Programmation Impérative I

Projet 2018 : Forteresse (n°20)

Nous présentons ici le projet du jeu Forteresse en langage C, réalisé par le binôme BERHOUMA Dhia Eldine et PAN Julien.

1 Introduction

Le jeu de Forteresse consiste à ce que deux joueurs s'affrontent sur un plateau carré. Chacun joue à tour de rôle. Jouer consiste à construire une tour sur une case. Un joueur peut jouer jusqu'à trois fois sur la même case de façon à obtenir une tour de hauteur 3. Toute tour posée donne une domination sur les cases adjacentes (y compris sur la case où elle est posée). Quand une tour d'un joueur est sur une case dominée majoritairement par l'autre joueur, elle est dominée et s'écroule.

2 Rapport

2.1 Présentation

Le travail demandé est de créer un programme en C qui permet de jouer au jeu Forteresse à deux, d'ajouter à ce programme le jeu par l'ordinateur, d'étudier et de définir des stratégies de gain et de pouvoir jouer contre l'ordinateur, mais que l'ordinateur joue bien.

2.2 Choix

Nous avons décidé de créer deux tableaux à deux dimensions, un tableau *char* pour y inscrire les forteresses des joueurs (a ou b) et un autre tableau *int* pour y mettre les hauteurs de chaque forteresse. Ces deux tableaux prennent en paramètre deux constantes : ligne et colonne qui reçoivent la même valeur afin d'obtenir un tableau carré. Nous avons donc été amenés à faire une structure pouvant contenir deux variables : un *int* pour la hauteur et un *char* pour le pion. Ces deux variables seront utilisées dans toutes les autres fonctions du programme pour manipuler les forteresses et les hauteurs.
Nous avons ensuite créer deux fonctions pour les règles, une qui vérifie si la hauteur d'une forteresse est supérieur à celle d'une forteresse adverse adjacente, et une autre qui vérifie si une forteresse est encerclée par deux forteresses adverse. Ces deux fonctions prennent en paramètre une ligne, une colonne, un même joueur et un joueur adverse.
Après les règles deux fonctions ont été créées, une pour le joueur a et une pour le joueur b qui servent à rentrer une forteresse dans le plateau et d'également empêcher au joueur de "mal jouer" : jouer sur une case en dehors du plateau, jouer sur une case déjà occupée par l'adversaire ou de dépasser la hauteur maximale d'une forteresse.
Une fonction ordinateur a été créée qui permet à l'ordinateur de jouer, celui-ci remplit de bas en haut les cases du plateau.
Une autre fonction qui passe d'une fonction à une autre dans les deux sens qui permet le changement de tour. Celle-ci est appelée dans le *main*. Enfin, une dernière fonction qui à été créée pour vérifier si le plateau a été remplit par un joueur, et ce dernier gagne donc la partie.

2.3 Les difficultés

Au début de la rédaction du programme, nous voulions créer un seul tableau qui pourrait prendre une tour et sa hauteur directement, nous nous sommes rendu compte qu'il en fallait deux. Après cela nous voulions manipuler les valeurs du tableau, nous avons donc voulu intégrer des pointeurs, ce qui nous a prit énormément de temps et de travail, mais cette méthode n'était pas la bonne solution. Nous nous sommes ensuite attaqués aux règles, nous voulions créer une variable qui prendrait comme valeur une certaine pression que chaque case produit en fonction de sa hauteur

