Programmation Impérative – Projet 2016

Xuehui JIA

décembre 2016

1 Présentation du mon jeu

1.1 introduction du jeu

L e but de ce projet est d'implanter une plateforme de jeu de type morpion 3D.

C e jeu se présente sous la forme d'un tableau de taille n*n dans lequel sont placés des piles de jetons marquées par un X ou par un O. Initialement aucun jeton n'est présent. Le jeu se joue à deux joueurs, l'un possède les jetons X, l'autre les jetons O. Le but du jeu est de faire apparaître une ligne de n jetons de sa couleur.

1.2 Mes réalisations

J 'ai réalisé 4 modes pour ce jeu:
mode 1: le jeu en variante vue de dessus.
mode 2: le jeu en variante vue de dessus avec option séisme.
mode 3: le jeu en variante 3D.
mode 4: le jeu en variante 3D avec option séisme.

1.3 comment fonctionner ce jeu

D 'abord , on doit choisir le mode du jeu . L'interface est au dessous :
[jiaxuehui@localhost]$ ./projet
choisez un mode SVP:
2D—tapez 1
2D avec option séime—tapez 2
3D—tapez 3
3D avec option séisme—tapez 4

A près ,on doit taper un entier indiquant la taille du tableau . L'interface est au dessous :
[jiaxuehui@localhost]$ ./projet
choisez un mode SVP:
2D—tapez 1
2D avec option séime—tapez 2
3D—tapez 3
3D avec option séisme—tapez 4
2
tapez la taille du tableau
5
Dans ce exemple , on a choisi le mode 2 et un tableau de taille 5*5.

P our jouer ce jeu , il faut taper i, j, k et l pour déplace le curseur de sélection de la case respectivement vers le haut, la gauche, le bas et la droite. Et chaque fois qu'on choisit ,il faut taper la touche entrée . En dessous du tableau se trouvent :Une ligne de - de la largeur du tableau, En dessous une ligne indiquant de quel joueur c'est le tour, et un invite de choix d'action. Et aussi le contenu de la pile sur la case sélectionnée.

L'interface est dessous :

X . O . .
. . . . .
. . . . .
. . . . .
. . . . .

---

c'est tour de –O–,Quel est votre choix ?
le contenu de la pile sur la case sélectionnée('.'est le fond) X X .

2 La réalisation

2.1 Les structures

Un tableau 2D

J 'utilise un tableau 2D de taille (3*n)*(3*n), pour mettre les jetons , les '+' et les '*' autour les cases. Le contenu de chaque case est un pointeur vers un pile.

```
pile ** createTableau()
{
        int i,j;
        pile** t=(pile **)calloc(n*3,sizeof(pile*)) ;
        for(i=0;i<n*3;i++)
        {
                t[i]=(pile *)calloc(n*3,sizeof(pile));
        }
        for(i=0;i<3*n;i++)
        {
                for(j=0;j<n*3;j++)
                {
                        t[i][j]=createPile();
                        push('␣',&t[i][j]);
                }
        }
        return t;
}
```

Les piles

J 'utilise les piles pour mettre les jetons.

Je choisis le pile parce que la procédure de poser et de retirer un jeton est correspondant aux lesquelles d'un pile. En plus, les piles sont flexibles. C'est plus facile pour contrôler la hauteur et économiser le mémoire. Ici, j'utilise les piles statiques. Si besoin , on peut les changer aux piles dynamiques par "mailloc" facilement .

```
struct pile
{
        char val[TAILLE];
        int sommet;
};
typedef struct pile* pile;
```

