Thème : statistiques

Ce dé est-il truqué ?

TI-83 Premium CE

python™

Résumé : cette activité propose une simulation classique de lancers de dé à six faces (dé équilibré ou non).

Mots-clés : bibliothèque `random` ; conditionnelle `if` ; simulation ; statistique ; loi des grands nombres

Compétences visées

Chercher : « observer, s'engager dans une démarche, expérimenter en utilisant éventuellement des outils logiciels » avec ici des résultats numériques à considérer avec un regard critique.

Modéliser : « utiliser, comprendre, élaborer une simulation numérique ».

Calculer : « mettre en œuvre des algorithmes simples », en utilisant ici une structure conditionnelle.

Situation déclenchante

Si un dé à six faces est donné, comment savoir s'il est bien équilibré ou non ? L'observation de lancers donnera des indications, pas tout à fait des certitudes !

Et avec combien de lancers ?

Image libre de droits d'après Pixabay

Problématique

Comment simuler un lancer de dé à six faces :

- S'il est bien équilibré ?
- S'il est pipé ?

Au passage, il est intéressant de se questionner sur la simulation numérique : remplace-t-elle réellement un dé ?

Thème : statistiques

Ce dé est-il truqué ?

TI-83 Premium CE

python™

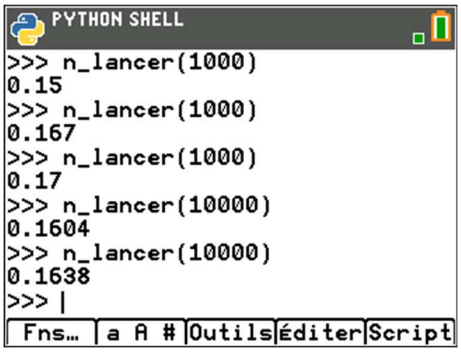
Scénario pédagogique

- **Avec la classe :** lancer une recherche par binôme ou trinôme en distribuant des dés à six faces, avec dans l'idéal, certains de ces dés pipés.
- **Tests :** se mettre d'accord sur un protocole pour déterminer si un dé est bien équilibré ou pas. Après plusieurs essais, basculer vers une simulation numérique nécessitant la création d'un code.
- **Utilisation d'un code :** rédaction de ce code avec la classe avec mise en évidence de l'instruction conditionnelle.
- **Des défis :** les élèves peuvent par la suite modifier le code pour simuler un dé bien équilibré ou pas, et demander à un camarade de « deviner » si le dé est bien équilibré en lançant quelques simulations ; la loi des grands nombres est alors évoquée en pleine classe.
- **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

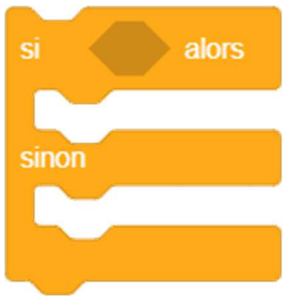
avec la TI-83 Premium CE Edition Python



Ce dé est-il truqué ?

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
	Tirer un nombre entier aléatoire entre les bornes définies par l'utilisateur, 1 et 6 ici.	`randint(0,6)` L'instruction `randint(a,b)` renvoie un nombre entier entre a et b inclus.

	Créer une structure de contrôle conditionnelle.	`if condition:`
		`••instructions si vraie`
		`else:`
		`••instructions si faux`
		
		L'indentation avec •• est nécessaire pour définir les instructions à exécuter dans un bloc du code Python.

	Incrémenter de 1 la valeur de la variable « reussite ».	`c+=1` Cette instruction comporte un opérateur d'affectation augmenté. Il est aussi possible d'écrire `x=x+pas`
	Effectuer le quotient des valeurs des variables « reussite » et « nb_lancers ».	`c/n`

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

Thème : statistiques

Ce dé est-il truqué ?

TI-83 Premium CE

python™

Avec Python

Le code complet est construit avec les élèves. Il est composé de deux fonctions.

- La fonction `un_lancer`, sans paramètre, qui retourne un booléen : `True` ou `False`, selon la condition requise.

  C'est à cet endroit que l'on pourra modifier la fréquence d'apparition d'une face lors du lancer du dé en modifiant la condition `if randint(1,6)==1`.

  La condition `if randint(1,25)<=4` a été commentée, par l'appui de la touche `[2nde]` suivie de `[3]` par exemple.

  Cette ligne pourra intervenir dans un second temps de l'activité pour simuler la réalisation d'une issue de probabilité égale à $\frac{4}{25}$, valeur proche de $\frac{1}{6}$.

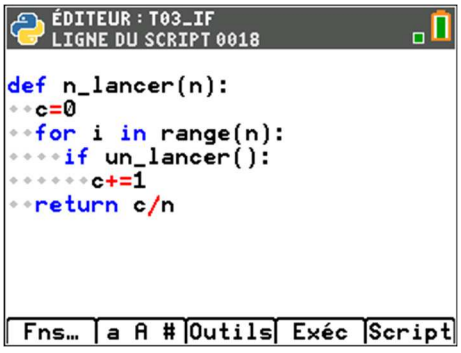
  A noter la nécessité d'importer la bibliothèque `random` pour pouvoir utiliser la fonction `randint`.

- La fonction `n_lancer` de paramètre `n`, le nombre de répétitions, qui va simuler n lancers de dés et retourner la fréquence de réussite de la fonction `un_lancer`.

  A noter dans ce code l'utilisation d'un compteur : la variable `c`, initialisée à 0, qui va cumuler les réussites de la fonction `un_lancer`.

  La valeur `c/n` correspond bien à la fréquence de réussite de la fonction `un_lancer` après n répétitions.

  La syntaxe `if un_lancer()` est très efficace : il faut bien se rappeler que `un_lancer()` est un booléen (`True` ou `False`) ; il est donc inutile de saisir : `if un_lancer()==True`.



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

Prolongements possibles

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

- demander des explications sur le fait que `n_lancer(1000)` ne donnera pas toujours la même valeur ;
- représenter graphiquement, grâce à la bibliothèque `ti_plotlib`, la fréquence de réussite de la fonction `un_lancer` en fonction du nombre de lancers ;
- travailler sur la somme des numéros des faces supérieures de deux dés lancés simultanément ;
- varier le nombre de faces ou le nombre de dés raisonnablement.