DM n°8 : Probabilités

CORRECTION

On effectue une collection d'objets choisis au hasard parmi N objets et on s'intéresse au temps moyen d'attente pour obtenir la collection complète. Par exemple, on tire au hasard un chiffre (entre 0 et 9) et on s'arrête lorsqu'on a obtenu les 10 chiffres au moins une fois chacun.

I] Exemples

Déterminer le nombre de chiffres à examiner dans les décimales de ces trois nombres irrationnels :

π = 3,14159265358979323846264338327950288419716939937510582097494459230781640628620899862803482534211...

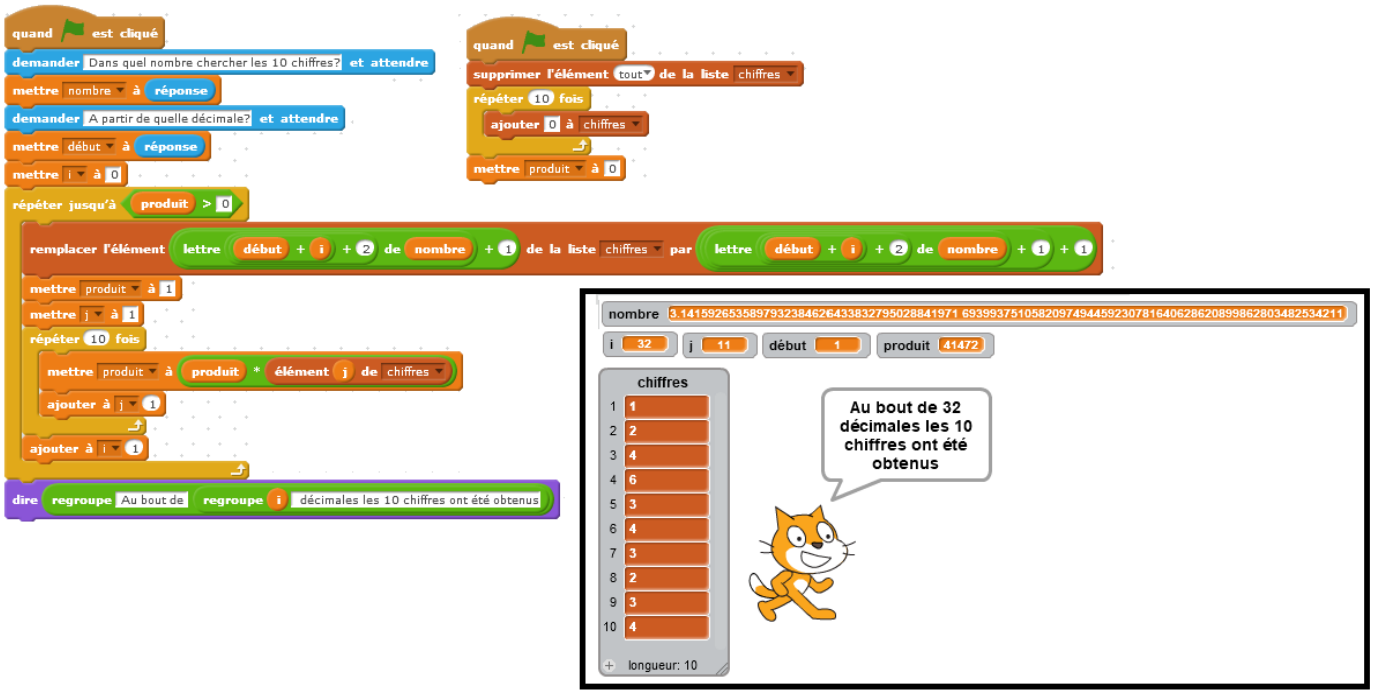
φ = 1,61803398874989484820458683436563811772030917980576286213544862270526046281890244970720720418939...

$\sqrt{2}$ =1,41421356237309504880168872420969807856967187537694807317667973799073247846210703885038753432 76...

Effectuer ces dénombrements en partant ① : du 1er chiffre après la virgule - ② : du 40ème chiffre après la virgule

>> Commenter/comparer ces résultats.

J'ai effectué les dénombrements demandés en utilisant le programme en Scratch (pour changer) ci-dessous. Les résultats sont consignés dans le tableau qui suit. On pouvait bien sûr se contenter de faire cela directement : j'ai surligné les décimales qui apportent un nouveau chiffre ① : en jaune à partir du 1er chiffre après la virgule - ② : en vert à partir du 40ème chiffre après la virgule (désolé, j'avais mis un espace après le 40ème chiffre : il fallait donc commencer le décompte juste avant cet espace). Il suffit alors de compter le nombre de chiffres qu'il y a entre le chiffre du début et le dernier chiffre surligné.



