**Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации**

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Сибирский государственный университет телекоммуникаций и информатики»**

**(СибГУТИ)**

**КУРСОВАЯ РАБОТА**

**По дисциплине « Программирование (часть 2)»**

**Игра «Змейка» на два пользователя**

Выполнили: Дмитриев Антон Александрович

Дмитриев Егор Александрович

Группа: ИП-213

Проверил: Агалаков Антон Александрович

**Новосибирск, 2021**

Задание:

Разработать игру «Змейка» на пользователя. Два игрока должны одновременно управлять своей змейкой.

Оглавление

[Введение 2](#_Toc136002059)

[1. Постановка комплекса задач 2](#_Toc136002060)

[2. Проектный раздел 3](#_Toc136002061)

[3. Исходный код программы 4](#_Toc136002062)

[4. Результат тестирования программы 9](#_Toc136002063)

# Введение

Данная работа посвящена разработке программ на языке программирования высокого уровня Си. Целью курсовой работы является закрепление основ и углубление знаний приемов программирования на языке Си, получение практических навыков в создании программного продукта.

Задачей курсовой работы является разработка программы для игры в «Змейка» на два пользователя.

# Постановка комплекса задач

Задача: Разработка программы для игры в «Змейка» на два пользователя, удовлетворяющей следующим условиям:

1. Отрисовка игрового поля с помощью библиотеки graphics.h.

2. Одновременная игра и управление двумя пользователями.

3. Змеи должны развиваться независимо друг от друга и быть способны уничтожать друг друга.

# Проектный раздел

При решении данных задач интерфейс пользователя реализован в основной программе. При запуске на экран выводится меню, в котором последовательно выводятся действия из списка:

1. Выбор цвета поля.

2. Выбор цвета стен.

3. Выбор цвета яблока.

4. Выбор цвета первой змейки.

5. Выбор цвета второй змейки.

После выбора пользователем цветов интерфейса запускается графическое окно.

Управление левой змейки реализовано на клавиши WASD, а правой змейки на «Стрелки»

**Описание отдельных функций**

Функция endf() отвечает за вывод результата игры

Функция check\_walls() отвечает за проверку на столкновение со стенами

Функция check\_tail() отвечает за проверку на столкновение с хвостом противоположной змейки

Функция check\_own\_tail() отвечает за проверку на столкновение с собственным хвостом

### Исходный код программы

#include <graphics.h>

#include <time.h>

#include <iostream>

#include <windows.h>

#include <stdlib.h>

using namespace std;

int endf(int first\_snake\_score, int second\_snake\_score, int winer); // функция вывода результата

int check\_walls(int x[], int y[]); // функция проверки на столкновение с стенами

int check\_tail(int x[], int y[], int x2[], int y2[], int length2); // функция проверки на столкновение с хвостом противоположной змейки

int check\_own\_tail(int x[], int y[], int length); // функция проверки на столкновение с собственным хвостом

int main(){

int s1\_collor, s2\_collor, place\_collor, whals\_collor, apple\_collor;

cout << "Collors: BLACK 0, BLUE 1, GREEN 2, CYAN 3, RED 4, MAGENTA 5, BROWN 6, LIGHTGRAY 7, DARKGRAY 8,\n LIGHTBLUE 9, LIGHTGREEN 10, LIGHTCYAN 11, LIGHTRED 12, LIGHTMAGENTA 13, YELLOW 14, WHITE 15\n";

cout << "\nEnter collor for place: ";

cin >> place\_collor;

cout << "\nEnter collor for walls: ";

cin >> whals\_collor;

cout << "\nEnter collor for apple: ";

cin >> apple\_collor;

cout << "\nEnter collor for 1st snake: ";

cin >> s1\_collor;

cout << "\nEnter collor for 2nd snake: ";

cin >> s2\_collor;

int rx=180, ry=180, x[200], y[200], d, d2;

int x2[200], y2[200];

for(int i = 0; i < 200 ; i++){

x[i]=0;

y[i]=0;

}

for(int i = 0; i < 200 ; i++){

x2[i] = 0;

y2[i] = 0;

}

int first\_snake\_score = 0; // начальный счёт

int second\_snake\_score = 0;

int dir = 1; // направления

int dir2 = 1;

initwindow(647, 492);

x[0] = 200, y[0] = 200; // начальное положение з-мей

x2[0] = 400, y2[0] = 200;

d = 1; // сторона движения

d2 = 1;

int length = 1;

int length2 = 1;

for(;;){

setfillstyle(1, place\_collor); // рисование поля

bar(0, 0, 640, 480);

setfillstyle(1, whals\_collor);

bar(0, 0, 640, 10);

bar(0, 0, 10, 480);

bar(0, 480, 640, 470);

bar(630, 10, 640, 480);

if(x[0]==rx && y[0]==ry){ // проверка на сЪедание яблока 1 змеёй

length += 1;

first\_snake\_score += 1;

rx = ((rand() % 62) + 1) \* 10;

ry = ((rand() % 46) + 1) \* 10;

