# 《程序设计综合实验》期末大作业

### 课题名称

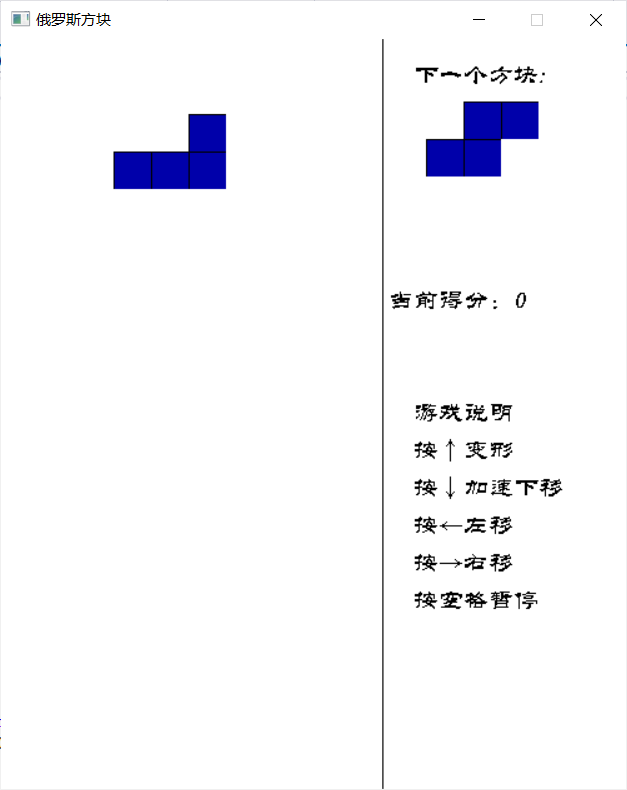
俄罗斯方块

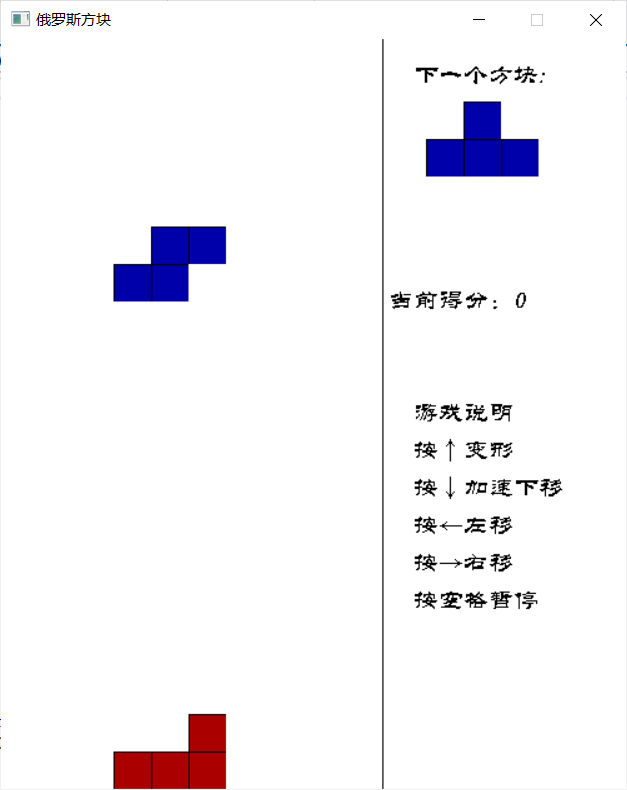
### 主要功能描述

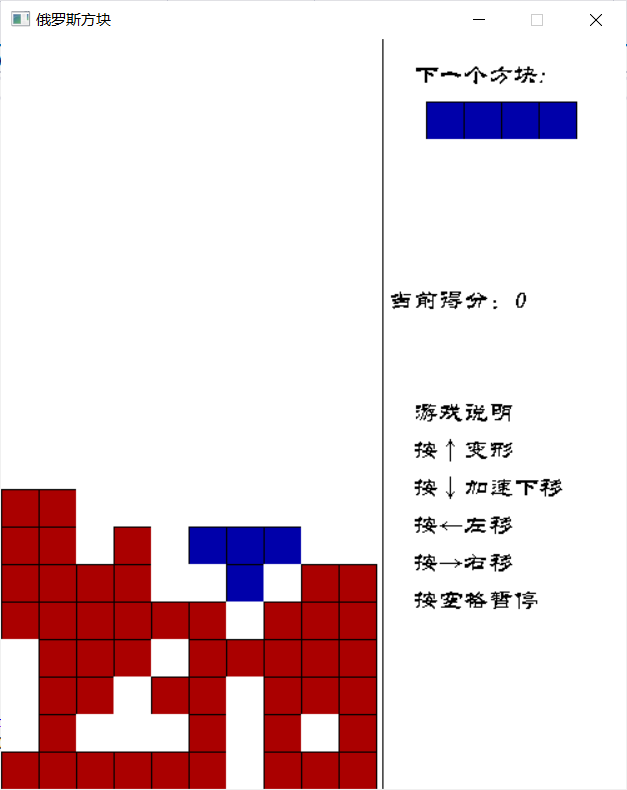
通过二维数组的知识，实现了7种俄罗斯方块，即28种转换形态的俄罗斯方块的随机生成和下落和计分，画出了下一个方块的模型，通过上下左右键分别控制方块旋转，加速下落，向左移动，向右移动的功能，空格键是暂停游戏。