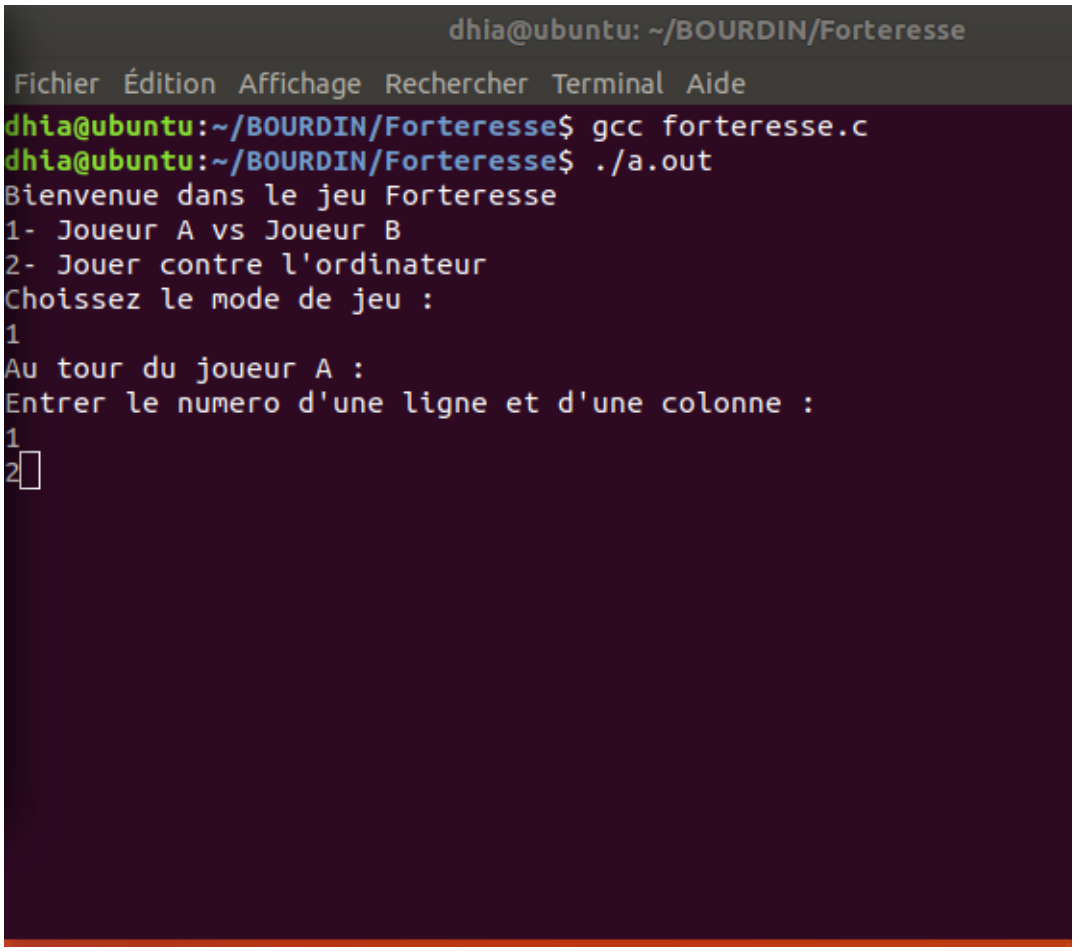
pour ensuite la comparer aux cases adjacentes. Manipuler cette variable aurait été plus facile mais nous n'avons pas réussi à la produire. Nous nous sommes donc contenter de définir tout les cas possibles lorsque le joueur pose sa forteresse, cette méthode pour créer les règles fut très contraignante et longue mais elle nous a permit d'avancer. Créer une fonction joueur au lieu de deux était également une étape qui nous a bloqué et, par faute de temps et d'incompréhension, nous sommes restés avec les deux fonctions. Enfin, ajouter un ordinateur n'a pas été facile, l'ordinateur qui joue sur ce programme présente beaucoup de problèmes et créer un ordinateur qui joue bien n'a pas été possible à cause du manque de temps qu'il nous restait. Chercher à régler les différents problèmes de ce programme fut laborieux et même à présent il en comprend encore beaucoup.

