ECE 1 - Année 2017-2018
Lycée français de Vienne
Mathématiques - F. Gaunard
`http://frederic.gaunard.com`

T.P. n°7

Premières simulations

1 Introduction. Loi uniforme et instruction `rand()`

En mathématiques, on travaille souvent avec des *variables aléatoires* (*i.e* des fonctions dont l'argument est une issue de l'univers modélisant l'expérience aléatoire), mais en informatique, on **simule** une **réalisation** de ces variables aléatoires (ce qui correspond à la notion d'observation en statistique).

L'instruction `rand()` permet de générer un nombre réel au hasard entre 0 et 1 (on dit que la fonction `rand()` simule la loi uniforme sur $[0;1]$, qui sera introduite avec plus de détails dans le chapitre sur les v.a.r à densité).

Exemple. Voici le résultat obtenu après quatre exécutions de la commande `rand()`. Essayez, vous obtiendrez à chaque fois un réel entre 0 et 1 différent!

```
--> [rand(), rand(), rand(), rand()]
 ans  =

   0.7263507   0.1985144   0.5442573   0.2320748
```

Ainsi, pour simuler un évènement dont la probabilité est p, on génère un nombre au hasard entre 0 et 1 à l'aide de `rand()`, si ce nombre est entre 0 et p (ce qui se passe avec probabilité p), on interprète cela comme un succès (c'est à dire une réalisation de l'évènement), si ce nombre est supérieur à p (ce qui se passe avec probabilité $1-p$), on considère que c'est un échec (l'évènement n'a pas lieu).

Exemple. Pile ou *Face*.
Le programme suivant permet de simuler le lancer d'une pièce équilibrée et affiche le résultat obtenu. On représente donc l'évènement "obtenir *Pile*" (de probabilité 1/2) par un *autre* évènement (celui que le nombre généré aléatoirement par `rand()` soit inférieur (ou égal) à 1/2.

```
function []=tirage()
    if rand() <= 0.5 then
        disp("Pile")
    else
        disp("Face")
    end
endfunction
```

☞ Ici la fonction ne renvoie pas de valeur numérique (on veut seulement un affichage), il y a donc un élément vide `[]` à la place de l'argument de sortie. Elle ne prend également pas d'argument d'entrée.

☞ Modifier le programme précédent pour simuler le lancer d'une pièce truquée dont la probabilité d'obtention de *Face* est de 2/3. Modifier encore la fonction si cette probabilité est p où p devient le paramètre de la fonction.

Exercice 1. (Tirages dans une urne)

On considère une urne U contenant 3 boules bleues, 4 boules blanches et 5 boules rouges. On effectue n tirages dans cette urne et on s'intéresse à la couleur de la boule à chacun des tirages.

(1) Créer un programme qui demande à l'utilisateur un nombre de tirages n et simule l'expérience en affichant les couleurs successives obtenues.
(2) Écrire une fonction `y=urne(n)` qui renvoie 1 si on a tiré au moins une boule rouge lors des n premiers tirages et 0 sinon.
(3) Illustrer graphiquement que lors d'une infinité de tirages, presque sûrement, on va tirer une boule rouge.

Exercice 2. (Un dé à 6 faces)

(1) Écrire une instruction qui permet de simuler le jet d'un dé (non truqué) à six faces.
(2) Écrire ensuite un programme qui, à partir d'un nombre n de lancers entré par l'utilisateur renvoie la fréquence d'obtention d'un 6.
(3) Faire tourner le programme précédent pour des valeurs de plus en plus grandes de n. Que semble-t-on observer?

Exercice 3. (Un jeu et un compteur)

Écrire un programme qui, après avoir tiré (secrètement) au hasard un nombre entier entre 1 et 10, demande à l'utilisateur de le deviner et compte le nombre de coups nécessaires.

Exercice 4. (Le lièvre et la tortue)

Un lièvre et une tortue partent d'un point O, origine d'un axe gradué, et se déplacent uniquement vers la droite. On lance un dé équilibré: si le résultat est inférieur ou égal à 5, la tortue avance d'une unité. Sinon, le lièvre avant de 6 unité. Le gagnant est celui qui arrive le premier au point d'abscisse 6.

(1) Déterminer mathématiquement la probabilité que la tortue gagne la course.
(2) Créer une fonction `course()` qui simule l'expérience en renvoyant 1 si la tortue gagne et 0 si c'est le lièvre.
(3) Améliorer le programme précédent pour qu'il affiche la fréquence des courses gagnées par la tortues lors de la réalisations de n expériences. Comparer le résultat à celui obtenu à la première question.

Exercice 5. (D'après **EDHEC 2000**)

Soit $n \geq 2$. On lance n fois une pièce de monnaie, dont la probabilité d'apparition de *Pile* est $p \in]0;1[$. Pour tout entier naturel $k \geq 2$, on dira que le $k-$ième lancer est un *changement* s'il amène un résultat différent du $(k-1)-$ième lancer.

(1) Écrire une fonction `CountCh(n,p)` simule l'expérience et renvoie le nombre de changements au cours de n lancers.
(2) Écrire une suite d'instructions permettant de calculer `CountCh(10000,p)` pour `p` allant de $0,1$ à $0,9$ avec un pas de $0,1$ et qui affiche ensuite le diagramme en rectangles.

Exercice 6. (La ruine du joueur)

Notre ami José se rend au Casino avec n euros en poche. Il s'installe à la table de roulette américaine. À chaque partie remportée, il gagne un euro, sinon il perd un euro. On suppose que le casino dispose d'une fortune de N euros. José ne s'arrête que lorsque il est ruiné (ou que le casino est ruiné). Simuler l'évolution de sa fortune. (On écrira une fonction `casino(n,N,p)` renvoyant un vecteur dont les composantes seront les valeurs successives de sa fortune.)