TP9
Simulation de variables aléatoires finies

1 Loi uniforme discrète

Exercice 1 **Simulations de lancers d'un dé équilibré**

1. **Avec rand**
   Écrire des instructions Scilab permettant de construire une matrice-ligne (ou liste) A, 1 ligne, 100 colonnes, dont les coefficients sont des simulations indépendantes de lancers d'un dé équilibré à 6 faces. On utilisera **rand**.

```
A=
```

2. **Avec grand**
   Même question, mais avec **grand** cette fois.

```
B=
```

3. À l'aide de la fonction `sum`, qui calcule la somme des coefficients d'une matrice, obtenir une valeur approchée de l'espérance de X pour $X \hookrightarrow \mathcal{U}(\llbracket 1,6 \rrbracket)$. Comparer les résultats donnés par A et par B.

```

```

2 Loi de Bernoulli

Exercice 2 **Simulations de lancers d'une pièce truquée**

1. **Avec rand**
   Écrice une fonction `Bernoulli(r,s,p)` renvoyant une matrice X possédant r lignes et s colonnes et dont les coefficients suivent (indépendamment) une loi de Bernoulli de paramètre p.

```
function X=Bernoulli(r,s,p)
    X=zeros(r,s)
    for i=1:r
        for j=1:s
            x=rand()
            if ....... then
                X(i,j)=...
            else
                X(i,j)=...
            end
        end
    end
endfunction
```

2. Définir alors une matrice-ligne $A \in \mathscr{M}_{1,1000}(\mathbb{R})$ contenant les simulations de 1000 lancers d'une pièce de monnaie amenant Pile (considéré comme un succès) avec la probabilité $\frac{1}{4}$.

```
A=
```

3. **Avec grand**
Définir une matrice-ligne $B \in \mathscr{M}_{1,1000}(\mathbb{R})$ contenant les simulations de 1000 lancers d'une pièce de monnaie amenant Pile (considéré comme un succès) avec la probabilité $\frac{1}{4}$ (en utilisant `grand`).

```
B=
```

4. Grâce aux simulations précédentes, obtenir une valeur approchée de l'espérance d'une variable aléatoire suivant une loi $\mathscr{B}(\frac{1}{4})$. Comparer avec le résultat du cours.

3 Loi binomiale

Exercice 3 On lance 10 fois une pièce truquée amenant Pile avec la probabilité $\frac{1}{4}$. Soit X la variable aléatoire égale au nombre de Pile obtenus. X suit alors la loi $\mathscr{B}(10, \frac{1}{4})$. On souhaite simuler 1000 réalisations de cette variable aléatoire.

1. **Avec la fonction Bernoulli**
Écrice une fonction `Binomiale(r,s,n,p)` renvoyant une matrice X possédant r lignes et s colonnes et dont les coefficients suivent (indépendamment) une loi binomiale de paramètres n et p.

```
function X=Binomiale(r,s,n,p)
    X=zeros(r,s)
    for i=1:r
        for j=1:s
            Y=Bernoulli(1,n,p) // n simulations de B(p)
            X(i,j)= ......     // nombre de succès
        end
    end
endfunction
```

2. Définir alors une matrice-ligne $A \in \mathscr{M}_{1,1000}(\mathbb{R})$ contenant 1000 simulations de l'expérience décrite au début de l'exercice.

```
A=
```

3. **Avec grand**
Définir une matrice-ligne $B \in \mathscr{M}_{1,1000}(\mathbb{R})$ contenant 1000 simulations de l'expérience (avec grand).

```
B=
```

4. Grâce aux simulations précédentes, obtenir une valeur approchée de l'espérance d'une variable aléatoire suivant une loi $\mathscr{B}(10, \frac{1}{4})$. Comparer avec le résultat du cours.

```

```

5. On souhaite compter le nombre de 2 obtenus dans la matrice A.
   (a) Écrire un programme utilisant un compteur et une boucle for.

```

```

   (b) Avec la fonction **sum**. La commande `sum(A==2)` permet de compter directement le nombre de 2. Essayer.

```
>>> sum(A==2)
```

Exercice 4 On dispose de 2 pièces de monnaie équilibrées. On effectue des lancers successifs selon le protocole suivant :

- à l'étape 1, on lance les 2 pièces,
- à l'étape 2, on lance les pièces ayant amené Pile à l'étape 1 (s'il en existe),
- à l'étape 3, on lance les pièces ayant amené Pile à l'étape 2 (s'il en existe),

et ainsi de suite. Compléter le programme pour qu'il simule cette expérience aléatoire. On pourra considérer que Pile= 1 et Face= 0. À chaque étape on affichera une liste de la forme [pièce1, pièce2].

```
// Premier lancer
piece1 = ...
piece2 = ...
NbPile = .......
disp(........)

// Lancers suivants
while ...........
    if piece1 == ... then
        piece1 = ...
    end
    if ...................... then
        ............
    end
    NbPile = ...
    ...
end
```