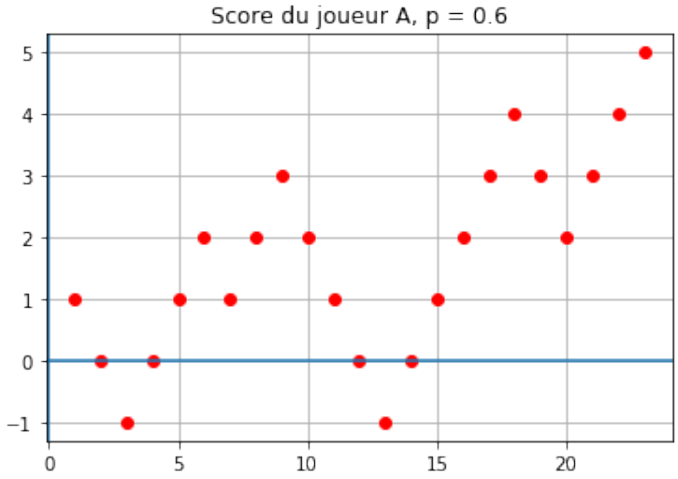
Corrigé disponible sur http://www.frederic-junier.org/TS2020/Progression/TS_2020.html.

Deux joueurs A et B s'affrontent dans un jeu vidéo. Un joueur marque +1 s'il gagne une partie, −1 s'il perd. Le vainqueur du duel est le premier qui arrive à 5.
On suppose qu'à chaque partie, le joueur A possède trois chances sur 5 de gagner. Le déroulement du jeu peut être assimilé au déplacement aléatoire d'un pion sur un axe vertical : au départ, le pion est à l'origine : si A gagne, le pion monte d'un cran ; si B gagne il descend d'un cran.

1 Exemple de marche aléatoire

Le graphique ci-contre représente une marche aléatoire associée à un duel, en abscisse est représenté le nombre de parties et en ordonnée le score du joueur A.



1. Qui a gagné le duel représenté ? Après combien de parties ?
2. Quel est le nombre de parties gagnées par A ?
3. Combien de fois le score de A est-il revenu à 0 ? Interprétez ces éventualités.

2 Nombre de parties jouées

On note N la variable aléatoire qui indique le nombre de parties jouées jusqu'à ce qu'il y ait un gagnant.

4. Quelle est la plus petite valeur que N peut prendre ? Avec quelle probabilité ?
5. Justifier que N peut prendre une infinité de valeurs.

3 Algorithme de simulation du jeu

6. Compléter la fonction `Python` contenu le fichier `cadeau.py`, pour qu'elle affiche le graphique d'évolution du score du joueur A comme ci-dessus et qu'elle retourne le nombre de parties nécessaires pour obtenir un vainqueur. `p` est la probabilité que le joueur A marque un point sur une partie.

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


def simulation(p):
    """Simule un duel du jeu suivant : joueur A + 1 et joueur B -1 avec une
        probabilite de p
       sinon joueur A +1 et joueur B -1
```

```
    Le duel s'arrete des qu'un joueur atteint 5
    Retourne le nombre de parties necessaires pour obtenir un vainqueur
"""
n = 0
yA = 0
listn = []
listyA = []
while yA != 0: #modifier la condition d'entree de boucle
    nombre_aleatoire = random() #experience aleatoire
    #modification de yA
    if nombre_aleatoire < p:
        "a completer"
        yA = .......
    else:
        "a completer"
        yA = .......
    n = n + 1
    listn.append(n)
    listyA.append(yA)
#trace du graphique
plt.plot(listn, listyA, 'ro')
plt.savefig("simulation-p{}.png".format(p))
plt.grid()
plt.axhline(0)
plt.axvline(0)
plt.title('Score du joueur A, p = {}'.format(p))
plt.show()
return n
```

7. Quel est le rôle des variables `n` et `yA` ?

8. A quelle condition le duel s'arrête-t-il? Déduisez-en la modification qu'il faut apporter à la condition d'entrée de boucle.

9. Tester plusieurs appels de fonction `simulation(0.6)`. Existe-t-il toujours un vainqueur du duel?

10. A partir de la fonction précédente, écrire une fonction `vainqueur(p)` qui simule un duel et retourne 1 si le joueur A est vainqueur et 0 sinon. La fonction ne doit pas générer et afficher de graphique.

11. Compléter la fonction ci-dessous pour qu'elle retourne la fréquence de victoires du joueur A sur un échantillon de `nbexp` duels.

```
def frequenceA_echantillon(nbexp, p):
    victoireA = 0
    for k in range(nbexp):
        ......................
        ......................
        ......................
        ......................
    return ....................
```

12. Les événements S_A = « A vainqueur » et S_B = « B vainqueur » forment-ils une partition de l'univers ?

13. Tester plusieurs appels de fonctions `frequenceA_echantillon(10000, 0.6)` puis `frequenceA_echantillon(100000, 0.6)`.

On considère que la fréquence f de succès de A, observée sur un échantillon de taille $n = 10\,000$, est une approximation de la probabilité p de S_A à $\frac{1}{\sqrt{10000}} = 10^{-2}$ près, puisque $\left[f - \frac{1}{\sqrt{n}} ; f + \frac{1}{\sqrt{n}}\right]$ est un intervalle de confiance permettant d'estimer la probabilité p de S_A au niveau de confiance de 95 % (c'est-à-dire qu'environ 95 % des intervalles de confiance obtenus à partir d'échantillons de taille n contiennent cette probabilité).

Quelle conjecture peut-on faire sur la probabilité de victoire du joueur A ?

14. Ecrire une fonction qui retourne la fréquence de victoires du joueur B sur un échantillon de `nbexp` duels où `p` est la probabilité que le joueur A marque un point sur une partie.

Tester plusieurs appels de fonctions `frequenceB_echantillon(10000, 0.6)` puis `frequenceB_echantillon(100000, 0.6)` et conjecturer la probabilité de victoire du joueur B.

15. Ecrire une fonction `nombre_moyen_partie(nbexp, p)` qui retourne le nombre moyen de parties sur un duel où `p` est la probabilité que le joueur A marque un point sur une partie. La fonction calculera le nombre moyen de parties sur un échantillon de `nbexp` duels de taille suffisamment grande.

4 Réponses

Réponse 1: B après 14 parties Réponse 2: 13 Réponse 3: 4

Réponse 4: $\left(\frac{2}{5}\right)^5 + \left(\frac{3}{5}\right)^5 \approx 0,088$

Réponse 7: `(y-5)*(y+5) != 0` ou `abs(y) != 5` ou `y != 5 and y != -5`

Réponse 11: on peut montrer que la probabilité de S_A est $\frac{(\frac{3}{2})^5 - (\frac{3}{2})^{10}}{1 - (\frac{3}{2})^{10}} \approx 0,884$

Réponse 12: d'un point de vue ensembliste S_A et S_B ne sont pas complémentaires, il pourrait ne pas y avoir de vainqueur sur un nombre infini de parties, mais on peut montrer que la probabilité de S_B est $\frac{(\frac{2}{3})^5 - (\frac{2}{3})^{10}}{1 - (\frac{2}{3})^{10}} \approx 0,116$ soit $1 - \mathbb{P}(S_A)$

Réponse 15: on peut observer que le nombre moyen de parties durant un duel est environ de 19