Thème : le Web

TI-83 Premium CE EDITION PYTHON

Classe enseignée : 2nde SNT

Page rank

F. GIROD

Compétences visées

Un des objectifs de l'enseignement de SNT est de comprendre que les requêtes menées par un moteur de recherche peuvent donner des résultats différents d'un moteur de recherche à l'autre, puisque ceux-ci utilisent des algorithmes qui leur sont propres. Cette activité vise à comprendre sur des exemples simples la manière dont on peut mesurer la popularité d'une page en utilisant le principe du "surfeur aléatoire".

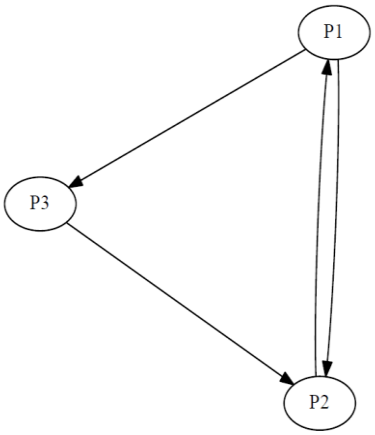
Elle s'inscrit dans la thématique "**le Web**" pour l'activité proposée : "Calculer la popularité d'une page à l'aide d'un graphe simple puis programmer l'algorithme". Nous pouvons notamment travailler les compétences suivantes dans l'activité proposée :

- Modéliser une situation par un graphe.
- Écrire et développer des algorithmes pour résoudre une problématique.

Situation déclenchante

Comment classer une page web en termes de popularité afin d'établir un classement qui permettra de hiérarchiser des réponses à apporter lors d'une recherche menée sur un moteur de recherche ?

Problématique



Voici un modèle (extrêmement) réduit du Web puisqu'il ne comporte que trois pages !

Les liens entre ces pages sont modélisés par un graphe.

On suppose qu'un 'surfeur aléatoire' est actif sur ce réseau.

À quelle fréquence ce surfeur se retrouvera-t-il sur chacune des pages P_1, P_2 et P_3 ?

Déroulement possible du projet

En amont du projet :

Les propositions suivantes permettent d'intégrer l'activité dans le cadre d'une réflexion à mener autour du thème des moteurs de recherche.

- Un groupe d'élèves peut réaliser un travail montrant que les résultats d'une requête ne sont pas tout à fait les mêmes selon le moteur de recherche utilisé.
- Un groupe d'élèves peut présenter une recherche d'une page par un moteur de recherche, en particulier le 'page rank'.
- Un groupe d'élèves peut présenter la notion de référencement.
- Un groupe d'élèves peut préparer un exposé sur le modèle du 'surfeur aléatoire'.

Différentes évolutions possibles pour le projet :

Les propositions suivantes donnent des pistes pour gérer les différences de vitesse d'exécution des élèves.
On peut aussi considérer ces différentes étapes comme une manière de séquencer le projet pour ceux qui en auraient besoin.

- 1ère étape : la première situation est mise en actes par des lancers de dés afin que tous les élèves comprennent le principe utilisé permettant de mettre en place un algorithme.
- 2ème étape : l'algorithme précédent est codé en PYTHON.
- 3ème étape : d'autres situations (d'autres graphes) sont proposées pour s'interroger sur diverses situations et adapter algorithmes et codes.

A la suite du projet :

Synthétiser ce travail par rapport au cours de SNT :

- Pourquoi est-il important de comprendre le principe de l'algorithme de recherche d'un moteur de recherche ?
- Pourquoi le référencement d'un page est-il important ?
- Comment est-il possible de rendre 'artificiellement' populaire une page et comment un moteur de recherche peut-il contourner ces pratiques ?

Pour profiter de tutoriels vidéos, Flasher le QRCode ou cliquer dessus !

Thème : le Web

TI-83 Premium CE EDITION PYTHON

Classe enseignée : 2nde SNT

Fiche méthode

F. GIROD

Proposition de résolutions

Avec des dés :

- On propose aux élèves de 'jouer au surfeur aléatoire' à l'aide de dés quand cela est nécessaire (pour la première page et lorsque l'on tombe sur la page P_1).
- On collecte les résultats de l'ensemble des groupes (en étant critique sur les résultats donnés).
- On détermine la fréquence d'apparition de chaque page, ce qui donne la popularité de chacune d'elle.

