```python
# correction proba.py

from random import *
###Exercice1
def var_uniform(a,b):
    x=random()
    return((b-a)*x+a)

#ou juste
def unif(a,b):
    return(uniform(a,b))

#2a : 2/5

#2b :

def experience1():
    x=var_uniform(5,10)
    if x>=6 and x<=8:
        return(True)
    else:
        return(False)

def frequence1(N):
    c=0 #Compteur
    for i in range(N): #on itère N fois l'expérience
        if experience1(): #Pas besoin de mettre
experience1()==True,
                    #car experience1 retourne déjà un booléen.
            c=c+1
    return(c/N)


###Exercice2

#1
a,b=1,12
randint(a,b)

#2 inutile.
from math import floor
def var_ent(a,b):
    return(floor(uniform(a,b+1)))

#3
def experience2():
    x=randint(1,12)
    if x%3==0:
        return(True)
    else:
        return(False)

def frequence2(N):
    c=0 #Compteur
    for i in range(N): #on itère N fois l'expérience
```

```
053|         if experience2(): #Pas besoin de mettre
experience1()==True,
054|                       #car experience1 retourne déjà un booléen.
055|             c=c+1
056|     return(c/N)
057|
058|
059| ###Exercice3
060| #1
061| def pile_ou_face(p):
062|     x=random()
063|     if p<0 or p>1:
064|         return('Argh une proba est entre 0 et 1')
065|     if x<p: #probabilité p que x soit plus petit que p
066|         return(1) #1 correspond à Pile
067|     else:
068|         return(0) #0 corrrespond à Face
069|
070| def frequence(N,p):
071|     pile=0 #Compteur
072|     for i in range(N): #on itère N fois l'expérience
073|         if pile_ou_face(p)==1:
074|             pile=pile+1
075|     return(pile/N)
076|
077| print(frequence(10000,1/5))
078| ###Exerice 4:
079| #1
080| def lancer(n,p):
081|     pile=0 #Compteur
082|     for i in range(n): #on itère N fois l'expérience
083|         if pile_ou_face2(p)==1:
084|             pile=pile+1
085|     return(pile)
086|
087| #2
088| def trois_PF(p):
089|     p0=(1-p)**3
090|     p1=3*p*(1-p)**2
091|     p2=3*((1-p)**2)*p
092|
093|     x=random()
094|     if x<p0:
095|         return(0)
096|     elif x<p0+p1:
097|         return(1)
098|     elif x<p0+p1+p2:
099|         return(2)
100|     else:
101|         return(3)
102|
103|
104| def tirage_succefis(N,p):
105|     c0,c1,c2,c3=0,0,0,0 #Compteur pour chacuns des evenements (0
pile, 1 Pile//
```

```
106|
107|     for i in range(N):
108|         k=trois_PF(p)
109|         if k==0:
110|             c0=c0+1
111|         if k==1:
112|             c1=c1+1
113|         if k==2:
114|             c2=c2+1
115|         if k==3:
116|             c3=c3+1
117|
118|     return(c0/N,c1/N,c2/N,c3/N)
119|
120|
121| ###Exercice 5
122| def Anniversaire(n):
123|     liste_anniversaire=[]
124|     for i in range(n):
125|         liste_anniversaire =liste_anniversaire+[randint(1,365)]
#au lieu de jour/Mois on met le numero du jour dans l'année
126|         #ceci donne une liste aléatoire d'anniversaire de n
personnes.
127|
128|     return(liste_anniversaire)
129|
130| def verif_2_identique(L):
131|     #On va vérifier si la liste L contient deux élements
identiques.
132|     n=len(L)
133|     for i in range(n):
134|         for j in range(i+1,n):
135|             if L[i]==L[j]: #Il faut vérifier tous les éléments
de L avec tous les éléments suivants
136|                 return(True)  #Si on tombe sur deux anniversiare
identiques on renvoit True et on s'arrete
137|     return(False) #Si a la fin du parcours de la liste aucun
doublon n'a été détecté on retourne False
138|
139|
140| def frequence_anniv_identique(n,N):
141|     #n correspond au nombre d'éléves considérés,
142|     #N correspond au nombre de fois où on itère l'expérience
pour calculer une valeur approchée de notre probabilité
143|     c=0 #Compteur
144|     for i in range(N): #On itère N fois l'expérience.
