ETUDE DE LA LOI BINOMIALE À TRAVERS LE JEU DE   
SOCIÉTÉ LE PERUDO

OBJECTIF

L’objectif de ce projet est d’étudier et de mettre en application le calcul de probabilité associé à la loi binomiale, en utilisant le langage informatique Python. La mise en application de ce calcul se fera à travers l’exemple du jeu de société nommé « Perudo », son principe de fonctionnement étant basé sur cette loi statistique.

SOMMAIRE

[I\_ PRINCIPE - 2 -](#_Toc524980024)

[I.1\_ La loi binomiale - 2 -](#_Toc524980025)

[I.1.1\_ Probabilité de la loi binomiale - 2 -](#_Toc524980026)

[I.1.2\_ Coefficient binomial - 2 -](#_Toc524980027)

[I.1.3\_ L’opérateur factoriel - 2 -](#_Toc524980028)

[I.2\_ Le Perudo - 3 -](#_Toc524980029)

[I.2.1\_ Principe du jeu - 3 -](#_Toc524980030)

[I.2.2\_ Règles du jeu et statistiques - 3 -](#_Toc524980031)

[II\_ DÉVELOPPEMENT - 5 -](#_Toc524980032)

[II.1\_ Calcul de la probabilité associée à la loi binomiale - 5 -](#_Toc524980033)

[II.1.1\_ Produit factoriel - 5 -](#_Toc524980034)

[II.1.2\_ Coefficient binomial - 6 -](#_Toc524980035)

[II.1.3\_ Probabilité binomiale - 7 -](#_Toc524980036)

[II.2\_ Calculs des statistiques du Perudo - 8 -](#_Toc524980037)

[II.2.1\_ Calculs des probabilités - 8 -](#_Toc524980038)

[II.2.2\_ Calculs des probabilités cumulées - 9 -](#_Toc524980039)

[II.2.3\_ Extraction des résultats - 10 -](#_Toc524980040)

[III\_ RÉSULTATS & ANALYSES - 11 -](#_Toc524980041)

[III.1\_ Résultats des calculs de probabilités - 11 -](#_Toc524980042)

[III.2\_ Résultats des calculs de probabilités cumulées - 13 -](#_Toc524980043)

# PRINCIPE

## La loi binomiale

### Probabilité de la loi binomiale

La loi binomiale est une loi statistique qui permet notamment de calculer la probabilité d’occurrence d’un événement aléatoire qui répond aux trois conditions suivantes :

1. L’événement est un processus aléatoire qui consiste en une série d’essais ;
2. Chaque essai ne peut résulter qu’en une des deux issues possibles qui sont le succès ou l’échec ;
3. La probabilité de succès pour chaque essai indépendant est une constante fixe.

*E.g. : nous cherchons à déterminer la probabilité d’obtenir 60 fois « pile » sur une série de 100 lancers (les essais) d’une pièce de monnaie. Le résultat du lancer d’une pièce de monnaie est un processus qui peut être considéré comme aléatoire. Il ne peut résulter qu’en deux issues à savoir « pile » que nous associons ici à un succès de l’essai, et « face » qui est considéré considéré dans cette expérience comme un échec. Enfin, la probabilité pour chaque lancer d’obtenir « pile » (donc un succès) est connue et fixe et est de ½.*

Les trois conditions inhérentes à l’utilisation de la loi binomiale sont donc réunies. Cet exemple simple est uniquement présenté ici pour illustrer une utilisation possible de la loi binomiale, mais il ne sera pas traité dans ce document.

En pratique et lorsque les conditions d’application de la loi binomiale présentées ci-dessus sont réunies, la probabilité que la variable aléatoire binomiale *X* soit égale à une valeur choisie *x* est donnée par la formule :

Avec *X, une variable aléatoire binomiale dont la valeur est empiriquement déterminée par le nombre de succès au cours d’une série d’essais ;*

*x, nombre de succès choisis lors d’une série d’essais ;*

*, le coefficient binomial (voir II.1.2\_;*

*n, le nombre d’essais ;*

*p, la probabilité de succès pour un essai.*

### Coefficient binomial

Le coefficient binomial (d’où la loi tire son nom) permet de déterminer le nombre de combinaisons qu’il est possible de créer lorsque sont choisis *x* éléments parmi un ensemble d’un total de *n* éléments. Dans un tel cas, l’ordre dans lequel sont choisis les *x* éléments parmi *n* n’a pas d’importance. Le coefficient binomial est donné par la formule :

### L’opérateur factoriel

Le calcul du coefficient binomial fait intervenir l’opérateur factoriel ‘*!’*. Cet opérateur correspond au produit de chaque entier positif de la suite inférieure ou égale au nombre auquel il est appliqué tel que :

Avec pour exception et convention mathématique :

N.B. : Les notions présentées ci-dessus à propos du calcul de probabilité associé à la loi binomiale, du coefficient binomial et de l’opérateur factoriel sont certes simples pour un public initié aux mathématiques ; mais il nous apparaît utile de les rappeler pour la pédagogie du document car elles représentent les trois étapes à compléter pour atteindre l’objectif d’automatiser le calcul de la probabilité d’occurrence d’un événement aléatoire selon la loi binomiale.

## Le Perudo

### Principe du jeu

Le Perudo est un jeu de dés pour lequel chaque joueur a besoin d’un gobelet opaque et de cinq dés à six faces ordinaires, exceptée pour la face de valeur 1 qui est remplacée par une illustration différente. Bien qu’il n’y ait en théorie pas de limite du nombre de joueurs, le jeu tel que distribué est prévu pour 6 joueurs, soit un maximum de 30 dés en jeu. Le projet tel que présenté ici s’appuie sur ce nombre « maximal » de dés, mais peut être adapté très facilement pour n’importe quel nombre maximal de dés en jeu comme nous le montrons en XX.XX.XX.

Au cours d’une partie, chaque joueur jette ses dés aléatoirement sous son gobelet, ignorant les résultats des dés des autres joueurs. Le principe est que le joueur, ne connaissant que ses propres dés, doit faire un pari sur le nombre de dés au total ayant une certaine valeur (*e.g. : cinq 4*). Le joueur suivant, qui ne connaît donc que ses propres dés, peut alors surenchérir soit sur le nombre de dés de même valeur (*e.g. : sept 4*) soit sur la valeur des dés (*e.g. : cinq 5*). Les joueurs parient chacun à leur tour de jeu dans le sens horaire, jusqu’à ce qu’un joueur décide à son tour de jeu que le pari du joueur précédent est faux. Il dit alors « dudo » pour dénoncer ce pari et tous les joueurs dévoilent leurs dés. S’il y a au moins le nombre de dés de la valeur donnée par le précédent parieur, le joueur ayant dénoncé le pari perd un dé, sinon c’est le dernier parieur qui en perd un.

Enfin, la face illustrée du dé est appelée « paco », elle apporte une spécificité car cette face peut prendre n’importe quelle valeur parmi 2, 3, 4, 5 ou 6. Un joueur qui dispose donc après son lancer de deux pacos et quatre 4 a donc potentiellement dans son jeu deux 2, deux 3, six 4, deux 5 ou deux 6.

### Règles du jeu et statistiques

Pour l’ensemble des explications à suivre, nous présentons des exemples concrets en considérant le cas de six joueurs jouant tous avec 5 dés réguliers chacun au démarrage du jeu.

#### Règle 1

***La règle -*** Le premier jouer fait une enchère qui ne peut pas porter sur le nombre de pacos. Il doit donc annoncer un nombre de dés d’une valeur de son choix (2, 3, 4, 5 ou 6) parmi la totalité des 30 dés en jeu. Puisque les pacos présents parmi les 30 dés peuvent prendre n’importe quelle valeur de 2 à 6, le pari du joueur doit en tenir compte.

*E.g. : le joueur annonce neuf 4*.

***La stat -*** Chaque face de chaque dé régulier a un probabilité d’1/6 d’apparaître. Sur un tirage de trente dés, il y a statistiquement cinq dés de chaque valeur, soient cinq pacos, cinq 2, cinq 3, cinq 4, cinq 5 et cinq 6. Comme décrit en II.2.1, n’importe quelle valeur de 2 à 6 peut être attribuée au paco. La statistique est donc modifiée de telle sorte qu’il y a statistiquement, sur un tirage de 30 dés, dix 2 (cinq pacos + cinq 2), dix 3, dix 4, dix 5 ou dix 6. En résumé, puisque deux faces de chaque dé peuvent compter pour la valeur annoncée, chaque face à une probabilité de 1/6 d’apparaître, mais chaque valeur a une probabilité de 1/3 d’apparaître.

#### Règle 2

***La règle -*** Pour les joueurs suivant le premier, chacun peut soit surenchérir sur le nombre de dés de la valeur donnée, soit surenchérir sur la valeur du nombre de dés donnés, soit parier sur le nombre de pacos en jeu. Si ce dernier choix est fait, le nombre de pacos annoncé doit être au moins égal à la moitié du nombre de dés annoncé au tour précédent.

*E.g. : le joueur suivant le premier ayant annoncé neuf 4 décide d’annoncer un nombre de pacos, il doit annoncer au minimum 9/2 = 4,5 soient 5 pacos*.

***La stat -*** Lorsqu’un joueur décide d’annoncer des pacos, seules les dés sur lesquels apparaissent effectivement des pacos compteront. La probabilité pour chaque dé redevient donc de 1/6, et pour 30 dés il y a donc statistiquement de nouveau cinq pacos.

N.B. : Après qu’un joueur ait décidé de parier sur des pacos, il est possible de revenir à des paris ordinaires sur des valeurs. Le joueur le joueur doit alors annoncer un nombre de dés au minimum égal à 2 fois le nombre de pacos précédemment annoncés plus 1.

