// SnakeGame.cpp

// игра змейка на С++

#include <iostream>

#include <conio.h> // для \_getch, \_kbhit()

using namespace std;

/////// глобальные переменные /////////////////////////////////////////

bool gameOver; // переменная для выхода из игры

const int width = 20; // переменная для ширины поля, типа const, чтобы

// не изменялся этот параметр

const int height = 20; // переменная для высоты поля, типа const, чтобы

// не изменяля этот параметр

int x, y; // переменные для положения змейки в игре

int FruitX, FruitY; // переменнаые для положения фрукта в игре

int score; // переменная для счёта

int tailX[100]; // массив для хвоста змеи, их координаты по ох

int tailY[100]; // массив для хвоста змеи, их координаты по oy

int nTail; // переменная для количества ячеек под длину

// хвоста змеи

int key = 0; // ключ для меню

//////// перечисления для движения змейки /////////////////////////////

enum eDirection{STOP=0,LEFT, RIGHT, UP, DOWN};

eDirection dir; // переменная типа eDirection - перечисления

//////////// void Setup() /////////////////////////////////////////////

// при вызове данной функции игра будет настраивать все нужные нам пара-

// метры игры

void Setup()

{

gameOver = false; // первоначальное значение false, в начале игры

// мы не проиграли, игра должна запустится, как

// только переменная меняется на true, игра за-

// вершается

dir = STOP;

x = width / 2 - 1; // координаты положения змейки по оси ох

y = height / 2 - 1; // координаты положения змейки по оси oy

FruitX = rand() % width; // положение фракта задаётся рандомом,

// деление на wight, и выделение остат-

// ка позволяет не выходить за рамки

// игры, это по оси ох

FruitY = rand() % height; // положение фракта задаётся рандомом,

// деление на height, и выделение остат-

// ка позволяет не выходить за рамки

// игры, это по оси оy

score = 0; // в начальный момент времени счёт игры

// равен 0;

}

/////////// void Drow /////////////////////////////////////////////////

// функция Drow - рисует карту игры

void Drow()

{

system("cls"); // очистка карты для действий

// верхняя граница поля

for (int i = 0; i < width + 1; i++)

cout << "#";

cout << endl;

// граница по бокам поля

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

{

for (int j = 0; j < width; j++)

{

if (j == 0 || j == width - 1)

cout << "#";

// задаём положение змеи на игровом поле

if (i == y && j == x)

cout << "0";

// задаём положение фрукта

else if (i == FruitY && j == FruitX)

cout << "F";

// во всех остальных случаях пустое пространство

else

{

bool print = false; // переменная для правильности

// растановки пробелов, если

// неправильно расставлять

// пробелы, то хвост змеи будет

// находится на верхней строчке

// цикл для прорисовки хвоста змеи

for (int k = 0; k < nTail; k++)

{

if (tailX[k] == j && tailY[k] == i)

{

print = true;

cout << "o";

}

}

// условие расстановки пробелов только в том случае,

// когда print = false

if(!print)

cout << " ";

}

}

// переход на следующую строку при проверке

cout << endl;

}

// нижняя граница поля

for (int i = 0; i < width +1 ; i++)

cout << "#";

cout << endl;

// вывод результа (счёта)

cout << "Score: " << score << endl;

}

/////////// void Input ////////////////////////////////////////////////

// отслеживает нажатия клавишь

void Input()

{

if (\_kbhit()) // функция \_kbhit() возвращает true, если

{ // пользователь нажал на клавишу

// работа по перемещению, через клавишу, без отображения на поле,

// так как используется \_getch, а не \_getche

switch (\_getch())

{

// движение влево

case 'a':

dir = LEFT;

break;

// движение вправо

case 'd':

dir = RIGHT;

break;

// движение вниз

case 's':

dir = DOWN;

break;

// движение вверх

case 'w':

dir = UP;

break;

// выход из игры

case 'x':

gameOver = true;

break;

}

}

}

////////// void Logic /////////////////////////////////////////////////

// логика игры

void Logic()

{

int prefX = tailX[0]; // переменная по ох, для запоминания преды-

// дущего положения хвоста, чтобы змея дви-

// галась правильно, по оси х

int prefY = tailY[0]; // переменная по ох, для запоминания преды-

// дущего положения хвоста, чтобы змея дви-

// галась правильно, по оси y

int pref2X, pref2Y; // для предыдущих элементов хвоста

tailX[0] = x; // первый элемент для x, чтобы хвост был за

// головой, а не в начале строки, поэтому

// присваиваем значение начальное х

tailY[0] = y; // первый элемент для y, чтобы хвост был за

// головой, а не в начале строки, поэтому

// присваиваем значение начальное y

// перемещение хвоста за головой

for (int i = 1; i < nTail; i++)

{

pref2X = tailX[i];

pref2Y = tailY[i];

tailX[i] = prefX;

tailY[i] = prefY;

prefX = pref2X;

prefY = pref2Y;

}

// уменьшение координаты при нажатии клавиш

switch (dir)

{

case LEFT:

x--;

break;

case RIGHT:

x++;

break;

case DOWN:

y++;

break;

case UP:

y--;

break;

}

// логика при съедании фрукта

if (x == FruitX && y == FruitY)

{

score += 10;

FruitX = rand() % width;

FruitY = rand() % height;

nTail++;

}

// логика при попадании в хвост змеи

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

{

if (tailX[i] == x && tailY[i] == y)

gameOver = true;

}

if (key == 1)

{

// логика при попадание на стенку змейки

if (x > width || x < 0 || y > height || y < 0)

gameOver = true;

}

if (key == 2)

{

// логика для перехода через стенку по х

if (x >= width - 1)

x = 0;

else if (x < 0)

x = width - 2;

// логика для перехода через стенку по y

if (y >= height)

y = 0;

else if (y < 0)

y = height - 1;

}

}

//////////// int main() ///////////////////////////////////////////////

int main()

{

setlocale(LC\_ALL, "Russian");

cout << "\nДобро пожаловать в змейку!" << endl;

cout << "Выберите один из вариантов игры:" << endl;

cout << "---------------------------------" << endl;

cout << "1. Игра с твёрдыми стенками " << endl;

cout << "2. Игра со стенкой зеркало " << endl;

cin >> key;

Setup();

while (!gameOver)

{

Drow();

Input();

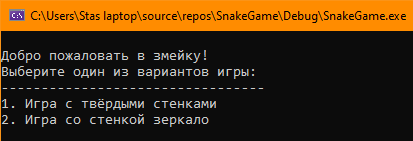
Logic();

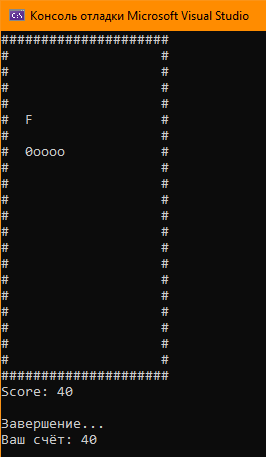
}

cout << "\nЗавершение...\n" << "Ваш счёт: " << score << endl;

return 0;

}

Результат:

НЕ ЗАБЫТЬ ПРО ФУНКЦИЮ НОВУЮ, ДЛЯ ОТСЛЕЖИВАНИЯ НАЖАТИЯ КЛАВИШ, <CONIO.H>, \_KBHIT()