Nombres / début du compte	1	10	20	30	40	50
π	32	28	18	14	17	20
φ	22	25	23	29	19	27
$\sqrt{2}$	18	17	27	40	30	34

On voit sur ce tableau que les valeurs sont assez dispersées. Pour mieux montrer cela, j'ai effectué les dénombrements à partir du 10ème, 20ème, 30ème et 50ème chiffre, en plus de ce qui était demandé. Les valeurs s'étalent entre 14 et 40 (entre 18 et 32 avec les valeurs demandées). La moyenne de ces dix-huit valeurs est 24,44 (23 pour les six valeurs demandées). Entre les différents nombres irrationnels, il n'y a pas de différence notable. Ils ont l'air d'avoir une répartition des chiffres à peu prés aléatoire. On observe une accumulation du chiffre 4 dans les premières décimales de π (il y en a 6 sur les 32 premières décimales) et un manque de 0 (le 1er se fait attendre pendant 32 décimales). De même, on observe une surreprésentation du 8 et une sous-représentation du 5 dans les premières décimales de φ. Mais toutes ces irrégularités semblent s'estomper par la suite. Ces trois nombres sont de bons candidats pour la normalité (*simple normalité* : avoir 10% de chaque sorte de chiffre dans l'écriture décimale infinie en base dix), mais cela reste à prouver.

II] Simulations

a) Écrire un programme qui prend en argument N le nombre d'objets à collecter et qui réalise des tirages d'un nombre entier aléatoire entre 1 et N jusqu'à avoir obtenu tous les nombres (c'est-à-dire tous les objets). Le programme affiche alors le nombre de tirages T qu'il a fallu effectuer.

On veut réaliser ce programme en utilisant une liste. Les indications ci-dessous donne la syntaxe de quelques instructions en Python concernant les listes (pour la traduction dans un autre langage voir sur internet).

L=[] / *L*=10*[0]	*Initialise une liste vide nommée L / initialise une liste nommée L de 10 valeurs égales à 0*
if R in L / if R not in L	*Teste si la valeur contenue dans la variable R est/n'est pas dans la liste L*
L.append(R) / L[0]=*R*	*Place la valeur de R en dernière position dans la liste L / affecte la valeur de R en position 0 dans L*

>> Tester votre programme avec la valeur *N*=10 (la valeur obtenue est aléatoire mais elle doit être du même ordre de grandeur que les valeurs obtenues au I).

Voici notre programme et deux exécutions pour *N*=10.

```
from random import randint
N=int(input("Combien d'objets? "))
S=0 #nombre d'objets obtenu
T=0 #nombre de tirages
L=[]
while S<N :
    T+=1
    R=randint(1,N)
    if R not in L:
        L.append(R)
        S+=1
        print("Liste={} - {}/{} objets obtenus en {} tirages".format(L,S,N,T))
```

```
Combien d'objets? 10
Liste=[2] - 1/10 objets obtenus en 1 tirages
Liste=[2, 7] - 2/10 objets obtenus en 2 tirages
Liste=[2, 7, 5] - 3/10 objets obtenus en 3 tirages
Liste=[2, 7, 5, 9] - 4/10 objets obtenus en 4 tirages
Liste=[2, 7, 5, 9, 6] - 5/10 objets obtenus en 5 tirages
Liste=[2, 7, 5, 9, 6, 10] - 6/10 objets obtenus en 6 tirages
Liste=[2, 7, 5, 9, 6, 10, 4] - 7/10 objets obtenus en 9 tirages
Liste=[2, 7, 5, 9, 6, 10, 4, 8] - 8/10 objets obtenus en 11 tirages
Liste=[2, 7, 5, 9, 6, 10, 4, 8, 1] - 9/10 objets obtenus en 17 tirages
Liste=[2, 7, 5, 9, 6, 10, 4, 8, 1, 3] - 10/10 objets obtenus en 20 tirages

Combien d'objets? 10
Liste=[10] - 1/10 objets obtenus en 1 tirages
Liste=[10, 4] - 2/10 objets obtenus en 2 tirages
Liste=[10, 4, 1] - 3/10 objets obtenus en 3 tirages
Liste=[10, 4, 1, 3] - 4/10 objets obtenus en 4 tirages
Liste=[10, 4, 1, 3, 8] - 5/10 objets obtenus en 7 tirages
Liste=[10, 4, 1, 3, 8, 7] - 6/10 objets obtenus en 9 tirages
Liste=[10, 4, 1, 3, 8, 7, 9] - 7/10 objets obtenus en 10 tirages
Liste=[10, 4, 1, 3, 8, 7, 9, 5] - 8/10 objets obtenus en 22 tirages
Liste=[10, 4, 1, 3, 8, 7, 9, 5, 2] - 9/10 objets obtenus en 33 tirages
Liste=[10, 4, 1, 3, 8, 7, 9, 5, 2, 6] - 10/10 objets obtenus en 46 tirages
```