*E.g. : le joueur précédent a annoncé 5 pacos, il faut donc annoncer au moins 5 x 2 + 1 = 11 dés de la valeur de son choix*. Les statistiques autour du nombre de dés annoncés retrouvent donc une configuration initiale, avec une probabilité de 1/3 d’apparaître pour n’importe quelle valeur (autre que le paco).

#### Règle 3

***La règle -*** Lorsqu’un joueur vient de perdre son avant-dernier dé (il n’en a donc plus qu’un seul), les règles de la manche en cours sont modifiées. Ainsi les pacos ne peuvent plus prendre n’importe quelle valeur souhaitée, et la valeur des dés du pari annoncé ne peut plus être modifiée. Les parieurs suivant ne pourront donc qu’augmenter le nombre de dés de cette valeur annoncée. De plus le premier parieur peut directement parier sur le nombre de pacos.

*E.g. : le premier parieur annonce cinq 4, le suivant ne peut qu’annoncer un nombre supérieur de 4.*

***La stat -*** La configuration du jeu reprend une statistique classique ou chaque valeur des dés à une probabilité égale d’apparaître soit 1/6.

#### Résumé du principe du jeu

Chaque pari au cours du jeu tente donc de déterminer un nombre *x* de dés ayant la même valeur, parmi la totalité *n* des dés en jeu, quelque soit la valeur annoncée. Selon la phase du jeu en cours, chaque valeur peut avoir une probabilité *p* d’apparaître de 1/3 ou de 1/6. Pour parier au mieux, nous devrions donc disposer des tables de probabilités pour tous les cas possibles de nombre de dés que l’on peut annoncer (*x* =1 à *n*), parmi le nombre de dès présents en jeu (*n* = 2 à 30). Ces tables doivent être produites pour les deux probabilités que chaque valeur a d’apparaître selon la phase du jeu (*p* = 1/3 ou *p* = 1/6).

Le projet est développé en deux phases :

1. Développement de la fonction permettant le calcul de la probabilité associée à la loi binomiale ;
2. Développement d’un script permettant la création automatique des tables pour tous les cas de figure.

# DÉVELOPPEMENT

L’objectif est de créer une fonction qui permette de calculer la probabilité recherchée en fonction de *x*, le nombre de dés pariés pour une valeur donnée, de *n*, le nombre de dés total en jeu au cours de la manche, et de *p*, la probabilité que chaque valeur a d’apparaître.

Une fois la fonction créée, il suffira de l’intégrer dans une série de boucles qui permettront l’itération automatique sur le nombre de dés pariés, puis sur le nombre de dés en jeu dans une manche.

Enfin, il suffira de réaliser ces calculs pour deux cas, selon que la probabilité *p* qu’une valeur apparaisse est de 1/3 ou 1/6.

## Calcul de la probabilité associée à la loi binomiale

Le calcul de la probabilité associée à la loi binomiale se fait selon la formule présentée en II.1.1\_. Ce calcul fait intervenir le coefficient binomial présenté en II.1.2\_, lui-même faisant intervenir le produit factoriel présenté en II.1.3\_.

Concrètement, il faut donc développer en premier lieu une fonction permettant de réaliser un produit factoriel, puis une fonction permettant de calculer le coefficient binomial, et enfin la fonction permettant calculer la probabilité binomiale.

### Produit factoriel

***But -*** Calculer le produit factoriel d’un nombre *x*. Une des fonctions du produit factoriel est de déterminer le nombre d’arrangements possibles d’un groupe d’objets.

***Nom -*** Nous définissons le nom qui sera utilisé pour appeler la fonction : *factorial( )*.

***Argument(s) -*** La fonction prend un argument : *x*.

La fonction ne fait appel à aucune fonction, aucune librairie, et est développée telle que :

1 **def factorial**(*x*)**:**2 num **=** 1  
3 **if** *x* **==** 0**:**4 **return** num  
5 **else:**6  **while** *x* **>** 0**:**7 num **=** num **\*** *x*8  *x* **-=** 19 **return** num

***L1 -*** Déclaration de la fonction qui prend un argument nommé *x*.

***L2 -*** Déclaration de la variable *num* permettant le stockage du calcul par incrémentation, assignation de la valeur de départ *num = 1*.

***L3 -*** Initialisation du cas si *x = 0*.

***L4 -*** La fonction retourne la variable *num* de valeur 1 (selon II.1.3).

***L5 -*** Initialisation des cas pour *x > 0*.

***L6 -*** Initialisation d’une boucle while, tant que *x > 0*.

***L7 -*** La variable *num = 1* au départ de la boucle est multipliée par *x* donc *num = x.*

***L8 –*** La valeur de *x* est diminuée de 1 et la boucle recommence si *x > 0* (au 2e tour de la boucle,   
*num = x* est multiplié par *x – 1* donc *num = x \* (x – 1)* et ainsi de suite jusque *x = 1*).

***L9 -*** La fonction retourne la valeur résultante de *num* lorsque la boucle est conclue.

Ci-dessous, le résultat de l’exécution de la fonction en forçant l’affichage des valeurs de *x* et de *num* pour chaque itération de la boucle while. L’exécution de la fonction est conforme à l’attendu :

>>> factorial(3)

x = 3

num = 3 # Départ de la boucle, *num = 1* est multipliée par *x = 3*

x = 2 # La variable *x* décroit de 1, *x = 3 – 1 = 2*

num = 6 # *x > 0*, la boucle continue, *num = 3* est multipliée par *x = 2*

x = 1 # La variable *x* décroit de 1, *x = 2 – 1 = 1*

num = 6 # *x > 0*, la boucle continue, *num = 6* est multipliée par *x = 1*

x = 0 # La variable *x* décroit de 1, *x = 1 – 1 = 0*. La boucle est terminée.

Ci-dessous, le résultat de l’exécution de la fonction sans affichage intermédiaire :

>>> factorial(8)  
40320

>>> factorial(0)  
1

>>> factorial(100)  
30414093201713375576366966406747986832057064836514787179557289984

### Coefficient binomial

***But -*** Calculer le coefficient binomial entre deux nombres. Une des fonctions du coefficient binomial est de déterminer le nombre de combinaisons possibles en choisissant *x* objets parmi un ensemble de *n* objets.

***Nom -*** Nous définissons le nom qui sera utilisé pour appeler la fonction : *coeff\_binom( )*.

***Argument(s) -*** La fonction prend deux arguments : *x,* *n*.

La fonction fait appel à la fonction *factorial( ),* à aucune librairie, et est développée telle que :

1 **def coeff\_binom**(*x*, *n*)**:**2 fac\_x **=** factorial(*x*)3 fac\_n **=** factorial(*n*)4 fac\_nx **=** factorial(n-x)  
5 comb **=** fac\_n **/** (fac\_x **\***(fac\_nx)6 **return** int(comb)

***L1 -*** Déclaration de la fonction qui prend deux arguments nommés *x* et *n*.

***L2 -*** Déclaration de la variable *fac\_x* permettant le stockage du calcul *x!* par appel de la fonction *factorial( )* sur l’argument *x*.

***L3 -*** Déclaration de la variable *fac\_n* permettant le stockage du calcul *n!* par appel de la fonction *factorial( )* sur l’argument *n*.

***L4 -*** Déclaration de la variable *fac\_nx* permettant le stockage du calcul *(n - x)!* par appel de la fonction *factorial( )* sur la soustraction d’arguments *(n - x)*.

***L5 -*** Déclaration de la variable *comb* permettant le stockage du calcul du coefficient binomial (selon II.1.2).

***L6 -*** La fonction retourne la valeur résultante de *comb* issue du calcul sous forme d’un nombre entier.

Ci-dessous, le résultat de l’exécution de la fonction en forçant l’affichage des valeurs intermédiaires du calcul. L’exécution de la fonction est conforme à l’attendu :

>>> coeff\_binom(5, 8)

fac\_x = 120 # Le résultat de *5!* est assigné à la variable *fac\_x*

fac\_n = 40320 # Le résultat de *8!* est assigné à la variable *fac\_n*

fac\_nx = 6 # Le résultat de *(8-5)!* est assigné à la variable *fac\_nx*

comb = 56.0 # Le résultat du quotient est assigné à la variable *comb* de type décimal

56 # La valeur de la variable *comb* est convertie au type entier

Ci-dessous, le résultat de l’exécution de la fonction sans affichage intermédiaire :

>>> coeff\_binom(5, 49)

1906884

### Probabilité binomiale

***But -*** Calculer la probabilité associée à la loi binomiale d’avoir *x* succès parmi *n* essais pour un événement ayant une probabilité *p*.

***Nom -*** Nous définissons le nom qui sera utilisé pour appeler la fonction : *proba\_binom( )*.

***Argument(s) -*** La fonction prend trois arguments : *x*, *n* et *p*.

La fonction fait appel à la fonction *coeff\_binom( ),* à aucune librairie, et est développée telle que :

1 **def proba\_binom**(*x*, *n*, *p*)**:**2 prob **=** coeff\_binom(*x*, *n*) **\*** (*p***\*\****x*) **\*** ((1**-***p*)**\*\***(*n***-***x*))3 **return {:.4E}"**.format(prob)

***L1 -*** Déclaration de la fonction qui prend trois arguments nommés *x*, *n* et *p*.

***L2 -*** Déclaration de la variable *prob* permettant le stockage du calcul de la probabilité associée à la loi binomiale (selon II.1.1).

***L3 -*** La fonction retourne la valeur résultante de *prob* issue du calcul sous forme d’un nombre décimal arrondi au 4e chiffre en écriture scientifique.

