具体代码详细分工：

白智崴：1、4、9

郭艺博：2、6、8

徐国鹏：3

于书昶：5、7

1.#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#include <conio.h>

#include <time.h>

#include <windows.h>

#include <stdbool.h>

//MAXWIDTH、MAXHEIGHT、INITLEN 以字符记

#define Width (30)

#define Length Width

#define INITLEN (3) //贪吃蛇的初始长度

struct

{

char \*ch;

int color;

char type;

}

Border = { "▓", 15, 1 }, //边框

Background = { "■", 0, 2 }, //背景

Snake = { "●", 11, 3 }, //贪吃蛇节点

Food = { "★", 12, 4 }, //食物

Food\_spec = { "◆",12, 5 }, //为了实现贪吃蛇每吃三个food，第四个food有特殊加分的功能，需要设计特殊食物

Head = { "○",11, 6 };

int Gx, Gy;//

int TIMES;

int foodSpeed = 4;

int snakespeed = 200;

int scores = 0; //得分也就是吃苹果的数量

int snakeMapLength = (Width - 2)\*(Length - 2); //snakemaplength是snakemap一维数组的长度

int headerIndex, tailIndex; //蛇头蛇尾对应的snakeMap中的索引（下标）

int eat\_count = 0; //用来记录贪吃蛇吃的苹果的数量

int snakespeed;

HANDLE Handle; //控制台句柄

2.struct

{

char type;

int index;

}WholeMap[Width][Length];

//WholeMap是对应整个地图区域二维数组，用type来确定地图此处是什么东西（食物or蛇or背景or墙），index是SnakeMap一维数组的下标

//贪吃蛇有效活动范围地图的索引

struct {

int x;

int y;

} SnakeMap[(Width - 2)\*(Length - 2)], ApplePosition;

void setPosition(int x, int y)

{

COORD coord; //COORD结构中X和Y与显示中是相反的

coord.X = 2 \* y; //乘2是因为定义窗口大小的cols需要（试出来的）

coord.Y = x;

SetConsoleCursorPosition(Handle, coord);

}

// 设置颜色

void setColor(int color)

{

SetConsoleTextAttribute(Handle, color);

}

3.//创建食物

void createFood()

{

int index, rang, x, y;

//产生随机数，确定 snakeMap 数组的索引

srand((unsigned)time(NULL)); //以当前时间time为产生随机数的种子

if (tailIndex<headerIndex)

{ //当蛇头下标大于蛇尾下标，产生食物的随机数

rang = headerIndex - tailIndex - 1;

index = rand() % rang + tailIndex + 1;

}

else //当蛇头下标小于蛇尾下标，产生食物的随机数

{

rang = snakeMapLength - (tailIndex - headerIndex + 1);

index = rand() % rang;

if (index >= headerIndex) {

index += (tailIndex - headerIndex + 1);

}

}

eat\_count++;

x = SnakeMap[index].x;

y = SnakeMap[index].y;

Gx = x;

Gy = y;

setPosition(x, y);

if (eat\_count % 4 != 0 && eat\_count != 0)

{

setColor(Food.color);

printf("%s", Food.ch);

}

else

{

int x1, y1;

index = 0;

while (WholeMap[SnakeMap[index].x][SnakeMap[index].y].type != 2) {

if (tailIndex<headerIndex)

{ //当蛇头下标大于蛇尾下标，产生食物的随机数

rang = headerIndex - tailIndex - 1;

index = rand() % rang + tailIndex + 1;

}

else //当蛇头下标小于蛇尾下标，产生食物的随机数

{

rang = snakeMapLength - (tailIndex - headerIndex + 1);

index = rand() % rang;

if (index >= headerIndex) {

index += (tailIndex - headerIndex + 1);

}

}

}

x1 = SnakeMap[index].x;//新蛇头的坐标

y1 = SnakeMap[index].y;

setPosition(x1, y1);

setColor(Food\_spec.color);

printf("%s", Food\_spec.ch);

WholeMap[x1][y1].type = Food\_spec.type;

}

WholeMap[x][y].type = Food.type;

}

4.void moveFood()

{

srand((unsigned)time(NULL));

bool flag = true;

int cx, cy;

int x, y;

int count = 0, maxCount = 1000;

while (flag)

{

int dir = rand() % 4 + 1;

if (dir % 2 == 0) {

cx = 0;

cy = dir - 3;

}

else {

cx = dir - 2;

cy = 0;

}

if ((Gx + cx)>1 && (Gx + cx)<(Width - 2) && (Gy + cy)>1 && (Gy + cy)<(Length - 2)) {

if (WholeMap[Gx + cx][Gy + cy].type != 2) {

continue;

}

flag = false;

}

count++;

if (count > maxCount) {

cx = 0;

cy = 0;

break;

}

}

//抹去原来的食物

setPosition(Gx, Gy);

setColor(Background.color);

printf("%s", Background.ch);

WholeMap[Gx][Gy].type = Background.type;

//显示新的食物

Gx += cx;

Gy += cy;

setPosition(Gx, Gy);

setColor(Food.color);

printf("%s", Food.ch);

