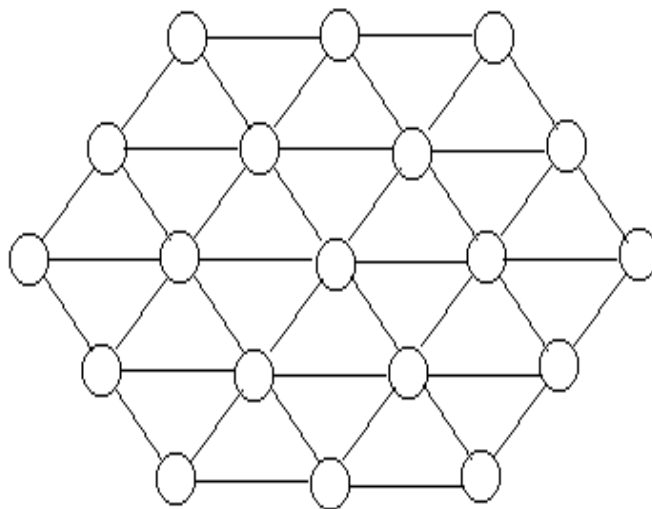


IA 1 – Devoir n° 1

Je vais vous proposer ma réponse au problème posé en présentant le cheminement de mon raisonnement, c'est à dire en montrant l'évolution de l'efficacité de mon algorithme, pour finalement terminer par son application et aboutir au résultat final.

Commençons par attribuer à chaque nœud de la grille une lettre pour le reconnaître. Ainsi, nous obtenons la grille suivante :



Pour commencer, on peut directement trouver la somme des entiers situés sur n'importe quelle ligne en calculant la somme de tous les entiers de 1 à 19 puis en divisant par 5. En effet :
 $A+B+C = D+E+F+G = H+I+J+K+L = M+N+O+P = Q+R+S = \text{Somme}$, par hypothèses, et de plus la somme de ces 5 sommes est égale à la somme des entiers 1 à 19.
On obtient donc $\text{Somme} = 38$.

Pour résoudre notre problème on peut directement utiliser cette information.

Modélisation :

Le premier réflexe est de faire un placement de 0 à 19 entiers sur la grille en ajoutant un à un les entiers sur des nœuds vides ; puis de vérifier l'égalité de chaque somme de ligne entre elles.
De cette manière, on a 19! états possibles que le futur algorithme regardera 1 par 1 pour un jour arriver à une bonne solution, un jour ...!

Il faut donc dès le départ imposer que $\text{Somme} = 38$.
Mais cela ne suffit pas car les 19 entiers étant placés, si $\text{Somme} \neq 38$, alors l'algorithme va rechercher une nouvelle façon de placer ces 19 entiers, puis vérifier que $\text{Somme} = 38$ et ainsi de suite, le problème est donc toujours le même ou presque et il faudra attendre un long moment avant d'arriver à quelque chose.

Ce qui nous amène à vérifier que $\text{Somme} = 38$ après chaque nouvelle création de ligne et si ce n'est pas le cas, alors seule cette ligne est changée. On procède donc ligne par ligne.

Nombre de solutions attendues :

Une fois une première solution donnée, il faut revenir en arrière pour explorer les autres solutions possibles mais en fait une seule solution est possible et toutes les autres ne seront que transformations de la première. En effet, si on a « la première solution », on obtiendra les autres en faisant des rotations et symétries sur la première.

On peut donc s'attendre à avoir 11 solutions en plus de la première. (rotation de 60° à droite et à gauche, rotation de 120° à droite et à gauche, rotation de 180°, et 5 symétries axiales par rapport à la première solution).

On pourra donc choisir de coder la résolution du problème, de telle façon que le programme backtrack, après avoir trouvé une solution, pour trouver les « autres » (qui sont en fait les mêmes) ou empêche de développer une nouvelle solution si elle est une transformation d'une solution déjà donnée. Avec cette deuxième méthode, l'obtention des résultats sera plus rapide et on donnera pour chaque solution ces 11 images directement en empêchant le programme de les chercher lui-même.