Ci-dessous, le résultat de l’exécution de la fonction en forçant l’affichage des valeurs intermédiaires du calcul. L’exécution de la fonction est conforme à l’attendu :

>>> proba\_binom(5, 30, 1/3)

coeff\_binom = 142506

px = 0.004115226337448558

(1-p)(n-x) = 3.96021280423662e-05

prob = 0.02322444797862319

2.3224E-02 # La valeur de la variable *prob* est arrondie et formatée

Ci-dessous, le résultat de l’exécution de la fonction sans affichage intermédiaire :

>>> proba\_binom(10, 30, 1/6)

1.2961E-02

## Calculs des statistiques du Perudo

### Calculs des probabilités

Pour chaque pari fait par un joueur, l’enjeu est de déterminer la probabilité que pour *x* dés parmi *n* apparaissent la valeur *k* avec une probabilité *p* pour cette valeur d’apparaître pour chaque dé. Comme nous l’avons présenté en II.2.2, *p* peut prendre deux valeurs : 1/3 et 1/6. Nous commençons par traiter le cas pour lequel p = 1/3.

Nous choisissons de développer notre code de la façon suivante :

1. Initialiser les deux variables *n* (nombre total de dés) et *p* (probabilité d’apparition de chaque valeur) en leur assignant leur valeur ;
2. Initialiser un *DataFrame* vide de la librairie *Pandas* dans lequel sont stockées les données issues des calculs de probabilité, pour lequel les lignes correspondent au nombre de dés pariés par un joueur (*x*), et les colonnes correspondent au nombre de dés totaux en jeu (*n*) ;
3. Développer deux boucles itératives imbriquées qui permettent de calculer pour chaque valeur de *x* et *n* possibles la probabilité associée à la loi binomiale en faisant appel à la fonction développée en III.1.3.

Le code permettant ces trois étapes est développé tel que :

1  **import** pandas **as** pd  
2  **import** numpy **as** np  
3  **import** Functions **as** F45  n **=** 30  
6  p **=** 1**/**3  
7  
8  df\_proba **=** pd.DataFrame()  
9  
10 **for** j **in** range(n, 1, **-**1)**:**11data **=** {str(j) **:** [F.proba\_binom(i, j, p) **for** i **in** np.arange(j**+**1)]}12  
13 df\_proba **=** pd.concat([df\_proba, pd.DataFrame(data)],axis**=**1)  
14  
15 df\_proba.index.name **= '# of dice bet'**16 print(df\_proba, **'\n'**)

***L5 -*** Déclaration de la variable *n* permettant le stockage du nombre maximal de dés en jeu, assignation de la valeur *n = 30*.

***L6 -*** Déclaration de la variable *p* permettant le stockage de la probabilité pour chaque valeur de dé d’apparaitre, assignation de la valeur *p = 1/3*.

***L8 -*** Déclaration de la variable *df\_proba* permettant le stockage des résultats des calculs de probabilité pour chaque cas possible de *x* et *n*, assignation d’un *DataFrame* vide.

***L10 -*** Initialisation d’une boucle itérative décroissante *for* sur la variable *j* faisant varier le nombre total de dés en jeu, de *n = 30* à *2* par pas de *1*.

***L11 -*** Déclaration d’une variable *data* permettant le stockage intermédiaire des calculs de probabilité sous forme d’un dictionnaire possédant une seule paire clé-valeur. La clé prend la valeur de *j* (soit le nombre total de dés pour les calculs en cours) sous forme de caractère. Les valeurs associées à la clé sont les résultats des calculs de probabilité effectués dans une boucle itérative croissante *for* sur la variable *i* interne au dictionnaire. Cette boucle permet de faire varier le nombre *x* de dés sur lequel les paris des joueurs sont faits. Pour chaque valeur de *j* (donc de *n* dés), *data* contient donc les probabilités que *x* dés apparaissent pour tout *x* de *0* à *j*.

***L13 -*** Le DataFrame *df\_proba* est mis à jour en lui accolant les résultats contenus dans le dictionnaire *data*, sous forme d’une nouvelle colonne dont l‘index est le nombre total de dés en jeu *j* pour lequel les calculs sont effectués, et dont les valeurs sont les probabilités pour chaque pari possible du nombre de dés. Lorsque les calculs pour une colonne sont terminés et que les résultats sont transférés dans le tableau, la valeur de *j* est diminuée de 1 et le processus recommence pour cette nouvelle valeur, remplissant de nouveau le dictionnaire *data* des résultats des calculs, puis les accolant à *df\_proba* qui grandit d’une colonne à chaque nouvelle itération de la boucle sur *j*.

***L15 -*** L’index de df\_proba est renommé par ‘# of dice bet’.

***L16 -*** Affichage de*df\_proba*.

Ci-dessous la description de l’itération des boucles *for* sur *j* et *for* sur *i* imbriquées, pour le calcul de toutes les probabilités :

>>> n = 30

>>> p = 1/3

j = 30 # Départ de la boucle sur j

i = 0 # Pour j = 30, départ de la boucle sur i

proba\_binom(0, 30, 1/3) = 5.2151E-06 # Calcul de la probabilité

i = 1 # Première itération de la boucle sur i

proba\_binom(1, 30, 1/3) = 7.8226E-05

i = 2 # Deuxième itération de la boucle sur i

proba\_binom(2, 30, 1/3) = 5.6714E-04

... # Les itérations sur i se poursuivent

i = 30 # Dernière sur itération sur i pour j = 30

proba\_binom(30, 30, 1/3) = 4.8569E-15 # Fin de la boucle sur i pour j = 30

j = 29 # Première itération de la boucle sur j

i = 0 # Pour j = 29, départ de la boucle sur i

proba\_binom(0, 29, 1/3) = 7.8226E-06

... # Les itérations sur i et j se poursuivent

j = 2 # Dernière itération sur j, j = 2

i = 1 # Pour j = 2, dernière boucle sur i

proba\_binom(1, 2, 1/3) = 4.4444E-01

i = 2 # Dernière itération de la boucle sur i

proba\_binom(2, 2, 1/3) = 1.1111E-01 # Fin des boucles sur i et j

Nous obtenons donc un tableau de 31 lignes pour le nombre de dés pariés (de 0 à 30) et 29 colonnes pour le nombre total de dés en jeu (de 2 à 30). Les tableaux des résultats des calculs de probabilité sont présentés en section III.

### Calculs des probabilités cumulées

Le script développé jusqu’ici permet d‘obtenir les probabilités de tous les cas possibles en fonction du nombre total de dés en jeu, et du nombre de dés pariés pour lesquels apparaissent une valeur donnée. Cependant la probabilité de voir *x* dés parmi *n* prendre la valeur *k* est différente de la probabilité associée aux paris faits au cours du jeu. En effet les paris annonçant x dés parmi n sont gagnants si, sur l’ensemble des dés dévoilés, il y a au moins le nombre de dés pariés.

*E.g. : il y a 26 dés en jeu, le joueur 1 parie qu’il y a sept 4. D’après les résultats en* ***Tableau 1****, la probabilité de ce pari est de 0,136. Cependant si le joueur 2 dénonce ce pari, le joueur 1 gagne s’il y a au moins sept dés de valeurs 4, i.e. qu’il gagne également s’il y en a 8, 9, 10 etc.*

La véritable probabilité que le joueur 1 gagne son pari est donc la somme des probabilités pour le nombre de dés annoncés, et tous les nombres de dés supérieurs. En pratique, il nous faut donc créer un second tableau calculant les probabilités cumulées par ordre décroissant du nombre de dés pariés (du 30 à 0).

Le code permettant la création de ce tableau est présenté ci-dessous :

18 df\_stat **=** df\_proba.copy()  
19   
20 **for** j **in** range(0, n**-**1)**:**21  **for** i **in** range(0, n**+**1)**:**22 df\_stat.iloc[i, j] **=** (np.sum(df\_stat.iloc[i**:**, j]))  
23   
24 print(df\_stat, **'\n'**)

***L18 -*** Déclaration de la variable *df\_stat* permettant le stockage des résultats des calculs de probabilités cumulées, assignation des valeurs de *df\_proba* par copie complète du tableau.

***L20 -*** Initialisation d’une boucle itérative croissante *for* sur la variable *j* faisant varier l’indice de position des éléments d’un tableauselon les colonnes.

***L21 -*** Initialisation d’une boucle itérative croissante *for* sur la variable *i* faisant varier l’indice de position des éléments d’un tableau selon les lignes.

***L22 -*** Chaque élément de *df\_stat* de position *(i, j)* prend la valeur de la somme des éléments de la colonne *j* à partir de *i* : .

***L24 -*** Affichage de *df\_stat*.

La description de l’itération sur deux boucles imbriquées est présentée en III.2.1.

Nous obtenons donc un tableau de 31 lignes pour le nombre de dés pariés (de 0 à 30) et 29 colonnes pour le nombre total de dés en jeu (de 2 à 30). Le tableau des résultats des calculs de probabilité cumulées est présenté en section III.

### Extraction des résultats

La dernière étape de notre script consiste à exporter les deux tableaux contenant les résultats des calculs afin de pouvoir les utiliser en post-traitement, ou en présentation comme par exemple dans ce document. Compte tenu de la nature de ce document et des dimensions des tableaux, nous choisissons de les exporter au format d’un document Excel, d’extension .xlsx.