2.2 Les fonctions

La fonction main

```
int main()
{
        printf(informations pour choisir un mode);
        scanf("%d",&m);
        while(m!=1&&m!=2&&m!=3&&m!=4)
        {
                printf("————————erreur————————————\n");

    /*si le joueur a choisi un mode qui n'exsite pas,erreur */
```

```
10              printf(informations pour choisir un mode);
11                  scanf("%d",&m);
12          }
13          printf("tapez la taille du tableau\n");
14          scanf("%d",&n);
15                  while(n>50||n<0)
16          {
17                  printf("tapez un entier entre 1-50 S.V.P");
18                  scanf("%d",&n);
19          }
20          printf("\n\n\n\n");
21          pile **t=createTableau();
22          int j,i;
23          for(i=1;i<3*n;)
24            /*mettre un '.' dans chaque case,faire le tableau plus clair */
25          {
26                  for(j=1;j<n*3;)
27                  {
28                  push('.',&t[i][j]);
29                          j=j+3;
30                  }
31                  i=i+3;
32          }
33          flag='X';
34          while(key!=QUITTER)
35          /*quand le joueur taper 'q',on quitte ce jeu*/
36          {
37                  t=poser(t,flag);
38                  alterJoueur();
39          }
40          system("clear");
41          printf("=======game over=====\n");
42          return 0;
43  }
```

L a ligne 3-12

Imprimer les informations pour les joueurs choisir un mode(1-4). Si les joueurs choisissent un mode qui n'exsite pas, donner un message d'erreur et redonner les informations de choisir.

L a ligne 13-19

Imprimer les informations pour les joueurs taper un entier indiquant la taille du tableau,et la ligne 17 crée un tableau de taille (3*n)*(3*n).

Vérifier est-ce que la taille est entre 1-50 , sinon ,répéter .

L a ligne 23-32

Mettre un '.' dans chaque case pour faire le tableau plus clair. Les cases avec un '.' sont pour mettre un jeton , les autres cases avec un espace sont pour mettre les '+' et les '*'.

L a ligne 34-41

Jouer ce jeu , jusqu'à les joueurs tapent un 'q', imprimer "===gameover===".

Dans cette procédure , la fonction poser (pile **t,char flag) est un tour du jeu , dans ce tour, les joueurs peuvent poser ou retirer un jeton . La fonction alterJoueur() est pour changer le joueur .

La fonction poser

```
pile ** poser (pile **t,char flag)
{
    int *g;int i;
    if(m==2||m==4)
        g=seisme(t);
    int x=1;int y=1;
        choisi(x,y,t,flag);
    int s=1;
    while(s)
    {
        scanf("%1s",buffer);
        key=buffer[0];
        switch(key)
        {
                case GAUCHE:
                        if(y>=3)
                        {
                                y=y-3;
                                choisi(x,y,t,flag);
                        }
                        break;
                case DROITE:
                        if(y<3*n-3)
                        {
                                y=y+3;
                                choisi(x,y,t,flag);
                        }
                        break;
                case BAS:
                        if(x<3*n-3)
                        {
                                x=x+3;
                                choisi(x,y,t,flag);
                        }
                        break;
                case HAUT:
                        if(x>=3)
                        {
                                x=x-3;
                                choisi(x,y,t,flag);
                        }
                        break;
                case RETIRER:
                if(t[x][y]->val[t[x][y]->sommet]!='.')
                        {
                                pop(&t[x][y]);
                                if(m==1||m==2)
                                        gagner(t,x,y);
                                else
                                        gagner3D(t,x,y);
                                system("clear");
                                choisi(x,y,t,flag);
                                printf("\n vous avez retire un jeton");
                                break;
```

```
55                              }
56                      else
57                              printf("vous ne pouvez pas retirer un jeton");
58 //si t[x][y]->val[t[x][y]->sommet]=='.',on ne peut pas retirer un jeton
59                                      break;
60                      case POSER:
61                              s=0;
62                              break;
63                      case QUITTER:
64                              s=0;
65                              break;
66              }
67      }
68
69      if(flag=='X')
70              push('X',&t[x][y]);
71      else
72              push('O',&t[x][y]);
73      affiche(t);
74      system("clear");
75      if(m==1||m==2)
76          gagner(t,x,y);
77      if(m==4||m==3)
78              gagner3D(t,x,y);
79      if(m==2||m==4)
80      {
81              if(g[0]>1)
82              {
83                      for(i=1;i<g[0];i=i+2)
84                      {
85                              deletoile(g[i],g[i+1],t);
86                      if(t[g[i]][g[i+1]]->val[t[g[i]][g[i+1]]->sommet]!='.')
87                                      gagner(t,g[i],g[i+1]);
88                      }
89              }
90      }
91      return t;
92 }
```

L a ligne 4-5

Avant commencer ce tour , vérifier est-ce que ce mode avec l'option séisme , si oui ,exécuter la fonction séisme (), et retour un tableau contenant des coordonnées des cases qui s'effondrent.