Je m'arrêterai finalement sur la solution donnant tous les résultats sans leurs images au départ (il n'y en aura en fait qu'un possible) et en donnant « moi-même » les 11 transformations de chacun de ces résultats.

Spécifications et Codage :

On appellera repartition le prédicat prenant en argument une liste Liste qui sera la liste de 1 résultat avec ces 11 images, 1 résultat étant une liste d'entiers de 1 à 19 de la forme [A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S] vérifiant les conditions du problème.

Pour coder le corps de ce prédicat, on aura besoin de prédicat auxiliaire sur les manipulations de liste :

-appartient(X,L) vrai si X est un élément de la liste L.

-retire(X,L1,L2) vrai si X est un élément de la liste L1 et que L2 est la liste L1 avec l'élément X en moins.

L'idée est de tirer, d'une liste contenant les 19 premiers entiers, les 19 nœuds de la grille et de vérifier que la somme des entiers de chaque ligne est égale à 38.

Si on se limite à cette idée, on donnera comme réponse, le codage que j'ai appelé repart1 et que vous trouverez dans les pages suivantes :

On commence par donner une liste L0 des entiers de 1 à 19 puis on utilise 18 fois le prédicat retire pour donner une valeur aux nœuds A à R de la grille en supprimant de la liste, à chaque fois, la valeur affectée à un nœud. Pour S, il ne reste plus qu'un élément dans la liste des entiers (L18) et on utilise donc appartient pour donner une valeur à S.

Il y a en tout 15 sommes différentes possibles dans la grille, on écrit alors à quoi sont égales ces sommes en rappelant que chacune d'elles doit être égale à 38.

Comme je l'expliquais, lors de la modélisation, cette façon de faire n'est pas efficace et on préférera vérifier la somme d'une ligne dès que les nœuds qui la forment seront affectés par des entiers.

On obtient alors le code repart2 :

Pour la première ligne A B C, on retire A, B et C puis on vérifie que la somme est 38, si ce n'est pas le cas, il ne sert à rien de continuer et on backtrack sur les précédents points de choix, sinon on continue et ainsi de suite. Pour l'ordre dans lequel je construis les lignes, je pense qu'il est préférable de commencer par les lignes à 3 nœuds, puis à 4 et enfin à 5 : cela permet de limiter le nombre de points de choix au début alors que la liste contient encore 19 éléments.

(Remarque : pour repart1 et repart2, repartition(Liste) prédicat où Liste est la liste des 19 nœuds de la grille de A à S vérifiant les conditions.)

Les 2 codes précédents renvoient toutes les solutions en comptant les rotations et symétries d'une même solution ; cette recherche des images est inutile car on sait d'avance que pour une solution, 11 mêmes solutions images existent. On veut donc empêcher Prolog de développer des solutions qui sont déjà apparues sous une autre forme. Pour cela, il faut comprendre que, pour une solution, 12 formes différentes sont possibles :

- la première forme (1)
- les rotations à droite et à gauche de 60° et 120° de la 1^{ère} (4)
- la rotation de 180° de la 1^{ère} (1)
- la symétrie d'axe AEJOS (1)
- les rotations à droite et à gauche de 60° et 120° de la symétrie (4)
- la rotation de 180° de la symétrie (1)

Quelle que soit la transformation, la liste des 6 sommets de l'hexagone [A,C,L,S,Q,H] est toujours la même si on ne compte pas l'ordre des éléments. Il suffit donc d'imposer que A, par exemple, soit le plus petit entier des 6 sommets et on élimine alors toutes les rotations ; puis, pour la symétrie d'axe AEJOS, on aura C qui deviendra H et H deviendra C. Imposons alors $C > H$, et on élimine ainsi la symétrie.

Cette méthode apparaît dans les pages suivantes avec le nom `repart3` où j'ai fait une dernière modification : à chaque fois qu'on est sur le point de finir la construction d'une ligne, au lieu de prendre le dernier nœud au hasard dans la liste des entiers, il est plus rapide de dire à Prolog que le nœud est égal à 38 moins les autres nœuds de la ligne, en précisant qu'il doit être inférieur à 20.