La librairie Pandas permet d’exporter simplement un tableau de données vers un document Excel, nécessairement créé en amont de l’exécution du code. La portion de code utilisée pour exporter les tableaux vers le fichier ‘Perudo\_res.xlsx’ est présentée ci-dessous :

26 writer **=** pd.ExcelWriter(**'Perudo\_res.xlsx'**)  
27 df\_proba.to\_excel(writer, sheet\_name**='Sheet1'**)  
28 df\_stat.to\_excel(writer, sheet\_name**='Sheet2'**)  
29 writer.save()

***L26 -*** Déclaration de l’objet *writer* permettant l’ouverture d’un flux vers le fichier Excel de destination.

***L27 -*** Extraction du DataFrame *df\_proba* par le biais de l’objet *writer* et à destination de la feuille *‘Sheet1’*.

***L28 -*** Extraction du DataFrame *df\_stat* par le biais de l’objet *writer* et à destination de la feuille *‘Sheet2’*.

***L29 -*** Fermeture du flux de l’objet *writer*.

# RÉSULTATS & ANALYSES

## Calculs de probabilités

### Cas 1 : p = 1/3

**Tableau 1** **- Probabilité que x dés (lignes) apparaissent parmi un total de n dés (colonnes) pour p = 1/3.**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **x DÉS** | **NOMBRE TOTAL DE DÉS EN JEU POUR p = 1/3** | | | | | | | | | | | | | |
| **n = 30** | **n = 29** | **n = 28** | **n = 27** | **n = 26** | **n = 25** | **n = 24** | **n = 23** | **n = 22** | **n = 21** | **n = 20** | **n = 19** | **n = 18** | **n = 17** |
| **0** | 5,22E-06 | 7,82E-06 | 1,17E-05 | 1,76E-05 | 2,64E-05 | 3,96E-05 | 5,94E-05 | 8,91E-05 | 1,34E-04 | 2,00E-04 | 3,01E-04 | 4,51E-04 | 6,77E-04 | 1,01E-03 |
| **1** | 7,82E-05 | 1,13E-04 | 1,58E-04 | 2,29E-04 | 3,43E-04 | 4,75E-04 | 6,83E-04 | 1,02E-03 | 1,47E-03 | 2,11E-03 | 3,01E-03 | 4,29E-03 | 6,09E-03 | 8,63E-03 |
| **2** | 5,67E-04 | 7,92E-04 | 1,11E-03 | 1,54E-03 | 2,15E-03 | 2,96E-03 | 4,10E-03 | 5,64E-03 | 7,72E-03 | 1,05E-02 | 1,43E-02 | 1,93E-02 | 2,59E-02 | 3,45E-02 |
| **3** | 2,65E-03 | 3,57E-03 | 4,81E-03 | 6,44E-03 | 8,58E-03 | 1,14E-02 | 1,50E-02 | 1,97E-02 | 2,57E-02 | 3,33E-02 | 4,29E-02 | 5,46E-02 | 6,90E-02 | 8,63E-02 |
| **4** | 8,93E-03 | 1,16E-02 | 1,50E-02 | 1,93E-02 | 2,47E-02 | 3,13E-02 | 3,95E-02 | 4,93E-02 | 6,11E-02 | 7,50E-02 | 9,11E-02 | 1,09E-01 | 1,29E-01 | 1,51E-01 |
| **5** | 2,32E-02 | 2,90E-02 | 3,60E-02 | 4,44E-02 | 5,43E-02 | 6,58E-02 | 7,89E-02 | 9,37E-02 | 1,10E-01 | 1,27E-01 | 1,46E-01 | 1,64E-01 | 1,81E-01 | 1,96E-01 |
| **6** | 4,84E-02 | 5,81E-02 | 6,91E-02 | 8,14E-02 | 9,50E-02 | 1,10E-01 | 1,25E-01 | 1,41E-01 | 1,56E-01 | 1,70E-01 | 1,82E-01 | 1,91E-01 | 1,96E-01 | 1,96E-01 |
| **7** | 8,29E-02 | 9,54E-02 | 1,09E-01 | 1,22E-01 | 1,36E-01 | 1,49E-01 | 1,61E-01 | 1,71E-01 | 1,78E-01 | 1,82E-01 | 1,82E-01 | 1,78E-01 | 1,68E-01 | 1,54E-01 |
| **8** | 1,19E-01 | 1,31E-01 | 1,42E-01 | 1,53E-01 | 1,61E-01 | 1,67E-01 | 1,71E-01 | 1,71E-01 | 1,67E-01 | 1,59E-01 | 1,48E-01 | 1,33E-01 | 1,16E-01 | 9,64E-02 |
| **9** | 1,46E-01 | 1,53E-01 | 1,58E-01 | 1,61E-01 | 1,61E-01 | 1,58E-01 | 1,52E-01 | 1,42E-01 | 1,30E-01 | 1,15E-01 | 9,87E-02 | 8,14E-02 | 6,43E-02 | 4,82E-02 |
| **10** | 1,53E-01 | 1,53E-01 | 1,50E-01 | 1,45E-01 | 1,37E-01 | 1,26E-01 | 1,14E-01 | 9,96E-02 | 8,44E-02 | 6,91E-02 | 5,43E-02 | 4,07E-02 | 2,89E-02 | 1,93E-02 |
| **11** | 1,39E-01 | 1,32E-01 | 1,23E-01 | 1,12E-01 | 9,96E-02 | 8,62E-02 | 7,24E-02 | 5,88E-02 | 4,60E-02 | 3,45E-02 | 2,47E-02 | 1,66E-02 | 1,05E-02 | 6,13E-03 |
| **12** | 1,10E-01 | 9,91E-02 | 8,72E-02 | 7,47E-02 | 6,23E-02 | 5,03E-02 | 3,92E-02 | 2,94E-02 | 2,11E-02 | 1,44E-02 | 9,25E-03 | 5,55E-03 | 3,07E-03 | 1,53E-03 |
| **13** | 7,62E-02 | 6,48E-02 | 5,36E-02 | 4,31E-02 | 3,35E-02 | 2,51E-02 | 1,81E-02 | 1,24E-02 | 8,12E-03 | 4,98E-03 | 2,85E-03 | 1,49E-03 | 7,08E-04 | 2,95E-04 |
| **14** | 4,63E-02 | 3,70E-02 | 2,87E-02 | 2,15E-02 | 1,56E-02 | 1,08E-02 | 7,11E-03 | 4,44E-03 | 2,61E-03 | 1,42E-03 | 7,11E-04 | 3,20E-04 | 1,26E-04 | 4,21E-05 |
| **15** | 2,47E-02 | 1,85E-02 | 1,34E-02 | 9,34E-03 | 6,23E-03 | 3,95E-03 | 2,37E-03 | 1,33E-03 | 6,96E-04 | 3,32E-04 | 1,42E-04 | 5,34E-05 | 1,68E-05 | 4,21E-06 |
| **16** | 1,16E-02 | 8,10E-03 | 5,45E-03 | 3,50E-03 | 2,14E-03 | 1,23E-03 | 6,67E-04 | 3,33E-04 | 1,52E-04 | 6,23E-05 | 2,22E-05 | 6,67E-06 | 1,58E-06 | 2,63E-07 |
| **17** | 4,77E-03 | 3,10E-03 | 1,92E-03 | 1,13E-03 | 6,29E-04 | 3,27E-04 | 1,57E-04 | 6,86E-05 | 2,69E-05 | 9,15E-06 | 2,62E-06 | 5,89E-07 | 9,29E-08 | 7,74E-09 |
| **18** | 1,72E-03 | 1,03E-03 | 5,87E-04 | 3,15E-04 | 1,57E-04 | 7,26E-05 | 3,05E-05 | 1,14E-05 | 3,73E-06 | 1,02E-06 | 2,18E-07 | 3,27E-08 | 2,58E-09 |  |
| **19** | 5,43E-04 | 2,99E-04 | 1,55E-04 | 7,45E-05 | 3,31E-05 | 1,34E-05 | 4,82E-06 | 1,50E-06 | 3,93E-07 | 8,03E-08 | 1,15E-08 | 8,60E-10 |  |  |
| **20** | 1,49E-04 | 7,47E-05 | 3,48E-05 | 1,49E-05 | 5,80E-06 | 2,01E-06 | 6,02E-07 | 1,50E-07 | 2,94E-08 | 4,02E-09 | 2,87E-10 |  |  |  |
| **21** | 3,56E-05 | 1,60E-05 | 6,62E-06 | 2,48E-06 | 8,28E-07 | 2,39E-07 | 5,73E-08 | 1,07E-08 | 1,40E-09 | 9,56E-11 |  |  |  |  |
| **22** | 7,28E-06 | 2,91E-06 | 1,05E-06 | 3,39E-07 | 9,41E-08 | 2,17E-08 | 3,91E-09 | 4,89E-10 | 3,19E-11 |  |  |  |  |  |
| **23** | 1,27E-06 | 4,43E-07 | 1,37E-07 | 3,68E-08 | 8,18E-09 | 1,41E-09 | 1,63E-10 | 1,06E-11 |  |  |  |  |  |  |
| **24** | 1,85E-07 | 5,54E-08 | 1,43E-08 | 3,07E-09 | 5,11E-10 | 5,67E-11 | 3,54E-12 |  |  |  |  |  |  |  |
| **25** | 2,21E-08 | 5,54E-09 | 1,15E-09 | 1,84E-10 | 2,05E-11 | 1,18E-12 |  |  |  |  |  |  |  |  |
| **26** | 2,13E-09 | 4,26E-10 | 6,59E-11 | 6,82E-12 | 3,93E-13 |  |  |  |  |  |  |  |  |  |
| **27** | 1,58E-10 | 2,36E-11 | 2,36E-12 | 1,31E-13 |  |  |  |  |  |  |  |  |  |  |
| **28** | 8,45E-12 | 8,45E-13 | 4,37E-14 |  |  |  |  |  |  |  |  |  |  |  |
| **29** | 2,91E-13 | 1,46E-14 |  |  |  |  |  |  |  |  |  |  |  |  |
| **30** | **4,86E-15** |  |  |  |  |  |  |  |  |  |  |  |  |  |

