Cet article est rédigé par des élèves. Il peut comporter des oublis et imperfections, autant que possible signalés par nos relecteurs dans les notes d'édition.

Le jeu des boîtes

Année 2018-2019

Auteurs : Adèle PIVETEAU, Anna SAFARIAN (Première S).
Établissement : Lycée Paul Guérin, Niort (79).
Encadrés par : Fabien AOUSTIN, Thomas FORGET.
Chercheur : Abdallah EL HAMIDI, Laboratoire des Sciences de l'Ingénieur pour l'Environnement, LaSIE, UMR CNRS 7356, La Rochelle Université.

Dans cet article, nous proposons différentes stratégies pour gagner au « jeu des boîtes » et présentons plusieurs programmes permettant de calculer des taux de réussite suivant les stratégies choisies.

1) Présentation du problème :

Le plateau d'un jeu télévisé est constitué de 3 salles :

- dans la salle 1, il y a 100 joueurs ;
- dans la salle 2, il y a 100 boîtes dans lesquelles se trouvent le nom des 100 joueurs (un nom par boîte) ;
- dans la salle 3, il n'y a rien.

Chaque joueur entre à tour de rôle dans la salle 2 :

- il ouvre la moitié des boîtes ;
- s'il trouve son nom, il passe dans la salle 3 ;
- sinon, il a perdu.

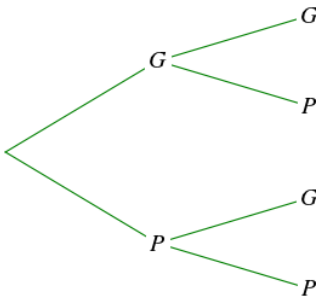
Les joueurs remportent la partie s'ils sont tous arrivés dans la salle 3.
Les joueurs peuvent se concerter avant le jeu mais ne peuvent plus communiquer pendant le jeu.

Notre but pour la suite a été de trouver une stratégie pour augmenter au maximum les chances de gagner.

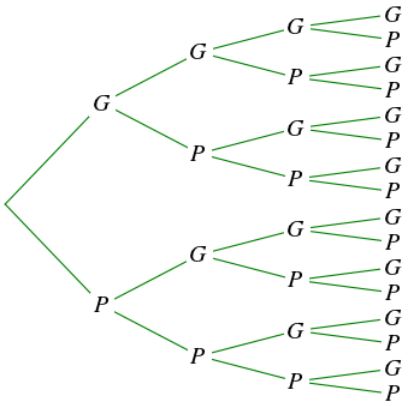
2) Premières recherches :

Nous avons tout d'abord étudié la situation sans appliquer de stratégie particulière et avec un plus petit nombre de joueurs. Chaque joueur choisit les boîtes au hasard. Le premier joueur a donc une chance sur deux de gagner, le deuxième aussi et ainsi de suite.

L'arbre ci-contre montre que pour 2 joueurs, il y a une chance sur 4 de gagner.



L'arbre ci-contre montre que pour 4 joueurs, il y a une chance sur 16 de gagner.



De façon générale, avec n joueurs, ils ont une chance sur 2^n de gagner.
Pour 100 joueurs, ils ont donc une chance sur 2^{100} de gagner, soit environ :

$7{,}9 \times 10^{-31} = 0{,}000\ 000\ 000\ 000\ 000\ 000\ 000\ 000\ 000\ 000\ 79.$

Ils n'ont donc pratiquement aucune chance de gagner !

