```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#define gridSize 4


/* Pour gagner :
    b - d - h - d - h - g -b -g - h - d - b - d - b - b - g - g - g - h - d - d - b - b
 */


int keepPlaying = 1;


int initialGrid[gridSize][gridSize] =
{
        {7,1,3,3},
        {7,4,4,5},
        {8,0,10,9},
        {11,2,12,6}
};



//Le tableau à obtenir.
int finalGrid[gridSize][gridSize] =
{
        {1,2,3,3},
        {4,4,5,6},
        {7,8,9,10},
        {7,11,12,0}
};


//Le tableau pour distinguer les cases paires
int backGrid[gridSize][gridSize];



// Cette structure nous permettra d'enregistrer les coordonnées d'une case
typedef struct
{
        int line;
        int col;
}Compartment;
```

```
// Pour enregistrer notre case vide en global, ainsi on pourra la changer à chaque fois

Compartment currentEmptyCompartment;



//initialiser notre tableau special
void initBack(int grid[gridSize][gridSize])
{
	for (int line = 0; line < gridSize; ++line)
	{
		for (int col = 0; col < gridSize; ++col)
		{
			if(initialGrid[line][col] == 3)
			{
				backGrid[line][col] = 1;
			}
			else if(initialGrid[line][col] == 4)
			{
				backGrid[line][col] = 2;
			}
			else if(initialGrid[line][col] == 7)
			{
				backGrid[line][col] = 3;
			}
			else
			{
				backGrid[line][col] = 0;
			}
		}
	}
}




//Afficher le plateau
void printGrid(int grid[gridSize][gridSize])
{
	printf("                    -------------------------\n");
	for (int line = 0; line < gridSize; ++line)
	{
		printf("                    |");
		for (int col = 0; col < gridSize; ++col)
		{
			if (col == 3)
			{
				if (grid[line][col] == 0)
				{
					printf("  ");
```

```
                        }
                        //pour que les cases a deux chiffre ne décale pas le plateau
                        else if (grid[line][col] >= 10)
                        {
                                printf("%d", grid[line][col]);
                        }

                        else
                        {
                                //Pour que le dernier élément de la ligne ne fasse pas de tab
après lui (pour un affichage ergonomique)
                                printf(" %d", grid[line][col]);
                        }
                }

                 // Pour que ça nous affiche une case vide
                else if (grid[line][col] == 0)
                {
                        printf("          ");
                }

                else
                {
                        printf(" %d     ", grid[line][col]);
                }
        }
        printf(" |");
        printf("\n");
    }
    printf("                    -------------------------\n");

}




// Rechercher la position de notre case vide pour l'enregistrer
void saveEmptyCompartment(int grid[gridSize][gridSize])
{
    for (int line = 0; line < gridSize; ++line)
    {
        for (int col = 0; col < gridSize; ++col)
        {
            // après avoir parcouru le tableau, lorsqu'on a trouvé la case vide, on
enregistre ses nouvelles coordonnées
            if (grid[line][col] == 0)
            {
                currentEmptyCompartment.line = line;
                currentEmptyCompartment.col = col;
            }
        }
    }
```

```
}



// Fonction permettant d'afficher les coordonnées d'une case
void printCompartment(Compartment x)
{
	printf("{%d, %d}\n", x.line,x.col);
}




// Fonction permettant de comparer 2 grilles
int compareGrid(int grid1[gridSize][gridSize], int grid2[gridSize][gridSize])
{
	// Renvoie 1 si les grilles sont identiques et 0 sinon
	int i = 0;

	for (int line = 0; line < gridSize; ++line)
	{
		for (int col = 0; col < gridSize; ++col)
		{
			if (grid1[line][col] == grid2[line][col])
			{
				i = 1;
			}
			else
			{
				return 0;
			}
		}

	}
	return i;
}




void movement()
{
	// On enregistre la case vide du début
	saveEmptyCompartment(initialGrid);
	char mov , toWin[22] =
{'B','D','H','D','H','G','B','G','H','D','B','D','B','B','G','G','G','H','D','D','B','B'};
	int score = 0;
	int i = 0;
```

```
	while (keepPlaying == 1)
	{
		score = score + 1;
		printf("\n		À toi de jouer : \n");
		printf("\n		PS : Pour gagner, je te propose d'appuyer sur %c \n",
toWin[i]);



		scanf("%c", &mov);

		// Si l’emplacement vide se trouve en haut de la pièce que tu veux déplacer : appuie
sur B pour la faire monter.

		if (mov == 'b' || mov == 'B')
		{

			// pour éviter que ça sorte de la grille
			if (currentEmptyCompartment.line != 3)
			{
				// si la pièce d'en bas est de taille 1*1
				if(backGrid[currentEmptyCompartment.line + 1]
[currentEmptyCompartment.col] == 0)
				{
					initialGrid[currentEmptyCompartment.line]
[currentEmptyCompartment.col] = initialGrid[currentEmptyCompartment.line + 1]
[currentEmptyCompartment.col];
					initialGrid[currentEmptyCompartment.line + 1]
[currentEmptyCompartment.col] = 0;
				}

				// Si la pièce d'en bas est de taille 1*2 verticalement
				else if (backGrid[currentEmptyCompartment.line + 1]
[currentEmptyCompartment.col] == 3)
				{
					initialGrid[currentEmptyCompartment.line]
[currentEmptyCompartment.col] = initialGrid[currentEmptyCompartment.line + 2]
[currentEmptyCompartment.col];
					initialGrid[currentEmptyCompartment.line + 2]
[currentEmptyCompartment.col] = 0;
				}
			}

		}

		// Si l’emplacement vide se trouve à gauche de la pièce que tu veux déplacer : appuie
sur d.
