**一、設計思路**

**1、游戲規則**

想要制作游戲，首先需要了解游戲的規則，下面就來介紹2048的游戲規則

2048游戲共有16個格子，初始時初始數字由2或者4構成。

* 手指向一個方向滑動，所有格子會向那個方向運動。
* 相同數字的兩個格子，相撞時數字會相加。
* 每次滑動時，空白處會隨機刷新出一個數字的格子。
* 當界面不可運動時（當界面全部被數字填滿時），游戲結束；當界面中最大數字是2048時，游戲勝利。

**2、思路**

了解了游戲規則接下來就好辦多了，規則中最重要的是：向某個方向滑動格子就會向哪兒滑，並且相撞的數字相加

根據最重要的規則，我偷了一個懶，想到了以下策略

* 首先完成向左滑動功能
* 向右滑動的功能可以借助向左滑動的功能
  + 將游戲界面逆時針旋轉180°
  + 向左滑動
  + 還原界面，將界面逆時針旋轉180°，歸回原位
* 向上滑動的功能可以借助向左滑動的功能
  + 將游戲界面逆時針旋轉270°
  + 向左滑動
  + 還原界面，將界面逆時針旋轉90°
* 向下滑動的功能可以借助向左滑動的功能
  + 將游戲界面逆時針旋轉90°
  + 向左滑動
  + 還原界面，將界面逆時針旋轉270°

通過以上分析，可以得知向左滑動的功能是最重要，其他方向都可以借助它來完成，因此接下來將分析怎么完成向左滑動的功能

* 游戲中說共有4行4列16個格子，有4行我們就可以采用循環，一次合並一行，總共4個循環，所以向左滑動又可以分解為向左合並一行
* 向左合並，首先不管有沒有相同的數字，都要將所有的數字緊湊到最左邊，兩個數字中間不能有空
* 最后采用合並算法，將一行中相同的數字合並

至此這個游戲的基本就可以設計好了，下面開始實現

**二、代碼實現**

**1、存儲結構**

根據游戲規則很容易想到使用二維數組來存儲游戲數據，如下

int data[4][4];

**2、初始化游戲數據**

規則說初始時初始數字由2或者4構成，這里偷個懶，設置所有的初始數字都是2，接下來要解決的就是初始數字的位置坐標，所以獲取隨機左邊的函數就行。

獲取隨機數可以使用以下函數

//獲取隨機數

void randArray(int a[], int n) {

for (int i = 0; i < n; i++) {//產生隨機坐標

srand((unsigned)time(NULL) + rand() + i);

a[i] = rand() % 4;

}

}

要產生不一樣的隨機數，首先需要置不一樣的隨機數種子，置隨機數種子可以采用srand()函數，然后我們把當前時間作為參數也就是srand((unsigned)time(NULL))，但是在后期時16個坐標產生相同的隨機坐標的幾率非常大，而計算機的速度又非常快，相同的隨機種子產生的隨機數序列是一樣的，這樣隨機性不高，很容易產生相同的坐標，所以參數還要加上rand()+i，也就是srand((unsigned)time(NULL) + rand() + i)，不信的同學可以試試按之前的寫法游戲玩到后期每產生一塊新的坐標都需要不少時間。

接着按以下寫法就可以產生兩個不一樣的隨機坐標

//初始化數據，initNum初始數字

void init(int data[4][4], int initNum) {

int random[4];

randArray(random, 4);

if (random[0] == random[2] && random[1] == random[3]) {

init(data, 2);//若坐標相同,則重新生成坐標

} else {

data[random[0]][random[1]] = initNum;

data[random[2]][random[3]] = initNum;

}

}

**3、向左合並**

按照之前的思路，首先要將所有的方塊移到最右側且中間不能有空，可以通過如下代碼實現

//緊湊數組(向左就湊)

void compact(int data[4]) {

int i, j = 0;

for (i = 0; i < 4; i++) {

if (data[i] != 0) {

data[j] = data[i];

j++;

}

}

for (; j < 4; j++)data[j] = 0;

}

接着完成向左合並函數

//向左合並

void mergerLeft(int data[4][4]) {

for (int i = 0; i < 4; i++) {

compact(data[i]);

for (int j = 0; j < 3; j++) {

if (data[i][j] == data[i][j + 1]) {

data[i][j] = 2 \* data[i][j];

data[i][j + 1] = 0;

compact(data[i]);

}

}

}

}

**4、其他方向合並**

雖然有了向左合並的函數，但是怎么模擬出旋轉界面呢，這里只要編寫一個可以旋轉矩陣的函數就行，如下

//拷貝數組

void copyArray(int data[4][4], int src[4][4]) {

for (int i = 0; i < 4; i++) {

for (int j = 0; j < 4; j++) {

data[i][j] = src[i][j];

}

}

}

//旋轉矩陣，count旋轉次數

void rotateMatrix(int data[4][4], int count) {

int temp[4][4];

copyArray(temp, data);

for (int i = 0; i < 4; i++) {

for (int j = 0; j < 4; j++) {

data[i][j] = temp[4 - j - 1][i];

}

}

if (count > 1)rotateMatrix(data, count-1);

}

有了旋轉矩陣的函數，那么其他方向就很好解決了

//向右合並

void mergerRight(int data[4][4]) {

rotateMatrix(data, 2);

mergerLeft(data);

rotateMatrix(data, 2);

}

//向上合並

void mergerUp(int data[4][4]) {

rotateMatrix(data, 3);

mergerLeft(data);

rotateMatrix(data, 1);