3 Mode d'emploi

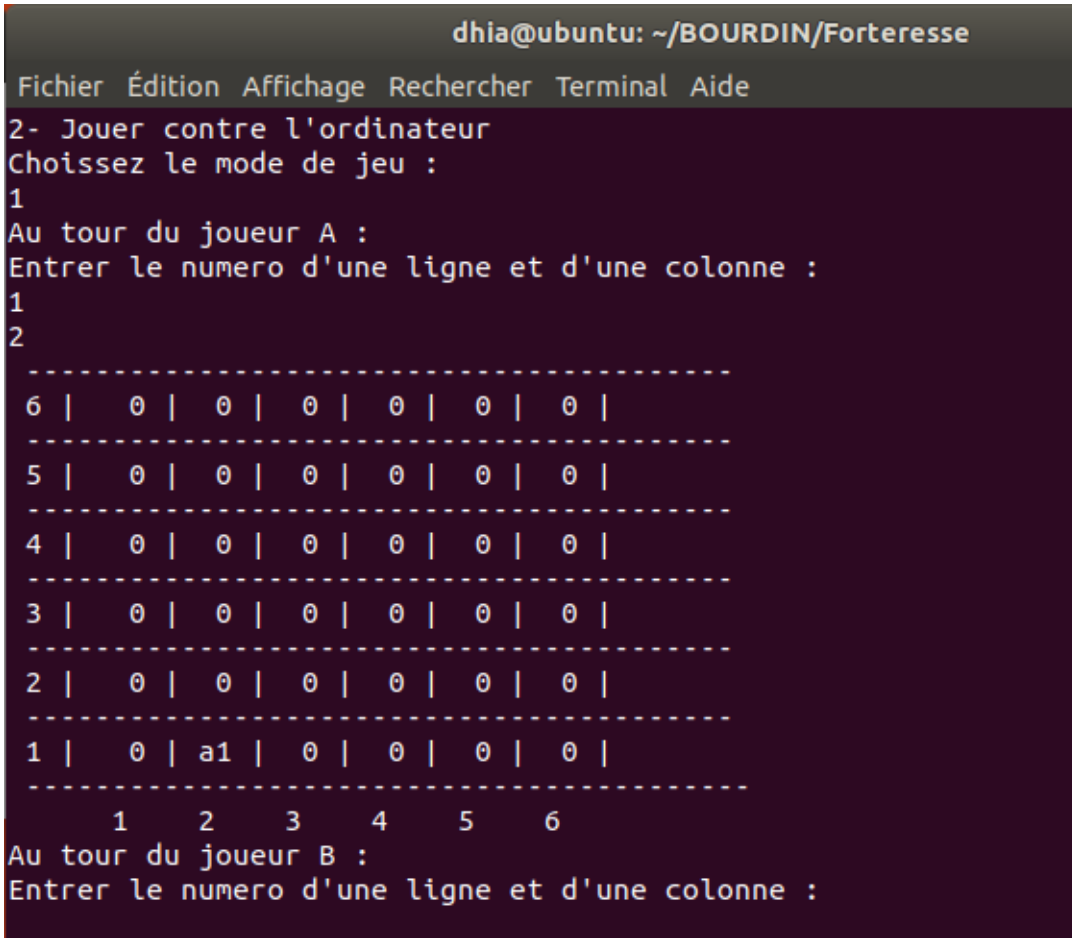
Lors de la compilation, un petit menu demandera à l'utilisateur de choisir entre deux modes de jeu, le jeu à deux joueurs et le jeu contre l'ordinateur. Dans le premier mode, le joueur A sera le premier à commencer la partie. Il lui sera alors demander de rentrer 2 valeurs, une pour la ligne et une pour la colonne, ces deux valeurs serviront à placer une forteresse, il devra appuyer sur entrer après chaque valeur qu'il entre. Ce sera ensuite le tour du joueur B qui devra faire de même et le programme continuera jusqu'à ce que l'un des deux joueurs gagne la partie. Pour le jeu contre l'ordinateur, l'ordinateur prendra la place du joueur B, il n'y aura que le joueur A qui rentrera ses valeurs comme dans l'autre mode et l'ordinateur jouera automatiquement après le joueur A.