3) Étude du jeu à 4 joueurs :

3-a) Quelques essais à la main :

On suppose maintenant que les joueurs A, B, C et D choisissent à l'avance les boîtes qu'ils vont ouvrir.
Il y a 24 façons d'agencer les noms dans les boîtes car il y a 4 noms répartis dans 4 boîtes.
Si, dans la boîte 1 on met le nom A alors il ne reste plus que 3 noms possibles pour la boîte 2, puis il restera 2 noms pour la boîte 3 et plus qu'un seul pour la boîte 4.
Le nombre total d'agencements est donc égal à : $4 \times 3 \times 2 \times 1 = 24 = 4!$

Voici les 24 possibilités d'agencement des noms dans les boîtes :

ABCD	BACD	CABD	DABC
ABDC	BADC	CADB	DACB
ACBD	BCAD	CBAD	DBAC
ACDB	BCDA	CBDA	DBCA
ADBC	BDAC	CDAB	DCAB
ADCB	BDCA	CDBA	DCBA

Considérons maintenant l'exemple suivant :
- A choisit les boîtes 1 et 3 ;
- B choisit les boîtes 2 et 4 ;
- C choisit les boîtes 1 et 2 ;
- D choisit les boîtes 2 et 3 ;

Sur les 24 agencements possibles, les joueurs ne gagnent que deux fois :

ABCD	BACD	CABD	DABC
ABDC	BADC	CADB	DACB
ACBD	BCAD	CBAD	DBAC
ACDB	BCDA	CBDA	DBCA
ADBC	BDAC	CDAB	DCAB
ADCB	BDCA	CDBA	DCBA

Les joueurs ont donc 2 chances sur 24 de gagner.
Tester les 24 agencements pour chaque stratégie adoptée est hélas très long à la main.

3-b) Traitement du problème avec un tableur et première conjecture :

Pour aller plus vite, nous avons utilisé un tableur.
Voici notre feuille de calcul :

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
1	Choix JA	Choix JB	Choix JC	Choix JD		Taux de réussite:		8,33 %														
2	1	3	2	1																		
3	4	4	3	4																		
4																						
5						Vérif 1 de A	Vérif 2 de A	Trouvé		Vérif 1 de B	Vérif 2 de B	Trouvé		Vérif 1de C	Vérif 2 de C	Trouvé		Vérif 1 de D	Vérif 2 de D	Trouvé		Résumé
6	A	B	C	D		1	0	1		0	0	0		0	1	1		0	1	1		0
7	A	B	D	C		1	0	1		0	0	0		0	0	0		0	0	0		0
8	A	C	B	D		1	0	1		1	0	1		1	0	1		0	1	1		1
9	A	C	D	B		1	0	1		0	1	1		1	0	1		0	0	0		0
10	A	D	B	C		1	0	1		1	0	1		0	0	0		0	0	0		0
11	A	D	C	B		1	0	1		0	1	1		0	1	1		0	0	0		0
12	B	A	C	D		0	0	0		0	0	0		0	1	1		0	1	1		0
13	B	A	D	C		0	0	0		0	0	0		0	0	0		0	0	0		0
14	B	C	A	D		0	0	0		0	0	0		1	0	1		0	1	1		0
15	B	C	D	A		0	1	1		0	0	0		1	0	1		0	0	0		0
16	B	D	A	C		0	0	0		0	0	0		0	0	0		0	0	0		0
17	B	D	C	A		0	1	1		0	0	0		0	1	1		0	0	0		0
18	C	A	B	D		0	0	0		1	0	1		0	0	0		0	1	1		0
19	C	A	D	B		0	0	0		0	1	1		0	0	0		0	0	0		0
20	C	B	A	D		0	0	0		0	0	0		0	0	0		0	1	1		0
21	C	B	D	A		0	1	1		0	0	0		0	0	0		0	0	0		0
22	C	D	A	B		0	0	0		0	1	1		0	0	0		0	0	0		0
23	C	D	B	A		0	1	1		1	0	1		0	0	0		0	0	0		0
24	D	A	B	C		0	0	0		1	0	1		0	0	0		1	0	1		0
25	D	A	C	B		0	0	0		0	1	1		0	1	1		1	0	1		0
26	D	B	A	C		0	0	0		0	0	0		0	0	0		1	0	1		0
27	D	B	C	A		0	1	1		0	0	0		0	1	1		1	0	1		0
28	D	C	A	B		0	0	0		0	1	1		1	0	1		1	0	1		0
29	D	C	B	A		0	1	1		1	0	1		1	0	1		1	0	1		1
30																						