Notez la simplicité d'utilisation des listes. Il suffit de trois ou quatre instructions :

- *L*=[] pour initialiser une liste vide
- *L.append(R)* pour placer la valeur de *R* en dernière position dans la liste *L*
- *if R not in L* pour tester si la valeur *R* est dans *L*
- j'ai ajouté dans mon programme l'affichage de la liste que l'on peut simplement obtenir avec *print(L)*

Pour bien voir ce qui se passe, nous avons fait afficher les valeurs obtenues dans la liste. Il va de soi que la liste obtenue suit un ordre aléatoire (dans le 1er cas il s'agit de 2, 7, 5, 9, 6, 10, 4, 8, 1, 3 et dans le 2d cas l'ordre est différent 10, 4, 1, 3, 8, 7, 9, 5, 2, 6). Les différents objets ont été obtenus à des moments qui sont, eux aussi, aléatoires (dans le 1er cas, les moments sont 1, 2, 3, 4, 5, 6, 9, 11, 17, 20 et dans le 2d cas les moments sont différents 1, 2, 3, 4, 7, 9, 10, 22, 33, 46). Seul le premier tirage est certain de figurer en premier dans cette dernière liste car le 1er objet est toujours accepté (le 2d peut, lui, être un doublon du 1er).

On remarque que les moments d'apparition d'un nouvel objet sont de plus en plus espacés : au début on a un objet nouveau à chaque tirage, puis cela devient plus rare d'avoir un nouvel objet, le dernier se faisant souvent attendre (dans le 1er cas, 3 tirages seulement sont nécessaires après le 9ème objet pour obtenir le 10ème objet, mais dans le 2d cas il faut 13 tirages).

Notez encore qu'on peut faire autrement, au niveau de l'algorithme. On peut utiliser la fonction *len*() qui renvoie la longueur d'une liste (son nombre d'éléments) pour éviter d'avoir à compter les éléments : *while len(L)<n :*

On peut aussi initialiser la liste *L* avec l'instruction *L*=10*[0] et ensuite, au lieu de *L.append(R)* qui place *R* en dernière position sur la liste, on peut écrire *L*[*R*]=*L*[*R*]+1 ce qui s'écrit aussi *L*[*R*]+=1, ce qui conduit à enregistrer le nombre d'objets *R* obtenu. Attention ici car le rang du 1er objet est 0 en Python, donc si on écrit *L*[*R*] il faut que *R* aille de 0 à 9 au lieu de 1 à 10 (sinon on peut écrire *L*[*R*] avec *R* qui va de 1 à 10 mais il faut avoir initialisé *L* avec *L*=11*[0] et penser que *L*[0] vaut 0). Le programme qui suit tient compte de cette subtilité pour obtenir un résultat assez intéressant.

```
from random import randint
N=int(input("Combien d'objets? "))
T=0 #nombre de tirages
P=0 #produit des rangs obtenus pour les T objets
L=N*[0]
while P==0 :
    T+=1
    R=randint(0,N-1)
    L[R]+=1
    P=1
    for i in range(N) : P*=L[i]
print("Liste={} - {} objets obtenus en {} tirages".format(L,N,T))
```

```
Combien d'objets? 10
Liste=[3, 5, 1, 2, 1, 2, 5, 3, 1, 1] - 10 objets obtenus en 24 tirages
```

