie pas ce qui est prévu, c'est à dire :

— `True` si le nombre est pair,

— `False` si le nombre est impair.

Question 2 Recopier et corriger le code précédent pour que la fonction `est_pair(n)` ne renvoie `True` que si le nombre est pair.

On souhaite réaliser la figure 2 dont les pixels sont colorés selon le principe suivant :

— si `entier` est pair alors le pixel doit être noir : `imagebase[coords[0]+dec,coords[1]+dec]=[0,0,0]`,

— si `entier` est impair alors on ne fait rien, on laisse le pixel blanc.

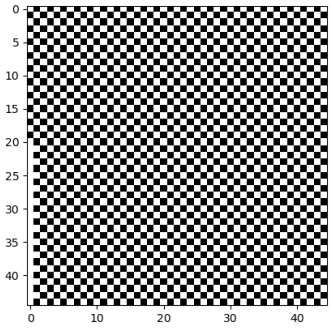


FIGURE 2 – Coloration de la figure en fonction de la parité de `entier`

Question 3 Recopier et modifier les lignes 15 à 46 (sans les commentaires) afin de réaliser la figure 2

3 La spirale d'Ulam

En mathématiques, la spirale d'Ulam, ou spirale des nombres premiers (dans d'autres langues, elle est appelée aussi horloge d'Ulam) est une méthode simple pour la représentation des nombres premiers qui révèle un motif qui n'a jamais été pleinement expliqué. Elle fut découverte par le mathématicien Stanislaw Ulam, lors d'une conférence scientifique en 1963.

Un algorithme pour décider si un nombre est premier (appelés tests de primalité) consiste à essayer de le diviser par tous les nombres en commençant par 2, qui n'excèdent pas sa racine carrée : s'il est divisible par l'un d'entre eux, il est composé, et sinon, il est premier.

Ainsi la fonction à coder est la suivante :

1 Si n est strictement inférieur à 2, il est premier `return True`,

2 Sinon :

3.1 on initialise d : `d=2`,

3.2 tant que d est inférieur ou égal à la racine carrée de n : `d<=n**0.5`,

i. si le reste de la division euclidienne de n par d faut 0 : `n%d==0`,

ii. alors le nombre n'est pas premier, je retourne un False : `return False`,

iii. sinon j'ajoute 1 à d : `d+=1`.

3.3 Le nombre est premier si aucun d n'est un diviseur de n `return True`.

Question 4 Coder la fonction `est_premier(n)` à partir de cet algorithme. Elle doit renvoyer `True` si `n` est premier, `False` sinon.

On souhaite réaliser la figure 3 dont les pixels sont colorés selon le principe suivant :

— si `entier` est premier alors le pixel doit être noir : `imagebase[coords[0]+dec,coords[1]+dec]=[0,0,0]`,

— si `entier` est composé alors on ne fait rien, on laisse le pixel blanc.

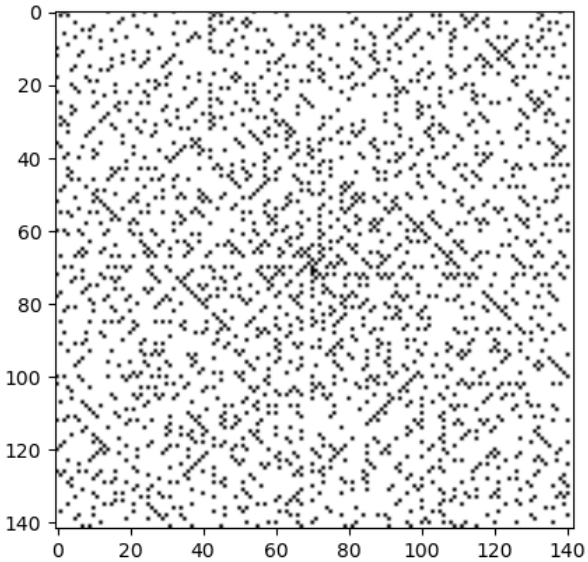


FIGURE 3 – Coloration de la figure en fonction du fait que `entier` soit premier

Question 5 Recopier les modifier les lignes 15 à 46 (sans les commentaires) afin de réaliser la figure 3. La valeur max devra être de 20000 pour cette question.

FIN

Correction

Question 1

Recopier le code.

Question 2

```
def est_pair(n):
    return n%2==0

print(est_pair(3),est_pair(4))
```

Question 3

```
# Parcours de la spirale a partir du centre
# en fonction de la valeur de entier
max=2000
imagebase = np.ones([int(max**0.5)+1,int(max**0.5)+1,3])
coords=[0,0]
i=0
entier=0
while entier<max:
    entier+=1
    # le pixel de imagebase qui a pour coordonnées
    # [coords[0]+(int(max**0.5))//2,coords[1]+(int(max**0.5))//2]
    # est coloré avec une couleur déterminée par la fonction colorFader
    if est_pair(entier):
        dec=(int(max**0.5))//2
        imagebase[coords[0]+dec,coords[1]+dec]=[0,0,0]
    if coords==[i,-i]:
        coords[0]+=1
        i+=1
    elif coords[0]==i and coords[1]<i:
        coords[1]+=1
    elif coords[0]>-i and coords[1]==i:
        coords[0]-=1
    elif coords[0]==-i and coords[1]>-i:
        coords[1]-=1
    elif coords[0]<i and coords[1]==-i:
        coords[0]+=1

# Tracé de la figure de la réponse
plt.imshow(imagebase)
plt.show()
```

Question 4

```
def est_premier(n):
    if n <2:
        return True
    d=2
    while d<=n**0.5:
        if n%d==0:
            return False
        else:
            d+=1
    return True

print(est_premier(6),est_premier(37))
```

Question 5

```
# Parcours de la spirale a partir du centre
# en fonction de la valeur de entier
max=20000
imagebase = np.ones([int(max**0.5)+1,int(max**0.5)+1,3])
coords=[0,0]
i=0
entier=0
while entier<max:
    entier+=1
    # le pixel de imagebase qui a pour coordonnées
    # [coords[0]+(int(max**0.5))//2,coords[1]+(int(max**0.5))//2]
    # est coloré avec une couleur déterminée par la fonction colorFader
    if est_premier(entier):
        dec=(int(max**0.5))//2
        imagebase[coords[0]+dec,coords[1]+dec]=[0,0,0]
    if coords==[i,-i]:
        coords[0]+=1
        i+=1
    elif coords[0]==i and coords[1]<i:
        coords[1]+=1
    elif coords[0]>-i and coords[1]==i:
        coords[0]-=1
    elif coords[0]==-i and coords[1]>-i:
        coords[1]-=1
    elif coords[0]<i and coords[1]==-i:
        coords[0]+=1

# Tracé de la figure de la réponse
plt.imshow(imagebase)
plt.show()
```