Avec la calculatrice :

- Il s'agit à présent de coder cette situation en créant une fonction qui retournera la fréquence d'apparition de chaque page après **n** 'clics' du surfeur.

Etapes de résolution

L'importation de la bibliothèque *random* permet d'utiliser la fonction **randint**. On a choisi ici une syntaxe qui n'importe que la fonction **randint** parmi toutes les fonctions que contient cette bibliothèque.

La fonction **pagesuiv1** va permettre de suivre le graphe proposé précédemment:

- Si on se trouve sur la page 1, la page suivante est, au hasard, la page 2 ou la page 3 ;
- Si on se trouve sur la page 2, la page suivante est la page 1 ;
- Si on se trouve sur la page 3, la page suivante est la page 2.

Ces trois cas impliquent l'utilisation des conditions **if** / **elif** / **elif** qui sont accessibles par **Fns** / **Ctl**.

```
ÉDITEUR : PAGERANK
LIGNE DU SCRIPT 0002
from random import randint

#cas 1
def pagesuiv1(pa):
  if pa == 1:
    ps = randint(2,3)
  elif pa == 2:
    ps = 1
  elif pa == 3:
    ps = 2
  return ps
Fns… | a A # | Outils | Exéc | Script
```

On crée une fonction **rep** qui prend pour paramètre **n**, le nombre de répétitions de l'application du principe du 'surfeur aléatoire' ; elle renvoie la fréquence d'apparition de chacune des pages.

On utilise des compteurs stockés dans les variables **c1, c2 et c3** qui compteront le nombre de fois où l'on passe - respectivement - sur les pages P_1, P_2 et P_3. Ces compteurs sont initialisés à 0.
Sur la même ligne, on initialise la variable **pa** (qui représente le numéro de la page) à un nombre aléatoire entier compris entre 1 et 3.

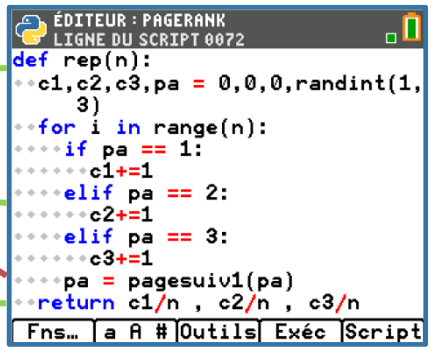
```
ÉDITEUR : PAGERANK
LIGNE DU SCRIPT 0072
def rep(n):
  c1,c2,c3,pa = 0,0,0,randint(1,
    3)
  for i in range(n):
    if pa == 1:
      c1+=1
    elif pa == 2:
      c2+=1
    elif pa == 3:
      c3+=1
    pa = pagesuiv1(pa)
  return c1/n , c2/n , c3/n
Fns… | a A # | Outils | Exéc | Script
```

Pour profiter de tutoriels vidéos, Flasher le QRCode ou cliquer dessus

La variable **pa** est égale au numéro de la page visitée par le surfer.
On répète **n** fois le fait d'incrémenter (augmenter de 1) le compteur de la page visitée par le surfer aléatoire.

Une fois sortie des trois conditions, la variable **pa** prend la valeur **pagesuiv1(pa)** c'est-à-dire que l'on suit la règle édictée par le graphe.
C'est l'utilisation successive de fonctions simples en python, en particulier le fait d'utiliser une fonction dans une fonction qui rend ce langage efficace.

On retourne la fréquence d'apparition de la page P_1, de la page P_2 et de la page P_3.

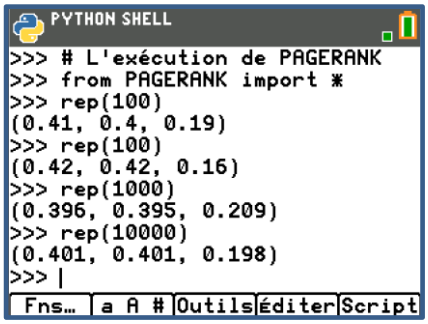


Rappel : `c+=1` est un raccourci pour `c=c+1`

Remarque

Une fois le programme fonctionnel, on peut formuler plusieurs remarques :

- Les résultats sont cohérents avec ce qui avait été collecté suite aux expérimentations avec les dés ;
- Les pages P_1 et P_2 ont une popularité similaire ;
- Plusieurs essais avec la même valeur de **n** ne donnent pas tout à fait les mêmes résultats ;
- Plus **n** est grand, plus on semble obtenir des résultats proches.