Dans les cellules A2 à D3, on indique les numéros des boîtes choisies par chaque joueur.
Dans les cellules A6 à D29, on indique les 24 agencements des noms dans les boîtes possibles.
Dans les cellules F6 à F29, on vérifie si le joueur A trouve son nom dans la première boîte qu'il a choisie. La formule saisie en F6 est :

=OU(SI(A$2=1;A6="A");SI(A$2=2;B6="A");SI(A$2=3;C6="A");SI(A$2=4;D6="A"))

Dans les cellules G6 à G29, on vérifie si le joueur A trouve son nom dans la deuxième boîte qu'il a choisie.
Dans les cellules H6 à H29, on vérifie si le joueur A a trouvé son nom dans une des deux boîtes qu'il a ouverte. La formule saisie en H6 est :

=OU(F6:G6)

On procède de même pour les joueurs B, C et D.
Dans les cellules V6 à V29, on vérifie si tous les joueurs on trouvé leur nom. La formule saisie en V6 est :

=ET(H6;L6;P6;T6)

Enfin, on calcule en H1 le taux de réussite pour la stratégie saisie dans les cellules A2 à D3. La formule saisie en H1 est :

=NB.SI(V6:V29;1)/24

En testant plusieurs choix au hasard, nous avons conjecturé que le meilleur score est de 4 chances sur 24, soit environ 16,67 %.
C'est par exemple le cas si A choisit les boîtes 1 et 3, B les boîtes 2 et 4, C, les boîtes 2 et 4 et D les boîtes 1 et 3. Ils gagnent alors avec les agencements ABDC, ACDB, DBAC et DCAB.

Nous avons calculé le nombre de stratégies par joueur.
Au départ le joueur peut ouvrir une des 4 boîtes, il faut ensuite qu'il ouvre une des 3 boîtes restante. Il y a donc 4×3 combinaisons possibles si on tient compte de l'ordre des boîtes. Mais ouvrir la boîte 1 puis la boîte 2 ou la boîte 2 puis la boîte 1 revient au même. Dans les 12 combinaisons ordonnées, seule la moitié est à prendre en compte.
Il y a donc 6 choix pour le joueur A : 1 et 2, 1 et 3, 1 et 4, 2 et 3, 2 et 4, 3 et 4.

Pour 4 joueurs il y a donc 6^4 c'est-à-dire 1 296 combinaisons à tester.

Même avec le tableur, c'était encore trop long !

3-c) Traitement du problème avec un programme et une deuxième conjecture :

Nous avons écrit un programme pour tester tous les cas envisagés précédemment.
Pour cela, nous avons utilisé le logiciel XCas.
Les noms des joueurs sont 0, 1, 2 et 3 car le logiciel ne prend pas en compte les lettres et commence les numérotations à 0.

Les noms dans les boîtes sont représentés par une liste de 4 éléments.
Les choix de chaque joueur sont représentés par une liste de 2 éléments.

Le programme **`Stratégie_correcte`** prend en entrée :

- **`Noms`** : la liste des noms dans les boîtes,
- **`Choix0`** : la liste des choix du joueur 0,
- **`Choix1`** : la liste des choix du joueur 1,
- **`Choix2`** : la liste des choix du joueur 2,
- **`Choix3`** : la liste des choix du joueur 3.

