Thème : probabilités/statistique

Marche aléatoire

TI-83 Premium CE

python™

Résumé : une situation « pseudo réelle » est proposée sur un modèle de marche aléatoire : à quelle valeur peut-on estimer la probabilité de réussite de cette marche aléatoire ?

Mots-clés : bibliothèque `turtle` ; probabilités ; statistique ; marche aléatoire ; loi des grands nombres

Compétences visées

Chercher : « observer, s'engager dans une démarche, expérimenter en utilisant éventuellement des outils logiciels » ; le but ici est de démarrer le travail de manière autonome et de poursuivre par l'outil informatique.

Représenter : « changer de registre » en passant d'un travail « à la main » à un travail informatique en effectuant des simulations.

Calculer : « mettre en œuvre des algorithmes simples », en utilisant et faisant évoluer un programme.

Situation déclenchante

Un cycliste débutant doit traverser un pont de 6 m de large et 10 m de long. On modélise son avancement de la manière suivante : à chaque fois qu'il avance d'un mètre, ce cycliste va vers la droite ou vers la gauche de manière aléatoire ; son écart (vers la droite ou vers la gauche) est égal à 1 mètre.

Au départ, le cycliste est situé au début du pont, à 3 m de chaque bord : quelles sont ses chances de réussite ?

Image libre de droits d'après Pixabay

Problématique

Comment estimer la probabilité de réussite de ce cycliste dans sa traversée du pont ?

Thème : probabilités/statistique

Marche aléatoire

TI-83 Premium CE

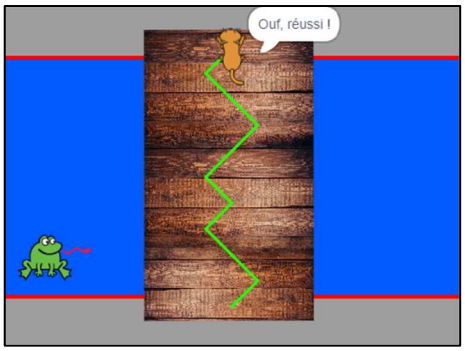
python™

Scénario pédagogique

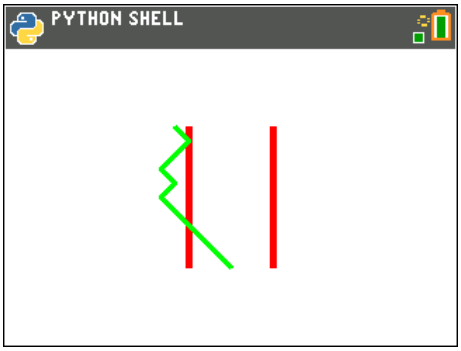
- **Avec la classe :** pour que chacun comprenne bien le sujet, il est intéressant de faire faire ce travail « à la main » à l'aide d'une pièce (pile ou face) sur une feuille de papier.
- **Approche algorithmique :** un travail par groupe peut être pertinent pour la mise en place d'un algorithme permettant de modéliser cette situation.
- **Utilisation d'un code :** un code peut être fourni à l'élève. La partie graphique sera laissée telle quelle et une partie du code, la fonction correspondant à la marche aléatoire en elle-même, peut être à la charge des élèves.
- **Mise en commun** (éventuellement au sein de petits groupes) : on observe que le passage est réussi ou pas à l'aide du graphique... mais est-ce suffisant ?
- **Approche fréquentiste :** mise en place en commun d'un algorithme permettant de donner la fréquence de réussite de la traversée du pont. Les fonctions en lien peuvent être fournies en partie aux élèves selon les choix de l'enseignant.
- **Pour les élèves les plus en avance :** il est possible de leur proposer un ou plusieurs prolongements possibles, décrit en fin de fiche.

