Fiche cours 10 : Probabilités

1 Bibliothèque et fonctions

Pour pouvoir simuler des expériences aléatoires en python, nous aurons besoin de la bibliothèque suivante : `random`. Pour pouvoir l'importer :

```
import random
```

Dans cette bibliothèque, il est possible de simuler la plupart des expériences aléatoires à l'aide des fonctions suivantes :

1. `random.randint` 2. `random.random`

La fonction `randint` permet de tirer un entier de manière aléatoire. Par exemple, si on effectue la commande `random.randint(1,5)` la valeur de sortie sera un entier $n \in [1, 5]$. Par construction de cette fonction, la valeur est tirée de manière uniforme sur l'ensemble qu'on lui donne.

La fonction `random` permet d'obtenir un réel compris entre $[0, 1]$. Le tirage est lui aussi effectué de manière uniforme.

2 Exemples

Tirages On considère une urne comportant 3 boules rouges, 2 boules blanches, 4 boules noires. On cherche à modéliser les expériences aléatoires suivantes :

— On tire 5 boules successivement avec remise.

— On tire 5 boules successivement et sans remise.

Dans les deux cas, on s'intéresse aux couleurs obtenues pour chacun des tirages. La première expérience peut être modéliser à l'aide de la fonction

```
import random
def avecRemise() :
     urne=["R","R","R","B","B","N","N","N","N"]  # on représente l'urne par une liste
     tirage=[]
     for i in range(5) :
         k=random.randint(0,8)
         tirage.append(urne[k])
     return tirage
```

Pour la deuxième expérience, il nous suffit d'enlever des éléments dans la liste au fur et à mesure.

```
import random
def sansRemise() :
     urne=["R","R","R","B","B","N","N","N","N"]  # on représente l'urne par une liste
     tirage=[]
     for i in range(5) :
         k=random.randint(0,len(urne)-1)
         tirage.append(urne[k])
         urne.pop(k)       # supprime l'élément en indice k de urne
     return tirage
```

Lancer d'une pièce truquée On considère une pièce où il y a une chance sur 3 d'obtenir face. On peut utiliser la fonction `random` pour modéliser cette expérience.

```
import random
def pieceTruquee() :
    x=random.random()
    if x<(1/3) :
        return "face"
    else :
        return "pile"
```

Modélisation générale On considère le modèle probabiliste suivant :

pour tout i $\{1, 2, \cdots, n\}$ la probabilité d'avoir i est de p_i

que l'on cherche à simuler.

```
import random
def proba(L) : # L=[p1,p2,...,pn], c'est la liste des différentes probabilités.
    S=0
    x=random.random()
    for i in range(len(L)) :
        S=S+L[i]
        if x<=S :
            return (i+1)
```