Министерство образования Российской Федерации

ФГБОУ ПО «Пензенский государственная университета»

Факультет вычислительной техники

Кафедра «Вычислительная техника»

Пояснительная записка

к курсовой работе

по курсу «Программирование»

на тему: «Разработка программы сложной структуры методом нисходящего программирования. Игра Тетрис»

01.06.2025 Генералова

Выполнил: ст. гр. 24ВВВ4

Чернышевский Е.И.

Проверил: к.т.н., доцент

Генералова А.А.

Пенза, 2025

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Факультет Вычислительной техники

Кафедра "Вычислительная техника"

"УТВЕРЖДАЮ"

Зав. кафедрой ВТ

_____ М.А. Митрохин

«14» февраля 2015г

ЗАДАНИЕ

на курсовое проектирование по курсу

Программирование

Студенту Черниневскому Евгению Игоревичу Группа 24ВВВ4

Тема проекта: ***Разработка программы сложной структуры методом нисходящего программирования.***

Исходные данные (технические требования) на проектирование

Игра „Петушс"
Работа с файлами. Таблица рекордов

Обязательные требования к программе:

1. Многомодульность.
2. Использование сложных типов данных – структур, списков, файлов.
3. Режим работы видеосистемы – текстовый / графический.
4. Устройство ввода информации – клавиатура, мышь.
5. Пользовательский интерфейс должен быть построен на основе меню и панели инструментов.
6. Наличие заставки.
7. Операционная система MS Windows.
8. Язык программирования – Си и Ассемблер.
9. Среда разработки ПО – Microsoft Visual Studio.

Объем работы по курсу

1. **Расчетная часть**

Разработка программы.

2. **Графическая часть**

Схема данных, схема ресурсов системы, схема работы системы, иерархическая структура программы, схема взаимодействия программы.

3. **Экспериментальная часть**

Отладка программы.

Срок выполнения проекта по разделам

В соответствии с графиком.

Дата выдачи задания "14" февраля 25

Дата защиты проекта "10" июня 25

Руководитель Тенеголова А. А

Задание получил "14" февраля 2025 г.

Студент Черношивский Евгений Игоревич

Министерство образования Российской Федерации
ФГБОУ ПО Пензенский государственный университет
Кафедра «Вычислительная техника»

Пояснительная записка

к курсовой работе

по курсу «Программирование»

на тему: «Разработка программы сложной структуры методом нисходящего программирования. Игра Тетрис»

Выполнил: ст. гр. 24ВВВ4
Чернышевский Е.И.
Проверил: к.т.н. доцент
Генералова А.А.

Пенза, 2025

Содержание

Введение 3
1.Постановка задачи 4
2.Выбор решения 5
3.Описание разработки программы 6
4.Отладка и тестирование 7
5.Описание программы 8
5.1 Схема функции main 8
5.2 Схема функции Game_over 9
5.3 Схема функции DelLine 9
5.4 Описание модулей. 10
6. Руководство пользователя 12
6.1 Заставка 12
6.2 Главное меню 13
6.3 Начало игры 14
6.4 Удаление линии 15
6.5 Проигрыш 16
6.6 Сохранение в файл 17
Заключение 18
Список используемых материалов 19
Приложение А. Листинг программы. 20
Файл Tetris.cpp 20
Файл Asm.cpp 28
Файл Data.h 28

Введение

Компьютерные технологии уже давно укоренились в нашей жизни. Многие привычные для нас вещи были бы невозможны без программирования.

Язык программирования Си — это универсальный язык с богатым набором операторов и компактным способом записи выражений. Благодаря гибкости, выразительности и компактности своих конструкций Си завоевал наибольшую популярность в среде профессиональных программистов и широко используется при разработке системных и прикладных программ.

В данной работе продемонстрированы базовые возможности языка СИ вкупе с ассемблерными вставками. Разработка велась в среде Microsoft Visual Studio 2022.

1.Постановка задачи

Необходимо разработать игру тетрис на языке программирования Си «С» с использованием ассемблерной вставки. Игра должна включать в себя основные элементы и правила, характерные для классической версии тетриса. Кроме того, стоит уделить внимание реализации функций управления, реализации функций подсчета очков. Для успешного выполнения задачи, необходимо провести тестирование игры на различных тестовых данных и проанализировать работу игры в различных условиях. Данная программа осуществляет вывод на экран случайным образом падение различных фигур. Входными данными является управление падающими фигурами.