**Tableau 1 (suite) - Probabilité que x dés (lignes) apparaissent parmi un total de n dés (colonnes) pour p = 1/3.**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **x DÉS** | **NOMBRE TOTAL DE DÉS EN JEU POUR p = 1/3** | | | | | | | | | | | | | | |
| **n = 16** | **n = 15** | **n = 14** | **n = 13** | **n = 12** | **n = 11** | **n = 10** | **n = 9** | **n = 8** | **n = 7** | **n = 6** | **n = 5** | **n = 4** | **n = 3** | **n = 2** |
| **0** | 1,52E-03 | 2,28E-03 | 3,43E-03 | 5,14E-03 | 7,71E-03 | 1,16E-02 | 1,73E-02 | 2,60E-02 | 3,90E-02 | 5,85E-02 | 8,78E-02 | 1,32E-01 | 1,98E-01 | 2,96E-01 | **4,44E-01** |
| **1** | 1,22E-02 | 1,71E-02 | 2,40E-02 | 3,34E-02 | 4,62E-02 | 6,36E-02 | 8,67E-02 | 1,17E-01 | 1,56E-01 | 2,05E-01 | 2,63E-01 | 3,29E-01 | 3,95E-01 | 4,44E-01 | **4,44E-01** |
| **2** | 4,57E-02 | 5,99E-02 | 7,79E-02 | 1,00E-01 | 1,27E-01 | 1,59E-01 | 1,95E-01 | 2,34E-01 | 2,73E-01 | 3,07E-01 | 3,29E-01 | 3,29E-01 | 2,96E-01 | 2,22E-01 | 1,11E-01 |
| **3** | 1,07E-01 | 1,30E-01 | 1,56E-01 | 1,84E-01 | 2,12E-01 | 2,38E-01 | 2,60E-01 | 2,73E-01 | 2,73E-01 | 2,56E-01 | 2,19E-01 | 1,65E-01 | 9,88E-02 | 3,70E-02 |  |
| **4** | 1,73E-01 | 1,95E-01 | 2,14E-01 | 2,30E-01 | 2,38E-01 | 2,38E-01 | 2,28E-01 | 2,05E-01 | 1,71E-01 | 1,28E-01 | 8,23E-02 | 4,12E-02 | 1,23E-02 |  |  |
| **5** | 2,08E-01 | 2,14E-01 | 2,14E-01 | 2,07E-01 | 1,91E-01 | 1,67E-01 | 1,37E-01 | 1,02E-01 | 6,83E-02 | 3,84E-02 | 1,65E-02 | 4,12E-03 |  |  |  |
| **6** | 1,90E-01 | 1,79E-01 | 1,61E-01 | 1,38E-01 | 1,11E-01 | 8,35E-02 | 5,69E-02 | 3,41E-02 | 1,71E-02 | 6,40E-03 | 1,37E-03 |  |  |  |  |
| **7** | 1,36E-01 | 1,15E-01 | 9,18E-02 | 6,89E-02 | 4,77E-02 | 2,98E-02 | 1,63E-02 | 7,32E-03 | 2,44E-03 | 4,57E-04 |  |  |  |  |  |
| **8** | 7,65E-02 | 5,74E-02 | 4,02E-02 | 2,58E-02 | 1,49E-02 | 7,45E-03 | 3,05E-03 | 9,14E-04 | 1,52E-04 |  |  |  |  |  |  |
| **9** | 3,40E-02 | 2,23E-02 | 1,34E-02 | 7,18E-03 | 3,31E-03 | 1,24E-03 | 3,39E-04 | 5,08E-05 |  |  |  |  |  |  |  |
| **10** | 1,19E-02 | 6,70E-03 | 3,35E-03 | 1,44E-03 | 4,97E-04 | 1,24E-04 | 1,69E-05 |  |  |  |  |  |  |  |  |
| **11** | 3,25E-03 | 1,52E-03 | 6,09E-04 | 1,96E-04 | 4,52E-05 | 5,65E-06 |  |  |  |  |  |  |  |  |  |
| **12** | 6,76E-04 | 2,54E-04 | 7,61E-05 | 1,63E-05 | 1,88E-06 |  |  |  |  |  |  |  |  |  |  |
| **13** | 1,04E-04 | 2,93E-05 | 5,85E-06 | 6,27E-07 |  |  |  |  |  |  |  |  |  |  |  |
| **14** | 1,12E-05 | 2,09E-06 | 2,09E-07 |  |  |  |  |  |  |  |  |  |  |  |  |
| **15** | 7,43E-07 | 6,97E-08 |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **16** | 2,32E-08 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Nous disposons de plusieurs outils pour analyser les résultats issus des calculs. La méthode *.describe( )* appliquée à un objet de type *DataFrame* permet notamment de retourner un tableau fournissant pour chaque colonne les valeurs suivantes : nombre de variables, la moyenne, l’écart-type, la valeur minimale, le premier quartile, le deuxième quartile (médiane), le troisième quartile et enfin la valeur maximale. Cette méthode offre donc une première vue d’ensemble rapide des sets de valeurs de chaque colonne. Un exemple de cette méthode est présenté ci-dessous :

>>> df\_proba.loc[ : , ‘n = 30’:’n = 26’].describe()

n = 30 n = 29 n = 28 n = 27 n = 26

count 3.100000e+01 3.000000e+01 2.900000e+01 2.800000e+01 2.700000e+01

mean 3.225806e-02 3.333322e-02 3.448240e-02 3.571397e-02 3.703704e-02

std 5.054976e-02 5.167319e-02 5.285838e-02 5.411068e-02 5.543625e-02

min 4.856936e-15 1.457081e-14 4.371242e-14 1.311373e-13 3.934118e-13

25% 3.240364e-06 4.138876e-06 6.624923e-06 1.180065e-05 1.609912e-05

50% 1.720697e-03 2.064836e-03 1.922434e-03 2.523029e-03 2.145115e-03

75% 4.733645e-02 5.280348e-02 5.363097e-02 5.197799e-02 5.826083e-02

max 1.530152e-01 1.530152e-01 1.582916e-01 1.611183e-01 1.611183e-01

Dans le cas présent, nous ne jugeons pas nécessaire de rassembler toutes ces informations. Nous recherchons dans un premier temps les valeurs de probabilité minimale et maximale pour l’ensemble du tableau. Pour cela, l’objet *DataFrame* dispose des attributs *.min( )* et *.max( )*, qui retournent chacun un objet de type *Series* contenant respectivement les valeurs minimales et maximales pour chaque colonne. Un exemple de l’utilisation de ces attributs est présenté ci-dessous :

>>> df\_proba.min()

n = 30 4.856936e-15 # valeur minimale pour la colonne ‘30’

n = 29 1.457081e-14 # valeur minimale pour la colonne ‘29’

...

n = 3 3.703704e-02 # valeur minimale pour la colonne ‘3’

n = 2 1.111111e-01 # valeur minimale pour la colonne ‘2’

>>> df\_proba.max()

n = 30 0.153015 # valeur maximale pour la colonne ‘30’

n = 29 0.153015 # valeur maximale pour la colonne ‘29’

...

n = 3 0.444444 # valeur maximale pour la colonne ‘3’

n = 2 0.444444 # valeur maximale pour la colonne ‘2’

Obtenir l’unique valeur minimale du *DataFrame* consiste simplement à obtenir la valeur minimale de la *Series* retournant les valeurs minimales pour chaque colonne. Nous y parvenons en chainant deux fois l’attribut *.min( )*. De la même manière, nous obtenons l’unique valeur maximale du *DataFrame* en chainant l’attribut .max( ). Un exemple est présenté ci-dessous :

>>> df\_proba.min().min()

4.856936e-15 # valeur minimale pour l’ensemble du DataFrame df\_proba

>>> df\_proba.max().max()

0.444444444444 # valeur maximale pour l’ensemble du DataFrame df\_proba

Ces valeurs sont celles dont la mise en forme diffère dans le **Tableau 1**. On peut donc y lire que la probabilité la plus faible du tableau est celle que 30 dés sur 30 en jeu possèdent la même valeur (attention : nous sommes dans le cas où les pacos peuvent prendre n’importe quelle valeur). C’est un résultat qui semble logique, nous comprenons intuitivement qu’il est hautement improbable que les 5 dés des 6 joueurs fassent tous apparaître, après un lancé aléatoire, des pacos ou des dés d’une unique valeur (e.g. : que des pacos et des 4).

La probabilité la plus forte du tableau est la même pour les deux cas de voir 0 ou 1 dé d’une même valeur (pacos compris) apparaître, lorsqu’il reste un total de deux dés en jeu. Nous pouvons l’expliquer par le fait que la somme de toutes les probabilités par colonne doit être 1. Autrement dit toutes les valeurs d’une colonne se partagent autant de fractions de valeurs entre 0 et 1 qu’il y a de dés en jeu. Lorsqu’il ne reste que deux dés en jeu, il ne reste que 3 issues possibles, soit trois fractions de valeurs. Nous pouvons donc penser que chaque fraction a donc une probabilité plus grande de survenir. Nous pouvons le résumer à un effet de dilution des probabilités parmi les intervalles entre 0 et 1. Les plus fortes probabilités se retrouvent pour les cas les moins dilués, donc ayant le moins de dés en jeu.