145|         L=Anniversaire(n)
146|         E=verif_2_identique(L) # E est un booléen, True si deux
anniversaires commun et false sinon
147|         if E:
148|             c=c+1
149|
150|     return(c/N)
151|
152|
```

```
153| print(frequence_anniv_identique(45,1000))
154|
155| ###Exercice 6
156|
157| def lancer_3_de():
158|     De=[]
159|     for i in range(3):
160|         De=De+[randint(1,6)]
161|     return(De)
162|
163|
164| def Evenement_A():
165|     L=lancer_3_de()
166|     if L[0]==L[1] and L[1]==L[2]:
167|         return(True)
168|     else:
169|         return(False)
170|
171| def Evenement_B():
172|     L=lancer_3_de()
173|     if L[0]==3 or L[1]==3 or L[2]==3:
174|         return(True)
175|     else:
176|         return(False)
177|
178| def Evenement_C():
179|     L=lancer_3_de()
180|     if L[0]+L[1]+L[2]==4:
181|         return(True)
182|     else:
183|         return(False)
184|
185| def au_moins_un():
186|     if Evenement_A() or Evenement_B() or Evenement_C():
187|         return(True)
188|     else:
189|         return(Fale)
190|
191| def frequence(N):
192|     c=0
193|     for i in range(N):
194|         if au_moins_un():
195|             c=c+1
196|     return(c/N)
197|
198| ### Exercice 7
199|
200|
201| def lancer_6_de():
202|     L=[]
203|     for i in range(6):
204|         n=randint(1,6)
205|         L=L+[n]
206|     return(L)
207|
```

```
208| def verif_different(L):
209|     l=len(L)
210|     for i in range(l):
211|         for j in range(i+1,l):
212|             if L[i]==L[j]:
213|                 return(False)
214|     return(True)
215|
216|
217|
218| def experience_7_2():
219|     L=lancer_6_de()
220|     resultat=verif_different(L)
221|     return(resultat)
222|
223| def repete_7(N):
224|     c=0
225|     for i in range(N):
226|         if experience_7_2():
227|             c=c+1
228|     return(c/N)
229|
230| print('6', repete_7(1000))
231|
232|
233| def au_moins_deux_distincts(L):
234|     l0=L[0]
235|     for element in L:
236|         if element!=l0:
237|             return(True)
238|     return(False)
239|
240| def experience_7_3():
241|     L=lancer_6_de()
242|     resultat=au_moins_deux_distincts(L)
243|     return(resultat)
244|
245|
246|
247| def repete_7_3(N):
248|     c=0
249|     for i in range(N):
250|         if experience_7_3():
251|             c=c+1
252|     return(c/N)
253| print(1-1/(6**5))
254| print(repete_7_3(100000))
255|
256|
257| ##Exercice 8
258|
259| def jeton(n):
260|     L=[i for i in range(1,n+1)]
261|     sac=L+L
262|
```

```
    alea_A=randint(0,2*n-1)
    A=sac[alea_A]

    alea_B=randint(0,2*n-1)
    B=sac[alea_B]

    if A>= 2*B:
        return(True)
    else:
        return(False)


def repete_jeton(N,n):
    c=0
    for i in range(N):
        if jeton(n):
            c=c+1
    return(c/N)

print(repete_jeton(1000,100))


##Exercice 9

def abeille(n,p):
    fleur='A'  # au jour 0 l'abeille va sur la fleur A
    liste_des_fleurs=[fleur]
    for i in range(n):
        x=random()
        if x<p and fleur=='A':
            fleur='A'
        elif x<p and fleur=='B':
            fleur='B'
        elif x>p and fleur=='A':
            fleur='B'
        elif x>p and fleur=='B':
            fleur='A'

        liste_des_fleurs=liste_des_fleurs+[fleur]
    return(liste_des_fleurs)


def compte_fleur_A(p):
    list_fleur=abeille(2000,p)[1000:] #on prend les jours de
1000 jusqu'a 2000
    cA=0
    cB=0
    for fleur in list_fleur:
        if fleur=='A':
            cA=cA+1
        else:
            cB+=1

    return(cA/1000,cB/1000)

```

```
317| print(compte_fleur_A(0.01))
318|
319|
320|
321|
322|
323|
324|
325|
326|
327|
328|
329|
330|
331|
332|
```