L a ligne 6-7

L'initiation . A chaque changement de tour,on commence toujours à t[1][1].

L a ligne 8-67

s=1,mais quand le joueur tape 'q' ou 'p', s=0, le boucle while se termine

Dans cette boucle ,le joueur choisit une case par taper "i,l,k,j", chaque fois il choisit , exécuter la fonction choisir () pour ajouter les étoiles autour la case.

Quand le joueur tape r, vérifier est-ce qu'il y a jeton , sinon , imprimer "erreur ". S'il a retiré un jeton ,vérifier si une des conditions de victoire(ligne 47-59).

Quand le joueur tape p, ajouter un jeton dans le sommet du pile, vérifier si une des conditions de victoire.

Quand le joueur tape q, terminer la boucle dans la fonction main , c'est la fin du jeu.

L a ligne 69-73
Poser un jeton 'X' ou 'O'

L a ligne 75-78
Vérifier si une des conditions de victoire par rapport à la mode sélectionné .

L a ligne 79-90
Supprimer les étoiles et verifier des conditions de victoire après effondrer

La fonction gagner 3D

C ette fonction est pour vérifier les conditions de victoire (la mode 3D)
Il est composé par 4 parties:
— n jetons identiques consécutifs dans une même pile ;
— une ligne, une colonne ou une diagonale sur un même niveau ;
— une ligne "en escalier montant " : sur une ligne, une colonne, ou une diagonale, on a les mêmes jetons en augmentant de 1 le niveau entre chaque pile. Exemple vu de côté, pour n = 5 :
— idem pour un ligne en escalier descendant.

partie 1:

```
1  if(t[x][y]->sommet>=n+1)
2  {
3  cnt=0;
4  for(k=2;k<(t[x][y]->sommet);k++)
5  {
6              if(t[x][y]->val[k]==t[x][y]->val[t[x][y]->sommet])
7                      cnt++;
8              if(t[x][y]->val[k+1]!=t[x][y]->val[t[x][y]->sommet])
9                      cnt=0;
10 }
11 if(t[x][y]->val[t[x][y]->sommet]==t[x][y]->val[t[x][y]->sommet])
12     cnt++;
13 if(cnt>=n)
14             printf("**%c a gagne !!!!!**", t[x][y]->val[t[x][y]->sommet]);
15 }
```

Parcourir tout les jetons dans un pile (t[x][y]->val[0]==' ',t[x][y]->val[1]=='.',donc k commence à 2)

L a ligne 1
Si la hauteur du pile supérieur ou égale n.

L a ligne 4-10
cnt est le nombre des piles identités , si il apparaît un jeton différent, cnt=0;

L a ligne 11-15
Si cnt >= n, le joueur a gagné.

