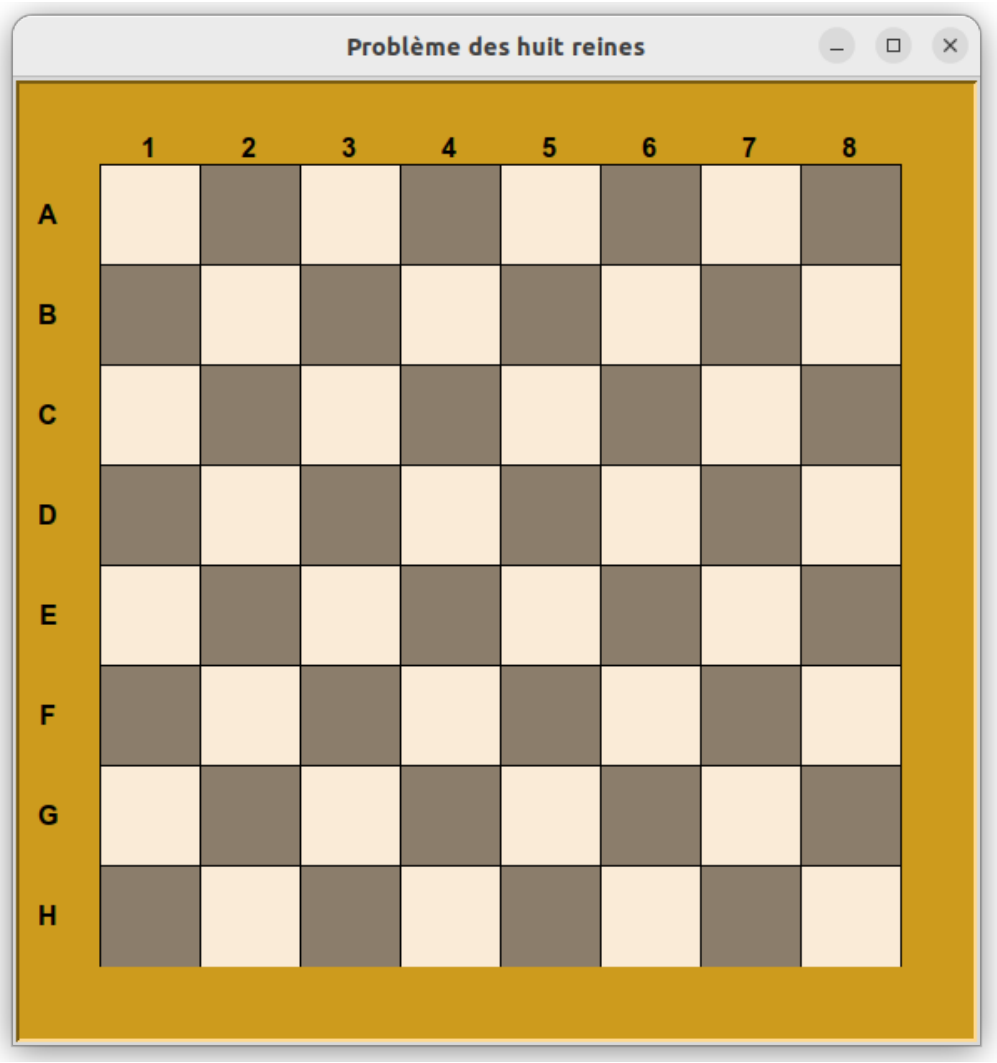
Le problème des huit reines

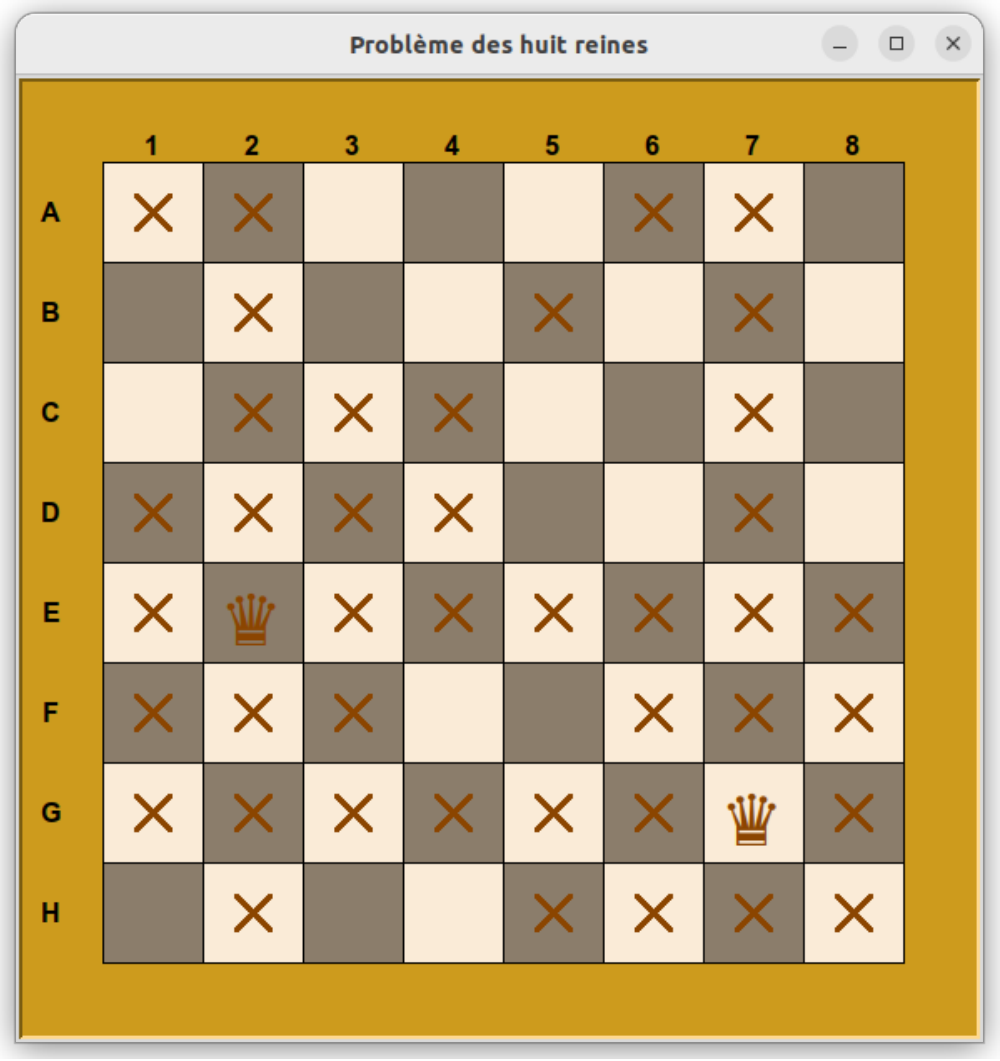
Le but du projet est d'écrire un programme permettant de résoudre le problème des huit reines (ou dames). Ce problème est un grand classique de la programmation récursive. Dans ce projet, on s'intéresse aussi à la réalisation d'une sortie graphique (avec `turtle`). Les trois premières étapes de ce projet sont **indépendantes**.

1. Etape 1 : visualisation

Réaliser en turtle, l'interface graphique permettant de représenter un échiquier comme ci-dessous :