}

//向下合並

void mergerDown(int data[4][4]) {

rotateMatrix(data, 1);

mergerLeft(data);

rotateMatrix(data, 3);

}

**5、產生新的方塊**

通過之前的函數很容易編寫出來

//生成新數字

void newNum(int data[4][4], int initNum) {

int random[2];

while (1) {

int flag = 1;

randArray(random, 2);//隨即產生坐標

for (int i = 0; i < 4; i++) {//檢測坐標是否重復

for (int j = 0; j < 4; j++) {

if (data[i][j] != 0 && i == random[0] && j == random[1]) {

flag = 0;

}

}

}

if (flag)break;

}

data[random[0]][random[1]] = initNum;

}

**6、源代碼**

#include<stdio.h>

#include<time.h>

#include<stdlib.h>

#include<conio.h>

//函數聲明

void randArray(int a[], int n);

void rotateMatrix(int data[4][4], int count);

void copyArray(int data[4][4], int src[4][4]);

//全局變量定義

int data[4][4];

//初始化數據

void init(int data[4][4], int initNum) {

int random[4];

randArray(random, 4);

if (random[0] == random[2] && random[1] == random[3]) {

init(data, 2);//若坐標相同,則重新生成坐標

} else {

data[random[0]][random[1]] = initNum;

data[random[2]][random[3]] = initNum;

}

}

//緊湊數組(向左就湊)

void compact(int data[4]) {

int i, j = 0;

for (i = 0; i < 4; i++) {

if (data[i] != 0) {

data[j] = data[i];

j++;

}

}

for (; j < 4; j++)data[j] = 0;

}

//向左合並

void mergerLeft(int data[4][4]) {

for (int i = 0; i < 4; i++) {

compact(data[i]);

for (int j = 0; j < 3; j++) {

if (data[i][j] == data[i][j + 1]) {

data[i][j] = 2 \* data[i][j];

data[i][j + 1] = 0;

compact(data[i]);

}

}

}

}

//向右合並

void mergerRight(int data[4][4]) {

rotateMatrix(data, 2);

mergerLeft(data);

rotateMatrix(data, 2);

}

//向上合並

void mergerUp(int data[4][4]) {

rotateMatrix(data, 3);

mergerLeft(data);

rotateMatrix(data, 1);

}

//向下合並

void mergerDown(int data[4][4]) {

rotateMatrix(data, 1);

mergerLeft(data);

rotateMatrix(data, 3);

}

//生成新數字

void newNum(int data[4][4], int initNum) {

int random[2];

while (1) {

int flag = 1;

randArray(random, 2);//隨即產生坐標

for (int i = 0; i < 4; i++) {//檢測坐標是否重復

for (int j = 0; j < 4; j++) {

if (data[i][j] != 0 && i == random[0] && j == random[1]) {

flag = 0;

}

}

}

if (flag)break;

}

data[random[0]][random[1]] = initNum;

}

//獲取隨機數

void randArray(int a[], int n) {

for (int i = 0; i < n; i++) {//產生隨機坐標

srand((unsigned)time(NULL) + rand() + i);

a[i] = rand() % 4;

}

}

//顯示界面

void show(int data[4][4]) {

printf("\n\t2048小游戲");

for (int i = 0; i < 4; i++) {

printf("\n|-----|-----|-----|-----|\n|");

for (int j = 0; j < 4; j++) {

if (data[i][j] == 0) {

printf("%5c|", '\*');

} else {

printf("%5d|", data[i][j]);

}

}

}

printf("\n|-----|-----|-----|-----|\n");

}

//拷貝數組

void copyArray(int data[4][4], int src[4][4]) {

for (int i = 0; i < 4; i++) {

for (int j = 0; j < 4; j++) {

data[i][j] = src[i][j];

}

}

}

//旋轉矩陣

void rotateMatrix(int data[4][4], int count) {

int temp[4][4];

copyArray(temp, data);

for (int i = 0; i < 4; i++) {

for (int j = 0; j < 4; j++) {

data[i][j] = temp[4 - j - 1][i];

}

}

if (count > 1)rotateMatrix(data, count-1);

}

//保存文件

void saveFile() {

FILE\* f = fopen("2048.dat", "w");

for (int i = 0; i < 4; i++) {

for (int j = 0; j < 4; j++) {

fprintf(f, "%d ", data[i][j]);

}

fprintf(f, "\n");

}

fclose(f);

}

//讀取文件

void readFile() {

FILE\* f = fopen("2048.dat", "r");

if (f == NULL) {

printf("\n無游戲存檔，開始新游戲\n");

init(data, 2);

return;

}

for (int i = 0; i < 4; i++) {

for (int j = 0; j < 4; j++) {

fscanf(f, "%d ", &data[i][j]);

}

fscanf(f, "\n");

}

fclose(f);

}

int main() {

char ch;//方向

printf("1、新游戲\n");

printf("2、繼續游戲\n");

ch = getch();

if(ch == '1'){

init(data, 2);

} else {

readFile();

}

while (1) {

show(data);

ch = getch();

switch (ch) {

case 72://上

mergerUp(data);

newNum(data, 2);

break;

case 77://右

mergerRight(data);

newNum(data, 2);

break;

case 80://下

mergerDown(data);

newNum(data, 2);

break;

case 75://左

mergerLeft(data);

newNum(data, 2);

break;

case 27://ESC

saveFile();

return 0;

}

saveFile();

system("cls");

}

return 0;

}