Informatique en CPGE (2017-2018)
Exercices divers corrigés

Exercice 1

```
from random import randint

def simule(n):
    etagere=n*[0]
    for i in range(n):
        c=randint(0,n-1)
        etagere[c]+=1
    cpt=0
    for u in etagere:
        if u>1:
            cpt+=1
    return cpt/n
```

Remarque : $1-2/e \simeq 0,264$.

Ce nombre est la limite quand n tend vers $+\infty$ de la probabilité qu'une étagère choisie au hasard parmi les n étagères supporte plusieurs livres . Pour $n=2$ cette probabilité vaut 1/4.

```
>>> simule(1000)
0.272
>>> simule(10000)
0.2649
```

Exercice 2

```
from random import random
import matplotlib.pyplot as plt

# simulation de 5 marches

def marche(n,p):
    for i in range(5):
        x=0
        liste=[0]
        for k in range(n):
            if random()<p:
                x=x+1
            else:
                x=x-1
            liste.append(x)
        axe=list(range(n+1))
        plt.plot(axe,liste)
    plt.plot([0,n],[0,(2*p-1)*n],"k") # E(X_n)
    plt.grid()
    plt.show()
```

```
marche(100,0.5)
marche(100,0.6)
```

Exercice 3

1. Produit de deux matrices.

```
def produit(A,B):
    n=len(A)
    P=[[0 for j in range(n)] for i in range(n)]
    for i in range(n):
        for j in range(n):
            for k in range(n):
                P[i][j]+=A[i][k]*B[k][j]
    return P
```

2. Puissance d'une matrice.

```
# version non récursive
def puissance1(A,k):
    n=len(A)
    P=[[0 for j in range(n)] for i in range(n)]
    for i in range(n):
        P[i][i]=1
    for i in range(k):
        P=produit(P,A)
    return P

# version récursive
def puissance2(A,k):
    if k==0:
        n=len(A)
        I=[[0 for j in range(n)] for i in range(n)]
        for i in range(n):
            I[i][i]=1
        return I
    else:
        return produit(A,puissance2(A,k-1))
```

Exercice 4

Nous avons besoin de la matrice inverse pour le déchiffrage.
Le déterminant de la matrice est 3 dont l'inverse modulo 59 est 20, puisque $3 \times 20 = 60 \equiv 1 \quad [59]$.
Calcul de l'inverse : la comatrice de `mat` est `[[11,-20],[-7,13]]`,
sa transposée est `[[11,-7],[-20,13]]`.
On multiplie par 20 soit `[[220,-140],[-400,260]]`.
Modulo 59, nous obtenons la matrice inverse `inv=[[43,37],[13,24]]`.

```
from random import randint
import numpy as np
```

```
def code_hill(phrase,choix='code'):
    mat=np.array([[13,7],[20,11]])
    if choix=='decode':
        mat=np.array([[43,37],[13,24]])
    n=len(phrase)
    if n%2: # ajout d'une lettre au hasard
        phrase=phrase+chr(randint(32,90))
        n=n+1
    codage=''
    for i in range(1,n,2):
        car1=phrase[i-1]
        car2=phrase[i]
        n1=ord(car1)-32 # 0<=n<=58
        n2=ord(car2)-32 # 0<=n<=58
        u=np.array([n1,n2])
        v=mat @ u
        n1=v[0]%59
        n2=v[1]%59
        car1=chr(n1+32)
        car2=chr(n2+32)
        codage=codage+car1+car2
    return codage
```

Test :

```
>>> test=code_hill('BONJOUR, COMMENT ALLEZ-VOUS ?')
>>> code_hill(test,'decode')
'BONJOUR, COMMENT ALLEZ-VOUS ?G'
>>>
```

Exercice 5

```
a,b,c,d=0.55,0.7,0.45,0.3

def systeme(x0,y0,n,dt):
    x,y=x0,y0
    liste_t=list(range(n+1))
    liste_x=[x]
    liste_y=[y]
    for k in range(n):
        x_temp=x+dt*(a*x-b*x*y)
        y=y+dt*(-c*y+d*x*y)
        x=x_temp
        liste_x.append(x)
        liste_y.append(y)
    return liste_t,liste_x,liste_y

import matplotlib.pyplot as plt

t,x,y=systeme(0.8,1.3,50000,0.001) # dt=0.001
plt.plot(x,y)
plt.grid()
plt.show()
```