Многомодульность программы. Программа должна быть поделена на логические модули. Это упростит поиск ошибок при отладке и тестировании программы, а также позволит легко расширять функционал программы.

Использование сложных типов данных – массивов, структур, файлов.

Режим работы видеосистемы – текстовый/графический. Для начала необходимо определиться с типом интерфейса и с элементами управления, затем необходимо изучить способы их реализации. Устройство ввода-вывода – клавиатура и мышь. Необходимо различать и идентифицировать действия, произведенные с их помощью, это облегчит использование программы.

Интерфейс должен быть построен на основе меню. Это необходимо для создания интуитивно понятного интерфейса для пользователя.

2.Выбор решения

При запуске программы открывается приветственный экран. Далее появляется меню, состоящее из 2 пунктов:

1. Кнопка начать игру. Пользователь нажимает кнопку «R», после чего генерируется карта и фигуры. Далее начинается игра. Управление происходит с помощью кнопок «WASD», где кнопка «W» переворачивает фигуру, кнопка «S» ускоряет спуск фигуры вниз, кнопка «A» перемещает фигуру влево, кнопка «D» перемещает фигуру вправо. Данный выбор управления является стандартом для большинства игр.

2. Кнопка выхода из игры. Пользователь нажимает кнопку «F», и программа завершает работу. ы

3.Описание разработки программы

Для написания данной программы будет использован язык программирования Си. Этот язык является распространённым языком программирования.

В качестве среды программирования был выбран программный продукт Visual Studio 2022.

4.Отладка и тестирование

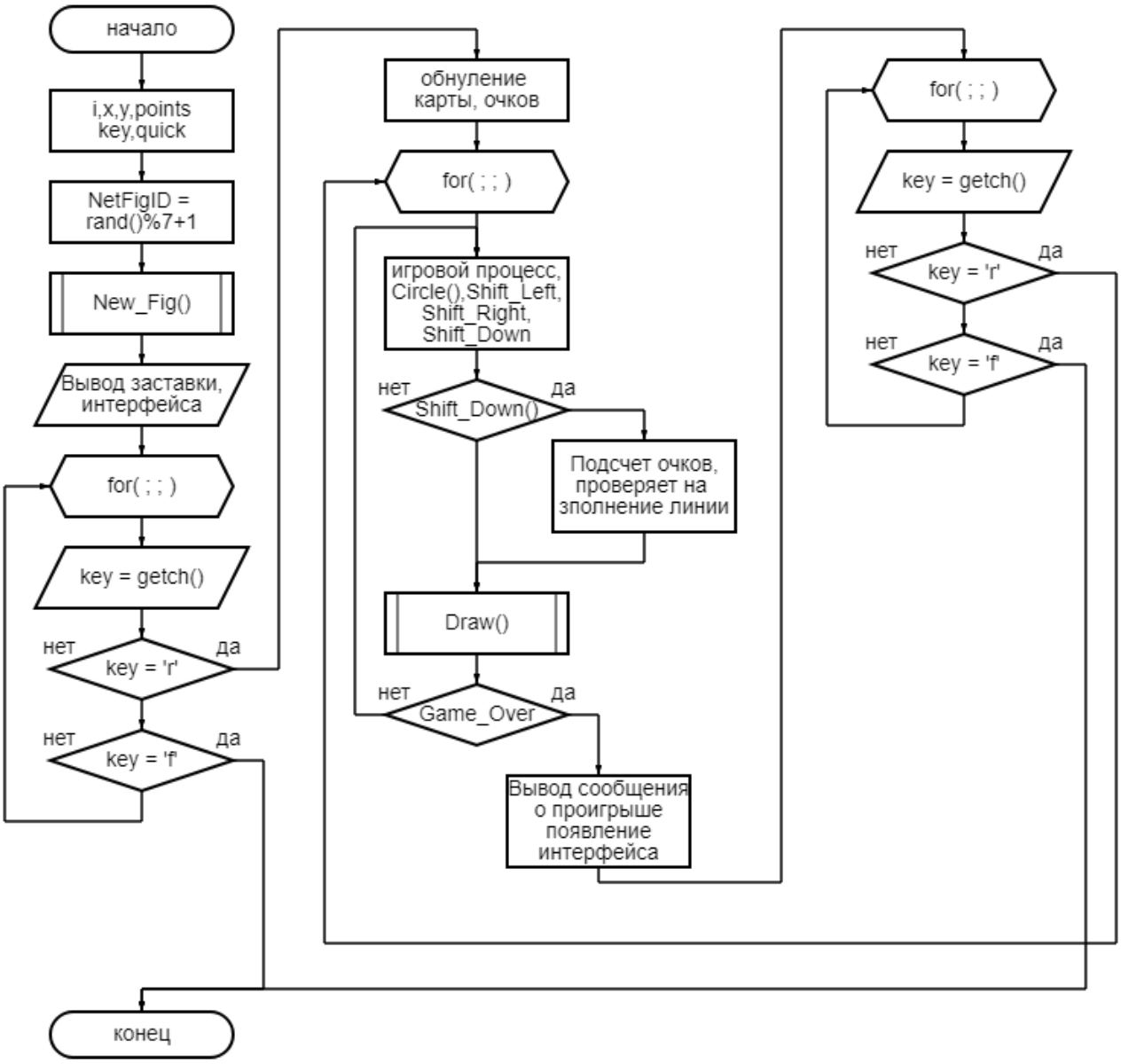
В качестве среды разработки была выбрана программа Microsoft Visual Studio 2022. Программа обладает всеми средствами необходимыми при разработке и отладке программы.