}

if(x2[0]==rx && y2[0]==ry){ // проверка на сЪедание яблока 2 змеёй

length2 += 1;

second\_snake\_score += 1;

rx = ((rand() % 62) + 1) \* 10;

ry = ((rand() % 46) + 1) \* 10;

}

setfillstyle(1,apple\_collor);

bar(rx,ry,rx+10,ry+10);

if(GetAsyncKeyState('D')) {

d = 1; // движение вправо

}

else if(GetAsyncKeyState('A')) {

d = 2; // движение влево

}

else if(GetAsyncKeyState('W')) {

d = 3; // движение вверх

}

else if(GetAsyncKeyState('S')) {

d = 4; // движение вниз

}

else{

d = 0;

}

if(GetAsyncKeyState(VK\_RIGHT)){

d2 = 1;

}

else if(GetAsyncKeyState(VK\_LEFT)){

d2 = 2;

}

else if(GetAsyncKeyState(VK\_UP)){

d2 = 3;

}

else if(GetAsyncKeyState(VK\_DOWN)) {

d2 = 4;

}

else{

d2 = 0;

}

switch(d){

case 0:

if(dir == 1){

x[0] = x[0] + 10;

}

else if(dir == 2){

x[0] = x[0] - 10;

}

else if(dir == 3){

y[0] = y[0] - 10;

}

else if(dir == 4) {

y[0] = y[0] + 10;

}

else{

d = 0;

}

break;

case 1:

x[0] = x[0] + 10;

dir = 1;

break;

case 2:

x[0] = x[0] - 10;

dir = 2;

break;

case 3:

dir = 3;

y[0] = y[0] - 10;

break;

case 4:

dir = 4;

y[0] = y[0] + 10;

break;

}

switch(d2){

case 0:

if(dir2 == 1){

x2[0] = x2[0] + 10;

}

else if(dir2 == 2){

x2[0] = x2[0] - 10;

}

else if(dir2 == 3){

y2[0] = y2[0] - 10;

}

else if(dir2 == 4) {

y2[0] = y2[0] + 10;

}

else{

d2 = 0;

}

break;

case 1:

x2[0] = x2[0] + 10;

dir2 = 1;

break;

case 2:

x2[0] = x2[0] - 10;

dir2 = 2;

break;

case 3:

dir2 = 3;

y2[0] = y2[0] - 10;

break;

case 4:

dir2 = 4;

y2[0] = y2[0] + 10;

break;

}

for(int i = 0; i < length; i++){ // рисование хвоста 1 змейки

setfillstyle(1, s1\_collor); // цвет 2 змейки

bar(x[i], y[i], x[i]+10, y[i]+10);

}

for(int i = 199; i > 0; i--){ // движение хвоста 1 змейки

x[i] = x[i - 1];

y[i] = y[i - 1];

}

for(int i = 0; i < length2; i++) { // рисование хвоста 2 змейки

setfillstyle(1, s2\_collor); // цвет 2 змейки

bar(x2[i], y2[i], x2[i] + 10, y2[i] + 10);

}

for(int i = 199; i > 0; i--) { // движение хвоста 2 змейки

x2[i] = x2[i - 1];

y2[i] = y2[i - 1];

}

delay(100);

if(check\_walls(x, y) == 1){

if(check\_walls(x2, y2) == 1){

endf(first\_snake\_score, second\_snake\_score, 2);

}

if(check\_tail(x2, y2, x, y, length) == 1){

endf(first\_snake\_score, second\_snake\_score, 2);

}

if(check\_own\_tail(x2, y2, length2) == 1){

endf(first\_snake\_score, second\_snake\_score, 2);

}

endf(first\_snake\_score, second\_snake\_score, 1);

}

if(check\_walls(x2, y2) == 1){

if(check\_tail(x, y, x2, y2, length2) == 1){

endf(first\_snake\_score, second\_snake\_score, 2);

}

if(check\_own\_tail(x, y, length) == 1){

endf(first\_snake\_score, second\_snake\_score, 2);

}

endf(first\_snake\_score, second\_snake\_score, 0);

}

if(check\_own\_tail(x, y, length) == 1){

if(check\_own\_tail(x2, y2, length2) == 1){

endf(first\_snake\_score, second\_snake\_score, 2);