Nous nous intéressons maintenant à la distribution des résultats de probabilité parmi les intervalles dans une même colonne. Nous commençons par étudier la colonne ‘n = 30’ car elle dispose du plus grand nombre d’intervalles, et propose donc une distribution plus descriptive. La

35 df\_proba[**'n = 30'**].plot(legend**=True**)  
36 plt.ylabel(**'Probability'**)  
37 plt.title(**'Probability distribution to get x dice among n'**)  
38 plt.savefig(**'/Users/mattiou/Desktop/df\_proba\_n30.pdf'**)  
39 plt.show()

***L35 -*** Exécution de l’attribut de tracé *.plot( )* sur la colonne *‘n = 30’* de l’objet *df\_proba* avec affichage du nom de la colonne pour légende.

***L36 -*** Attribution du nom de l’axe y.

***L37 -*** Attribution du titre du graphique.

***L38 -*** Exportation du graphique obtenu sous forme d’une image au format .pdf.

***L39 -*** Affichage du graphique à l’exécution du code.

La courbe décrivant la distribution des probabilités pour chaque pari possible (x = 0 à 30) pour la colonne n = 30 est présentée par la **Figure 1**.



**Figure 1 - Fonction de distribution des probabilités pour n = 30.**

### Cas 2 : p = 1/6

**Tableau 2 - Probabilité que x dés (lignes) apparaissent parmi un total de n dés (colonnes) pour p = 1/6.**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **x DÉS** | **NOMBRE TOTAL DE DÉS EN JEU POUR p = 1/6** | | | | | | | | | | | | | |
| **n = 30** | **n = 29** | **n = 28** | **n = 27** | **n = 26** | **n = 25** | **n = 24** | **n = 23** | **n = 22** | **n = 21** | **n = 20** | **n = 19** | **n = 18** | **n = 17** |
| **0** | 4,21E-03 | 5,06E-03 | 6,07E-03 | 7,28E-03 | 8,74E-03 | 1,05E-02 | 1,26E-02 | 1,51E-02 | 1,81E-02 | 2,17E-02 | 2,61E-02 | 3,13E-02 | 3,76E-02 | 4,51E-02 |
| **1** | 2,53E-02 | 2,93E-02 | 3,28E-02 | 3,79E-02 | 4,54E-02 | 5,03E-02 | 5,79E-02 | 6,94E-02 | 7,97E-02 | 9,13E-02 | 1,04E-01 | 1,19E-01 | 1,35E-01 | 1,53E-01 |
| **2** | 7,33E-02 | 8,19E-02 | 9,15E-02 | 1,02E-01 | 1,14E-01 | 1,25E-01 | 1,39E-01 | 1,53E-01 | 1,67E-01 | 1,83E-01 | 1,98E-01 | 2,14E-01 | 2,30E-01 | 2,45E-01 |
| **3** | 1,37E-01 | 1,48E-01 | 1,59E-01 | 1,70E-01 | 1,82E-01 | 1,93E-01 | 2,04E-01 | 2,14E-01 | 2,23E-01 | 2,31E-01 | 2,38E-01 | 2,43E-01 | 2,45E-01 | 2,45E-01 |
| **4** | 1,85E-01 | 1,92E-01 | 1,99E-01 | 2,04E-01 | 2,09E-01 | 2,12E-01 | 2,14E-01 | 2,14E-01 | 2,12E-01 | 2,08E-01 | 2,02E-01 | 1,94E-01 | 1,84E-01 | 1,72E-01 |
| **5** | 1,92E-01 | 1,92E-01 | 1,91E-01 | 1,88E-01 | 1,84E-01 | 1,78E-01 | 1,71E-01 | 1,63E-01 | 1,53E-01 | 1,42E-01 | 1,29E-01 | 1,16E-01 | 1,03E-01 | 8,93E-02 |
| **6** | 1,60E-01 | 1,54E-01 | 1,46E-01 | 1,38E-01 | 1,29E-01 | 1,19E-01 | 1,08E-01 | 9,75E-02 | 8,65E-02 | 7,55E-02 | 6,47E-02 | 5,44E-02 | 4,46E-02 | 3,57E-02 |
| **7** | 1,10E-01 | 1,01E-01 | 9,19E-02 | 8,27E-02 | 7,36E-02 | 6,45E-02 | 5,57E-02 | 4,74E-02 | 3,95E-02 | 3,24E-02 | 2,59E-02 | 2,02E-02 | 1,53E-02 | 1,12E-02 |
| **8** | 6,31E-02 | 5,55E-02 | 4,83E-02 | 4,14E-02 | 3,49E-02 | 2,90E-02 | 2,37E-02 | 1,89E-02 | 1,48E-02 | 1,13E-02 | 8,41E-03 | 6,06E-03 | 4,21E-03 | 2,81E-03 |
| **9** | 3,09E-02 | 2,59E-02 | 2,15E-02 | 1,75E-02 | 1,40E-02 | 1,10E-02 | 8,42E-03 | 6,32E-03 | 4,61E-03 | 3,27E-03 | 2,24E-03 | 1,48E-03 | 9,35E-04 | 5,61E-04 |
| **10** | 1,30E-02 | 1,04E-02 | 8,15E-03 | 6,29E-03 | 4,75E-03 | 3,51E-03 | 2,53E-03 | 1,77E-03 | 1,20E-03 | 7,85E-04 | 4,93E-04 | 2,96E-04 | 1,68E-04 | 8,98E-05 |
| **11** | 4,71E-03 | 3,58E-03 | 2,67E-03 | 1,94E-03 | 1,38E-03 | 9,57E-04 | 6,43E-04 | 4,18E-04 | 2,62E-04 | 1,57E-04 | 8,97E-05 | 4,85E-05 | 2,45E-05 | 1,14E-05 |
| **12** | 1,49E-03 | 1,07E-03 | 7,56E-04 | 5,18E-04 | 3,46E-04 | 2,23E-04 | 1,39E-04 | 8,36E-05 | 4,80E-05 | 2,62E-05 | 1,35E-05 | 6,46E-06 | 2,86E-06 | 1,14E-06 |
| **13** | 4,13E-04 | 2,81E-04 | 1,86E-04 | 1,20E-04 | 7,44E-05 | 4,47E-05 | 2,57E-05 | 1,41E-05 | 7,38E-06 | 3,62E-06 | 1,66E-06 | 6,96E-07 | 2,64E-07 | 8,79E-08 |
| **14** | 1,00E-04 | 6,42E-05 | 3,99E-05 | 2,39E-05 | 1,38E-05 | 7,66E-06 | 4,04E-06 | 2,02E-06 | 9,49E-07 | 4,14E-07 | 1,66E-07 | 5,96E-08 | 1,88E-08 | 5,02E-09 |
| **15** | 2,14E-05 | 1,28E-05 | 7,44E-06 | 4,15E-06 | 2,21E-06 | 1,12E-06 | 5,39E-07 | 2,43E-07 | 1,01E-07 | 3,87E-08 | 1,33E-08 | 3,98E-09 | 1,00E-09 | 2,01E-10 |
| **16** | 4,01E-06 | 2,25E-06 | 1,21E-06 | 6,22E-07 | 3,04E-07 | 1,40E-07 | 6,06E-08 | 2,43E-08 | 8,86E-09 | 2,90E-09 | 8,28E-10 | 1,99E-10 | 3,77E-11 | 5,02E-12 |
| **17** | 6,61E-07 | 3,44E-07 | 1,71E-07 | 8,05E-08 | 3,58E-08 | 1,49E-08 | 5,71E-09 | 2,00E-09 | 6,25E-10 | 1,71E-10 | 3,90E-11 | 7,02E-12 | 8,86E-13 | 5,91E-14 |
| **18** | 9,55E-08 | 4,58E-08 | 2,09E-08 | 8,94E-09 | 3,58E-09 | 1,32E-09 | 4,44E-10 | 1,33E-10 | 3,47E-11 | 7,58E-12 | 1,30E-12 | 1,56E-13 | 9,85E-15 |  |
| **19** | 1,21E-08 | 5,31E-09 | 2,20E-09 | 8,47E-10 | 3,01E-10 | 9,73E-11 | 2,80E-11 | 7,01E-12 | 1,46E-12 | 2,39E-13 | 2,74E-14 | 1,64E-15 |  |  |
| **20** | 1,33E-09 | 5,31E-10 | 1,98E-10 | 6,78E-11 | 2,11E-11 | 5,84E-12 | 1,40E-12 | 2,80E-13 | 4,39E-14 | 4,79E-15 | 2,74E-16 |  |  |  |
| **21** | 1,26E-10 | 4,55E-11 | 1,51E-11 | 4,52E-12 | 1,21E-12 | 2,78E-13 | 5,34E-14 | 8,01E-15 | 8,36E-16 | 4,56E-17 |  |  |  |  |
| **22** | 1,03E-11 | 3,31E-12 | 9,59E-13 | 2,46E-13 | 5,48E-14 | 1,01E-14 | 1,46E-15 | 1,46E-16 | 7,60E-18 |  |  |  |  |  |
| **23** | 7,19E-13 | 2,01E-13 | 5,00E-14 | 1,07E-14 | 1,91E-15 | 2,63E-16 | 2,43E-17 | 1,27E-18 |  |  |  |  |  |  |
| **24** | 4,20E-14 | 1,01E-14 | 2,08E-15 | 3,57E-16 | 4,76E-17 | 4,22E-18 | 2,11E-19 |  |  |  |  |  |  |  |
| **25** | 2,01E-15 | 4,03E-16 | 6,67E-17 | 8,57E-18 | 7,62E-19 | 3,52E-20 |  |  |  |  |  |  |  |  |
| **26** | 7,75E-17 | 1,24E-17 | 1,53E-18 | 1,27E-19 | 5,86E-21 |  |  |  |  |  |  |  |  |  |
| **27** | 2,30E-18 | 2,75E-19 | 2,20E-20 | 9,77E-22 |  |  |  |  |  |  |  |  |  |  |
| **28** | 8,45E-12 | 8,45E-13 | 4,37E-14 |  |  |  |  |  |  |  |  |  |  |  |
| **29** | 2,91E-13 | 1,46E-14 |  |  |  |  |  |  |  |  |  |  |  |  |
| **30** | 4,86E-15 |  |  |  |  |  |  |  |  |  |  |  |  |  |

