

# Ideas

## Générale

- Prendre la première pièce
- Mettre dans chaque case
- Voir la case de droite
  - Essayer de poser une des pièces
    - Tester avec chaque pièce
  - Si marche pour aucune pièce
    - Alors changer la première pièce
  - Si marche
    - passer à la suivante jusqu'à bloquer et faire marche arrière quand ça bloque
- S'arrêter quand full()
  - return True

```

bool estValide (int grille[9][9], LIST* position)
{
    nombre++;
    if (position == NULL)
        return true;

    int i = position->i;
    int j = position->j;
    for (k=0; k < 3; k++)
    {
        // Vérifie dans les tableaux si la valeur est déjà présente
        if ( ! existeSurLigne[i][k] && ! existeSurColonne[j][k] && ! existeSurBloc[3*(i/3)+(j/3)][k] )
        {
            // Ajoute k aux valeurs enregistrées
            existeSurLigne[i][k] = existeSurColonne[j][k] = existeSurBloc[3*(i/3)+(j/3)][k] = true;
            if ( estValide(grille, position->next) )
            {
                // Ecrit le choix valide dans la grille
                grille[i][j] = k+1;
                return true;
            }
            // Supprime k des valeurs enregistrées
            existeSurLigne[i][k] = existeSurColonne[j][k] = existeSurBloc[3*(i/3)+(j/3)][k] = false;
        }
    }
    return false;
}

```

## Plus précis

- **if** pos == end
  - return true
- **for** 0 -> 5\*5
  - **Si** (cas1 OR cas2 OR cas3) AND notUsed(piece) Alors
    - poserPiece(coordonées)
    - récursivité(nextPosition)
      - poser la pièce suivante
      - (RECURSION ICI)
    - **Si** la récursivité à échouer Alors
      - Retirer la pièce
      - Stop la branche
  - **Sinon**
    - Stop la branche

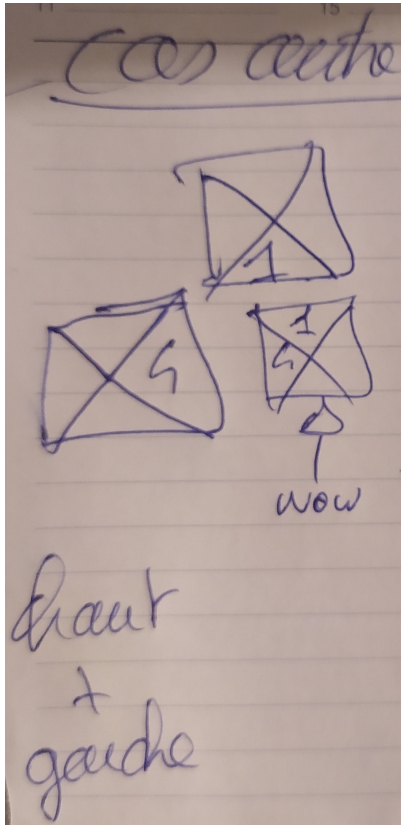
## Récursivité

- Passer en paramètre de la fonction récursive

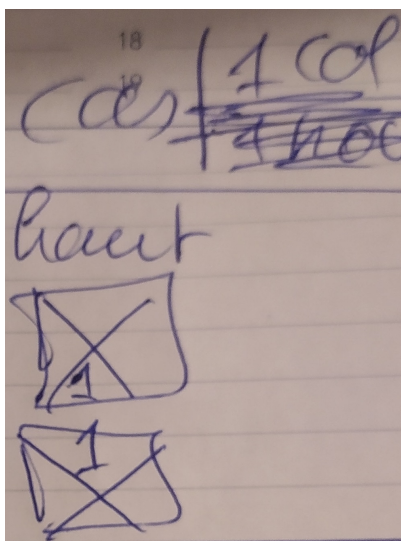
- La board
- Les coordonnées (x,y) du current empty
- Les available cards sont elles en Global et contiennent l'état d'utilisation.

## Différents cas

### Cas 1



### Cas 2



### Cas 3

