TP 09 – Révisions (II)

Exercice 1 Soit $f : \mathbb{R} \to \mathbb{R}$ la fonction définie par

$$\forall x \in \mathbb{R}, \qquad f(x) = \begin{cases} \ln(x) & \text{si } x > 0 \\ \exp(x) & \text{si } x \geqslant 0 \end{cases}$$

Écrire une fonction `f` qui prend en argument un réel `x` et qui renvoie la valeur de `f(x)`.

Entrée [1]:
```
import numpy as np

def f(x):
    if x>0:
        return(np.log(x))
    else:
        return(np.exp(x))
```

Exercice 2 Dans chacune des situations, reconnaître la loi de la variable aléatoire et simuler un tirage de cette variable aléatoire à l'aide de Python.

1. Une urne contient 4 boules, indiscernables au toucher, numérotées de 1 à 4. On tire aléatoirement une boule dans l'urne et on note X son numéro.

Entrée [2]:
```
rd.randint(1,5)
```

Out [2]:
```
3
```

2. Une urne contient 3 boules rouges et 5 boules noires, indiscernables au toucher. On tire, successivement et avec remise, 4 boules de l'urne. On note X le nombre de boules noires tirées.

Entrée [3]:
```
rd.binomial(4,5/8)
```

Out [3]:
```
2
```

3. Une urne contient 3 boules numérotées 1, 2 et 3, indiscernables au toucher. On tire successivement et avec remise 2 boules de l'urne. On note X le nombre de boules numérotées 1 obtenues.

Entrée [4]:
```
rd.binomial(2,1/3)
```

Out [4]:
```
0
```

Exercice 3 On considère une variable aléatoire X qui suit une loi uniforme sur $\{1, \dots, 7\}$.

1. Que vaut son espérance et sa variance théorique ?

$$E(X) = \frac{1+7}{2} = 4 \qquad \text{et} \qquad V(x) = \frac{7^2-1}{12} = 4$$

2. Calculer à l'aide de Python l'espérance et la variance empirique de la variable aléatoire X. *On rappelle que l'espérance empirique d'une variable aléatoire correspond à le moyenne d'un grand nombre de ces tirages. On vérifiera que les valeurs trouvées sont proches des valeurs théoriques.*

Entrée [5]:
```
import numpy as np
import numpy.random as rd

X = rd.randint(1,8,1000)
esp = np.mean(X)
print("L'esperance empirique de X est", esp)
var = np.var(X)
print("La variance empirique de X est", var)
```

Out [5]:
```
L'esperance empirique de X est 3.981
La variance empirique de X est 4.152639000000001
```

Exercice 4 On considère la suite $(u_k)_{k\in\mathbb{N}}$ définie par

$$u_0 = 0 \quad \text{et} \quad \forall k \in \mathbb{N}, \qquad u_{k+1} = \sqrt{u_k + 2}$$

1. Écrire une fonction, appelée `exo2`, qui prend en argument un entier `n` et qui renvoie le terme u_n de la suite. *On vérifiera que l'évaluation de* `exo2` *en* `4` *donne environ* `1.99`.

Entrée [6]:
```
import numpy as np

def exo2(n):
    u = 0
    for k in range(1, n+1):
        u = np.sqrt(u+2)
    return(u)
```

Entrée [7]:
```
exo2(4)
```

Out [7]:
```
1.9903694533443939
```

2. Écrire une fonction, appelée `listeexo2`, qui prend en argument un entier `n` et qui renvoie la liste de tous les termes de la suite (depuis u_0) jusqu'au `n`-ième. *On vérifiera que l'évaluation de* `listeexo2` *en* `4` *donne une liste finissant par environ* `1.96`.

Entrée [8]:
```
import numpy as np

def listeexo2(n):
    u=0
    L=[0]
    for k in range(1, n+1):
        u = np.sqrt(u+2)
        L.append(u)
    return(L)
```

Entrée [9]:
```
listeexo(3)
```

Out [9]:
```
[0, 1.4142135623730951, 1.8477590650225735, 1.9615705608064609]
```

3. Ecrire un programme qui renvoie le premier entier naturel n tel que $|u_n - 2| < 10^{-2}$. *On vérifiera que le programme renvoie* $n = 4$.

Entrée [10]:
```
import numpy as np

n = 0
u = 0

while np.abs(u-2)>= 10**-2:
    u = np.sqrt(u+2)
    n = n+1
print(n)
```

Out [10]:
```
4
```

Exercice 5 On considère la fonction f définie sur $]0,+\infty[$ par,

$$\forall x \in]0,+\infty[, \qquad f(x) = 2 - \frac{1}{2}\ln(x) - x$$

On admet que l'équation $g(x) = 0$ admet une unique solution sur $]0,+\infty[$, notée α. On peut par ailleurs montrer que $\alpha \in [1,\mathrm{e}]$. Recopier et compléter le programme suivant pour qu'il détermine une valeur approchée de α à 10^{-3} près par la méthode de la dichotomie.

Entrée [11]:
```
# définition de la fonction g
def g(x):
    return 2-1/2*np.log(x)-x

# dichotomie
a = 1
b = np.e
while b-a > 10**(-3) :
    m = (a+b)/2
    if g(a)*g(m)<0 :
        (a,b) = (a,m)
    else:
        (a,b) = (m,b)
print(a)
```

Out [11]:
```
1.7265781559792641
```