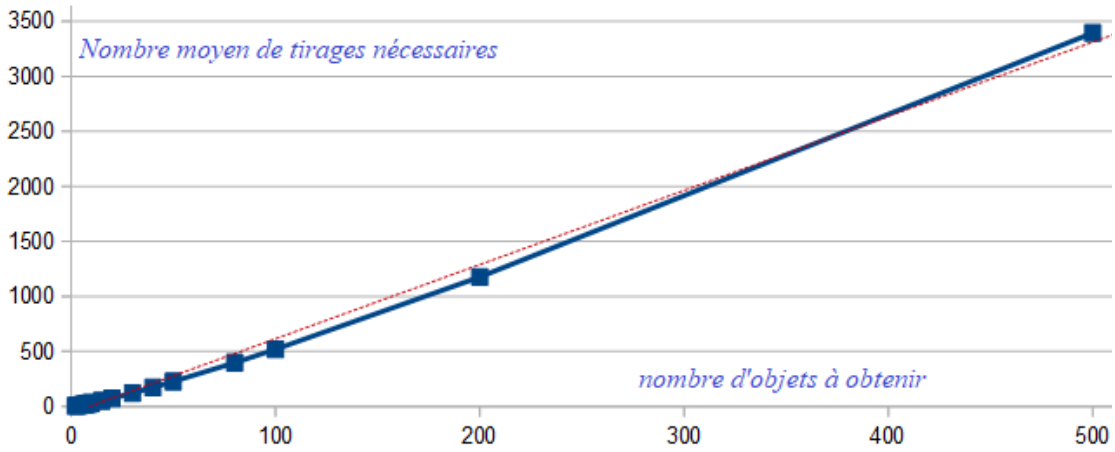
b) Écrire un nouveau programme qui relance M fois le programme précédent, de manière à constituer un échantillon de M collections pour lesquelles on détermine la valeur T. En effectuant la moyenne de ces M valeurs T obtenues, on obtient une valeur plus représentative de T. Le programme peut être une version améliorée du programme précédent, ou bien un programme indépendant qui appelle le programme précédent.
>> Donner la valeur moyenne obtenue pour T avec $M = 100$, lorsque N prend les valeurs 10, 20, 40 et 80.
Je reprends l'algorithme du début (avec *if R not in L*) en ajoutant la boucle qui permet de recommencer M fois le tirage. La variable X enregistre la somme du nombre de tirages (T) nécessaires pour obtenir les N objets. Pour obtenir la moyenne des T, il suffit de faire $X \div M$. Voici donc mon programme et une exécution pour M=100 et N=10.

```
from random import randint
M,N,X=100,10,0
for I in range(M):
    L,S,T=[],0,0
    while S<N :
        T+=1
        R=randint(1,N)
        if R not in L:
            L.append(R)
            S+=1
    X+=T
print("Nombre moyen de tirages de {} objets sur un échantillon de taille {} : {}".format(N,M,X/M))
```

```
Nombre moyen de tirages de 10 objets sur un échantillon de taille 100 : 28.89
```

Pour obtenir 10 objets, il faut donc en moyenne attendre environ 29 tirages. Je peux améliorer ce résultat en augmentant M. Pour M=10 000, je trouve T=29,3931. Je vais continuer à donner des résultats aussi précis (en gardant M=10000) mais arrondi au dixième. Le tableau ci-dessous donne les valeurs obtenues et le graphique en dessous représente ces données.

N	2	3	4	5	6	7	8	9	10	15	20	30	40	50	80	100	200	500
T	3	5,5	8,3	11,4	14,7	18	21,7	25,7	29,4	49,5	71,8	119,7	170,9	225,1	396,2	517,8	1176,2	3395,5



On observe que la croissance du nombre moyen T est presque linéaire. La droite tracée en pointillés rouges représente cette tendance à la linéarité. Il y a cependant un infléchissement certain : pour N=4, on a T qui vaut environ 2 fois N mais pour N=10, c'est déjà 3 fois N et pour N= 500, il faut presque 7 fois N. Cet infléchissement est lent et progressif mais inexorable.

III] L'approche probabiliste

a) Supposons qu'il y a N objets à obtenir et qu'on en a déjà obtenu N-1.
Quelle est la probabilité de tirer le dernier objet au coup suivant ?
Quelle est la probabilité de ne pas tirer le dernier objet au coup suivant ?
Quelle est la probabilité de tirer le dernier objet au $t^{ème}$ coup ?
Prendre N=10 et déterminer la moyenne théorique T du nombre t de coups à attendre pour obtenir le dernier objet en calculant la somme des produits $t \times P(t)$ pour t allant de 1 à 50 (faire cela avec un tableur ou un petit programme).
>>Vérifier que la valeur obtenue est très légèrement inférieure à N, la valeur théorique attendue dans cette situation.
Lorsque l'on a N-1 objets, la probabilité de tirer le dernier au coup suivant est $\frac{1}{N}$. La probabilité de ne pas tirer le dernier objet au coup suivant est $\frac{N-1}{N}$ (on obtient le double d'un des N-1 objets déjà obtenus). La probabilité de tirer le dernier objet au $t^{ème}$ coups est $\left(\frac{N-1}{N}\right)^{t-1} \times \frac{1}{N}$ (on ne tire pas le dernier objet pendant t-1 coups, puis on le tire).