4 Traces d'exécutions

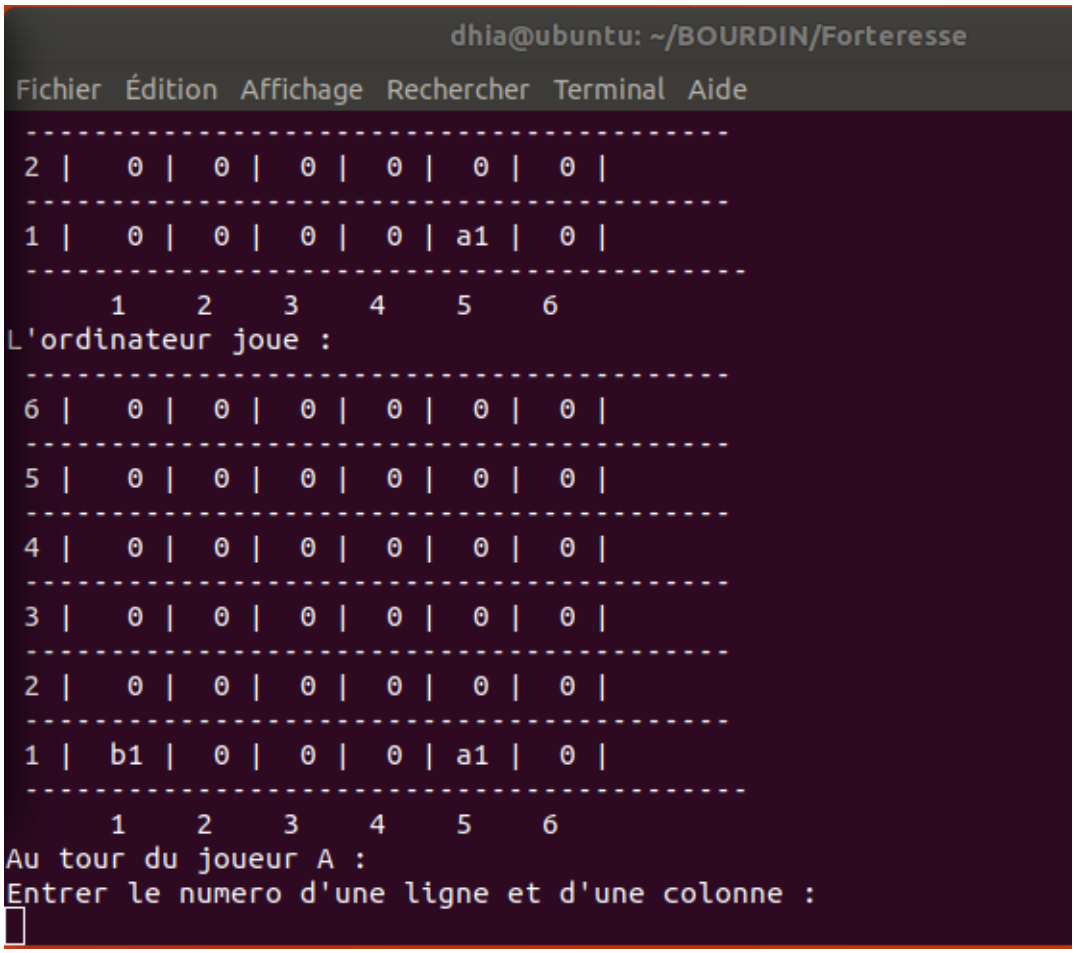
4.1 Joueur A vs Joueur B



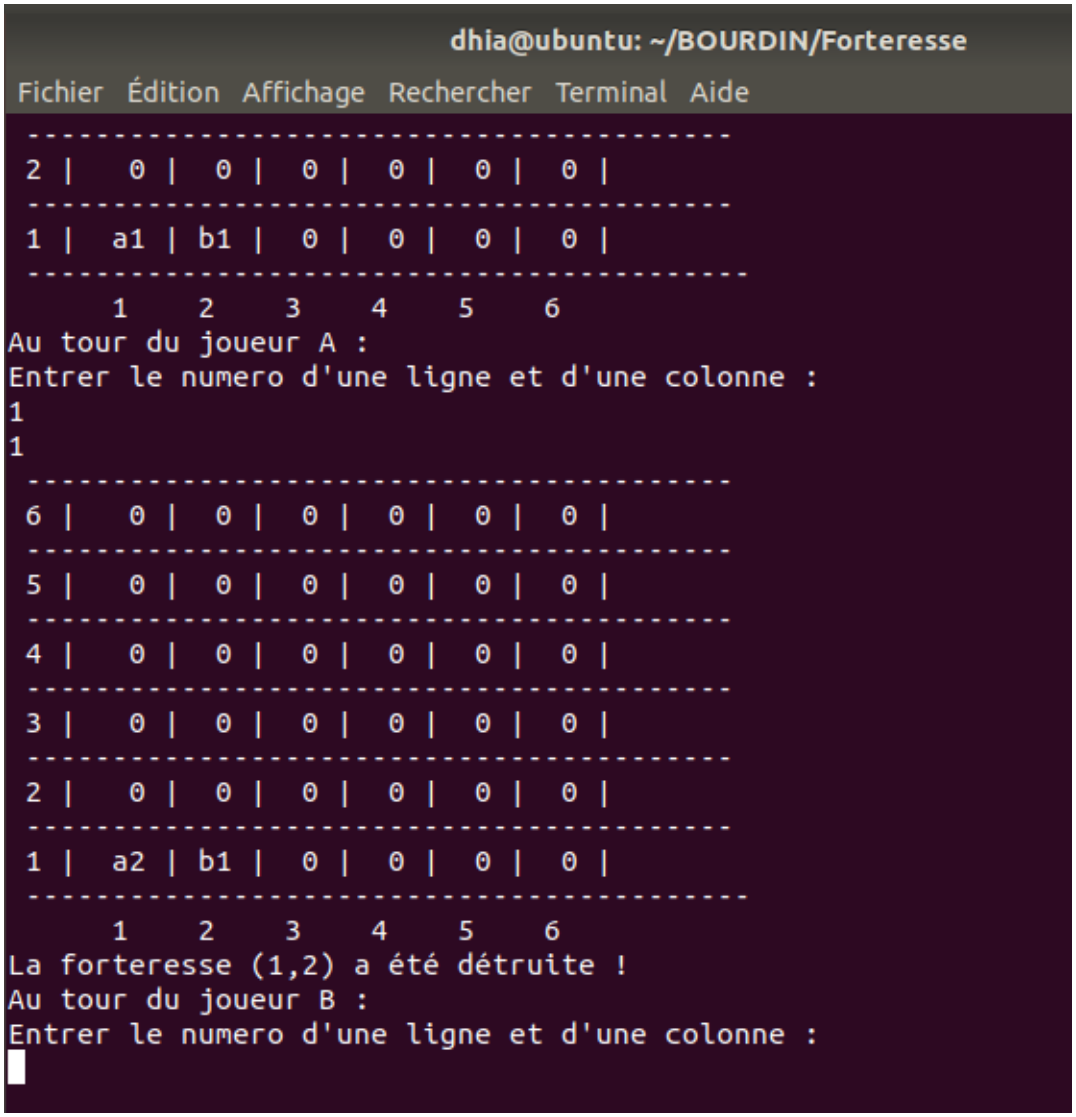
4.2 Le Joueur A joue



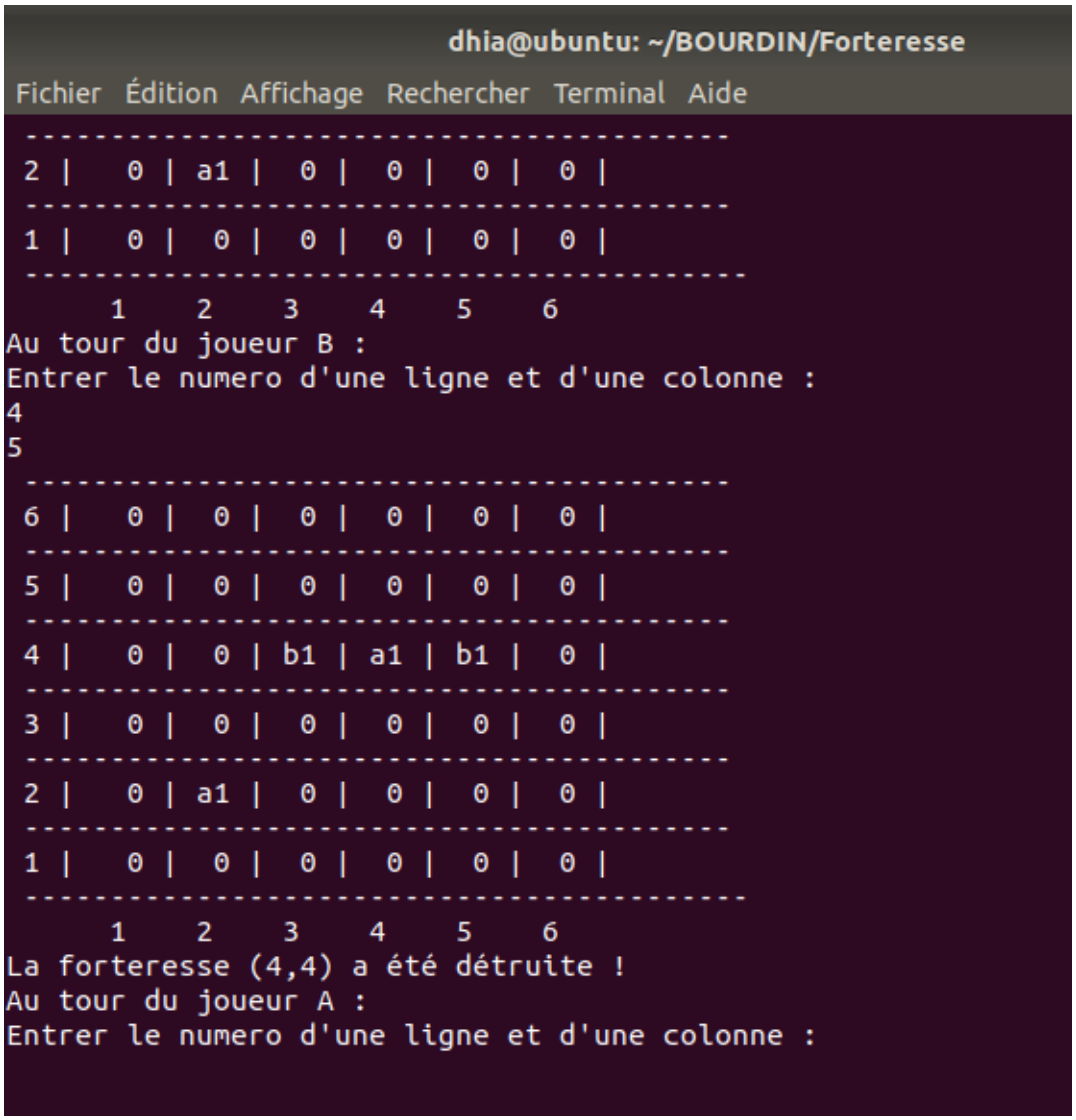
4.3 Le jeu contre l'ordinateur



4.4 Une forteresse qui domine une forteresse adverse



4.5 Une forteresse qui se fait encerclée



5 Conclusion

Au final, notre projet n'a pas été fini comme nous le souhaitions à cause des nombreuses difficultés que nous avons rencontré. Cependant, il nous a permit de développer notre expérience en matière de cohésion de groupe, de travail régulier et de programmation en C.