Il renvoie **`True`** si les joueurs ont gagné et **`False`** sinon.
Voici le programme :

```
Stratégie_correcte(Noms,Choix0,Choix1,Choix2,Choix3):={
  local Vérif0,Vérif1,Vérif2,Vérif3,Vérif_totale;
  Vérif0:=Noms[Choix0[0]]==0 ou Noms[Choix0[1]]==0;
  Vérif1:=Noms[Choix1[0]]==1 ou Noms[Choix1[1]]==1;
  Vérif2:=Noms[Choix2[0]]==2 ou Noms[Choix2[1]]==2;
  Vérif3:=Noms[Choix3[0]]==3 ou Noms[Choix3[1]]==3;
  Vérif_totale:=Vérif0 et Vérif1 et Vérif2 et Vérif3;
  return Vérif_totale;
}:;
```

Par exemple, si on saisit :

```
Stratégie_correcte([0,2,3,1],[0,2],[1,3],[0,1],[1,2])
```

le résultat obtenu est **`True`**.

Le logiciel XCas permet de tester facilement toutes les permutations dans l'ordre de **`[0,1,2,3]`** jusqu'à **`[3,2,1,0]`** avec la commande **`nextperm`**.

`[0,1,2,3]`	`[1,0,2,3]`	`[2,0,1,3]`	`[3,0,1,2]`
`[0,1,3,2]`	`[1,0,3,2]`	`[2,0,3,1]`	`[3,0,2,1]`
`[0,2,1,3]`	`[1,2,0,3]`	`[2,1,0,3]`	`[3,1,0,2]`
`[0,2,3,1]`	`[1,2,3,0]`	`[2,1,3,0]`	`[3,1,2,0]`
`[0,3,1,2]`	`[1,3,0,2]`	`[2,3,0,1]`	`[3,2,0,1]`
`[0,3,2,1]`	`[1,3,2,0]`	`[2,3,1,0]`	`[3,2,1,0]`

Le programme **`Nombre_succès`** prend en entrée :

- **`Choix0`** : la liste des choix du joueur 0,
- **`Choix1`** : la liste des choix du joueur 1,
- **`Choix2`** : la liste des choix du joueur 2,
- **`Choix3`** : la liste des choix du joueur 3.

Il teste ces choix sur les 24 façons d'agencer les noms dans les boîtes et renvoie le nombre de fois où les joueurs ont gagné.

Voici le programme :

```
Nombre_succès(Choix0,Choix1,Choix2,Choix3):={
  local Noms,Comptage,k;
  Noms:=[0,1,2,3];
  Comptage:=0;
  pour k de 1 jusque 24 faire
    si Stratégie_correcte(Noms,Choix0,Choix1,Choix2,Choix3)==vrai alors
        Comptage:=Comptage+1;
    fsi;
    Noms:=nextperm(Noms);
  fpour;
  return Comptage;
};;
```

Le programme **Meilleure_stratégie** teste tous les choix possibles des joueurs (6 choix pour chacun des 4 joueurs) sur chacun des 24 agencements des noms dans les boîtes et affiche les stratégies qui donnent les meilleurs résultats. (1)

```
Meilleure_stratégie():={
  local Taux_de_réussite,J0,J1,J2,J3,Choix0,Choix1,Choix2,Choix3;
  Taux_de_réussite:=0;
  pour J0 de 1 jusque 6 faire
    si J0==1 alors Choix0:= [0,1];fsi;
    si J0==2 alors Choix0:= [0,2];fsi;
    si J0==3 alors Choix0:= [0,3];fsi;
    si J0==4 alors Choix0:= [1,2];fsi;
    si J0==5 alors Choix0:= [1,3];fsi;
    si J0==6 alors Choix0:= [2,3];fsi;
  pour J1 de 1 jusque 6 faire
    si J1==1 alors Choix1:= [0,1];fsi;
    si J1==2 alors Choix1:= [0,2];fsi;
    si J1==3 alors Choix1:= [0,3];fsi;
    si J1==4 alors Choix1:= [1,2];fsi;
    si J1==5 alors Choix1:= [1,3];fsi;
    si J1==6 alors Choix1:= [2,3];fsi;
  pour J2 de 1 jusque 6 faire
    si J2==1 alors Choix2:= [0,1];fsi;
    si J2==2 alors Choix2:= [0,2];fsi;
    si J2==3 alors Choix2:= [0,3];fsi;
    si J2==4 alors Choix2:= [1,2];fsi;
    si J2==5 alors Choix2:= [1,3];fsi;
    si J2==6 alors Choix2:= [2,3];fsi;
  pour J3 de 1 jusque 6 faire
    si J3==1 alors Choix3:= [0,1];fsi;
    si J3==2 alors Choix3:= [0,2];fsi;
    si J3==3 alors Choix3:= [0,3];fsi;
    si J3==4 alors Choix3:= [1,2];fsi;
    si J3==5 alors Choix3:= [1,3];fsi;
    si J3==6 alors Choix3:= [2,3];fsi;
  si Nombre_succès(Choix0,Choix1,Choix2,Choix3)/24*100.0>=Taux_de_réussite alors
    Taux_de_réussite:=Nombre_succès(Choix0,Choix1,Choix2,Choix3)/24*100.0;
    afficher(Taux_de_réussite,Choix0,Choix1,Choix2,Choix3);
  fsi;
  fpour;
  fpour;
  fpour;
  fpour;
};;
```