}

if(check\_tail(x2, y2, x, y, length) == 1){

endf(first\_snake\_score, second\_snake\_score, 2);

}

endf(first\_snake\_score, second\_snake\_score, 1);

}

if(check\_own\_tail(x2, y2, length2) == 1){

if(check\_tail(x, y, x2, y2, length2) == 1){

endf(first\_snake\_score, second\_snake\_score, 2);

}

endf(first\_snake\_score, second\_snake\_score, 0);

}

if(check\_tail(x, y, x2, y2, length2) == 1){

if(check\_tail(x2, y2, x, y, length) == 1){

endf(first\_snake\_score, second\_snake\_score, 2);

}

endf(first\_snake\_score, second\_snake\_score, 1);

}

if(check\_tail(x2, y2, x, y, length) == 1){

endf(first\_snake\_score, second\_snake\_score, 0);

}

if(x2[0] == x[0] && y2[0] == y[0]){ // проверка на сталкивание головами

endf(first\_snake\_score, second\_snake\_score, 2);

}

}

}

int endf(int first\_snake\_score, int second\_snake\_score, int winer){

closegraph();

system("cls");

if(winer == 0){

cout << "The first snake won\n";

cout << "First snake score is : " << first\_snake\_score;

cout << "\nSecond snake score is : " << second\_snake\_score;

}

else if(winer == 1){

cout << "The second snake won\n";

cout << "First snake score is : " << first\_snake\_score;

cout << "\nSecond snake score is : " << second\_snake\_score;

}

else if(winer == 2){

cout << "Draw\n";

cout << "First snake score is : " << first\_snake\_score;

cout << "\nSecond snake score is : " << second\_snake\_score;

}

getch();

return 0;

}

int check\_walls(int x[], int y[]){

if(x[0] >= 640 || x[0] <= 0 || y[0] <= 0 || y[0] >= 480){

return 1;

}

return 0;

}

int check\_tail(int x[], int y[], int x2[], int y2[], int length2){

for(int i = 1; i < length2; i++){

if(x[0] == x2[i] && y[0] == y2[i]){

return 1;

}

}

return 0;

}

int check\_own\_tail(int x[], int y[], int length){

for(int i = 2; i < length; i++){

if(x[0] == x[i] && y[0] == y[i]){

return 1;