Для отладки использовались несколько возможностей Visual Studio: точка останова, трассировка, анализ содержимого переменных.

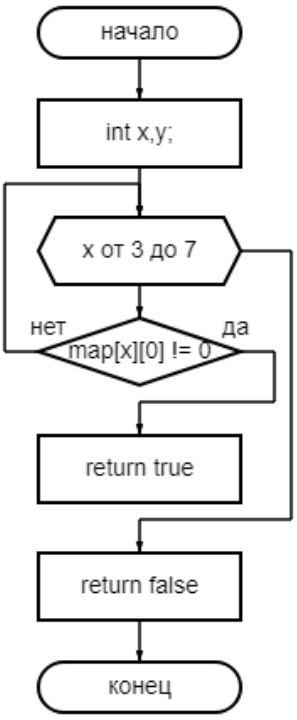
Тестирование проводилось во время разработки и также после завершения разработки. В ходе нее было выявлено огромное количество проблем, связанных с работой с файлами, работой с памятью, работой с полями структуры, с размерами самих массивов и т. д.

5.Описание программы

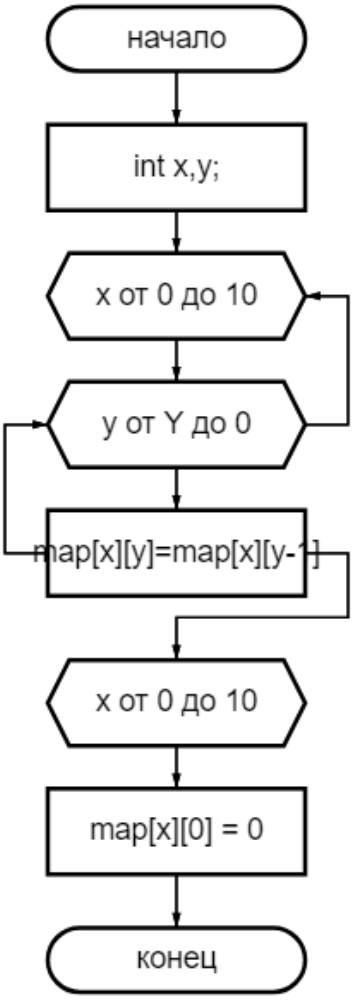
5.1 Схема функции main



5.2 Схема функции Game_over



5.3 Схема функции DelLine



5.4 Описание модулей.

1) Tetris.cpp – основной файл программы.

1.1) main() – функция, в которой реализовано основное меню и игровой процесс игры.

1.2) New_Fig() – функция, в которой реализовано создание новой фигуры для игры из семи возможных вариантов. Она также устанавливает тип текущей и следующей фигуры, чтобы они отображались на экране с разными цветами фона.

1.3) Draw() – функция, отвечает за рисование игрового поля, текущей фигуры, следующей фигуры и информации о текущем счете, рекорде и количестве собранных линий.

1.4) Circle() – функция, отвечает за поворот текущей фигуры на 90 градусов по часовой стрелке.