Voici les meilleurs résultats obtenus avec le programme **`Meilleure_stratégie`** :

16.6666666667,[0,1],[0,1],[2,3],[2,3]
16.6666666667,[0,1],[2,3],[0,1],[2,3]
16.6666666667,[0,1],[2,3],[2,3],[0,1]
16.6666666667,[0,2],[0,2],[1,3],[1,3]
16.6666666667,[0,2],[1,3],[0,2],[1,3]
16.6666666667,[0,2],[1,3],[1,3],[0,2]
16.6666666667,[0,3],[0,3],[1,2],[1,2]
16.6666666667,[0,3],[1,2],[0,3],[1,2]
16.6666666667,[0,3],[1,2],[1,2],[0,3]
16.6666666667,[1,2],[0,3],[0,3],[1,2]
16.6666666667,[1,2],[0,3],[1,2],[0,3]
16.6666666667,[1,2],[1,2],[0,3],[0,3]
16.6666666667,[1,3],[0,2],[0,2],[1,3]
16.6666666667,[1,3],[0,2],[1,3],[0,2]
16.6666666667,[1,3],[1,3],[0,2],[0,2]
16.6666666667,[2,3],[0,1],[0,1],[2,3]
16.6666666667,[2,3],[0,1],[2,3],[0,1]
16.6666666667,[2,3],[2,3],[0,1],[0,1]

Le résultat donné par le programme montre que la première conjecture est vraie : à quatre joueurs, les meilleures stratégies permettent de gagner 1 fois sur 6.

Nous avons remarqué que toutes ces stratégies consistaient à répartir les boîtes en deux groupes :

- deux joueurs ouvrent les deux mêmes boîtes,
- les deux autres joueurs ouvrent les deux autres boîtes.

Nous avons conjecturé que cela reste vrai pour plus de joueurs. Par exemple, à 10 joueurs, la meilleure stratégie consisterait à ce que 5 joueurs choisissent les mêmes numéros de boîtes (0, 1, 2, 3 et 4 par exemple) et les 5 autres joueurs choisissent les autres numéros (donc 5, 6, 7, 8 et 9 ici).

4) Étude du jeu à 6 joueurs :

Nous avons modifié nos programmes précédents pour étudier le jeu à 6 joueurs.
Il y a beaucoup plus de cas à prendre en compte.