Avec N=10, en supposant qu'on a tiré 9 objets. La probabilité de tirer le dernier en t coups est égale à $0{,}9^{t-1} \times 0{,}1$.
Pour $t \leq 20$, ces probabilités sont indiquées dans le tableau suivant (théoriquement t peut aller jusqu'à l'infini).

t	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
P(T=t)	0,1	0,09	0,081	0,073	0,066	0,059	0,053	0,048	0,043	0,039	0,035	0,031	0,028	0,025	0,023	0,021	0,019	0,017	0,015	0,014

Pour trouver la moyenne des valeurs de t, il faut calculer la somme théoriquement infinie :
$1\times0,9^0\times0,1+2\times0,9^1\times0,1+3\times0,9^2\times0,1+\ldots=0,1\times(1\times0,9^0+2\times0,9^1+3\times0,9^2+\ldots)$.
Si on fait la somme jusqu'à t=50, on trouve 9,69 environ.

t	1	2	3	4	5	…	48	49	50
P(T=t)	0,1	0,09	0,081	0,0729	0,06561	…	0,000706965	0,0006362685	0,0005726417
txP(t)	0,1	0,18	0,243	0,2916	0,32805	…	0,0339343224	0,0311771587	0,0286320845
somme		0,28	0,523	0,8146	1,14265	…	9,6309642444	9,6621414031	9,6907734876

En allant jusqu'à l'infini, on peut penser qu'on ne dépassera jamais 10. La valeur limite du temps d'attente moyen est 10.
Lorsqu'on a tiré 9 objets sur 10, il faut attendre en moyenne 10 tirages supplémentaires pour obtenir le dernier objet.

D'après l'énoncé cette observation est une loi générale qui fait que, lorsqu'on a tiré 19 objets d'une collection de 20 objets, on doit attendre en moyenne 20 tirages supplémentaires et lorsqu'on a tiré 199 objets d'une collection de 200 objets, on doit attendre en moyenne 200 tirages supplémentaires.

b) On considère maintenant la situation où, sur N objets à obtenir, on en a obtenu I (avec $o\leq I\leq N$).
Quelle est la probabilité de tirer un nouvel objet au coup suivant ?
Quelle est la probabilité de ne pas tirer un nouvel objet au coup suivant ?
Quelle est la probabilité de tirer un nouvel objet en t coups ?
Prendre N=10 et déterminer, avec la méthode algorithmique précédente, la moyenne théorique T du nombre t de coups à attendre pour obtenir un nouvel objet quand on en a déjà obtenu I=6.
Vérifier que la valeur obtenue est très légèrement inférieure à $\frac{N}{N-I}$, la valeur théorique attendue dans cette situation.
Lorsque l'on a déjà obtenu I objets, il en reste N-I à obtenir. La probabilité d'en tirer un nouveau est $\frac{N-I}{N}$ alors que la probabilité de ne pas en tirer un nouveau est $\frac{I}{N}$. La probabilité de tirer un nouvel objet seulement au $t^{ème}$ coup est $\left(\frac{I}{N}\right)^{t-1}\times\frac{N-I}{N}$ (on ne tire pas de nouvel objet pendant t-1 coups, puis on en tire un).

Avec N=10, en supposant qu'on a tiré I=6 objets. Il en reste N-I = 4 à tirer.
Les probabilités d'en tirer un nouveau en seulement t coups sont égales à $0,6^{t-1}\times0,4$.
Elles sont indiquées dans le tableau suivant, pour $t\leq13$ (théoriquement t peut aller jusqu'à l'infini).

t	1	2	3	4	5	6	7	8	9	10	11	12	13
P(T=t)	0,4	0,24	0,144	0,0864	0,05184	0,031104	0,0186624	0,01119744	0,006718464	0,0040310784	0,002418647	0,0014511882	0,0008707129

Pour trouver la moyenne des valeurs de t, il faut calculer la somme, théoriquement infinie :
$$0,4\times(1\times0,6^0+2\times0,6^1+3\times0,6^2+\ldots)$$
Si on fait la somme jusqu'à t=50, on trouve environ 2,5. Avec un tableur c'est facile d'aller aussi loin.

t	1	2	3	4	5	…	48	49	50
P(T=t)	0,4	0,24	0,144	0,0864	0,05184	…	1,49682E-011	8,98090E-012	5,38854E-012
txP(t)	0,4	0,48	0,432	0,3456	0,2592	…	7,18472E-010	4,40064E-010	2,69427E-010
somme		0,88	1,312	1,6576	1,9168	…	2,4999999989	2,4999999993	2,4999999996

En allant jusqu'à l'infini, on peut penser qu'on ne dépassera jamais 2,5. La valeur limite du temps d'attente moyen est 2,5.