6 Code

```
#include <stdio.h>
#define ligne 6
#define colonne 6

char pion[ligne][colonne];
int hauteur[ligne][colonne];


struct CASE {
  char pion;
  int hauteur;
};

typedef struct CASE CASE;



void initialise(){

  int l,c;
  for(l=0;l<=ligne;l++){
    for(c=0;c<=colonne;c++){
      hauteur[l][c] = 0;
      pion[l][c] = ' ';
    }
  }

}



void plusgrand(int lgn, int col, CASE jadverse){

  CASE h;
  int hbas, hhaut, hgauche, hdroite;
  int pbas, phaut, pgauche, pdroite;

  h.hauteur = hauteur[lgn][col];

  hbas = hauteur[lgn-1][col];
  hhaut = hauteur[lgn+1][col];
  hgauche = hauteur[lgn][col-1];
  hdroite = hauteur[lgn][col+1];

  pbas = pion[lgn-1][col];
```

```
phaut = pion[lgn+1][col];
pgauche = pion[lgn][col-1];
pdroite = pion[lgn][col+1];



//pression pion gauche plus grande que soi
if(h.hauteur<hgauche && hgauche >= 1 && pgauche == jadverse.pion){
    hauteur[lgn][col] = 0;
    pion[lgn][col] = ' ';
    printf("La forteresse (%d,%d) a été détruite !\n",lgn,col);
}

//pression pion droite plus grande que soi
if(h.hauteur<hdroite && hdroite >= 1 && pdroite == jadverse.pion){
    hauteur[lgn][col] = 0;
    pion[lgn][col] = ' ';
    printf("La forteresse (%d,%d) a été détruite !\n",lgn,col);
}

//pression pion bas plus grande que soi
if(h.hauteur<hbas && hbas >= 1 && pbas == jadverse.pion){
    hauteur[lgn][col] = 0;
    pion[lgn][col] = ' ';
    printf("La forteresse (%d,%d) a été détruite !\n",lgn,col);
}



//pression pion haut plus grande que soi
if(h.hauteur<hhaut && hhaut >= 1 && phaut == jadverse.pion){
    hauteur[lgn][col] = 0;
    pion[lgn][col] = ' ';
    printf("La forteresse (%d,%d) a été détruite !\n",lgn,col);
}


//pression plus grande que pion gauche
if(h.hauteur>hgauche && hgauche >= 1 && pgauche == jadverse.pion){
    hauteur[lgn][col-1] = 0;
    pion[lgn][col-1] = ' ';
    printf("La forteresse (%d,%d) a été détruite !\n",lgn,col-1);

}

//pression plus grande que pion droite
if(h.hauteur>hdroite  && hdroite >=1 && pdroite == jadverse.pion){
    hauteur[lgn][col+1] = 0;
    pion[lgn][col+1] = ' ';
    printf("La forteresse (%d,%d) a été détruite !\n",lgn,col+1);

}
```

```
  //pression plus grande que pion bas
  if(h.hauteur>hbas && hbas >=1  && pbas == jadverse.pion){
      hauteur[lgn-1][col] = 0;
      pion[lgn-1][col] = ' ';
      printf("La forteresse (%d,%d) a été détruite !\n",lgn-1,col);

  }


  //pression plus grande que pion haut
  if(h.hauteur>hhaut && hhaut >=1  && phaut == jadverse.pion){
      hauteur[lgn+1][col] = 0;
      pion[lgn+1][col] = ' ';
      printf("La forteresse (%d,%d) a été détruite !\n",lgn+1,col);

  }
}



void encercle(int lgn, int col, CASE mj, CASE jadverse){



  int bas, haut, gauche, droite;
  int hhg, hbg, hhd, hbd;
  int hbas2, hhaut2, hdroite2, hgauche2;

  char pd, pg, ph, pb;
  char phg, pbg, phd, pbd;
  char pbas2,phaut2,pdroite2,pgauche2;



  hhg = hauteur[lgn+1][col-1];
  hbg = hauteur[lgn-1][col-1];
  hhd = hauteur[lgn+1][col+1];
  hbd = hauteur[lgn-1][col+1];

  phg = pion[lgn+1][col-1];
  pbg = pion[lgn-1][col-1];
  phd = pion[lgn+1][col+1];
  pbd = pion[lgn-1][col+1];

  ph = pion[lgn+1][col];
  pb = pion[lgn-1][col];
  pg = pion[lgn][col-1];
  pd = pion[lgn][col+1];


  bas = hauteur[lgn-1][col];