- Le nombre de façon d'agencer les noms dans les boîtes est égal à : 6 × 5 × 4 × 3 × 2 × 1 = 720.
- Chacun des six joueurs à 20 choix possibles :

```
[0,1,2]   [0,2,3]   [0,3,5]   [1,2,5]   [2,3,4]
[0,1,3]   [0,2,4]   [0,4,5]   [1,3,4]   [2,3,5]
[0,1,4]   [0,2,5]   [1,2,3]   [1,3,5]   [2,4,5]
[0,1,5]   [0,3,4]   [1,2,4]   [1,4,5]   [3,4,5].
```

En arrêtant le programme au bout de 10 heures, les derniers résultats obtenus sont :

5.0,[0,1,2],[0,1,2],[0,1,2],[3,4,5],[3,4,5],[3,4,5]
5.0,[0,1,2],[0,1,2],[3,4,5],[0,1,2],[3,4,5],[3,4,5]
5.0,[0,1,2],[0,1,2],[3,4,5],[3,4,5],[0,1,2],[3,4,5]
5.0,[0,1,2],[0,1,2],[3,4,5],[3,4,5],[3,4,5],[0,1,2]

Sur ces premiers résultats, notre conjecture semble vérifiée : trois joueurs ouvrent les boîtes 0, 1, 2 et les trois autres ouvrent les boîtes 3, 4, 5.

Nous avons demandé à l'ordinateur combien de temps il mettait pour appliquer 100 fois :
`Nombre_succès([0,1,2],[0,1,2],[0,1,2],[3,4,5],[3,4,5],[3,4,5])`
et il a répondu : 8,72 secondes.

Nous avons estimé que pour finir tous les tests, le programme mettrait donc environ...

$20^6 \times 8{,}72 / 100 = 5\,580\,800$ secondes soit environ 65 jours !! (2)

5) Une autre approche du problème :

Pour le moment, nous avons choisi des stratégies fixées dès le début.
Nous avons donc pensé que chaque joueur pourrait tenir compte du nom qu'il découvre dans la boîte qu'il ouvre.

Par exemple, considérons que les noms sont répartis de la façon suivante : B, D, E, A, F, C.

- A ouvre la boîte 1, il trouve le nom B.
  Il ouvre donc la boîte 2, il trouve le nom D.
  Il ouvre donc la boîte 4 : il trouve son nom !
- B ouvre la boîte 2, il trouve le nom D.
  Il ouvre donc la boîte 4, il trouve le nom A.
  Il ouvre donc la boîte 1 : il trouve son nom !
- C ouvre la boîte 3, il trouve le nom E.
  Il ouvre donc la boîte 5, il trouve le nom F.
  Il ouvre donc la boîte 6 : il trouve son nom !
- On continue ainsi de suite...
- Dans cet exemple, tous les joueurs trouvent leur nom !

Nous avons testé quelques uns des 720 cas à la main et cette méthode nous a semblé nettement plus efficace !
Nous avons donc programmé cette seconde stratégie :

```
Stratégie2_4():={
  local k,Noms,Comptage,j0,j1,j2,j3;
  Noms:=[0,1,2,3];
  Comptage:=0;
  pour k de 1 jusque 24 faire
    j0:=Noms[0]==0 ou Noms[Noms[0]]==0;
    j1:=Noms[1]==1 ou Noms[Noms[1]]==1;
    j2:=Noms[2]==2 ou Noms[Noms[2]]==2;
    j3:=Noms[3]==3 ou Noms[Noms[3]]==3;
    si j0 et j1 et j2 et j3 alors
      Comptage:=Comptage+1;
      afficher(Noms);
    fsi;
    Noms:=nextperm(Noms);
 fpour;
 return(Comptage);
}:;
```

Nous avons utilisé différentes variables, comme **`Noms`** dans laquelle on trouve les noms de quatre joueurs ou **`Comptage`** qui sert à compter le nombre de victoires.
Le programme fonctionne comme lorsque nous le faisons à la main.

Par exemple, traitons le cas où les noms sont agencés de la façon suivante : **`[2,3,0,1]`**.
Pour le joueur 0 (nommé **`j0`**), le programme va aller chercher au rang 0 et vérifier si il trouve 0 (il vérifie si dans la boîte 0 se trouve le nom du joueur 0). Cela correspond à l'instruction « **`j0:=Noms[0]==0`** ». Dans notre exemple **`Noms[0]`** vaut 2 ; **`j0`** n'a pas encore trouvé son nom.
Ensuite on vérifie si on trouve le nom 0 au rang 2 (dans la boîte 2). Cela correspond à l'instruction : « **`Noms[Noms[0]]==0`** ». On trouve 0, donc **`j0`** a gagné.
On procède de même pour les 3 joueurs restants.