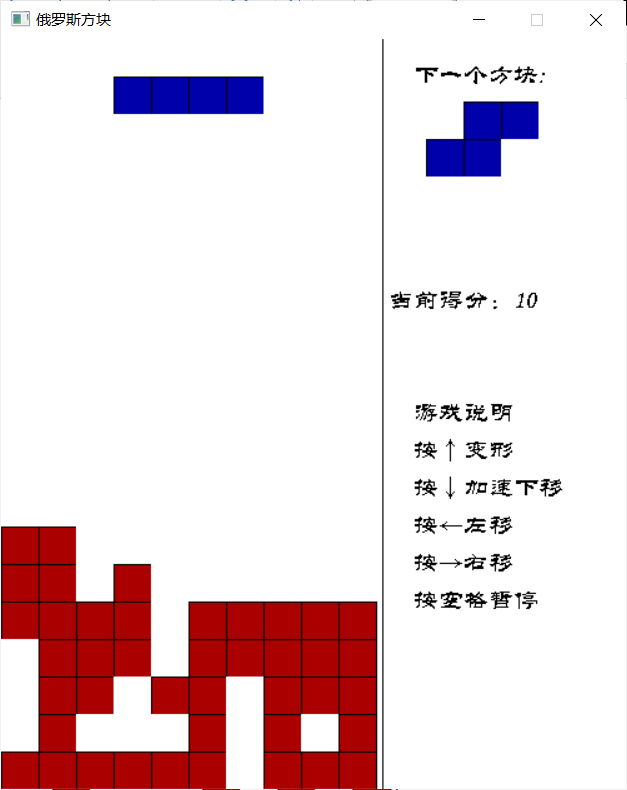
### 实现步骤（请写出核心代码和实现效果图）

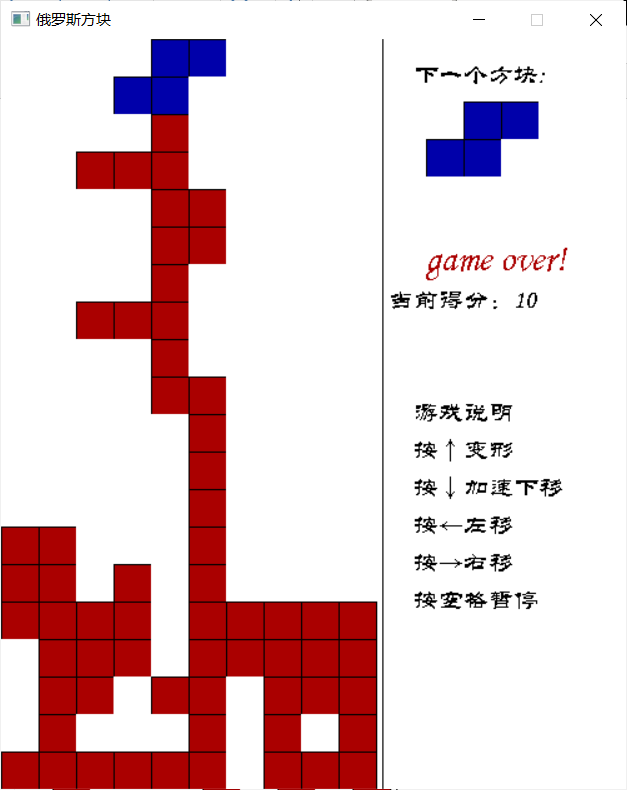
先绘制背景的静态的东西，那些文字和分割线，然后通过写好的俄罗斯方块的28种形态数组和随机函数，随机创建一个方块，然后循环下落，下落过程中通过按键操控其功能，中间夹杂着一系列边界判断，最后落到下面改变数组的值，即改变放块的状态，并判断最下面一行是不是可以消除，可以消除的画就得分，最后是循环的过程中要对整个数组的上边界进行判断，判断一下游戏是否结束。











#include<iostream>

#include<graphics.h>

#include<time.h>

#include<conio.h>

#include <mmsystem.h>

#pragma comment(lib,"winmm.lib")//音频

using namespace std;

int rotateIndex; //旋转方块的下标

int score = 0; //得分

int t = 300; //下落延迟的时间

int downFlg; //下落的旗帜

//整个地图

int map[20][10] = { 0 };