		else if (mov == 'd' || mov == 'D')
		{
```

```
                        // pour éviter que ça sorte de la grille
                        if (currentEmptyCompartment.col != 3)
                        {
                                // si la pièce de droite est de taille 1*1
                                if(backGrid[currentEmptyCompartment.line]
[currentEmptyCompartment.col + 1] == 0)
                                {
                                        initialGrid[currentEmptyCompartment.line]
[currentEmptyCompartment.col] = initialGrid[currentEmptyCompartment.line]
[currentEmptyCompartment.col + 1];
                                        initialGrid[currentEmptyCompartment.line]
[currentEmptyCompartment.col + 1] = 0;
                                }

                                // si la pièce de droite est de taille 1*2
                                else if ((backGrid[currentEmptyCompartment.line]
[currentEmptyCompartment.col + 1] == 1) || (backGrid[currentEmptyCompartment.line]
[currentEmptyCompartment.col + 1] == 2))
                                {
                                        initialGrid[currentEmptyCompartment.line]
[currentEmptyCompartment.col] = initialGrid[currentEmptyCompartment.line]
[currentEmptyCompartment.col + 2];
                                        initialGrid[currentEmptyCompartment.line]
[currentEmptyCompartment.col + 2] = 0;
                                }

                        }
                }

                // Si l'emplacement vide se trouve en bas de la pièce que tu veux déplacer : appuie
sur h pour la faire descendre.
                else if (mov == 'h' || mov == 'H')
                {

                        // pour éviter que ça sorte de la grille on vérifie si la case vide est tout en haut
de la grille
                        if (currentEmptyCompartment.line != 0)
                        {

                                // si la pièce d'en haut est de taille 1*1
                                if(backGrid[currentEmptyCompartment.line - 1]
[currentEmptyCompartment.col] == 0)
                                {
                                        initialGrid[currentEmptyCompartment.line]
[currentEmptyCompartment.col] = initialGrid[currentEmptyCompartment.line - 1]
[currentEmptyCompartment.col];
                                        initialGrid[currentEmptyCompartment.line - 1]
[currentEmptyCompartment.col] = 0;
                                }

                                // Si la pièce d'en haut est de taille 1*2 verticalement
```

```
                                else if (backGrid[currentEmptyCompartment.line - 1]
[currentEmptyCompartment.col] == 3)
                                {
                                        initialGrid[currentEmptyCompartment.line]
[currentEmptyCompartment.col] = initialGrid[currentEmptyCompartment.line - 2]
[currentEmptyCompartment.col];
                                        initialGrid[currentEmptyCompartment.line - 2]
[currentEmptyCompartment.col] = 0;
                                }
                        }
                }

                // Si l'emplacement vide se trouve à droite de la pièce que tu veux déplacer : appuie
sur D.
                else if (mov == 'g' || mov == 'G')
                {

                        // pour éviter que ça sorte de la grille
                        if (currentEmptyCompartment.col != 0)
                        {
                                // si la pièce de gauche est de taille 1*1
                                if(backGrid[currentEmptyCompartment.line]
[currentEmptyCompartment.col - 1] == 0)
                                {
                                        initialGrid[currentEmptyCompartment.line]
[currentEmptyCompartment.col] = initialGrid[currentEmptyCompartment.line]
[currentEmptyCompartment.col - 1];
                                        initialGrid[currentEmptyCompartment.line]
[currentEmptyCompartment.col - 1] = 0;
                                }

                                // si la pièce de gauche est de taille 1*2
                                else if ((backGrid[currentEmptyCompartment.line]
[currentEmptyCompartment.col - 1] == 1) || (backGrid[currentEmptyCompartment.line]
[currentEmptyCompartment.col - 1] == 2))
                                {
                                        initialGrid[currentEmptyCompartment.line]
[currentEmptyCompartment.col] = initialGrid[currentEmptyCompartment.line]
[currentEmptyCompartment.col - 2];
                                        initialGrid[currentEmptyCompartment.line]
[currentEmptyCompartment.col - 2] = 0;
                                }
                        }
                }


                else if (mov == 'e' || mov == 'E')
                {
                        keepPlaying = 0;
                        printf("                Tu pars déjà ? Dommage mais à bientôt ! \n");
                        break;
                }
```

```
                saveEmptyCompartment(initialGrid);
                initBack(backGrid);
                printGrid(initialGrid);


                if (compareGrid(initialGrid,finalGrid))
                {
                        printf("\n               Bravo, tu as réussi au bout de %d mouvement !\n", score);
                        keepPlaying = 0;
                }
                else
                {
                        while(getchar()!='\n'); // il faut vider le buffer pour réutiliser le scanf
                        i++;
                }

        }
}



int main()
{
        // Le Menu d'accueil

        printf("\n\n               BIENVENUE SUR QUEL TAQUIN !                    \n\n");
        printGrid(finalGrid);
        printf("\n         Je vais te mélanger cette grille et tu devra la remettre comme elle était.");
        printf("\n         Les commandes sont les suivantes : \n\n");
        printf("          ° Si tu veux déplacer la case vide en bas  : appuie sur B.");
        printf("\n        ° Si tu veux déplacer la case vide en haut : appuie sur H.");
        printf("\n        ° Si tu veux déplacer la case vide à droite: appuyerie sur D.");
        printf("\n        ° Si tu veux déplacer la case vide à gauche: appuie sur G.\n");
        printf("\n         Si tu veux abandonner, tu peux appuyer sur E pour sortir.\n          Bon
courage ! \n");

        printGrid(initialGrid);

        // On initialise la grille de fond
        initBack(backGrid);

        movement();

        return 0;
}
```