D'après l'énoncé cette observation est une loi générale qui fait que, lorsqu'on a tiré 6 objets d'une collection de 10 objets, on doit attendre en moyenne $\frac{10}{10-6}=\frac{10}{4}=2,5$ tirages supplémentaires pour obtenir le $7^{ème}$ objet et lorsqu'on a tiré 7 objets de cette même collection de 10 objets, on doit attendre en moyenne $\frac{10}{10-7}=\frac{10}{3}\approx3,3$ tirages supplémentaires. En fin du processus, après avoir tiré le $9^{ème}$ objet, il faut attendre $\frac{10}{10-9}=\frac{10}{1}=10$ tirages supplémentaires pour obtenir le $10^{ème}$ et dernier objet de la collection, comme on l'a vu dans la question précédente.

c) Faire la somme des temps moyens que l'on obtient pour N=10 et les différentes valeurs possibles de I. Vérifier que cette approche probabiliste conduit à un résultat comparable à ceux qui ont été obtenu dans les exemples et la simulation.
On peut donc donner les résultats du temps d'attente moyen qui est égal à $\frac{N}{N-I}$ du I+$1^{ème}$ objet lorsqu'on en a obtenu I sur les N objets de la collection. Ces temps d'attente moyen sont donnés dans le tableau ci-dessous pour N=10.

i	0	1	2	3	4	5	6	7	8	9	Somme
N-i	10	9	8	7	6	5	4	3	2	1	
T=N/(N-i)	1	1,111	1,25	1,429	1,667	2	2,5	3,333	5	10	29,2897

Si on veut déterminer le temps moyen d'attente pour obtenir les N objets de la collection, il suffit de faire la somme des temps moyens pour obtenir chacun des nouveaux objets. Pour N=10, on trouve environ 29,3. Ce résultat coïncide approximativement avec nos observations pour les nombres irrationnels où le temps moyen d'attente des dix premiers chiffres en partant de la virgule est 32 pour le nombre π, 22 pour le nombre φ et 18 pour le nombre $\sqrt{2}$. Par contre, ce résultat coïncide très précisément avec le résultat obtenu par la simulation pour M=10000.

On peut remarquer que faire la somme des $\frac{N}{N-I}$ pour I allant de 0 à N-1 revient à multiplier N par la somme des inverses des nombres entiers jusqu'à l'inverse de N : $T=\sum_{I=0}^{N-1}\frac{N}{N-I}=N\left(1+\frac{1}{2}+\frac{1}{3}+\ldots+\frac{1}{N}\right)$. Cette remarque va nous être utile.

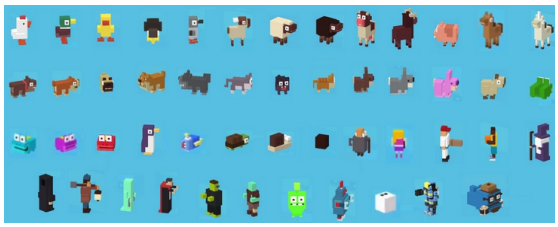
IIII] Applications concrètes

a) Je lance un dé à six faces jusqu'à ce que les six numéros soient sortis. Combien faut-il en moyenne de lancers pour que les six numéros sortent ? Joseph veut parier avec moi qu'il est capable de sortir les six numéros en moins de dix coups. Il suggère qu'on mise chacun 10 €. Le jeu est à l'avantage de qui ?

En moyenne il faut attendre $6\left(1+\frac{1}{2}+\frac{1}{3}+\frac{1}{4}+\frac{1}{5}+\frac{1}{6}\right)=6\left(\frac{60+30+20+15+12+10}{60}\right)=\frac{6\times147}{60}=14{,}7$.

Le jeu n'est donc pas avantageux pour Joseph qui perdra plus souvent ses 10 € qu'il ne gagnera les miens.

b) Il y a 200 personnages à obtenir dans le jeu « *Crossy road* » et Margaret en a déjà obtenus 50. Combien de temps* a t-elle dû attendre en moyenne pour en arriver là ? Combien de temps devra t-elle encore attendre en moyenne pour les avoir tous ?



* *il ne s'agit pas exactement de temps. Puisque les personnages s'obtiennent lorsqu'on a gagné 100 points au jeu, il s'agit de points. Mais pour avoir ces points, il faut y passer du temps...*

Voilà une question importante : tous ces jeux où on gagne des objets d'une collection sont particulièrement agaçants car, après un petit moment, on obtient de plus en plus d'objets en double et la collection semble ne jamais se terminer... On se demande aussi si certains objets ne sont pas plus rares que d'autres (le dernier est le plus long à obtenir, en moyenne il faut attendre N tirages pour obtenir le $N^{\text{ème}}$ objet).