1.5) Shift_Right() – функция, отвечает за сдвиг текущей фигуры на одну клетку вправо.

1.6) Shift_Left() – функция, отвечает за сдвиг текущей фигуры на одну клетку влево.

1.7) Shift_Down() – функция, отвечает за сдвиг текущей фигуры на одну клетку вниз.

1.8) Draw_Fig_Map() – функция, рисует текущую фигуру на игровом поле.

1.9) DelLine() – функция, удаляет заполненную линию на игровом поле.

1.10) CheckLine() – функция, проверяет каждую строку на заполнение, на игровом поле.

1.11) Game_Over() – функция, проверяет, проиграл ли игрок.

1.12) record() – функция, записывает в файл значение переменной, которое является рекордом игрока.

1.13) recordmax() – функция, считывает рекорд игрока из файла.

1.14) setpos() – функция, выводит информацию в нужном месте.

1.15) FigCpy() – функция, копирует содержимое одного массива в другой для дальнейшей работы с ним.

1.16) rect() – функция, отрисовывает прямоугольник заданным цветом на экране.

1.17) Vuvod() – функция, выводит символы на экран заданным цветом.

2) Data.h – заголовочный файл, в котором объявляются функции и подключаются библиотеки.

3) Asm.cpp – файл, в котором реализована ассемблерная вставка.

3.1) SumScore() – функция, прибавляет очки к текущему рекорду игрока.

```
#include "Data.h"

int SumScore(int score)
{
    _asm
    {
        mov eax, score;
        add eax, 10;
        mov score, eax;
    }
    return  score;
}
```

6. Руководство пользователя

Игра Tetris.exe предназначена для интересного проведения досуга. Тетрис представляет из себя классическую головоломку, в которой игроки манипулируют падающими разноцветными геометрическими фигурами, состоящими из четырех квадратных блоков, создавая горизонтальные линии без каких-либо пробелов. Игра бесконечна, и цель состоит в том, чтобы набрать как можно больше очков, заполняя линии, очищая тетрисы и комбинируя. Игра имеет понятный интерфейс и интуитивно понятное управление

6.1 Заставка

Рисунок 1 – Заставка

6.2 Главное меню

Главное меню. Игрок делает выбор и нажимает кнопку.