Voici les visuels à l'issue des programmes :

en Scratch



avec la TI-83 Premium CE Edition Python

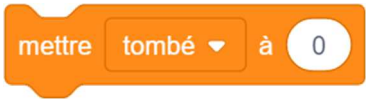
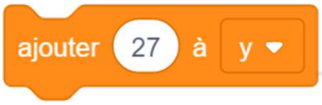


Marche aléatoire

TI-83 Premium CE

python

Avec Scratch

Les briques de codes principales en Scratch pour ce programme	Explications Cette instruction permet de :	Traduction en langage Python sur la TI-83 Premium CE Edition Python
	Affecter la valeur de la variable « alea » avec un nombre aléatoire qui est soit 0, soit 1, l'instruction ne traitant que les nombres entiers.	`a=randint(0,1)`
	Affecter la valeur de la variable « tombé » à 0 ; cette variable contrôle si le personnage tombe ou le nombre de fois où le personnage tombe lors de plusieurs simulations.	Cette variable n'est pas utile en l'état dans Python du fait du renvoi d'un booléen True/False.
	Ajouter à la valeur de la variable « x » la quantité « -27 ».	`x-=10` Cette instruction comporte un opérateur d'affectation augmenté. Il est aussi possible d'écrire `x=x-10`. Les valeurs sont différentes entre Scratch et Python car les tailles d'écran sont différentes : voir « Pour mieux lire le code Python ».
	Ajouter à la valeur de la variable « y » la quantité « 27 ».	`y+=10`
	Déplacer le lutin aux coordonnées indiquées, ici $(x\,;y)$.	`turtle.goto(x,y)`
	Effectuer le quotient de la valeur de la variable « reussite » par celle de « n ».	`c/n`

A noter que dans la dernière version de Scratch, il faut chercher ce qui concerne le stylo dans les extensions : 

Une programmation possible est disponible sur le site de Scratch : scratch.mit.edu/studios/27615196/

Thème : probabilités/statistique

Marche aléatoire

TI-83 Premium CE

python

Avec Python

Le code est fourni tout ou partie aux élèves, code dont voici quelques extraits.

Ce code est composé de plusieurs fonctions :

- La fonction `gt` de paramètres `x` et `y` qui permet au stylo d'aller directement au point de coordonnées $(x\,;y)$ sans écrire.
- La fonction `bord` de paramètres `x1` ; `y1` ; `x2`et `y2` qui trace des segments verts en trait épais joignant les points de coordonnées $(x1\,;y1)$ et $(x2\,;y2)$.
- La fonction `marche_aléatoire` (fonction sans paramètre) qui visualise une expérience de marche aléatoire.

  Cette fonction peut être fournie aux élèves en effaçant certaines lignes, celles qui concernent la condition par exemple.

  Il est important de bien maîtriser les syntaxes du type `x+=10` ; celle-ci signifie que la valeur de `x` augmente de 10.

  L'instruction `turtle.show()` permet de lancer la visualisation à la fin de l'appel à cette fonction.

- La fonction `reussi` : elle est sans paramètre et renvoie un booléen : `True` si la traversée du pont est faite, `False` en cas d'échec.

  On n'a pas de visualisation graphique avec cette fonction.

  La fonction `freq` ; elle a pour paramètre `n`, le nombre de répétitions d'essais de traversée de pont.

  Elle renvoie la fréquence de réussite.

  Elle est très simple dans sa syntaxe et utilise efficacement la fonction `reussi` précédemment décrite.

ÉDITEUR : T12_MAR1
LIGNE DU SCRIPT 0018

```
def bord(x1,y1,x2,y2):
  turtle.color(255,0,0)
  turtle.pensize(2)
  gt(x1,y1)
  turtle.goto(x2,y2)
  turtle.color(0,0,0)
  turtle.pensize(0)
  #turtle.show()
```

Fns… | a A # | Outils | Exéc | Script

ÉDITEUR : T12_MAR1
LIGNE DU SCRIPT 0029

```
from random import *
def marche_aléatoire():
  turtle.clear()
  bord(-30,-50,-30,50)
  bord(30,-50,30,50)
  gt(0,-50)
  x,y=0,-50
  turtle.color(0,255,0)
  turtle.pensize(1)
  for i in range(10):
    a=randint(0,1)
    if a==0:
      x+=10
    else:
      x-=10
    y+=10
    turtle.goto(x,y)
  turtle.show()
```

