документирование

1.Обзор проекты обеспечения

1.1 цели

Реализовать классическую игру Snake (змейка).

Управление змейкой осуществляется с клавиатуры. Пользователь управляет головой змейки (квадрат может делать перемещение влево, вправо, вверх и вниз). За головой следует тело змейки (цепочка квадратов). В течение раунда скорость змейки и длина тела фиксированы. Для успешного завершения раунда голова змейки должна пересечься с едой (квадрат). Переход на следующий раунд сопровождается увеличением длины тела змейки на квадрат. Пересечение головы змейки с границей игровой области или с телом змейки завершает игру.

1.2 особенности

Длины тела змейки увеличением после того,кушать продукт.

2.инструкции:

2.1 .импорт

клавиши ввода ：вверх вниз влева вправа

2.2. например импорта

Если ввозить вверх ,земя будет движить вверх , змея направление движения пищи, ест пищу, чтобы получить более длинный корпус

2.3 .экспорт

Змея изменяется движение направлении

3.Объяснение движения   
3.1. график движения (перечислять каждая возможная ситуация движения,и объснить цель движения.)  
1).направление движения змеи (вниз ,вверх,налево,направо,и змея ест питания)  
2).измерить направление движения змея，и увеличить длину змеи

4.осписание

#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

#include <time.h>

#include <windows.h>

1.(установить размер карты, длину и ширину)

const int W = 52;

const int H = 22;

2.(установить одну карту) ( сохранить,когда пользователь ударит по клавишам)

char map[22][52];

char key;

3.(установить настоящее направление змеи и положиние пищи и длуну змеи )

int direct = 4;

int food[2] = {9,11};

int head;

4. (все точки движения змеи )

int snake[400][3];

5.(змея двигает, возникнуть пищу случайно и рассудить ситуацию змеи)

void init(char map[22][52], int snake[400][3]);

void makeMap(char map[22][52],int snake[400][3],int food[]);move后的

void move(int snake[400][3],int direct);

void makeFood(int food[]);

void showView(char map[22][52]);

int ifEat(int head, int food[2]);

int ifReprat(int snake[400][3], int x, int y);

int ifBump(int head);

void getKey();

6.(инициализировать карту и выполнять фукцию)

int main() {

init(map, snake);

while (1) {

getKey();

system("cls");

Sleep(30);

move(snake, direct);

if (!food[0]&&!food[1]) {

makeFood(food);

}

makeMap(map, snake, food);

showView(map);

if (ifBump(head)) {

printf("GAME OVER：%d", head);

break;

}

getKey();

}

getchar();

return 0;

}

7.(инициализировать змею и карту и выполнять фукцию)

void init(char map[22][52], int snake[400][3]) {

snake[0][0] = 0, snake[0][1] = 9, snake[0][2] = 7;

snake[1][0] = 0, snake[1][1] = 9, snake[1][2] = 8;

snake[2][0] = 1, snake[2][1] = 9, snake[2][2] = 9;

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

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

if (i == 0 || j == 0 || i == H - 1 || j == W - 1) {

map[i][j] = '\*';

}

else {

map[i][j] = ' ';

}

}

}

}

8.Печать Карта

void showView(char map[22][52]) {

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

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

printf("%c", map[i][j]);

}

printf("\n");

}

}

9. Выполнение с помощью списка, чтобы змея двигаться

1》Змеи не едят пищу Перемещение в первоначальном направлении

2》Змеи едят пищу Прежде всего тела в положении в передней части

void move(int snake[400][3],int direct) {

int x, y;

for (int i = 0;i < 400;i++) {

if (snake[i][0] == 1) {

head = i;

break;

}

}

x = snake[head][1];

y = snake[head][2];

switch (direct){

case 1:

snake[head][1]--;

break;

case 2:

snake[head][1]++;

break;

case 3:

snake[head][2]--;

break;

case 4:

snake[head][2]++;

break;

}

if (ifEat(head, food)) {

snake[head + 1][0] = 1, snake[head + 1][1] = food[0], snake[head + 1][2] = food[1];

snake[head][0] = 0;

food[0] = 0, food[1] = 0;

}

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

int temp;

temp = x, x = snake[j][1], snake[j][1] = temp;

temp = y, y = snake[j][2], snake[j][2] = temp;

}

}

10. Снова положить координат пищи . Пища не должна совпадать с змеей

void makeFood(int food[]) {

srand(time(0));

int x = rand() % 49 + 2, y = rand() % 19 + 2;/

while (ifReprat(snake,x,y)) {

x = rand() % 49 + 2, y = rand() % 19 + 2;

}

food[0] = y;

food[1] = x;

}

void makeMap(char map[22][52], int snake[400][3], int food[]) {

int i;

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

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

if (i == 0 || j == 0 || i == H - 1 || j == W - 1) {

map[i][j] = '\*';

}

else {

map[i][j] = ' ';

}

}

}

for (i = 0;i < 400;i++) {

if (snake[i][0] == 1) break;

map[snake[i][1]][snake[i][2]] = '#';

}

map[snake[i][1]][snake[i][2]] = '@';

map[food[0]][food[1]] = 'o';

}

11.определить змея кушал протукты или нет

int ifEat(int head,int food[2]) {

if (snake[head][1] == food[0] && snake[head][2] == food[1])

return 1;

else

return 0;

}

int ifReprat(int snake[400][3],int x,int y) {

for (int i = 0;i < 400;i++) {

if (snake[i][0] == 1) break;

if ((snake[i][1] == x&&snake[i][2] == y)) { return 1;

}

}

return 0;

}

12.определить snake пересекаться с стеной или нет,с телом или нет.

int ifBump(int head) {

if ((snake[head][2]==0|| snake[head][2] == 51) || (snake[head][1] == 0|| snake[head][1] == 21))

return 1;

for (int i = 0;i < head-1;i++) {

if ((snake[i][1] == snake[head][1]&&snake[i][2] == snake[head][2])) {

return 1;

}

}

return 0;

}

13.определить направление движения

void getKey() {

if (\_kbhit()) {

key = \_getch();

}

switch (key) {

case 'H':

if (direct != 2)

direct = 1;

break;

case 'P':

if (direct != 1)

direct = 2;

break;

case 'K':

if (direct != 4)

direct = 3;

break;

case 'M':

if (direct != 3)

direct = 4;

break;

}

}