**Tableau 2 (suite) - Probabilité que x dés (lignes) apparaissent parmi un total de n dés (colonnes) pour p = 1/6.**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **x DÉS** | **NOMBRE TOTAL DE DÉS EN JEU POUR p = 1/6** | | | | | | | | | | | | | | |
| **n = 16** | **n = 15** | **n = 14** | **n = 13** | **n = 12** | **n = 11** | **n = 10** | **n = 9** | **n = 8** | **n = 7** | **n = 6** | **n = 5** | **n = 4** | **n = 3** | **n = 2** |
| **0** | 5,41E-02 | 6,49E-02 | 7,79E-02 | 9,35E-02 | 1,12E-01 | 1,35E-01 | 1,62E-01 | 1,94E-01 | 2,33E-01 | 2,79E-01 | 3,35E-01 | 4,02E-01 | 4,82E-01 | 5,79E-01 | 6,94E-01 |
| **1** | 1,73E-01 | 1,95E-01 | 2,18E-01 | 2,43E-01 | 2,69E-01 | 2,96E-01 | 3,23E-01 | 3,49E-01 | 3,72E-01 | 3,91E-01 | 4,02E-01 | 4,02E-01 | 3,86E-01 | 3,47E-01 | 2,78E-01 |
| **2** | 2,60E-01 | 2,73E-01 | 2,84E-01 | 2,92E-01 | 2,96E-01 | 2,96E-01 | 2,91E-01 | 2,79E-01 | 2,60E-01 | 2,34E-01 | 2,01E-01 | 1,61E-01 | 1,16E-01 | 6,94E-02 | 2,78E-02 |
| **3** | 2,42E-01 | 2,36E-01 | 2,27E-01 | 2,14E-01 | 1,97E-01 | 1,78E-01 | 1,55E-01 | 1,30E-01 | 1,04E-01 | 7,81E-02 | 5,36E-02 | 3,22E-02 | 1,54E-02 | 4,63E-03 |  |
| **4** | 1,58E-01 | 1,42E-01 | 1,25E-01 | 1,07E-01 | 8,88E-02 | 7,11E-02 | 5,43E-02 | 3,91E-02 | 2,60E-02 | 1,56E-02 | 8,04E-03 | 3,22E-03 | 7,72E-04 |  |  |
| **5** | 7,56E-02 | 6,24E-02 | 4,99E-02 | 3,85E-02 | 2,84E-02 | 1,99E-02 | 1,30E-02 | 7,81E-03 | 4,17E-03 | 1,88E-03 | 6,43E-04 | 1,29E-04 |  |  |  |
| **6** | 2,77E-02 | 2,08E-02 | 1,50E-02 | 1,03E-02 | 6,63E-03 | 3,98E-03 | 2,17E-03 | 1,04E-03 | 4,17E-04 | 1,25E-04 | 2,14E-05 |  |  |  |  |
| **7** | 7,92E-03 | 5,35E-03 | 3,42E-03 | 2,05E-03 | 1,14E-03 | 5,68E-04 | 2,48E-04 | 8,93E-05 | 2,38E-05 | 3,57E-06 |  |  |  |  |  |
| **8** | 1,78E-03 | 1,07E-03 | 5,99E-04 | 3,08E-04 | 1,42E-04 | 5,68E-05 | 1,86E-05 | 4,47E-06 | 5,95E-07 |  |  |  |  |  |  |
| **9** | 3,17E-04 | 1,66E-04 | 7,98E-05 | 3,42E-05 | 1,26E-05 | 3,79E-06 | 8,27E-07 | 9,92E-08 |  |  |  |  |  |  |  |
| **10** | 4,44E-05 | 2,00E-05 | 7,98E-06 | 2,74E-06 | 7,58E-07 | 1,52E-07 | 1,65E-08 |  |  |  |  |  |  |  |  |
| **11** | 4,84E-06 | 1,81E-06 | 5,81E-07 | 1,49E-07 | 2,76E-08 | 2,76E-09 |  |  |  |  |  |  |  |  |  |
| **12** | 4,03E-07 | 1,21E-07 | 2,90E-08 | 4,98E-09 | 4,59E-10 |  |  |  |  |  |  |  |  |  |  |
| **13** | 2,48E-08 | 5,58E-09 | 8,93E-10 | 7,66E-11 |  |  |  |  |  |  |  |  |  |  |  |
| **14** | 1,06E-09 | 1,60E-10 | 1,28E-11 |  |  |  |  |  |  |  |  |  |  |  |  |
| **15** | 2,84E-11 | 2,13E-12 |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **16** | 3,54E-13 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

## Calculs de probabilités cumulées

Pour p = 1/3, les résultats des calculs de probabilités cumulées tels que présentés en II.2.2 sont regroupés dans le **Tableau 3** ci-dessous :

**Tableau 3 - Probabilités en pourcentage qu'AU MOINS x dés (lignes) apparaissent parmi un total de n dés (colonnes) pour p = 1/3.**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **x DÉS** | **NOMBRE TOTAL DE DÉS EN JEU POUR p = 1/3** | | | | | | | | | | | | | |
| **n = 30** | **n = 29** | **n = 28** | **n = 27** | **n = 26** | **n = 25** | **n = 24** | **n = 23** | **n = 22** | **n = 21** | **n = 20** | **n = 19** | **n = 18** | **n = 17** |
| **0** | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% |
| **1** | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% |
| **2** | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 99% | 99% |
| **3** | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 99% | 99% | 99% | 98% | 98% | 97% | 96% |
| **4** | 100% | 100% | 99% | 99% | 99% | 99% | 98% | 97% | 96% | 95% | 94% | 92% | 90% | 87% |
| **5** | 99% | 98% | 98% | 97% | 96% | 95% | 94% | 92% | 90% | 88% | 85% | 81% | 77% | 72% |
| **6** | 96% | 95% | 94% | 93% | 91% | 89% | 86% | 83% | 79% | 75% | 70% | 65% | 59% | 52% |
| **7** | 92% | 90% | 87% | 85% | 81% | 78% | 74% | 69% | 64% | 58% | 52% | 46% | 39% | 33% |
| **8** | 83% | 80% | 77% | 72% | 68% | 63% | 58% | 52% | 46% | 40% | 34% | 28% | 22% | 17% |
| **9** | 71% | 67% | 62% | 57% | 52% | 46% | 41% | 35% | 29% | 24% | 19% | 15% | 11% | 8% |
| **10** | 57% | 52% | 46% | 41% | 36% | 30% | 25% | 21% | 16% | 12% | 9% | 6% | 4% | 3% |
| **11** | 42% | 36% | 31% | 27% | 22% | 18% | 14% | 11% | 8% | 6% | 4% | 2% | 1% | 1% |
| **12** | 28% | 23% | 19% | 15% | 12% | 9% | 7% | 5% | 3% | 2% | 1% | 1% | 0% | 0% |
| **13** | 17% | 13% | 10% | 8% | 6% | 4% | 3% | 2% | 1% | 1% | 0% | 0% | 0% | 0% |
| **14** | 9% | 7% | 5% | 4% | 2% | 2% | 1% | 1% | 0% | 0% | 0% | 0% | 0% | 0% |
| **15** | 4% | 3% | 2% | 1% | 1% | 1% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| **16** | 2% | 1% | 1% | 1% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| **17** | 1% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| **18** | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |  |
| **19** | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |  |  |
| **20** | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |  |  |  |
| **21** | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |  |  |  |  |
| **22** | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |  |  |  |  |  |
| **23** | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |  |  |  |  |  |  |
| **24** | 0% | 0% | 0% | 0% | 0% | 0% | 0% |  |  |  |  |  |  |  |
| **25** | 0% | 0% | 0% | 0% | 0% | 0% |  |  |  |  |  |  |  |  |
| **26** | 0% | 0% | 0% | 0% | 0% |  |  |  |  |  |  |  |  |  |
| **27** | 0% | 0% | 0% | 0% |  |  |  |  |  |  |  |  |  |  |
| **28** | 0% | 0% | 0% |  |  |  |  |  |  |  |  |  |  |  |
| **29** | 0% | 0% |  |  |  |  |  |  |  |  |  |  |  |  |
| **30** | 0% |  |  |  |  |  |  |  |  |  |  |  |  |  |