Fns… | a A # | Outils | Exéc | Script

ÉDITEUR : T12_MAR1
LIGNE DU SCRIPT 0039

```
def reussi():
  x=0
  for i in range(10):
    a=randint(0,1)
    if a==0:
      x+=1
    else:
      x-=1
    if x>3 or x<-3:
      return False
  return True
```

Fns… | a A # | Outils | Exéc | Script

ÉDITEUR : T12_MAR1
LIGNE DU SCRIPT 0060

```
def freq(n):
  c=0
  for i in range(n):
    if reussi():
      c+=1
  return c/n
```

Fns… | a A # | Outils | Exéc | Script

Une programmation possible est disponible sur le site TI : education.ti.com/fr/scratch-python

Thème : probabilités/statistique

Marche aléatoire

TI-83 Premium CE

python™

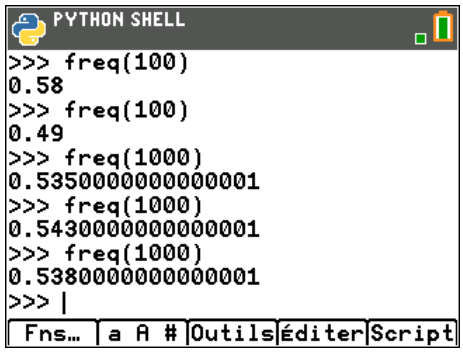
Mode opératoire

Une fois le script exécuté, il faut appuyer sur la touche [var] : les fonctions définies dans le script apparaissent.

Par les flèches directionnelles, il faut sélectionner la fonction `marche_aléatoire` qui permettra de visualiser une situation. Il faut alors appuyer sur la touche [on] pour sortir de cette visualisation graphique.

En sélectionnant la fonction `reussi`, on verra s'afficher `True` ou `False` mais il est difficile de se faire une idée de la fréquence de réussite.

C'est la fonction `freq` qui permettra de le faire : on verra à ce moment-là que les valeurs peuvent différer pour une même valeur de n, et que ces valeurs de fréquence semblent se stabiliser à mesure que n augmente.



Prolongements possibles

Voici des pistes pour les élèves les plus rapides ou qui ont envie de prolonger le travail :

- ajouter dans le graphique une phrase indiquant que la traversée est réussie ou non ;
- dans le graphique représentant la traversée, stopper le tracé dès que l'on bascule du pont ;
- modifier les valeurs numériques du problème de départ et adapter le code en conséquence.

Pour mieux lire le code Python

Il faut bien comprendre que dès que le script rencontre l'instruction `turtle.show`, l'écran propose une image : le dessin demandé par le script. Pour quitter cet écran, il faut appuyer sur la touche [on].

Cependant, il peut être intéressant de tester certaines fonctions graphiques, comme la fonction bord de ce code. Ainsi, on écrit `turtle.show()` à la fin de la fonction bord pour la tester et la valider et par la suite, on commente cette ligne par un `#,` en tapant sur la touche [2nde] suivie de [3] par exemple. Si cette fonction est appelée dans une autre fonction, les instructions se dérouleront sans se « stopper » à l'instruction `turtle.show()`. C'est ce qui a été fait dans le code proposé.

Pour information, sur Scratch, la taille d'écran est de 480 pixels par 360 pixels. Sur la calculatrice, elle est de 320 pixels par 240 pixels. Ces différences nécessitent donc des aménagements pour passer d'une programmation à l'autre, occasionnant des problèmes mathématiques qu'il peut être intéressant à développer pour les élèves les plus en avance.

Bien avoir en tête les coordonnées des extrémités de l'écran de la calculatrice : $(-160\,;115)$ pour le coin supérieur gauche et $(160\,;-115)$ pour le coin inférieur droit.