WholeMap[Gx][Gy].type = Food.type;

}

5.void die()

{

int xCenter = Length % 2 == 0 ? Length / 2 : Length / 2 + 1;

//xcenter、ycenter就是窗口的中心座标，这里用来显示游戏结束警告

int yCenter = Width % 2 == 0 ? Width / 2 : Width / 2 + 1;

setPosition(xCenter, yCenter - 5);

setColor(0xC);

printf("You lose! Game Over!");

\_getch();

system("cls");

setColor(15);

scores = 0;

eat\_count = 0;

main();

}

6.int start()

{

int a, b = 30;

getchar();

printf(" \n\n\n\n\n\n\n\n\n\n 贪 吃 蛇\n\n\n\n");

printf(" 请选择您的游戏难度: 1、 简单 2、 普通 3、 困难\n");

printf(" ");

scanf("%d", &a);

if (a == 1)

b = 250;

else if (a == 2)

b = 150;

else if (a == 3)

b = 80;

else

{

printf(" 干嘛啊，输1、2、3啊\n");

printf(" 既然你故意输错，那就将进入变态难度");

\_getch();

}

eat\_count = 0;

return b;

}

7.void move(char direction)

{

int newHeaderX, newHeaderY; //新蛇头的坐标

int Storevalue; //新蛇头坐标以前对应的索引

int StorevalueX, StorevalueY; //新蛇头的索引以前对应的坐标

int newHeaderPreType; //新蛇头以前的类型

int oldTailX, oldTailY; //老蛇尾坐标

int m, n; // -----------------------------------------------

m = SnakeMap[headerIndex].x;//新蛇头的坐标

n = SnakeMap[headerIndex].y;

switch (direction) {

case 'w':

newHeaderX = SnakeMap[headerIndex].x - 1;

newHeaderY = SnakeMap[headerIndex].y;

break;

case 's':

newHeaderX = SnakeMap[headerIndex].x + 1;

newHeaderY = SnakeMap[headerIndex].y;

break;

case 'a':

newHeaderX = SnakeMap[headerIndex].x;

newHeaderY = SnakeMap[headerIndex].y - 1;

break;

case 'd':

newHeaderX = SnakeMap[headerIndex].x;

newHeaderY = SnakeMap[headerIndex].y + 1;

break;

default:

printf("%c\n", direction);

exit(0);

break;

}

//新蛇头的索引

//newheaderX、newheaderY是蛇头将要出现位置在globalmap二维数组的坐标

// headerIndex = headerIndex == 0 ? snakeMapLen - 1 : headerIndex - 1;

if (headerIndex == 0)

headerIndex = snakeMapLength;

else

headerIndex = headerIndex - 1;

//贪吃蛇默认在snakemap这个一维数组中前进，所谓前进就是每次下标减1

// -----------------------------------------------

//新蛇头坐标以前对应的索引

Storevalue = WholeMap[newHeaderX][newHeaderY].index;

//Storevalue就是蛇头将要出现位置在snakemap一维数组中的下标

//新蛇头的索引以前对应的坐标

StorevalueX = SnakeMap[headerIndex].x;

StorevalueY = SnakeMap[headerIndex].y;

//StorevalueX、Y就是蛇头预计要出现位置在wholemap二维数组中的下标

//双向绑定新蛇头索引与坐标的对应关系

SnakeMap[headerIndex].x = newHeaderX;

SnakeMap[headerIndex].y = newHeaderY;

//把蛇头将要出现位置的坐标赋值给蛇头预计出现位置的坐标

WholeMap[newHeaderX][newHeaderY].index = headerIndex;

//把蛇头预计出现位置的下标赋值给蛇头将要出现位置的下标

//双向绑定以前的索引与坐标的对应关系

SnakeMap[Storevalue].x = StorevalueX;

SnakeMap[Storevalue].y = StorevalueY;

//把蛇头预计出现位置的坐标赋值给蛇头将要出现位置的坐标

WholeMap[StorevalueX][StorevalueY].index = Storevalue;

//把蛇头将要出现位置的下标赋值给蛇头预计出现位置的下标

//新蛇头以前的类型

newHeaderPreType = WholeMap[newHeaderX][newHeaderY].type;

//把蛇头将要出现位置填充图标的类型索引取出

//设置新蛇头类型

WholeMap[newHeaderX][newHeaderY].type = Snake.type;

//把蛇头将要出现位置填充图标的类型索引设置为蛇的图标索引

// 判断是否出界或撞到自己

if (newHeaderPreType == Border.type || newHeaderPreType == Snake.type)

{

die();

}

//输出新蛇头

setPosition(newHeaderX, newHeaderY);

setColor(Snake.color);

printf("%s", Head.ch);

setPosition(m, n);

setColor(Snake.color);

printf("%s", Snake.ch);

//判断是否吃到食物，如果吃到食物就不删除蛇尾；变向的增加了蛇尾的长度.

if (newHeaderPreType == Food.type)

{ //吃到食物

createFood();