Combien de temps a t-il fallu à Margaret pour obtenir ces 50 personnages ? Le résultat n'est pas $50\left(1+\frac{1}{2}+\frac{1}{3}+\frac{1}{4}+\frac{1}{5}+\ldots+\frac{1}{50}\right)\approx225$ tirages (extrait du tableur ci-dessous) car il ne s'agit pas d'obtenir tous les personnages d'une collection de 50 personnages...

n	1	2	3	4	5	6	...	47	48	49	50
$1/n$	1	0,5	0,3333333333	0,25	0,2	0,1666666667	...	0,0212765957	0,0208333333	0,0204081633	0,02
$\Sigma(1/n)$	1	1,5	1,8333333333	2,0833333333	2,2833333333	2,45	...	4,4379638417	4,4587971751	4,4792053383	4,4992053383

n	$\Sigma(1/n)$
224	224,96026692

Il ne faut attendre en moyenne que $200\left(\frac{1}{200}+\frac{1}{199}+\frac{1}{198}+\frac{1}{197}+\frac{1}{196}+\ldots+\frac{1}{151}\right)\approx57{,}4$ tirages (extrait du tableur ci-dessous) pour obtenir les 50 premiers personnages de la collection. Remarquons que ce nombre est très proche de 50 : au début, comme il y a beaucoup de personnages différents, il y a très peu de doubles (seulement 7 pour les 50 1ers personnages).

n	200	199	198	197	196	195	194	...	154	153	152	151
$1/n$	0,005	0,00503	0,0050505051	0,0050761421	0,0051020408	0,0051282051	0,0051546392	...	0,0064935065	0,0065359477	0,0065789474	0,0066225166
$\Sigma(1/n)$	0,005	0,01003	0,0150756307	0,0201517728	0,0252538136	0,0303820188	0,0355366579	...	0,2671129478	0,2736488956	0,2802278429	0,2868503595

n	$\Sigma(1/n)$
200	57,3700719

Combien de temps faut-il à Margaret pour obtenir les 200 personnages ?

En moyenne il faut attendre $200\left(1+\frac{1}{2}+\frac{1}{3}+\frac{1}{4}+\frac{1}{5}+\ldots+\frac{1}{200}\right)\approx1176$ tirages (extrait du tableur ci-dessous). Ce qui conduit à répondre à la question : combien de temps devra t-elle encore attendre en moyenne pour les avoir tous ? » Il lui faudra encore attendre $1176-57=1119$ tirages.

n	1	2	3	4	5	...	196	197	198	199	200
$1/n$	1	0,5	0,3333333333	0,25	0,2	...	0,0051020408	0,0050761421	0,0050505051	0,0050251256	0,005
$\Sigma(1/n)$	1	1,5	1,8333333333	2,0833333333	2,2833333333	...	5,8578791753	5,8629553174	5,8680058225	5,8730309481	5,8780309481

n	$\Sigma(1/n)$
1176	1175,6061896

Ces tirages seront obtenus après 111 900 points puisqu'il faut 100 points au jeu pour obtenir un tirage de personnage.

c) Donner une autre application de cette situation.

En voici une : combien faut-il réunir de personnes en moyenne pour que les 365 jours de l'année soient tous des jours d'anniversaires, le 29 février ne comptant pas ?

En moyenne il faut attendre $365\left(1+\frac{1}{2}+\frac{1}{3}+\frac{1}{4}+\frac{1}{5}+\ldots+\frac{1}{365}\right)$. On va utiliser le tableur pour trouver la valeur attendue :

n	1	2	3	4	5	...	362	363	364	365
$1/n$	1	0,5	0,3333333333	0,25	0,2	...	0,0027624309	0,0027548209	0,0027472527	0,002739726
$\Sigma(1/n)$	1	1,5	1,8333333333	2,0833333333	2,2833333333	...	6,4702404563	6,4729952772	6,47574253	6,478482256

2364,6460234

On constate que la somme vaut environ 6,5. Multiplié par 365, ce nombre donne environ 2364,65. Il faut donc en moyenne réunir 2365 personnes pour couvrir chaque jour du calendrier par la date d'anniversaire d'au moins une personne.

