Exercice 1 Combien y a-t-il de p-uplets strictement croissants de $[\![1,n]\!]$?

Exercice 2 Dans une urne on place $4n$ boules blanches et n boules noires puis on tire au hasard 4 boules. Déterminer :

$\boldsymbol{c_1} =$ le nombre de tirages ne contenant que des boules noires.

$\boldsymbol{c_2} =$ le nombre de tirages avec exactement 2 boules blanches.

dans les trois cas suivants :

1. Les tirages sont simultanés.
2. Les tirages sont successifs avec remise.
3. Les tirages sont successifs sans remise.

Pour chacun des deux derniers protocoles on proposera également deux simulations informatiques de l'expérience. La première simulation utilisera une urne virtuelle alors que la deuxième n'utilisera que des compteurs.

Exercice 3 On doit ranger n paires de chaussettes dans n tiroirs.

1. Calculer le nombre total de rangements.
2. Calculer le nombre de rangements avec une paire dans chaque tiroir.
3. Calculer le nombre de rangements avec exactement un tiroir vide.
4. Simuler un rangement aléatoire sous la forme d'une n-liste de nombres de chaussettes par tiroir.

Exercice 4

1. Trois enfants décident de se répartir 20 bonbons distincts. Ils décident que chacun d'eux en recevra au moins un. Combien y a-t-il de répartitions possibles ?
2. (a) Simuler la distribution aléatoire de 20 bonbons à 3 enfants sous la forme d'une fonction python *distri* qui renvoie trois listes d'entiers entre 1 et 20 telles que la $i^{\text{ème}}$ liste soit constituée des numéros des bonbons attribués au $i^{\text{ème}}$ enfant.
   (b) En déduire une fonction *distriSurj* qui simule la distribution aléatoire de 20 bonbons à 3 enfants telle que chacun d'eux en reçoive au moins un.
3. On note B l'ensemble des 20 bonbons. On appelle partition de B en trois sous-ensembles la donnée de trois parties de B : $\{B_1, B_2, B_3\}$ telles que les B_i soient non vides, deux à deux disjointes (pas d'élément en commun) et $B_1 \cup B_2 \cup B_3 = B$. Combien y a-t-il de partitions de B en trois sous-ensembles ?

Exercice 5 Montrer que dans un groupe de n personnes il existe au moins deux personnes ayant le même nombre d'amis. On supposera qu'une personne ne peut pas être amie avec elle-même et que l'amitié est une relation symétrique (A ne peut pas être ami avec B sans que B le soit avec A).

Exercice 6 Soit n_1, n_2, p des entiers naturels. Déterminer de deux façons le nombre de tirages simultanés de p boules dans une urne contenant n_1 boules blanches et n_2 boules noires. En déduire la formule de Vandermonde $\binom{n_1+n_2}{p} = \sum\limits_{k=0}^{p} \binom{n_1}{k}\binom{n_2}{p-k}$.

Écrire une fonction python qui simule cette expérience en renvoyant le nombre de boules blanches tirées et le nombre de boules noires tirées.

Exercice 7 Dans une urne on place deux jetons numérotés 1, deux jetons numérotés 2,. . ., deux jetons numérotés n. On tire alors simultanément deux jetons. Déterminer le nombre de tirages possibles et le nombre de tirages avec deux numéros identiques. En déduire la probabilité que les deux numéros soient identiques.
Retrouver cette probabilité par simulation pour $n = 2$.

Exercice 8 (Paradoxe des anniversaires) Soit $n \in \mathbb{N}^*$, on considère un groupe de n personnes choisies au hasard parmi la population des gens qui ne sont pas nés un 29 février.
On admettra qu'un même nombre de personnes naissent chaque jour de l'année.
On assimilera une date de l'année (non bissextile) à un nombre de $[\![1, 365]\!]$.

1. Calculer, en fonction de n, la probabilité qu'au moins deux personnes de ce groupe aient la même date d'anniversaire. On commencera par expliquer pourquoi la modélisation par des n-uplets est préférable celle obtenue avec des n-combinaisons.
2. (a) Écrire une fonction python *anniv* de paramètre n qui renvoie une n-liste de $[\![1, 365]\!]$.
   (b) Coder une fonction *repetition* d'argument *lst* qui renvoie le booléen indiquant une répétition dans la liste *lst*.
   (c) Écrire une fonction *proba* qui prend en entrée n,q et qui renvoie une approximation de la probabilité qu'au moins deux personnes parmi n aient la même date d'anniversaire à l'aide de q simulations.
   (d) Écrire une fonction *diag* de paramètres n, q qui affiche le diagramme en barres des probabilités *proba(k,q)* pour k variant de 0 à n.
   (e) Écrire une fonction *seuil* de paramètres p, q qui renvoie le plus petit entier n tel que *proba(n,q)* soit supérieure ou égale à p.