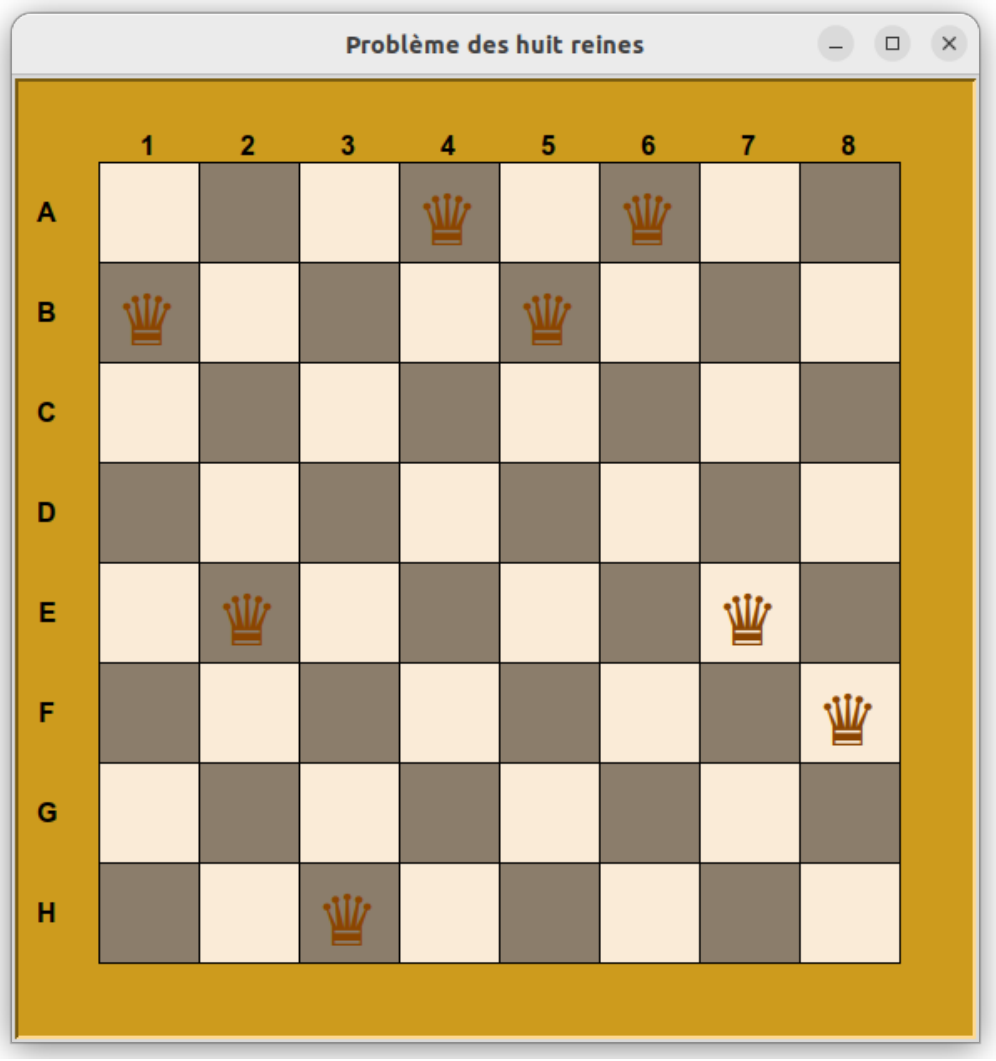


Ajouter la possibilité soit via un clic soit via un `textinput` de placer une reine sur cet échiquier. On peut éventuellement ajouter la possibilité de voir les cases menacées par une reine (en barrant ces cases). Dans l'illustation ci-dessous on a placé deux dames, l'une en **G1** et l'autre en **H7**, les cases menacées par ces reines portent une croix.



2. Etape 2 : résolution par force brute

On ne peut placer qu'une seule reine par colonne, on représente donc le placement de huit reines sur l'échiquier par une liste de huit entiers (les positions par colonne de chacune des huit reines). Par exemple la liste `[2, 5, 8, 1, 2, 1, 5, 6]` correspond à l'échiquier ci-dessous :



C'est à dire que le ième élément de la liste est la position de la reine de la colonne numéro i. Par exemples, la reine de la colonne 1 est situé sur la ligne 2, la reine de la colonne 2 sur la ligne 5. Justifier que les huit éléments de la liste doivent être différents pour que cette liste représente une solution du problème. En déduire qu'une solution est une permutation de la liste `[1, 2, 3, 4, 5, 6, 7, 8]`

L'idée de la résolution par force brute et donc de parcourir l'ensemble des permutations de la liste `[1, 2, 3, 4, 5, 6, 7, 8]` afin d'en extraire les solutions. Une permutation est une solution lorsqu'une deux reines ne sont pas sur la même diagonale de l'échiquier.

Aide

Pour générer les permutations de la liste `[1, 2, 3, 4, 5, 6, 7, 8]`, on pourra :

- dans un premier temps utiliser la fonction `permutations` du module `itertools`
- puis, écrire sa propre fonction. Pour cela penser récursif, et exprimer une permutation de `[1, 2, 3, 4, 5, 6, 7, 8]` comme l'un des éléments de cette liste suivie d'une permutation des éléments restants.

3. Etape 3 : résolution par *backtracking*

Faire des recherches sur un algorithme récursif appelé *backtracking* (en français : retour sur trace) qui permet de résoudre efficacement ce problème. On pourra par exemple consulter ce site

Programmer ce nouvel algorithme.

4. Etape 4 : échiquier de taille quelconque et comparaison

Le problème des 8 reines se généralise rapidement à un échiquier de dimensions $n \times n$ sur lequel on cherche à placer n reines sans qu'elles se menacent. Faire fonctionner les deux algorithmes (force brute et retour sur trace) en augmentant progressivement les valeurs de n. Que pouvez-vous en conclure ?

5. Etape 5 : aller plus loin

Résoudre par *backtracking* un problème similaire, par exemple le problème du cavalier