Si tous les joueurs trouvent leur nom la variable **Comptage** comptabilise un cas favorable de plus. Grâce à la commande « **nextperm** » le programme va tester cette stratégie sur tous les agencements de noms dans les boîtes pour 4 joueurs (il y en 24).

À la fin, le programme nous affiche la variable « **Comptage** ». Ici, on obtient 10 ce qui signifie que les joueurs gagnent pour 10 des 24 agencements possibles. Cela correspond à un taux de réussite d'environ 41,67 %.
Les 10 agencements qui permettent de gagner sont :

Noms:[0,1,2,3]
Noms:[0,1,3,2]
Noms:[0,2,1,3]
Noms:[0,3,2,1]
Noms:[1,0,2,3]
Noms:[1,0,3,2]
Noms:[2,1,0,3]
Noms:[2,3,0,1]
Noms:[3,1,2,0]
Noms:[3,2,1,0]

Hélas, nous n'avons rien remarqué de particulier dans ces agencements. (3)

Nous avons ensuite adapté ce programme pour 6, puis 8 et enfin 10 joueurs.
Voici par exemple le programme pour 6 joueurs :

```
Stratégie2_6():={
  local k,Noms,Comptage,j0,j1,j2,j3,j4,j5;
  Noms:=[0,1,2,3,4,5];
  Comptage:=0;
  pour k de 1 jusque 720 faire
    j0:=Noms[0]==0 ou Noms[Noms[0]]==0 ou Noms[Noms[Noms[0]]]==0;
    j1:=Noms[1]==1 ou Noms[Noms[1]]==1 ou Noms[Noms[Noms[1]]]==1;
    j2:=Noms[2]==2 ou Noms[Noms[2]]==2 ou Noms[Noms[Noms[2]]]==2;
    j3:=Noms[3]==3 ou Noms[Noms[3]]==3 ou Noms[Noms[Noms[3]]]==3;
    j4:=Noms[4]==4 ou Noms[Noms[4]]==4 ou Noms[Noms[Noms[4]]]==4;
    j5:=Noms[5]==5 ou Noms[Noms[5]]==5 ou Noms[Noms[Noms[5]]]==5;
    si j0 et j1 et j2 et j3 et j4 et j5 alors
      Comptage:=Comptage+1;
      afficher(Noms);
    fsi;
    Noms:=nextperm(Noms);
 fpour;
 return(Comptage);
};;
```

Voici les taux de réussite obtenus :
- pour 6 joueurs : environ 38,33 % ;
- pour 8 joueurs : environ 36,55 % ;
- pour 10 joueurs : environ 35,44 %.

Nous conjecturons que ce taux de réussite décroît quand le nombre de joueur augmente.
On pourrait penser que le taux limite sera d'environ 30 % ou 1/3.

Notes d'édition

(1) Il est possible de rendre ce code plus compact, par exemple en rangeant les noms des joueurs dans un tableau [J1, J2, J3, J4] et en effectuant deux boucles imbriquées, la première sur le tableau et la seconde de 1 à 6.

(2) Le temps d 'exécution dépend bien sûr de la machine utilisée… mais il sera clairement prohibitif si l'on veut tester toutes les combinaisons pour 100 joueurs !

(3) On peut remarquer que pour ces agencements, la règle énoncée page 7 est toujours vérifiée. Par exemple, dans [3,2,1,0], J0 ouvre la boîte 0 et trouve 3, il ouvre donc la boîte 3 et trouve 0, son numéro, etc… Ce constat vous mettra sur la piste d'une démonstration formelle de la conjecture énoncée en bas de page.