//俄罗斯方块的各种类型

int block[28][4][4] = {

{

{1,0,0,0},

{1,1,1,0},

{0,0,0,0},

{0,0,0,0},

},{

{1,1,0,0},

{1,0,0,0},

{1,0,0,0},

{0,0,0,0},

},{

{1,1,1,0},

{0,0,1,0},

{0,0,0,0},

{0,0,0,0},

},{

{0,1,0,0},

{0,1,0,0},

{1,1,0,0},

{0,0,0,0},

},{

{0,0,1,0},

{1,1,1,0},

{0,0,0,0},

{0,0,0,0},

},{

{1,0,0,0},

{1,0,0,0},

{1,1,0,0},

{0,0,0,0},

},{

{1,1,1,0},

{1,0,0,0},

{0,0,0,0},

{0,0,0,0},

},{

{1,1,0,0},

{0,1,0,0},

{0,1,0,0},

{0,0,0,0},

},{

{1,1,0,0},

{0,1,1,0},

{0,0,0,0},

{0,0,0,0},

},{

{0,1,0,0},

{1,1,0,0},

{1,0,0,0},

{0,0,0,0},

},{

{1,1,0,0},

{0,1,1,0},

{0,0,0,0},

{0,0,0,0},

},{

{0,1,0,0},

{1,1,0,0},

{1,0,0,0},

{0,0,0,0},

},{

{0,1,1,0},

{1,1,0,0},

{0,0,0,0},

{0,0,0,0},

},{

{1,0,0,0},

{1,1,0,0},

{0,1,0,0},

{0,0,0,0},

},{

{0,1,1,0},

{1,1,0,0},

{0,0,0,0},

{0,0,0,0},

},{

{1,0,0,0},

{1,1,0,0},

{0,1,0,0},

{0,0,0,0},

},{

{0,1,0,0},

{1,1,1,0},

{0,0,0,0},

{0,0,0,0},

},{

{1,0,0,0},

{1,1,0,0},

{1,0,0,0},

{0,0,0,0},

},{

{1,1,1,0},

{0,1,0,0},

{0,0,0,0},

{0,0,0,0},

},{

{0,1,0,0},

{1,1,0,0},

{0,1,0,0},

{0,0,0,0},

},{

{1,1,1,1},

{0,0,0,0},

{0,0,0,0},

{0,0,0,0},

},{

{1,0,0,0},

{1,0,0,0},

{1,0,0,0},

{1,0,0,0},

},{

{1,1,1,1},

{0,0,0,0},

{0,0,0,0},

{0,0,0,0},

},{

{1,0,0,0},

{1,0,0,0},

{1,0,0,0},

{1,0,0,0},

},{

{1,1,0,0},

{1,1,0,0},

{0,0,0,0},

{0,0,0,0},

}, {

{1,1,0,0},

{1,1,0,0},

{0,0,0,0},

{0,0,0,0},

}, {

{1,1,0,0},

{1,1,0,0},

{0,0,0,0},

{0,0,0,0},

}, {

{1,1,0,0},

{1,1,0,0},

{0,0,0,0},

{0,0,0,0},

}

};

//交换两个值

void swap (int& a, int& b) {

int t = a;

a = b;

b = t;

}

//绘制当前要掉落的方块

void createBlock (int n) {

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

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

if (map[i][j + 3] != 2)

map[i][j + 3] = block[n][i][j];

}

}

}

//绘制整个地图的方块

void drawMap () {

for (int i = 0; i < 20; i++) {

for (int j = 0; j < 10; j++) {