  hbas2 = hauteur[lgn-2][col];
  pbas2 = pion[lgn-2][col];
```

```
haut = hauteur[lgn+1][col];
hhaut2 = hauteur[lgn+2][col];
phaut2 = pion[lgn+2][col];

gauche = hauteur[lgn][col-1];
hgauche2 = hauteur[lgn][col-2];
pgauche2 = pion[lgn][col-2];




droite = hauteur[lgn][col+1];
hdroite2 = hauteur[lgn][col+2];
pdroite2 = pion[lgn][col+2];


//haut et gauche
if((hhg >= 1 && phg == mj.pion && ph == jadverse.pion ) && (hauteur[lgn+1][col] > 0)){

    hauteur[lgn+1][col] = 0;
    pion[lgn+1][col] = ' ';
    printf("La forteresse (%d,%d) a été détruite !\n",lgn+1,col);
}

//bas et gauche
if((hbg >= 1 && pbg == mj.pion && pb == jadverse.pion) && (hauteur[lgn-1][col] > 0)){

    hauteur[lgn-1][col] = 0;
    pion[lgn-1][col] = ' ';
    printf("La forteresse (%d,%d) a été détruite !\n",lgn-1,col);
}

//haut et droite
if((hhd >= 1 && phd == mj.pion && ph == jadverse.pion) && (hauteur[lgn+1][col] > 0)){

    hauteur[lgn+1][col] = 0;
    pion[lgn+1][col] = ' ';
    printf("La forteresse (%d,%d) a été détruite !\n",lgn+1,col);
}

//bas et droite
if((hbd >= 1 && pbd == mj.pion && pb == jadverse.pion) && (hauteur[lgn-1][col] > 0)){

    hauteur[lgn-1][col] = 0;
    pion[lgn-1][col] = ' ';
    printf("La forteresse (%d,%d) a été détruite !\n",lgn-1,col);
}




//haut et bas
if((hbas2 >= 1 && pbas2 == mj.pion && pb == jadverse.pion) && (hauteur[lgn-1][col] > 0)){
```

```
        hauteur[lgn-1][col] = 0;
        pion[lgn-1][col] = ' ';
        printf("La forteresse (%d,%d) a été détruite !\n",lgn-1,col);
    }


    //bas et haut
    if((hhaut2 >= 1 && phaut2 == mj.pion && ph == jadverse.pion) && (hauteur[lgn+1][col] > 0)){

        hauteur[lgn+1][col] = 0;
        pion[lgn+1][col] = ' ';
        printf("La forteresse (%d,%d) a été détruite !\n",lgn+1,col);
    }


    //droite et gauche
    if((hgauche2 >= 1 && pgauche2 == mj.pion && pg == jadverse.pion) && (hauteur[lgn][col-1] > 0))

        hauteur[lgn][col-1] = 0;
        pion[lgn][col-1] = ' ';
        printf("La forteresse (%d,%d) a été détruite !\n",lgn,col-1);
    }

    //gauche et droite
    if((hdroite2 >= 1 && pdroite2 == mj.pion && pd == jadverse.pion) && (hauteur[lgn][col+1] > 0)){

        hauteur[lgn][col+1] = 0;
        pion[lgn][col+1] = ' ';
        printf("La forteresse (%d,%d) a été détruite !\n",lgn,col+1);
    }

}




void majaff(int lgn, int col, CASE mj, CASE jadverse){

    int x,y;



    for (x = ligne; x >= 1; --x){
      printf(" ----------------------------------------");
      printf("\n %d | ", x);

      for (y = 1; y <= colonne; ++y){
        printf(" %c%d |",pion[x][y],hauteur[x][y]);
      }
        printf("\n");
    }

    printf(" ----------------------------------------");
```

```
  printf("\n");


   printf("    ");
  for (x = 1; x <= ligne; ++x){
    printf("  %d  ", x);
  }

  printf("\n");


  encercle(lgn,col,mj,jadverse);
  plusgrand(lgn,col,jadverse);

}




void joueurA(){

  int lgn, col;
  CASE joueura;
  CASE joueurb;
  joueura.pion = 'a';
  joueurb.pion = 'b';


  printf("Au tour du joueur A :\n");
  printf("Entrez le numéro d'une ligne et d'une colonne :\n");
  scanf("%d %d",&lgn, &col);


  if((col > colonne) || (col <= 0) || (lgn <= 0) || (lgn > ligne)){