partie 2:

```
for(i=1;i<3*n;i=i+3)                    /*niveau————horizontal*/
{
        if(t[x][y]->val[t[x][y]->sommet]=='X'||
                        t[x][y]->val[t[x][y]->sommet]=='O')
        {
                if(t[x][i]->val[t[x][y]->sommet]==
                                t[x][y]->val[t[x][y]->sommet])
                        cnt++;
                else
                        break;
        }
}
if(cnt==n)
        printf("***%c a gagne!!!!***",t[x][y]->val[t[x][y]->sommet]);
cnt=0;
for(i=1;i<3*n;i=i+3)                    /*niveau————verticale*/
{
        if(t[x][y]->val[t[x][y]->sommet]=='X'||
                        t[x][y]->val[t[x][y]->sommet]=='O')
        {
        if(t[i][y]->val[t[x][y]->sommet]==
                        t[x][y]->val[t[x][y]->sommet])
                cnt++;
        else
                break;
        }
}
if(cnt==n)
        printf("***%c a gagne!!!!***",t[x][y]->val[t[x][y]->sommet]);
cnt=0;
for(i=1;i<3*n;i=i+3)                    /*niveau————diagonale augmenter */
{
        if(t[x][y]->val[t[x][y]->sommet]=='X'||
                        t[x][y]->val[t[x][y]->sommet]=='O')
        {
        if(t[i][i]->val[t[x][y]->sommet]==
                        t[x][y]->val[t[x][y]->sommet])
                cnt++;
        }
}
if(cnt==n)
        printf("***%c a gagne!!!***",t[x][y]->val[t[x][y]->sommet]);
cnt=0;                          /*niveau————diagonale descendre */
for(i=3*n-2,j=1;j<3*n;i=i-3,j=j+3)
{
        if(t[x][y]->val[t[x][y]->sommet]=='X'||
                        t[x][y]->val[t[x][y]->sommet]=='O')
        {
        if(t[i][j]->val[t[x][y]->sommet]==
                        t[x][y]->val[t[x][y]->sommet])
                cnt++;
        }
}
if(cnt==n)
        printf("***%c a gagne!!!***",t[x][y]->val[t[x][y]->sommet]);
```

Cette partie est ressemble avec la fonction gagne(pile **t,int x,int y), mais sur un même niveau.

L a ligne 1-15
Une ligne horizontal de n jetons de sa couleur.

L a ligne 16-29
Une ligne verticale de n jetons de sa couleur.

L a ligne 30-42
Une diagonale augmenter de n jetons de sa couleur.

L a ligne 43-55
Une diagonale descendre de n jetons de sa couleur.

partie 3:

```
1  cnt=0;
2  for(i=1,j=t[x][y]->sommet-(y/3);i<3*n;i=i+3,j++)
3  {
4          if(t[x][i]->val[j]==t[x][y]->val[t[x][y]->sommet])
5                  cnt++;
6  }
7  if(cnt==n)
8          printf("**%c a gagne !!!!**",t[x][y]->val[t[x][y]->sommet]);
9  cnt=0;
10 for(i=1,j=t[x][y]->sommet-(x/3);i<3*n;i=i+3,j++)
11 {
12         if(t[i][y]->val[j]==t[x][y]->val[t[x][y]->sommet])
13                 cnt++;
14 }
15 if(cnt==n)
16         printf("**%c a gagne !!!!**",t[x][y]->val[t[x][y]->sommet]);
17 cnt=0;
18 for(i=1,j=t[x][y]->sommet-(y/3);i<3*n;i=i+3,j++)
19 {
20         if(t[i][i]->val[j]==t[x][y]->val[t[x][y]->sommet])
21                 cnt++;
22 }
23 if(cnt==n)
24         printf("**%c a gagne !!!!**",t[x][y]->val[t[x][y]->sommet]);
25 cnt=0;
26 for(i=1,k=3*n-2,j=t[x][y]->sommet-(y/3);i<3*n;i=i+3,j++,k=k-3)
27 {
28         if(t[k][i]->val[j]==t[x][y]->val[t[x][y]->sommet])
29                 cnt++;
30 }
31 if(cnt==n)
32         printf("**%c a gagne !!!!**",t[x][y]->val[t[x][y]->sommet]);
```

L a ligne 1-8
Une ligne "en escalier montant "(x constant).

L a ligne 9-16
Une ligne "en escalier montant "(y constant).

L a ligne 17-24
Une ligne "en escalier montant "(diagonale montant–"/").