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

setfillcolor (WHITE);

solidrectangle (30 \* j, 30 \* i, 30 \* (j + 1), 30 \* (i + 1));

}

if (map[i][j] == 1) {

setfillcolor (BLUE);

fillrectangle (30 \* j, 30 \* i, 30 \* (j + 1), 30 \* (i + 1));

}

if (map[i][j] == 2) {

setfillcolor (RED);

fillrectangle (30 \* j, 30 \* i, 30 \* (j + 1), 30 \* (i + 1));

}

}

}

}

//判断并消除一行

void clearLine () {

for (int i = 0; i < 20; i++) {

int k = 1;

for (int j = 0; j < 10; j++) {

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

k = 0;

}

}

if (k) {

score += 10;

for (int j = 0; j < 10; j++) {

map[i][j] = 0;

}

for (int x = i - 1; x >= 0; x--) {

for (int y = 0; y < 10; y++) {

swap (map[x][y], map[x + 1][y]);

}

}

}

}

}

//落到下面的方块变成红色，并修改其数组的值为2

void changeState () {

for (int i = 0; i < 20; i++) {

for (int j = 0; j < 10; j++) {

if (map[i][j] == 1)

map[i][j] = 2;

}

}

clearLine ();

}

//方块旋转

void rotateBlock (int n) {

int a = 20, b = 10;

for (int i = 0; i < 20; i++) {

for (int j = 0; j < 10; j++) {

if (map[i][j] == 1) {

if (i <= a) {

a = i;

}

if (j <= b) {

b = j;

}

break;

}

}

}

int r = 3;

if (n >= 20 && n < 24) {

r = 4;

}

//判断是否可以旋转

for (int i = r; i >= 0; i--) {

for (int j = r; j >= 0; j--) {

if (a + i > 19 || b + j > 10 || map[a + i - 1][b + j - 1] == 2) {

return;

}

}

}

//赋旋转后的值

for (int i = r - 1; i >= 0; i--) {

for (int j = r - 1; j >= 0; j--) {

map[a + i][b + j] = block[n][i][j];

}

}

}

//判断落下方块的状态

void judgeState () {

for (int i = 0; i < 20; i++) {

for (int j = 0; j < 10; j++) {

if (i != 19 && map[i][j] == 1 && map[i + 1][j] == 2 || i == 19 && map[i][j] == 1) {

changeState ();

downFlg = 1;

}

}

}

}

//移动方块

void move () {

if (!\_kbhit ()) {

judgeState ();

//向下移动

for (int i = 18; i >= 0; i--) {

for (int j = 9; j >= 0; j--) {

if (map[i][j] == 1) {

swap (map[i][j], map[i + 1][j]);