    printf("Cette case n'existe pas ou est hors du plateau de taille %d x %d \n",ligne,colonne);
    joueurA();
    printf("Réessayez:\n");
  }



  if(pion[lgn][col] != joueurb.pion){

    if(hauteur[lgn][col] <= 2){
      pion[lgn][col] = joueura.pion;
      hauteur[lgn][col]++;

    }

    else{
      printf("La hauteur maximale a été atteinte\n");
      printf("Réessayez:\n");
      joueurA();
    }
```

```
  }
  else{
      printf("Cette case est déjà occupée par le joueur adverse.\n");
      printf("Réessayez:\n");
      joueurA();
    }
  majaff(lgn,col,joueura,joueurb);
}



void joueurB(){

  int lgn, col;
  CASE joueurb;
  CASE joueura;
  joueura.pion = 'a';
  joueurb.pion = 'b';

  printf("Au tour du joueur B :\n");
  printf("Entrez le numéro d'une ligne et d'une colonne :\n");
  scanf("%d %d",&lgn, &col);

  if((col > colonne) || (col <= 0) || (lgn <= 0) || (lgn > ligne)){
    printf("Cette case n'existe pas ou est hors du plateau de taille %d x %d \n",ligne,colonne);
    printf("Réessayez:\n");
    joueurB();
     }

     if(pion[lgn][col] != joueura.pion){

       if(hauteur[lgn][col] <= 2){
 pion[lgn][col] = joueurb.pion;
 hauteur[lgn][col]++;
       }

       else{
 printf("La hauteur maximale a été atteinte\n");
 printf("Réessayez:\n");
 joueurB();
       }
     }

       else{
 printf("Cette case est déjà occupée par le joueur adverse.\n");
 printf("Réessayez:\n");
 joueurB();
       }
     majaff(lgn,col,joueurb,joueura);
}



void ordinateur(int lgn, int col){
```

```
  CASE joueura,joueurb;
  joueura.pion = 'a';
  joueurb.pion = 'b';


    if(pion[lgn][col] != joueura.pion){

      if(hauteur[lgn][col] <= 2){
pion[lgn][col] = joueurb.pion;
hauteur[lgn][col]++;
      }
    }


  printf("L'ordinateur joue : \n");
  majaff(lgn,col,joueurb,joueura);
}



void findujeu(){


  int nbpiona, nbpionb;
  int lgn,col;


  for(lgn = 1; lgn <= ligne; ++lgn){
    for(col = 1; col <= colonne; ++col){
      if(pion[lgn][col] == 'a'){
nbpiona++;
      }
      if(pion[lgn][col] == 'b'){
nbpionb++;
      }
    }
  }

  if(nbpiona == ligne*colonne){
    printf("Le joueur A gagne la partie\n");
  }
  if(nbpionb == ligne*colonne){
    printf("Le joueur B gagne la partie\n");
  }
}



void jouer(){

  int i;

  //cette fonction sert aussi a changer de tour
```

```
  for(i=2;i<100;i++){
    if(i%2 == 0){
      joueurA();
    }
    else{
      joueurB();
    }
    findujeu();
  }
}


 void jouerordi(){

   int i,lgn,col;
   lgn = 1;
   col = 1;

  //cette fonction sert aussi a changer de tour

  for(i=2;i<100;i++){
    if(i%2 == 0){
      joueurA();
    }
    else{
      if(lgn > 6){
col++;
lgn = 1;
      ordinateur(lgn++,col);
      }
      else{
ordinateur(lgn++,col);
      }
    }
    findujeu();
  }
}




void choisir_jeu(int nb){

  printf("Bienvenue dans le jeu Forteresse\n");
  printf("1- Joueur A vs Joueur B\n");
  printf("2- Jouer contre l'ordinateur\n");
  printf("Choisissez un mode de jeu :\n");
  scanf("%d",&nb);

  if(nb == 1){
    initialise();
    jouer();
  }
```

```
  if(nb == 2){
    initialise();
    jouerordi();
  }

  if(nb != 1 || nb != 2){
    printf("Veuillez entrer une valeur valide\n");
    choisir_jeu(nb);
  }
}




int main(){
  int nb;
  choisir_jeu(nb);
}
```