}

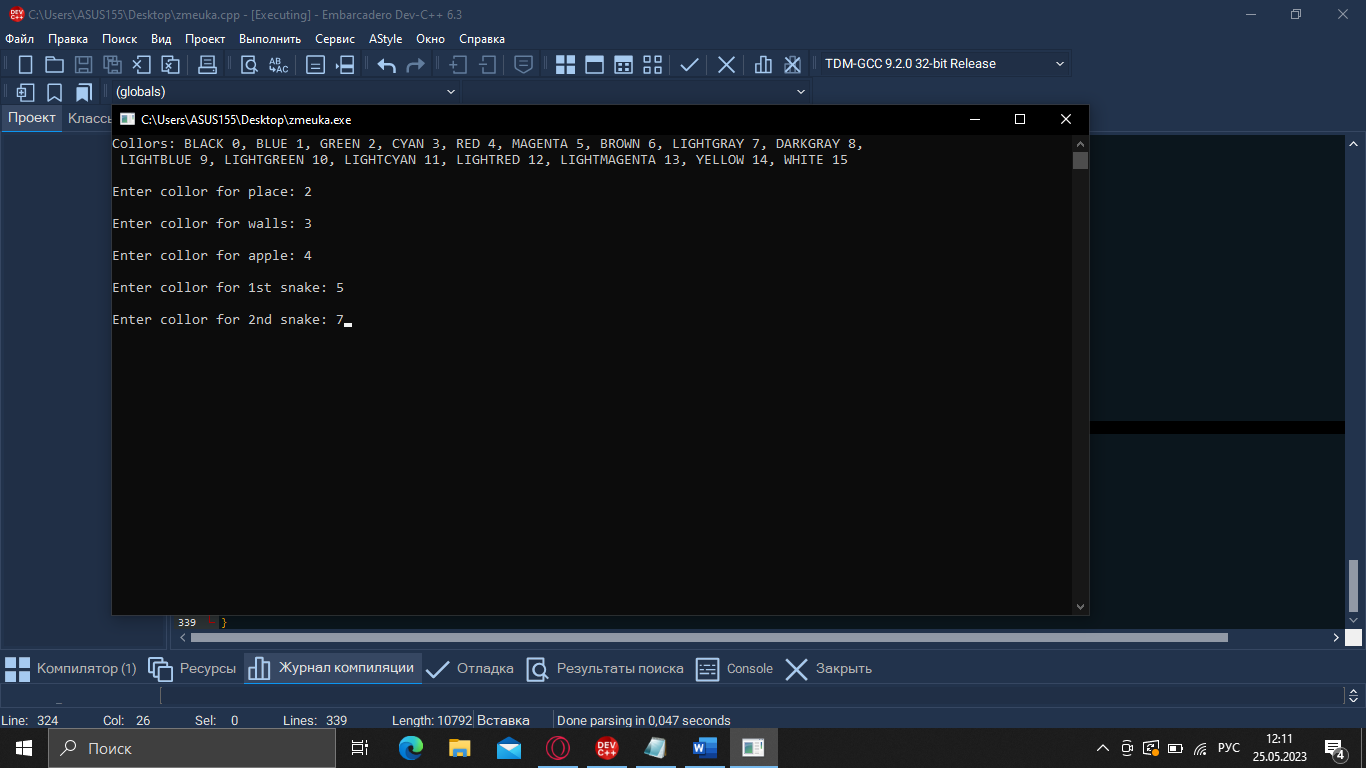
}

return 0;

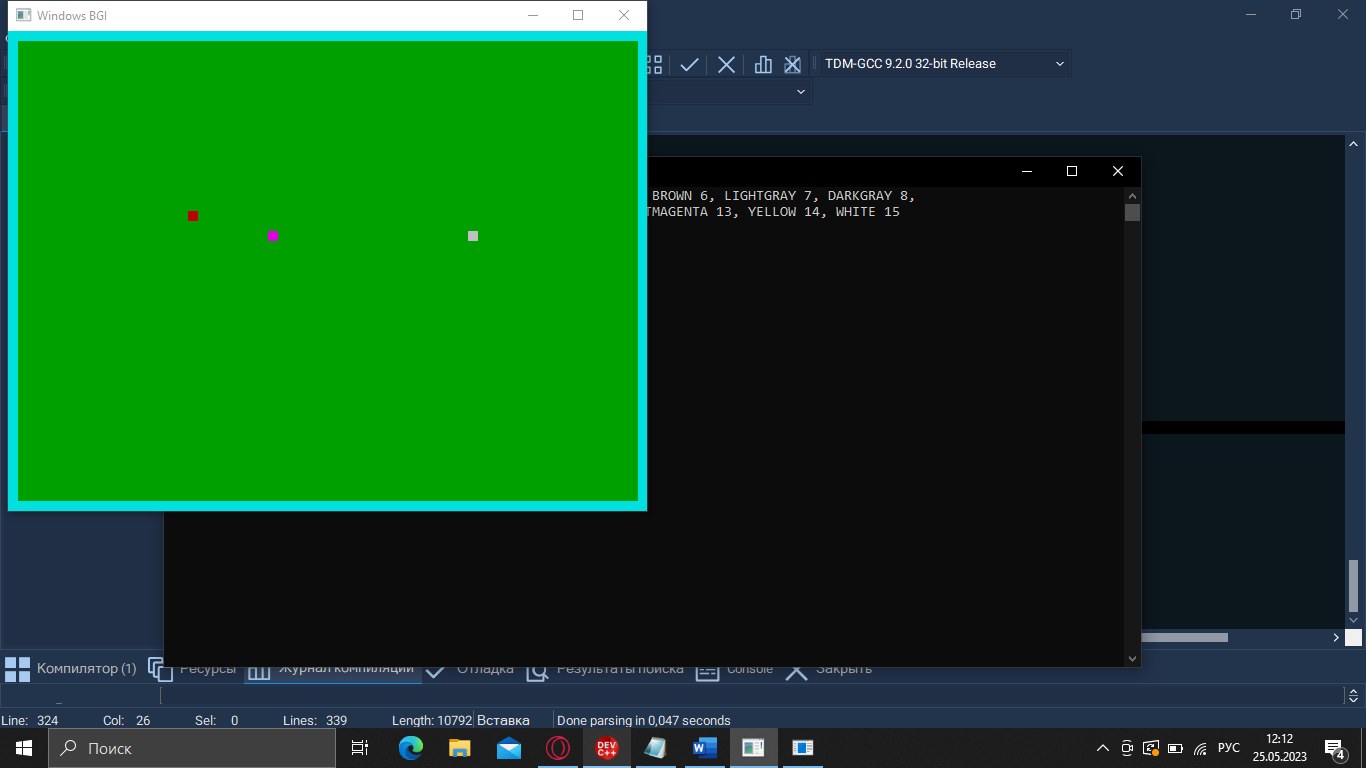
}

# Результат тестирования программы

Запуск программы и выбор цветов интерфейса



Запуск графического окна



Конец игры