D'une façon générale :

- vous pouvez consulter ce site pour de plus amples informations et prolongations.
- La référence Wikipédia pour ce problème est Problème du collectionneur de vignettes (*coupon-collector problem*). On y apprend que George Pólya (1887-1985, ce nom vous dit quelque chose je suppose... cf le DS8) étudia le problème dans les années 30 puis William Feller (1906-1970). Outre l'apparition de la suite harmonique (la somme des inverses), ce qui est amusant/intéressant, comme souvent en mathématiques, c'est la diversité des applications de ce type de situation. Nous avons étudié des temps d'attente d'un nouvel objet dans le but d'évaluer le temps d'attente pour la collection complète, mais on peut appliquer le même modèle (la même méthodologie, les mêmes résultats) pour la recherche des contraintes intéressantes d'un problème d'optimisation (on fait des tests au hasard jusqu'à ce qu'on estime en avoir fait assez) ou dans le domaine de la diffusion de rumeurs (on informe une personne qui peut déjà avoir été informée).

La plupart d'entre vous a été tenté d'écrire des programmes pour répondre à ces questions. Ils sont de trois types :

1. la méthode expérimentale, empirique, qui simule l'expérience M fois (M=100 est un minimum pour obtenir une bonne estimation) du tirage des N objets de la collection jusqu'à obtention de la collection complète.
2. La méthode probabiliste qui nécessite de faire une somme infinie pour trouver la valeur moyenne. Dans l'énoncé, j'avais limité à 50 les termes de cette somme pour obtenir une bonne estimation quand N=10. Mais si on continue avec cette valeur de 50 quand N=200, on va très largement sous-estimer la moyenne du nombre de tirages. Rappelez-vous qu'il faut environ N tirages pour obtenir le $N^{ème}$ objet. Du coup si on calcule la moyenne en se limitant à 50 coups à chaque fois cela ne va pas donner les résultats attendus. J'ai écrit sur les copies de ceux qui ont obtenu 640 tirages au lieu des 1119 attendu qu'il fallait utiliser 10×N au lieu de 50 (donc 2000 quand N=200) pour obtenir une bonne estimation du nombre de tirages.
3. La méthode finale qui se dégage de la remarque faite dans l'énoncé : si le nombre de tirages pour obtenir le I+1ème objet (on en a déjà obtenu I) est $\frac{N}{N-I}$, alors il suffit de faire la somme de ces nombres pour I allant de 0 à $N-1$ lorsqu'on débute la collection, ou bien de D (nombre d'objets Déjà collectés) à $N-1$ lorsqu'on a déjà D objets. Cette méthode est la plus simple à programmer et donne les résultats théoriques finaux (comme si on avait été jusqu'à une somme infinie dans la méthode 2 ou si on avait recommencé M=infini de fois dans la méthode 1).

Voici donc un programme, légèrement adapté de celui trouvé dans une copie (celle de Thoula et Roxane) :

```
N=200 #Nombre de personnages en tout
D=50  #nombre de personnages Déjà obtenus (le I de l'énoncé)
R=N-D #nombre de personnages Restant à obtenir
T=0   #nombre de Tirages restant à effectuer
for _ in range(R):
    T+=N/(N-D)
    D+=1
print("T= {} ".format(T))
```

```
T= 1118.2361177287753
```

NB : j'ai mis une variable nommée « _ » comme compteur de la boucle car on ne se sert pas de cette variable.
Pour mon exemple avec les 365 jours :

```
N=365 #Nombre de personnages en tout
D=0   #nombre de personnages Déjà obtenus (le I de l'énoncé)
R=N-D #nombre de personnages Restant à obtenir
T=0   #nombre de Tirages restant à effectuer
for _ in range(R):
    T+=N/(N-D)
    D+=1
print("T= {} ".format(T))
```

```
T= 2364.6460234363376
```

Pour les 6 faces du dé :

```
N=6   #Nombre de personnages en tout
D=0   #nombre de personnages Déjà obtenus (le I de l'énoncé)
R=N-D #nombre de personnages Restant à obtenir
T=0   #nombre de Tirages restant à effectuer
for _ in range(R):
    T+=N/(N-D)
    D+=1
print("T= {} ".format(T))
```

```
T= 14.7
```

Pour les 681 stickers de la coupe du monde de football 2018 (vendus par paquets de 5 stickers coûtant 0,90€ d'après Elsa mais j'ai trouvé une offre d'un lot de 13 paquets à 9,90€ ce qui fait 0,76€ par pochette) que veut obtenir Benjamin : il lui faut acheter 4836 stickers, soit 968 paquets coûtant en tout 871,20€ ou bien 75 lots de 13 pochettes coûtant en tout 742,50€. Dans tous les cas c'est une collection qui reviendra cher...


```
N=681 #Nombre de personnages en tout
D=0   #nombre de personnages Déjà obtenus (le I de l'énoncé)
R=N-D #nombre de personnages Restant à obtenir
T=0   #nombre de Tirages restant à effectuer
for _ in range(R):
    T+=N/(N-D)
    D+=1
print("T= {} ".format(T))
```

```
T= 4836.1296759167235
```