```c
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#include <string.h>

#define PION_A 'X'
#define PION_B 'O'
#define VIDE  '/'
#define INCONNU '/'
#define NBLIG 6
#define NBCOL 7
const int COLONNE_DEBUT = NBCOL/2;

typedef char Grille[NBLIG][NBCOL];

void initGrille(Grille tablo);
void afficher(Grille tablo, char pion, int colonne);
bool grillePleine(Grille tablo);
void jouer(Grille tablo, char pion, int *ligne,int *colonne);
int choisirColonne(Grille tablo, char pion, int colonne);
int trouverLigne(Grille tablo, int colonne);
bool estVainqueur(Grille tablo, int ligne, int colonne);
void finDePartie(char pion);

int main(){

char vainqueur;
int ligne, colonne;
Grille g;


initGrille(g);
vainqueur = INCONNU;
afficher(g, PION_A, COLONNE_DEBUT);
while ((vainqueur == INCONNU) && !(grillePleine(g))){
  jouer(g, PION_A, &ligne, &colonne);
  afficher(g, PION_B, COLONNE_DEBUT);
  if (estVainqueur(g, ligne,colonne)){
    vainqueur = PION_A;
  }else if(!grillePleine(g)){
    jouer(g, PION_B, &ligne, &colonne);
    afficher(g, PION_A, COLONNE_DEBUT);
    if (estVainqueur(g, ligne,colonne)){
      vainqueur = PION_B;
    }
  }

}
finDePartie(vainqueur);
}
```

```
void initGrille(Grille tablo){

int i, j;
for (i=0;i<NBLIG ; i++){
  for(j=0;j<NBCOL;j++){
    tablo[i][j]=VIDE;
  }
}
}


void afficher(Grille tablo, char pion, int colonne){

system("clear");

  for (int i = 0; i < colonne; i++){
    printf("  ");
  }
  printf("%3c\n", pion);
  printf("#|");

  for (int colonne = 0; colonne < NBCOL; colonne++){
    printf("%d|", colonne + 1);
  }
  printf("#\n");

  for (int i = 0; i < NBLIG; i++){
    printf("#|");
    for (int j = 0; j < NBCOL; j++){
        printf("%c|", tablo[i][j]);
    }

    printf("#\n");
  }
  printf("##");
  for (int colonne = 0; colonne < NBCOL; colonne++){
  printf("##");
  }
  printf("#\n");
}

bool grillePleine(Grille tablo){

bool max;
int ligne, colonne;
max = true;
ligne = 0;
colonne = 0;

  while (max && ligne < NBLIG){
    while (max && colonne < NBCOL){
```

```
      if (tablo[ligne][colonne] == VIDE){

        max = false;
      }

    colonne = colonne + 1;
    }
  ligne = ligne + 1;
  }
return max;
}


void jouer(Grille tablo, char pion, int *ligne, int *colonne){

  *colonne = choisirColonne(tablo, pion, COLONNE_DEBUT);
  *ligne = trouverLigne(tablo, *colonne);
  tablo[*ligne][*colonne] = pion;
}

int choisirColonne(Grille tablo, char pion, int colonne){
  char deplacement, entree;
  char rappel[60];
  int position = colonne;

  scanf("%c%c", &deplacement, &entree);
  while (deplacement != ' ' || trouverLigne(tablo, position) == -1){
    strcpy(rappel, "");
    switch (deplacement){
    case 'q':
      if (position > 0){
        position = position - 1;
      }
      break;
    case 'd':
      if (position < NBCOL - 1){
        position = position + 1;
      }
      break;
    case ' ':
      if (trouverLigne(tablo, position) == -1){
        strcpy(rappel, "Rejouer\n");
      }
      break;

    default:
      strcpy(rappel, "Jouer avec q (gauche), d (droite) et espace (poser un pion)\
n");
      break;
    }
    afficher(tablo, pion, position);
```

```
    printf("%s\n", rappel);
    scanf("%c%c", &deplacement, &entree);
  }
  return position;
}


int trouverLigne(Grille tablo, int colonne){

int ligne;
ligne = -1;

  while (ligne < NBLIG && tablo[ligne+1][colonne] == VIDE){

  ligne = ligne + 1;
  }
return ligne;
}

bool estVainqueur(Grille tablo, int ligne, int colonne){

int i = 0;
int avance = 0;
int victoire = 0;

  while (ligne+i < NBLIG && avance < 4 && tablo[ligne+i][colonne] == tablo[ligne]
[colonne]){
    i++;
    avance++;
  }
  if (victoire < avance){
    victoire = avance;
  }else{
    victoire = victoire;
  }
  avance = 1;
  i = 1;
  while (colonne+i < NBCOL && avance < 4 && tablo[ligne][colonne+i] == tablo[ligne]
[colonne]){
    i++;
    avance++;
  }

  i = 1;
  while (colonne-i >= 0 && avance < 4 && tablo[ligne][colonne-i] == tablo[ligne]
[colonne]){
    i++;
    avance++;
  }
  if (victoire < avance){
    victoire = avance;
```

```
  }else{
    victoire = victoire;
  }

  avance = 1;
  i = 1;
  while (colonne+i < NBCOL && ligne-i >= 0 && avance < 4 && tablo[ligne-i]
[colonne+i] == tablo[ligne][colonne]){
    i++;
    avance++;
  }

  i = 1;
  while (colonne-i >= 0 && ligne+i < NBLIG && avance < 4 && tablo[ligne+i][colonne-
i] == tablo[ligne][colonne]){
    i++;
    avance++;
  }
  if (victoire < avance){
    victoire = avance;
  }else{
    victoire = victoire;
  }
  avance = 1;
  i = 1;
  while (colonne-i >= 0 && ligne-i >= 0 && avance < 4 && tablo[ligne-i][colonne-i]
== tablo[ligne][colonne]){
    i++;
    avance++;
  }

  i = 1;
  while (colonne+i < NBCOL && ligne+i < NBLIG && avance < 4 && tablo[ligne+i]
[colonne+i] == tablo[ligne][colonne]){
    i++;
    avance++;
  }
  if (victoire < avance){
    victoire = avance;
  }else{
    victoire = victoire;
  }
  if(victoire >= 4){
    victoire =true;
  }else{
    victoire = false;
  }
return victoire;
}

void finDePartie(char pion){
```

```
  if (pion != VIDE){

    printf("Victoire des %c\n", pion);
  }else{

    printf("Match nul\n");
  }
}
```