}

}

}

}

else {

int moveFlg = 1;

int key = \_getch ();

//空格暂停

if (key == 32) {

\_getch ();

}

if (\_kbhit ()) {

int option = \_getch ();

switch (option) {

case 75://左

for (int i = 0; i < 20; i++) {

for (int j = 0; j < 10; j++) {

if (map[i][j] == 1) {

if (j == 0 || map[i][j - 1] == 2) {

moveFlg = 0;

}

}

}

}

if (moveFlg) {

for (int i = 0; i < 20; i++) {

for (int j = 1; j < 10; j++) {

if (map[i][j] == 1 && map[i][j - 1] == 0) {

swap (map[i][j], map[i][j - 1]);

}

}

}

}

break;

case 77://右

for (int i = 0; i < 20; i++) {

for (int j = 0; j < 10; j++) {

if (map[i][j] == 1) {

if (j == 9 || map[i][j + 1] == 2) {

moveFlg = 0;

}

}

}

}

if (moveFlg) {

for (int i = 0; i < 20; i++) {

for (int j = 8; j >= 0; j--) {

if (map[i][j] == 1 && map[i][j + 1] == 0) {

swap (map[i][j], map[i][j + 1]);

}

}

}

}

break;

case 72://上

rotateIndex++;

rotateBlock (rotateIndex); //旋转方块

if (rotateIndex % 4 == 3)

rotateIndex -= 4;

break;

case 80://下

t = 100; //缩短休眠时间

break;

}

}

}

}

//绘制下一个方块

void createNextBlock (int n) {

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

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

if (block[n][i][j] == 0) {

setfillcolor (WHITE);

solidrectangle (340 + 30 \* j, 50 + 30 \* i, 370 + 30 \* j, 80 + 30 \* i);

}

if (block[n][i][j] == 1) {

setfillcolor (BLUE);

fillrectangle (340 + 30 \* j, 50 + 30 \* i, 370 + 30 \* j, 80 + 30 \* i);

}

}

}

//判断游戏是否结束

int isEnd () {

for (int j = 0; j < 10; j++) {

if (map[0][j] == 2) {

return 1;

}

}

return 0;

}

//绘制游戏背景

void drawBackground () {

setbkcolor (WHITE);

cleardevice ();

//绘制直线

setlinecolor (BLACK);

settextcolor (BLACK);

settextstyle (20, 0, "华文隶书");

outtextxy (330, 20, "下一个方块:");

outtextxy (330, 290, "游戏说明");

outtextxy (330, 320, "按↑变形");

outtextxy (330, 350, "按↓加速下移");

outtextxy (330, 380, "按←左移");

outtextxy (330, 410, "按→右移");

outtextxy (330, 440, "按空格暂停");

//绘制分割线

line (305, 0, 305, 600);

}

int main () {

//背景音乐

mciSendString ("open music.mp3 alias music", nullptr, 0, nullptr);

mciSendString ("play music repeat", nullptr, 0, nullptr);

//绘制画布

initgraph (500, 600);

//绘制背景函数

drawBackground ();

//随机种子函数

srand ((unsigned int)time (NULL));

char str[30];

int n = rand () % 7;

while (1) {

sprintf\_s (str, "当前得分：%d", score);

outtextxy (310, 200, str);

downFlg = 0;

//生成一个方块

createBlock (4 \* n);

rotateIndex = 4 \* n;

//下一个要生成的方块

int nNext = rand () % 7;

createNextBlock (4 \* nNext);

n = nNext;

//判断游戏是否结束

if (isEnd () == 1) break;

//循环下落

t = 300;

while (true) {

//绘制整个地图

drawMap ();

move ();

Sleep (t);

if (downFlg) {

break;

}

}

}

settextcolor (RED);

setbkmode (TRANSPARENT); //文字背景透明

settextstyle (30, 0, "华文隶书");

outtextxy (340, 163, "game over!");

while (1);

return 0;

}