L a ligne 25-32
Une ligne "en escalier montant (diagonale montant–"\").
partie 4:

```
cnt=0;
for(i=1,j=t[x][y]->sommet+(y/3);i<3*n;i=i+3,j--)
{
        if(t[x][i]->val[j]==t[x][y]->val[t[x][y]->sommet])
                cnt++;
}
if(cnt==n)
        printf("**%c a gagne !!!!**",t[x][y]->val[t[x][y]->sommet]);
cnt=0;
for(i=1,j=t[x][y]->sommet+(x/3);i<3*n;i=i+3,j--)
{
        if(t[i][y]->val[j]==t[x][y]->val[t[x][y]->sommet])
                cnt++;
}
if(cnt==n)
        printf("**%c a gagne !!!!**",t[x][y]->val[t[x][y]->sommet]);
cnt=0;
for(i=1,j=t[x][y]->sommet+(y/3);i<3*n;i=i+3,j--)
{
        if(t[i][i]->val[j]==t[x][y]->val[t[x][y]->sommet])
                cnt++;
}
if(cnt==n)
        printf("**%c a gagne !!!!**",t[x][y]->val[t[x][y]->sommet]);
cnt=0;
for(i=1,k=3*n-2,j=t[x][y]->sommet+(y/3);i<3*n;i=i+3,j--,k=k-3)
{
        if(t[k][i]->val[j]==t[x][y]->val[t[x][y]->sommet])
                cnt++;
}
if(cnt==n)
        printf("**%c a gagne !!!!**",t[x][y]->val[t[x][y]->sommet]);
}
```

L a ligne 1-8
Une ligne "en escalier descendre "(x constant).

L a ligne 9-16
Une ligne "en escalier descendre "(y constant).

L a ligne 17-24
Une ligne "en escalier descendre "(diagonale descendre–"/").

L a ligne 25-32
Une ligne "en escalier descendre (diagonale descendre–"\").

les autres fonctions

pile createPile()
void push(char c,pile *p)
char pop(pile *p)
void affichePile(pile p)
pile ** createTableau()
void affiche(pile **t)
void alterJoueur()
void choisi (int x,int y,pile **t,char flag)
void gagner(pile **t,int x,int y)
void etoile (int x,int y,pile **t)
void deletoile (int x,int y,pile **t)
int * seisme (pile **t)
void gagner3D(pile **t,int x,int y)

3 Limites

1.Dans le programme , les piles sont statiques avec un constant TAILLE, donc le nombre du jeton dans un pile est limité .

2.Quand le joueur tape la taille du tableau, il faut forcément taper un entier.

3.Quand le joueur tape la taille du tableau, il faut taper un entier entre 1-50, si supérieur de 50, le terminal n'est pas assez large.

4 Les problèmes et les solutions

1. Quand'on vérifie les conditions de victoire, j'ai utilité le stratégie au dessous :

```
for(i=1;i<3*n;i=i+3)//horizontal
{
    if(t[x][i]->val[t[x][i]->sommet]==
                t[x][y]->val[t[x][y]->sommet])
            cnt++;
        else
            break;
}
```

Mais il souvent imprime que quelqu'un a gagné , c'est parce que ". . . . ." est un des situation de victoire , donc j'ai ajouter un condition au dessous :

```
for(i=1;i<3*n;i=i+3)                    /*horizontal*/
{
        if(t[x][y]->val[t[x][y]->sommet]=='X'||
                t[x][y]->val[t[x][y]->sommet]=='O')
        {
            if(t[x][i]->val[t[x][i]->sommet]==
                    t[x][y]->val[t[x][y]->sommet])
                cnt++;
            else
                break;
        }
}
```

2.Il a existé des erreur quand'on retirer un jeton ,c'est parce que quand il n'y a aucun jeton , si on continu de retirer les jeton , il aura des erreurs.

donc j'ai ajouter un condition au dessous :

```
case RETIRER:
        if(t[x][y]->val[t[x][y]->sommet]!='.')
        {

        }
        else
        printf("vous ne pouvez pas retirer un jeton");
        break;
```

5 Conclusion

J 'ai réalisé tous les questions, ce travail m'aide de mieux comprendre la programmation impérative. J'ai encore quelques chose à améliorer , il faut simplifier les codes, et utiliser plus d'interfaces. c'est très important pour les projets grands. Et aussi s'améliorer sur le rapport et la langue.