Jeu de mojette
2025
Basile Loiseau Naudot et Pierre Cottin
Établissement : Lycée public Fernand Renaudeau
Encadrés par : **Maëva Tauzin**, **Marie-Pierre Delhaye**
Chercheur : **Damien Gobin**, **université de Nantes**

Le jeu de mojette a été inventé par Jeanpierre Guédon. Il s'agit d'un jeu de logique semblable au sudoku et aux pyramides d'additions.

Présentation du sujet

Testez le jeu présenté ici : https://www.mojette.net./. On pourra coder ce jeu en Python et chercher à déterminer quelle(s) condition(s) sur les valeurs initiales il est possible de trouver une solution. On pourra commencer par étudier le cas du sudoku, plus classique et qui pose le même type de questions.

Règles :

Le jeu de la mojette consiste en la complétion d'une grille. Cette grille peut prendre plusieurs formes mais nous nous sommes intéressés à la forme diamant 24.

			I8		I1			
		I10	I9	R1	I2	I3		
	I12	I11	R2	R3	R4	I4	I5	
I14	I13	R5	R6	R7	R8	R9	I6	I7
	R10	R11	R12	R13	R14	R15	R16	
		R17	R18	R19	R20	R21		
			R22	R23	R24			
	I15	I16	I17	I18	I19	I20	I21	

Voici un exemple de grille diamant, avec en vert les valeurs de la grille à deviner, et autour les indices.

Pour compléter la grille il faut respecter plusieurs règles :

- Il faut utiliser que des chiffres entre 0 et 9
- Il ne faut pas utiliser plus de trois chiffres différents
- Il faut que la somme des chiffres des colonnes et des diagonales soit égale à l'indice correspondant

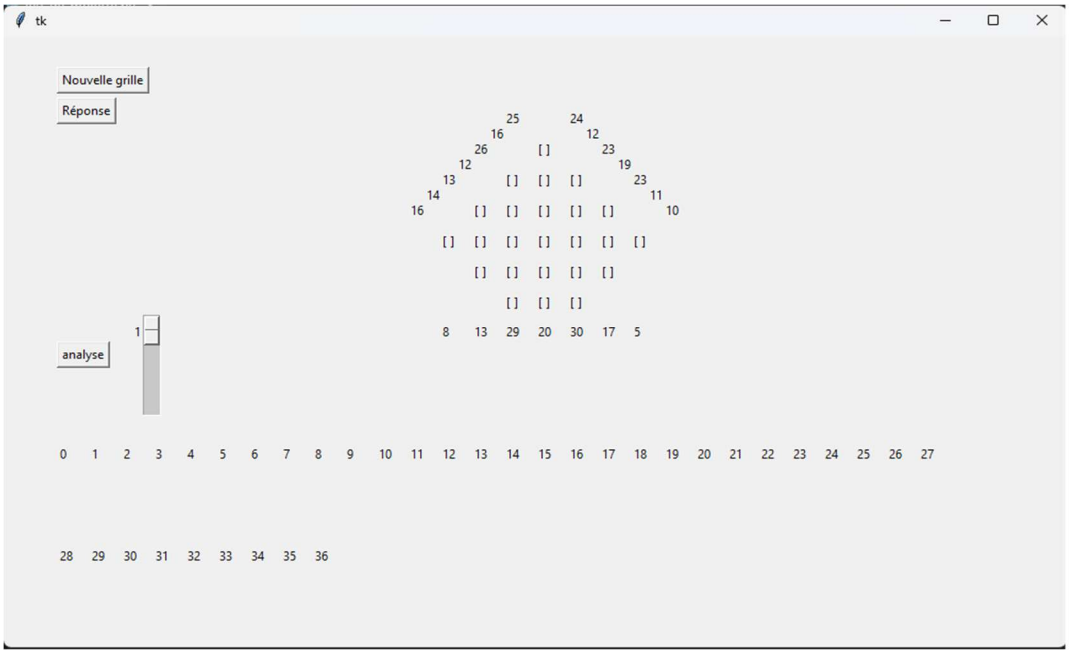
Pour des raisons de simplicité, nous avons décidé de ne pas utiliser la règle 2, la règle des trois chiffres différents.

