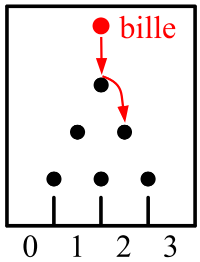
FLUCTUATION D'ÉCHANTILLONNAGE 17.2

La planche de Galton



Sur la planche ci-contre, on a planté de manière régulière trois rangées de clous en quinconce.
On lâche une bille au sommet et celle-ci rebondit de clou en clou jusqu'à l'une des cases numérotées du bas. A chaque clou, la bille a la même probabilité de rebondir à gauche ou à droite.

I) Taille de l'échantillon et fluctuation

1) On appelle p_0, p_1, p_2 et p_3 les probabilités qu'une bille tombe respectivement dans les cases 0, 1, 2 et 3. Calculer ces 4 probabilités à l'aide d'un arbre de probabilité.

2) La fonction ci-contre simule la chute d'une bille en renvoyant le numéro de la case dans laquelle elle tombe.
Compléter cette fonction, puis vérifier son fonctionnement en exécutant plusieurs fois l'instruction « RésultatChute() ».

```
from random import randint
def RésultatChute():
    somme = 0
    for i in range( …. ):
        rebond = randint(0,1)
        somme = somme + ….
    return somme
```

3) La fonction ci-contre simule la chute de « n » billes et renvoie la proportion des billes tombées dans la case « numéro ».
Compléter cette fonction, puis vérifier son fonctionnement en exécutant par exemple les instructions : « Echantillon(100,0) », puis « Echantillon(100,1) », « Echantillon(100,2) »,...

```
def Echantillon(n, numéro):
    somme = 0
    for i in range(n):
        if RésultatChute() == ….:
            somme = somme + ….
    proportion = somme / ….
    return proportion
```

4) a) En vous aidant de la fonction « Echantillon » ci-dessus, simuler la chute de 1000 billes en déterminant la proportion des billes tombées dans la case 0.
b) Même chose pour les cases 1, 2 et 3.
c) Les résultats obtenus sont-ils cohérents avec les probabilités trouvées en 1) ?

5) Le script ci-contre affiche un graphique grâce à la bibliothèque « pylab » :
L'instruction « plot(x, y, '+', color='blue') » affiche le point de coordonnées (x, y) en traçant un '+' de couleur bleue.
L'instruction « axis([x_{min}, x_{max} ,y_{min} ,y_{max}]) » configure les axes.
L'instruction « show() » affiche le graphique.
a) Combien d'échantillons la fonction va-t-elle simuler ?
b) Quel est la taille de ces échantillons ?
c) Décrire par une phrase précise ce que contient la variable f.
d) Quelle probabilité la variable f permet-elle d'estimer ?

```
from pylab import plot,axis,show
for i in range(100):
    f = Echantillon(1000, 1)
    plot(i, f, '+', color='blue')
axis([0, 100 ,0 ,1])
show()
```

6) Modifier ce script pour qu'il affiche sur un même graphique en bleu 100 échantillons de taille 100 et en rouge 100 échantillons de taille 1000. Est-ce que la taille des échantillons a une influence indiscutable sur l'amplitude de la fluctuation ?

II) Taille de l'échantillon et estimation de l'erreur

1) On s'intéresse maintenant à la fréquence des billes tombées dans la case 1. En s'appuyant sur le travail ci-dessus, écrire une fonction « ProportionBonsEchantillons(n) » qui simule 100 échantillons de n billes, et pour chaque échantillon, calcule l'écart entre cette fréquence et p_1 puis renvoie la proportion d'échantillons pour lesquels cet écart est inférieur à $\frac{1}{\sqrt{n}}$. On s'aidera des fonctions « sqrt » (racine carrée) et « fabs » (valeur absolue) du module « math ».

2) Tester cette fonction avec plusieurs valeurs de n.
Quel est environ le pourcentage des échantillons pour lesquels cet écart est inférieur à $\frac{1}{\sqrt{n}}$?