//更改分数

setColor(15);

setPosition(ApplePosition.x, ApplePosition.y);

printf("%d", ++scores);

}

else if (newHeaderPreType == Food\_spec.type) {

scores += 2;

setColor(15);

setPosition(ApplePosition.x, ApplePosition.y);

printf("%d", scores);

}

else

{

///NEW

TIMES += 1;

if (TIMES % foodSpeed == 0)

{

moveFood();

}

//老蛇尾坐标

oldTailX = SnakeMap[tailIndex].x;

oldTailY = SnakeMap[tailIndex].y;

//删除蛇尾

setPosition(oldTailX, oldTailY);

setColor(Background.color);

printf("%s", Background.ch);

WholeMap[oldTailX][oldTailY].type = Background.type;

// tailIndex = (tailIndex == 0) ? snakeMapLength - 1 : tailIndex - 1;

if (tailIndex == 0)

tailIndex = snakeMapLength;

else

tailIndex = tailIndex - 1;

}

}

//下次移动的方向

char nextDirection(char ch, char directionOld)

{

int sum = ch + directionOld;

ch = tolower(ch);

if ((ch == 'w' || ch == 'a' || ch == 's' || ch == 'd') && sum != 197 && sum != 234)

//197=a+d，234=w+s如果输入的方向和当前运动方向相反

{

return ch;

}

else

{

return directionOld;

}

}

8.//暂停

char pause()

{

return \_getch();

}

// 初始化

void init() {

// 设置相关变量

int x, y, i, index;

int xCenter = Length % 2 == 0 ? Length / 2 : Length / 2 + 1;

int yCenter = Width % 2 == 0 ? Width / 2 : Width / 2 + 1;

//xcenter、ycenter是窗口的中心座标

CONSOLE\_CURSOR\_INFO cci;

//控制台光标信息

//判断相关设置是否合理

if (Width<16)

{

printf("'MAXWIDTH' is too small!");

\_getch();

exit(0);

}

//设置窗口大小

system("mode con: cols=106 lines=32");

//这里cols=106对应setposition函数中COORD.x=2\*y

//隐藏光标

Handle = GetStdHandle(STD\_OUTPUT\_HANDLE);

GetConsoleCursorInfo(Handle, &cci);

cci.bVisible = 0;

SetConsoleCursorInfo(Handle, &cci);

//打印背景

for (x = 0; x<Length; x++)

{

for (y = 0; y<Width; y++)

{

if (y == 0 || y == Width - 1 || x == 0 || x == Length - 1)

{

WholeMap[x][y].type = Border.type; //设置填充图片的类型

setColor(Border.color);

printf("%s", Border.ch);

}

else

{

index = (x - 1)\*(Width - 2) + (y - 1);

//index是由二维数组wholemap中x、y计算出的一维数组snakemap的下标，从0开始

SnakeMap[index].x = x;

SnakeMap[index].y = y;

WholeMap[x][y].type = Background.type; //设置填充图片的类型

WholeMap[x][y].index = index;

//设置wholemap的成员index的值，index也就是一维数组snakemap的下标

setColor(Background.color);

printf("%s", Background.ch);

}

}

printf("\n");

}

//初始化贪吃蛇

WholeMap[xCenter][yCenter - 1].type = WholeMap[xCenter][yCenter].type = WholeMap[xCenter][yCenter + 1].type = Snake.type;

//设置贪吃蛇三节的填充类型

headerIndex = (xCenter - 1)\*(Width - 2) + (yCenter - 1) - 1;

//计算蛇头在snakemap这个一维数组中的下标

tailIndex = headerIndex + 2;

//蛇长默认为3，所以tailIndex=headerIndex+2

setPosition(xCenter, yCenter - 1);

setColor(Snake.color);

printf("%s", Head.ch);

for (y = yCenter; y <= yCenter + 1; y++)

{

printf("%s", Snake.ch);

}

//生成食物

createFood();

//设置程序信息

setPosition(xCenter - 1, Width + 2);

setColor(15);

printf(" Apples : ");

setPosition(xCenter, Width + 2);

printf(" Author:白智崴,郭艺博,徐国鹏,于书昶");

setPosition(xCenter + 1, Width + 2);

ApplePosition.x = xCenter - 1;

ApplePosition.y = Width + 8;

}

9.int main()

{

int b;

b = start();

char charInput, direction = 'a'; //默认贪吃蛇移动的方向

init(); //调用初始化函数

snakespeed = b;

charInput = tolower(\_getch()); //tolower把输入的字符转化为小写

direction = nextDirection(charInput, direction); //nextdirection方向判断函数

int pre = eat\_count;

while (1)

{

if (\_kbhit())

{ //kbhit检测是否有用户输入

charInput = tolower(\_getch());

if (charInput == ' ')

{ //如果用户输入空格，程序暂停

charInput = pause();

}

direction = nextDirection(charInput, direction);

}

move(direction); //move贪吃蛇移动函数

if (eat\_count > pre) {

snakespeed--;

pre = eat\_count;

}

Sleep(snakespeed);

}

\_getch();

return 0;

}