Recherches :

Code :

Nous avons donc choisi de suivre la proposition de notre sujet d'utiliser python pour modéliser le jeu de mojette. Notre objectif avec ce programme était de pouvoir étudier plusieurs grilles générées aléatoirement.

Voici une photo de ce que donne notre programme une fois exécuté :



Ce programme permet de construire des grilles générées aléatoirement.

Fonctionnalités :

Les cases de la grille sont remplies aléatoirement avec des chiffres entre 0 et 9, mais remplacées par des crochets afin de ne pas les dévoiler. Ces valeurs sont ensuite utilisées pour calculer les indices, la somme des colonnes et des diagonales qui permet de résoudre la grille. Ces derniers sont affichés en bout de ligne, afin de faciliter la compréhension et rester fidèle au jeu de base. Les valeurs peuvent être affichés à l'aide du bouton « Réponse », afin de vérifier la bonne réponse de la grille. Nous avons par ailleurs créé d'autres boutons, notamment le bouton « Nouvelle grille » qui permet d'effacer la grille actuelle afin de la remplacer par une nouvelle, elle aussi générée aléatoirement. Le dernier bouton nommé « Analyse » permet quant à lui de choisir un chiffre entre 1 et 1000, qui sera le nombre de grilles analysées par la suite. L'analyse compte toutes les valeurs et diagonales des grilles analysées, afin de pouvoir réaliser des statistiques par exemple.

Fonctionnement :

Le programme construit les grilles en partant de réponses générées aléatoirement. Les indices sont donc calculés en réalisant la somme des réponses de leur colonne ou diagonale.

```
143
144 def affichage_indice():
145     global nb_indices_diagonales,nb_indices_ligne_du_bas
146
147     nb_indices_diagonales=[
148         reponses[0]+reponses[3]+reponses[8]+reponses[15],
149         reponses[2]+reponses[7]+reponses[14],
150         reponses[1]+reponses[6]+reponses[13]+reponses[20],
151         reponses[5]+reponses[12]+reponses[19],
152         reponses[4]+reponses[11]+reponses[18]+reponses[23],
153         reponses[10]+reponses[17]+reponses[22],
154         reponses[9]+reponses[16]+reponses[21],
155
156         reponses[0]+reponses[1]+reponses[4]+reponses[9],
157         reponses[2]+reponses[5]+reponses[10],
158         reponses[3]+reponses[6]+reponses[11]+reponses[16],
159         reponses[7]+reponses[12]+reponses[17],
160         reponses[8]+reponses[13]+reponses[18]+reponses[21],
161         reponses[14]+reponses[19]+reponses[22],
162         reponses[15]+reponses[20]+reponses[23]]
163
164     nb_indices_ligne_du_bas=[
165         reponses[9],
166         reponses[4]+reponses[10]+reponses[16],
```

Pour afficher proprement le résultat, nous avons choisi d'utiliser Tkinter, un moteur graphique intégré à python. C'était la première fois que nous l'utilisions et notre programme n'est donc pas aussi optimisé et esthétique qu'il pourrait l'être.

```
54  def creation_grille():
55
56      if affichage_reponse :
57          if type_grille=="diamant24":
58
59              case = Label(fenetre, text="[ ]")
60              case.place(x=500,y=100)
61
62              for i in range (3):
63                  case = Label(fenetre, text="[ ]")
64                  case.place(x=470+30*i,y=130)
65
66              for i in range (5):
67                  case = Label(fenetre, text="[ ]")
68                  case.place(x=440+30*i,y=160)
69
70              for i in range (7):
71                  case = Label(fenetre, text="[ ]")
72                  case.place(x=410+30*i,y=190)
73
74              for i in range (5):
75                  case = Label(fenetre, text="[ ]")
76                  case.place(x=440+30*i,y=220)
77
```

