Voici une description de chaque fonction déjà développer

1. void PrintSodoku(GameInfo gameInfo) :

- Description : Cette fonction imprime la grille Sudoku avec la progression du joueur, y compris des informations telles que le nom du joueur, son âge, le niveau du jeu, la partie en cours et le nombre de points. Elle utilise une boucle pour formater et afficher la grille ainsi que les informations pertinentes.

2. bool IsSodoku(int SodokuMatrice[SIZE][SIZE], int ligne, int colone, int element) :

- Description : Cette fonction vérifie s'il est valide de placer l'élément spécifié à la ligne et à la colonne indiquées dans la grille Sudoku. Elle vérifie la ligne, la colonne et le sous-grille 3x3 pour s'assurer que le placement est valide.

3. bool SodokuSolution(int SodokuMatrice[SIZE][SIZE]) :

- Description : Cette fonction tente de résoudre le puzzle Sudoku en utilisant une approche de récursivité avec backtracking. Elle remplit les cases vides (0) avec des valeurs valides et continue jusqu'à ce qu'une solution soit trouvée ou déterminée comme impossible.

4. void GenerateDagonaleBlocks(int SodokuMatrice[SIZE][SIZE]) :

- Description : Cette fonction génère les blocs diagonaux initiaux pour la grille Sudoku. Elle remplit chaque bloc diagonal 3x3 avec des valeurs valides, en veillant à ce qu'il n'y ait pas de conflits.

5. void grillenonresolu(GameInfo *jeu) :

- Description : Cette fonction supprime un certain nombre de valeurs de la grille Sudoku en fonction du niveau de difficulté choisi. Elle randomise et efface des cellules pour créer une grille Sudoku non résolue.

6. bool solution(GameInfo jeu) :

- Description : Cette fonction vérifie si la grille Sudoku dans la structure `jeu` est entièrement résolue. Elle vérifie si toutes les cases de la grille contiennent des valeurs valides.

7. void printbravo(char bravo_message[], int index) :

- Description : Cette fonction imprime un message caractère par caractère, simulant une sortie à la machine à écrire avec un délai entre les caractères. Elle fournit également un message de chargement lorsque `index` est 0.

Description générale du code :

Ce code est une implémentation d'un jeu Sudoku en mode texte. Il permet aux joueurs de résoudre des puzzles Sudoku de différents niveaux de difficulté. Les fonctions du code gèrent la logique du jeu, génèrent des grilles Sudoku, vérifient les solutions et fournissent une interface conviviale pour les joueurs. Le code utilise la récursivité et le backtracking pour résoudre les puzzles. Il stocke également des informations sur le joueur, telles que le nom, l'âge, le niveau, la partie en cours et les points. Le code simule une sortie typewriter-like pour afficher les messages. En fin de compte, ce code crée une expérience de jeu où les joueurs peuvent interagir avec des puzzles Sudoku et suivre leur progression dans le jeu

---

Description des fonctions partie 2

---

1. void tovide(int SodokuMatrice[SIZE][SIZE]) :

- Description : Cette fonction initialise une matrice 2D `SodokuMatrice` avec des cellules vides (0) en parcourant l'ensemble de la matrice.

2. void printMatrix(int matrice[SIZE][SIZE]) :

- Description : Cette fonction imprime le contenu d'une matrice 2D sur la console.

3. char* create(char *folderName):

- Description : Cette fonction crée un répertoire portant le nom spécifié par `folderName` et renvoie le chemin vers le répertoire créé. Elle utilise des fonctions spécifiques à la plateforme pour gérer la création de répertoires.

4. void saveGameInfo(const char *folderPath, const GameInfo *gameInfo, int SodokuMatrice[SIZE][SIZE]) :

- Description : Cette fonction enregistre des informations liées au jeu, telles que le nom du joueur, l'âge, le niveau, la partie en cours, les points, et les matrices Sudoku, dans un fichier texte à l'intérieur du répertoire spécifié.

5. int loadGameInfo(const char *folderPath, const char *gameName, GameInfo *gameInfo, int SodokuMatrice[SIZE][SIZE])`:

- Description : Cette fonction charge les informations du jeu à partir d'un fichier sauvegardé précédemment. Elle remplit la structure gameInfo et les matrices Sudoku avec les données chargées.

6. void game(GameInfo jeu, int SodokuMatrice[SIZE][SIZE]) :

- Description : Cette fonction représente la boucle principale du jeu. Elle permet au joueur d'entrer des valeurs dans la grille Sudoku jusqu'à la résolution du puzzle. Elle gère également la sauvegarde du jeu, la révélation de solutions et la mise à jour de l'état du jeu.

7. void run(int sodokumatrice[SIZE][SIZE], char *folderName) :

- Description : Cette fonction est le point d'entrée pour démarrer ou continuer une partie. Elle invite l'utilisateur à choisir entre commencer une nouvelle partie ou reprendre une partie existante.

8. int main() :

- Description : La fonction main initialise le générateur de nombres aléatoires, crée un répertoire pour sauvegarder les données du jeu, initialise la grille Sudoku, puis appelle la fonction run pour démarrer ou reprendre une partie.

Description générale du code :

Ce code implémente un jeu Sudoku en mode texte. Il permet aux joueurs de jouer à des puzzles Sudoku, de sauvegarder et de charger leur progression, et de choisir entre commencer une nouvelle partie ou reprendre une partie existante. Les fonctions du code gèrent la logique du jeu, la sauvegarde des données du joueur et la manipulation des matrices Sudoku. Le code utilise des fonctions spécifiques à la plateforme pour créer des répertoires et gérer les fichiers. En fin de compte, ce code crée une expérience de jeu où les joueurs peuvent interagir avec des puzzles Sudoku, enregistrer leur progression et choisir différents niveaux de difficulté..