Ces deux derniers points peuvent éventuellement être réinvestis en cours de mathématiques.

Pour aller plus loin

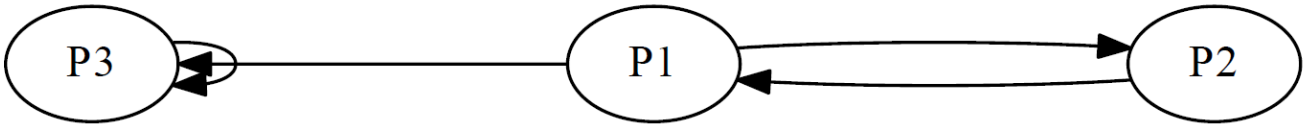
On peut alors proposer d'autres graphes modélisant les liens entre les pages et dans chaque cas, observer le problème posé et y apporter une solution.

Un nouveau graphe demande juste à créer une nouvelle fonction du type **pagesuiv** ; le cœur du programme restera le même.

Il faudra juste modifier dans la fonction **rep** l'appel à la fonction **pagesuiv** en la nommant en cohérence avec la situation étudiée.

Pour profiter de tutoriels vidéos, Flasher le QRCode ou cliquer dessus

Ci-dessous, des propositions de graphe avec le problème posé et la solution proposée :



Dans ce cas-là, la page P_3 va 'absorber' les autres : une fois que l'on s'y trouve, on ne la quitte plus ! On parle de 'page puits'.

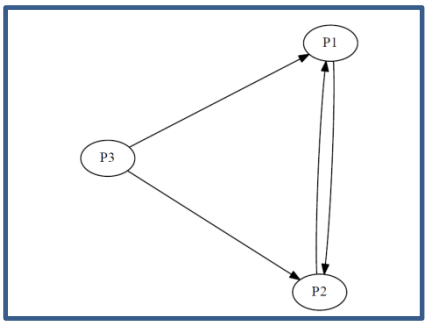
Proposition d'aménagement : lorsque l'on se trouve sur la page P_3, on pointe au hasard vers P_1, P_2 ou P_3. Cela donne la nouvelle fonction **pagesui22** :

```
ÉDITEUR : PAGERANK
LIGNE DU SCRIPT 0027
#cas 2 avant modif
def pagesuiv2(pa):
  if pa == 1:
    ps = randint(2,3)
  elif pa == 2:
    ps = 1
  elif pa == 3:
    ps = 3
  return ps

_
Fns... | a A # | Outils | Exéc | Script
```

```
EDITEUR : PAGERANK
LIGNE DU SCRIPT 0035
#cas 2 après modif
def pagesuiv22(pa):
  if pa == 1:
    ps = randint(2,3)
  elif pa == 2:
    ps = 1
  elif pa == 3:
    ps = randint(1,3)#on pointe
      au hasard une des trois pag
      es
  return ps_
Fns... | a A # | Outils | Exéc | Script
```

Dans ce troisième cas (présenté ci-contre), à moins de tomber au départ sur la page P_3, on n'y retourne jamais !



Pour pallier ce problème, on propose d'aller sur une des trois pages au hasard avec une probabilité p (de l'ordre de 0,15 en général) et le reste du temps, de suivre les indications du graphe.

Cela donne la fonction **pagesuiv32** ci-dessous, pour laquelle on a pris comme valeur par défaut de probabilité 0,15.

Par la suite, un appel de la fonction **rep** simulera n répétitions avec une probabilité d'aller sur une des trois pages au hasard de 0,15, et de suivre le cheminement du graphe avec une probabilité de 0,85.

```
ÉDITEUR : PAGERANK
LIGNE DU SCRIPT 0058
def pagesuiv32(pa, p=15):
  if randint(1,100)<p:
    ps = randint(1,3)
  else :
    if pa == 1:
      ps = 2
    elif pa == 2:
      ps = 1
    elif pa == 3:
      ps = randint(1,2)
  return ps
Fns... | a A # | Outils | Exéc | Script
```

Pour profiter de tutoriels vidéos, Flasher le QRCode ou cliquer dessus !