Pour observer les valeurs des indices pour les analyser, il nous a suffi de créer une boucle qui en génère plusieurs sans les afficher et qui additionne les valeurs.

```
13  def analyse():
14
15      global nb_indices_diagonales,nb_indices_ligne_du_bas
16
17      #analyse diagonales
18
19      nb_de_chaques_nb=[]
20
21      for i in range(55):
22          nb_de_chaques_nb.append(0)
23
24      for i in range (int(nb_test.get())):
25          construction_nouvelle_grille()
26
27          for j in range(37):
28              nb_de_chaques_nb.insert(j,nb_indices_diagonales.count(j)+nb_de_chaques_nb[j])
29              del nb_de_chaques_nb[j+1]
30
31      for i in range(37):
32          nb = Label(fenetre, text=nb_de_chaques_nb[i])
33          nb.place(x=50+i*30,y=420)
```

Formule :

Nous avons ensuite tenté de créer une formule qui permettrait de retrouver les réponses de la grille à partir des indices initiaux. Nous nous sommes demandé s'il existait une égalité entre les indices et les réponses.

Voici notre recherche de formule :

R1O=I15

On recherche R17 :

R17=I14-I15-R22

R17=I14-I15-(I17-R18-R12-R6-R2) *(formule pouvant contenir des erreurs)*

R17=I14-I15-I17+R18+R12+R6+R2

R17=I14-I15-I17+(I13-R11-R23) +R12+R6+R2

Comme on peut observer ci-dessus, on remarque que notre recherche tourne en rond, la formule augmente de taille à chaque étape.

Brute force :

Pour réussir à finir les grilles avec un programme python, nous nous sommes intéressés au brute force, technique qui consiste à tester toutes les possibilités pour remplir la grille. Nous avons donc recherché un nombre minimum de case à remplir aléatoirement pour en déduire le reste. Le programme teste à chaque fois si les valeurs aléatoires fonctionnent pour finir la grille.

			I8		I1			
		I10	I9	R1	I2	I3		
	I12	I11	R2	R3	R4	I4	I5	
I14	I13	R5	R6	R7	R8	R9	I6	I7
	R10	R11	R12	R13	R14	R15	R16	
		R17	R18	R19	R20	R21		
			R22	R23	R24			
	I15	I16	I17	I18	I19	I20	I21	

Les cases orange sont les cases à tirer aléatoirement au début pour la compléter. Les cases noires sont trouvables grâce aux indices I15 et I21. On peut donc déduire les valeurs des cases vertes à partir des cases orange et noire.

Conclusion :

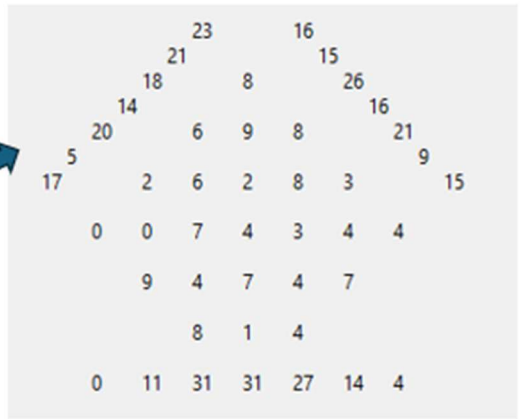
Notre sujet avait pour objectif de trouver des valeurs initiales qui permettent de trouver une solution. Nos recherches nous ont permis de mieux comprendre notre sujet et nous ont permis de créer des grilles tout le temps réalisable avec notre programme en python. Néanmoins, nos recherches ne nous ont pas permis de répondre à la question initiale.

Mais nous avons quand même trouvé une partie de la réponse :

Cas impossible :

. 2 lignes qui ne peuvent pas coexister

Si ce nombre est supérieur à 18, la grille est impossible.



Nous avons donc trouvé une règle qui empêche certains indices de dépasser un seuil. Ici, l'indice égal à 17 ne peut pas dépasser 18 car l'indice 0 empêche la somme de la diagonale de dépasser ce nombre.

Notre article se conclura sur cette pensée.