Tests :

Repart2 :

```
?- repartition(Liste).
```

```
Liste = [3, 17, 18, 19, 7, 1, 11, 16, 2, 5, 6, 9, 12, 4, 8, 14, 10, 13, 15] ;
Liste = [3, 19, 16, 17, 7, 2, 12, 18, 1, 5, 4, 10, 11, 6, 8, 13, 9, 14, 15] ;
Liste = [9, 11, 18, 14, 6, 1, 17, 15, 8, 5, 7, 3, 13, 4, 2, 19, 10, 12, 16] ;
Liste = [9, 14, 15, 11, 6, 8, 13, 18, 1, 5, 4, 10, 17, 7, 2, 12, 3, 19, 16] ;
Liste = [10, 12, 16, 13, 4, 2, 19, 15, 8, 5, 7, 3, 14, 6, 1, 17, 9, 11, 18] ;
Liste = [10, 13, 15, 12, 4, 8, 14, 16, 2, 5, 6, 9, 19, 7, 1, 11, 3, 17, 18] ;
Liste = [15, 13, 10, 14, 8, 4, 12, 9, 6, 5, 2, 16, 11, 1, 7, 19, 18, 17, 3] ;
Liste = [15, 14, 9, 13, 8, 6, 11, 10, 4, 5, 1, 18, 12, 2, 7, 17, 16, 19, 3] ;
Liste = [16, 12, 10, 19, 2, 4, 13, 3, 7, 5, 8, 15, 17, 1, 6, 14, 18, 11, 9] ;
Liste = [16, 19, 3, 12, 2, 7, 17, 10, 4, 5, 1, 18, 13, 8, 6, 11, 15, 14, 9] ;
Liste = [18, 11, 9, 17, 1, 6, 14, 3, 7, 5, 8, 15, 19, 2, 4, 13, 16, 12, 10] ;
Liste = [18, 17, 3, 11, 1, 7, 19, 9, 6, 5, 2, 16, 14, 8, 4, 12, 15, 13, 10] ;
```

No

Environ 2 minutes pour arriver jusqu'au No.

Repart3 :

```
?- repartition(Liste).
```

```
Liste = [[3, 17, 18, 19, 7, 1, 11, 16, 2, 5, 6, 9, 12, 4, 8, 14, 10, 13, 15], [16, 19, 3, 12, 2, 7, 17, 10, 4, 5, 1, 18, 13, 8, 6, 11, 15, 14, 9], [18, 11, 9, 17, 1, 6, 14, 3, 7, 5, 8, 15, 19, 2, 4, 13, 16, 12, 10], [10, 12, 16, 13, 4, 2, 19, 15, 8, 5, 7, 3, 14, 6, 1, 17, 9, 11, 18], [9, 14, 15, 11, 6, 8, 13, 18, 1, 5, 4, 10, 17, 7, 2, 12, 3, 19, 16], [15, 13, 10, 14, 8, 4, 12, 9, 6, 5, 2, 16, 11, 1, 7, 19, 18, 17, 3], [10, 13, 15, 12, 4, 8, 14, 16, 2, 5, 6, 9, 19, 7, 1, 11, 3, 17, 18], [9, 11, 18, 14, 6, 1, 17, 15, 8, 5, 7, 3, 13, 4, 2, 19, 10, 12, 16], [3, 19, 16, 17, 7, 2, 12, 18, 1, 5, 4, 10, 11, 6, 8, 13, 9, 14, 15], [18, 17, 3, 11, 1, 7, 19, 9, 6, 5, 2, 16, 14, 8, 4, 12, 15, 13, 10], [15, 14, 9, 13, 8, 6, 11, 10, 4, 5, 1, 18, 12, 2, 7, 17, 16, 19, 3], [16, 12, 10, 19, 2, 4, 13, 3, 7, 5, 8, 15, 17, 1, 6, 14, 18, 11, 9]] ;
```

No

Environ 8 secondes pour arriver jusqu'au No.

Vous trouverez ci-joint les 3 codes `repart1`, `repart2` et `repart 3` et une disquette les regroupant.