**Tableau 3 (suite) - Probabilités en pourcentage qu'AU MOINS *x* dés (lignes) apparaissent parmi un total de *n* dés (colonnes) pour p = 1/3.**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **x DÉS** | **NOMBRE TOTAL DE DÉS EN JEU POUR p = 1/3** | | | | | | | | | | | | | | |
| **n = 16** | **n = 15** | **n = 14** | **n = 13** | **n = 12** | **n = 11** | **n = 10** | **n = 9** | **n = 8** | **n = 7** | **n = 6** | **n = 5** | **n = 4** | **n = 3** | **n = 2** |
| **0** | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% |
| **1** | 100% | 100% | 100% | 99% | 99% | 99% | 98% | 97% | 96% | 94% | 91% | 87% | 80% | 70% | 56% |
| **2** | 99% | 98% | 97% | 96% | 95% | 92% | 90% | 86% | 80% | 74% | 65% | 54% | 41% | 26% | 11% |
| **3** | 94% | 92% | 89% | 86% | 82% | 77% | 70% | 62% | 53% | 43% | 32% | 21% | 11% | 4% |  |
| **4** | 83% | 79% | 74% | 68% | 61% | 53% | 44% | 35% | 26% | 17% | 10% | 5% | 1% |  |  |
| **5** | 66% | 60% | 52% | 45% | 37% | 29% | 21% | 14% | 9% | 5% | 2% | 0% |  |  |  |
| **6** | 45% | 38% | 31% | 24% | 18% | 12% | 8% | 4% | 2% | 1% | 0% |  |  |  |  |
| **7** | 26% | 20% | 15% | 10% | 7% | 4% | 2% | 1% | 0% | 0% |  |  |  |  |  |
| **8** | 13% | 9% | 6% | 3% | 2% | 1% | 0% | 0% | 0% |  |  |  |  |  |  |
| **9** | 5% | 3% | 2% | 1% | 0% | 0% | 0% | 0% |  |  |  |  |  |  |  |
| **10** | 2% | 1% | 0% | 0% | 0% | 0% | 0% |  |  |  |  |  |  |  |  |
| **11** | 0% | 0% | 0% | 0% | 0% | 0% |  |  |  |  |  |  |  |  |  |
| **12** | 0% | 0% | 0% | 0% | 0% |  |  |  |  |  |  |  |  |  |  |
| **13** | 0% | 0% | 0% | 0% |  |  |  |  |  |  |  |  |  |  |  |
| **14** | 0% | 0% | 0% |  |  |  |  |  |  |  |  |  |  |  |  |
| **15** | 0% | 0% |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **16** | 0% |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Pour p = 1/6, les résultats des calculs de probabilités cumulées tels que présentés en II.2.2 sont regroupés dans le **Tableau 4** ci-dessous :

**Tableau 4 - Probabilités en pourcentage qu'AU MOINS x dés (lignes) apparaissent parmi un total de n dés (colonnes) pour p = 1/6.**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **x DÉS** | **NOMBRE TOTAL DE DÉS EN JEU POUR p = 1/3** | | | | | | | | | | | | | |
| **n = 30** | **n = 29** | **n = 28** | **n = 27** | **n = 26** | **n = 25** | **n = 24** | **n = 23** | **n = 22** | **n = 21** | **n = 20** | **n = 19** | **n = 18** | **n = 17** |
| **0** | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% |
| **1** | 100% | 99% | 99% | 99% | 99% | 99% | 98% | 98% | 98% | 98% | 97% | 97% | 96% | 95% |
| **2** | 97% | 97% | 96% | 95% | 95% | 94% | 93% | 92% | 90% | 89% | 87% | 85% | 83% | 80% |
| **3** | 90% | 88% | 87% | 85% | 83% | 81% | 79% | 76% | 73% | 70% | 67% | 64% | 60% | 56% |
| **4** | 76% | 74% | 71% | 68% | 65% | 62% | 58% | 55% | 51% | 47% | 43% | 39% | 35% | 31% |
| **5** | 58% | 54% | 51% | 48% | 44% | 41% | 37% | 33% | 30% | 26% | 23% | 20% | 17% | 14% |
| **6** | 38% | 35% | 32% | 29% | 26% | 23% | 20% | 17% | 15% | 12% | 10% | 8% | 7% | 5% |
| **7** | 22% | 20% | 17% | 15% | 13% | 11% | 9% | 7% | 6% | 5% | 4% | 3% | 2% | 1% |
| **8** | 11% | 10% | 8% | 7% | 6% | 4% | 4% | 3% | 2% | 2% | 1% | 1% | 1% | 0% |
| **9** | 5% | 4% | 3% | 3% | 2% | 2% | 1% | 1% | 1% | 0% | 0% | 0% | 0% | 0% |
| **10** | 2% | 2% | 1% | 1% | 1% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| **11** | 1% | 1% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| **12** | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| **13** | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| **14** | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| **15** | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| **16** | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| **17** | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| **18** | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |  |
| **19** | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |  |  |
| **20** | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |  |  |  |
| **21** | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |  |  |  |  |
| **22** | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |  |  |  |  |  |
| **23** | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |  |  |  |  |  |  |
| **24** | 0% | 0% | 0% | 0% | 0% | 0% | 0% |  |  |  |  |  |  |  |
| **25** | 0% | 0% | 0% | 0% | 0% | 0% |  |  |  |  |  |  |  |  |
| **26** | 0% | 0% | 0% | 0% | 0% |  |  |  |  |  |  |  |  |  |
| **27** | 0% | 0% | 0% | 0% |  |  |  |  |  |  |  |  |  |  |
| **28** | 0% | 0% | 0% |  |  |  |  |  |  |  |  |  |  |  |
| **29** | 0% | 0% |  |  |  |  |  |  |  |  |  |  |  |  |
| **30** | 0% |  |  |  |  |  |  |  |  |  |  |  |  |  |

**Tableau 4 (suite) - Probabilités en pourcentage qu'AU MOINS *x* dés (lignes) apparaissent parmi un total de *n* dés (colonnes) pour p = 1/6.**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **x DÉS** | **NOMBRE TOTAL DE DÉS EN JEU POUR p = 1/3** | | | | | | | | | | | | | | |
| **n = 16** | **n = 15** | **n = 14** | **n = 13** | **n = 12** | **n = 11** | **n = 10** | **n = 9** | **n = 8** | **n = 7** | **n = 6** | **n = 5** | **n = 4** | **n = 3** | **n = 2** |
| **0** | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% |
| **1** | 100% | 100% | 100% | 99% | 99% | 99% | 98% | 97% | 96% | 94% | 91% | 87% | 80% | 70% | 56% |
| **2** | 99% | 98% | 97% | 96% | 95% | 92% | 90% | 86% | 80% | 74% | 65% | 54% | 41% | 26% | 11% |
| **3** | 94% | 92% | 89% | 86% | 82% | 77% | 70% | 62% | 53% | 43% | 32% | 21% | 11% | 4% |  |
| **4** | 83% | 79% | 74% | 68% | 61% | 53% | 44% | 35% | 26% | 17% | 10% | 5% | 1% |  |  |
| **5** | 66% | 60% | 52% | 45% | 37% | 29% | 21% | 14% | 9% | 5% | 2% | 0% |  |  |  |
| **6** | 45% | 38% | 31% | 24% | 18% | 12% | 8% | 4% | 2% | 1% | 0% |  |  |  |  |
| **7** | 26% | 20% | 15% | 10% | 7% | 4% | 2% | 1% | 0% | 0% |  |  |  |  |  |
| **8** | 13% | 9% | 6% | 3% | 2% | 1% | 0% | 0% | 0% |  |  |  |  |  |  |
| **9** | 5% | 3% | 2% | 1% | 0% | 0% | 0% | 0% |  |  |  |  |  |  |  |
| **10** | 2% | 1% | 0% | 0% | 0% | 0% | 0% |  |  |  |  |  |  |  |  |
| **11** | 0% | 0% | 0% | 0% | 0% | 0% |  |  |  |  |  |  |  |  |  |
| **12** | 0% | 0% | 0% | 0% | 0% |  |  |  |  |  |  |  |  |  |  |
| **13** | 0% | 0% | 0% | 0% |  |  |  |  |  |  |  |  |  |  |  |
| **14** | 0% | 0% | 0% |  |  |  |  |  |  |  |  |  |  |  |  |
| **15** | 0% | 0% |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **16** | 0% |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Cette première forme de la fonction semble pertinente et ne présente pas de faille mathématiques. Cependant son utilisation trouve des limites inhérentes au langage Python lui-même ; en effet en utilisant la fonction *factorial( )* sur des nombres élevés, le message d’erreur suivant est retourné :

*RuntimeWarning: overflow encountered in long\_scalars*

Concrètement, le produit d’un nombre élevé par les autres nombres de sa suite crée un nombre intermédiaire qui devient très grand et dépasse alors la limite de stockage assignée à la variable *num* qui porte ce calcul. L’explication se trouve dans la définition de la variable. En effet en assignant le chiffre 1 sous la forme d’un entier, la variable *num* adopte de fait le type entier, qui se trouve être, sans autre spécification, limité à 32 bits de mémoire par défaut. Il est donc nécessaire, pour s’accorder plus de flexibilité de calculs de définir un type de donnée pour la variable *num* qui lui adjoint plus d’espace de stockage. Or il se trouve que les nombres décimaux ont une limite de mémoire de 64 bits par défaut. Une simple modification de la fonction permet donc d’éliminer ce problème et la fonction devient :

**def factorial**(*x*)**:**~~num~~ **~~=~~** ~~1~~ num **=** 1. *# Initialisation de la variable num à 1.*  
 **if** *x* **==** 0**:** *# Pour x = 0, la fonction doit retourner 1 (car 0! = 1).* **return** num  
 **else:  
 while** *x* **>** 0**:** *# Pour x > 0, num prend la valeur de x, chaque itération* num **=** num **\*** *x diminue x de 1 et multiplie num par x en boucle, la  
 x* **-=** 1 *boucle continue jusque x = 1.* **return** int(num) *# La fonction retourne num.*

Le simple fait d’ajouter un ‘.’ derrière la valeur assignée à la variable *num* convertit son type d’entier à décimal, et donc étend le stockage associé de 32 à 64 bits. Lorsque la fonction retourne le résultat du produit factoriel, il est alors possible de convertir de nouveau la variable *num* en entier, Python allouera alors automatiquement à la variable un espace de stockage suffisant. Cette simple modification élimine le message d’erreur et assure la justesse des calculs.