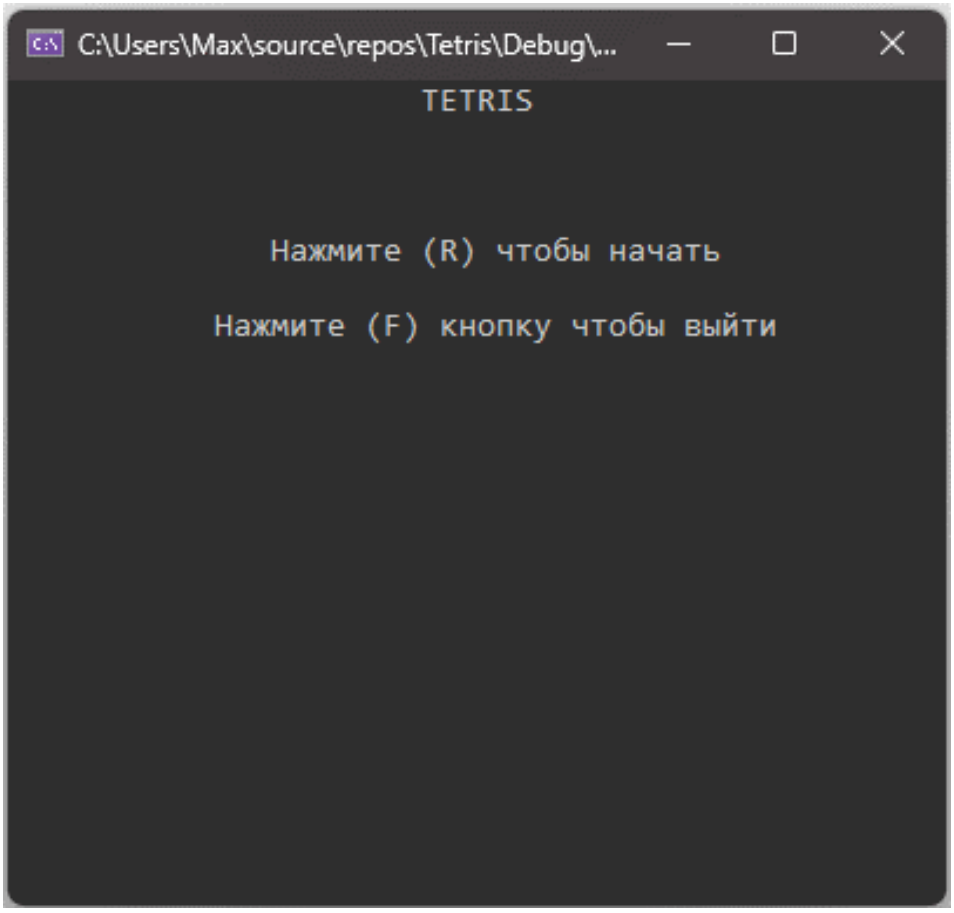


Рисунок 2 - Главное меню

6.3 Начало игры

Отображается поле игры, очки, рекорд, линии, главная и следующая фигура.

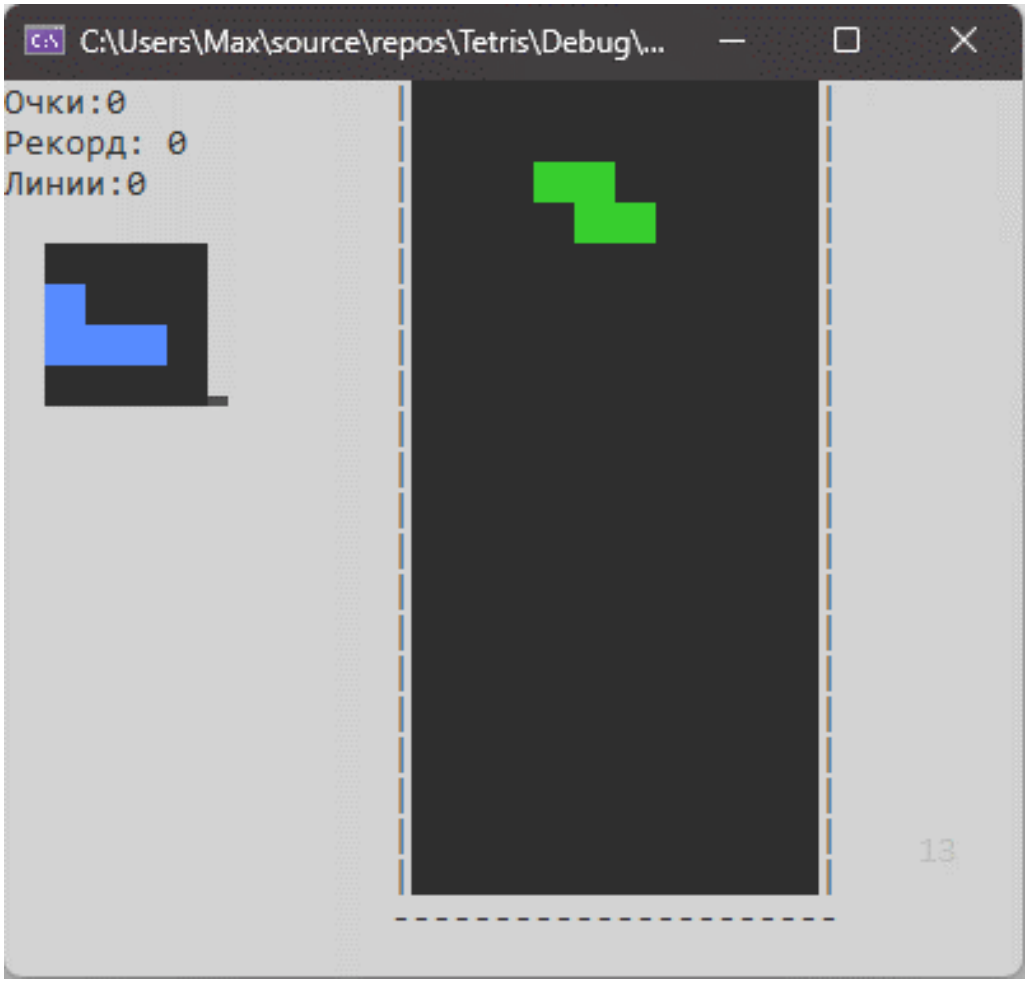
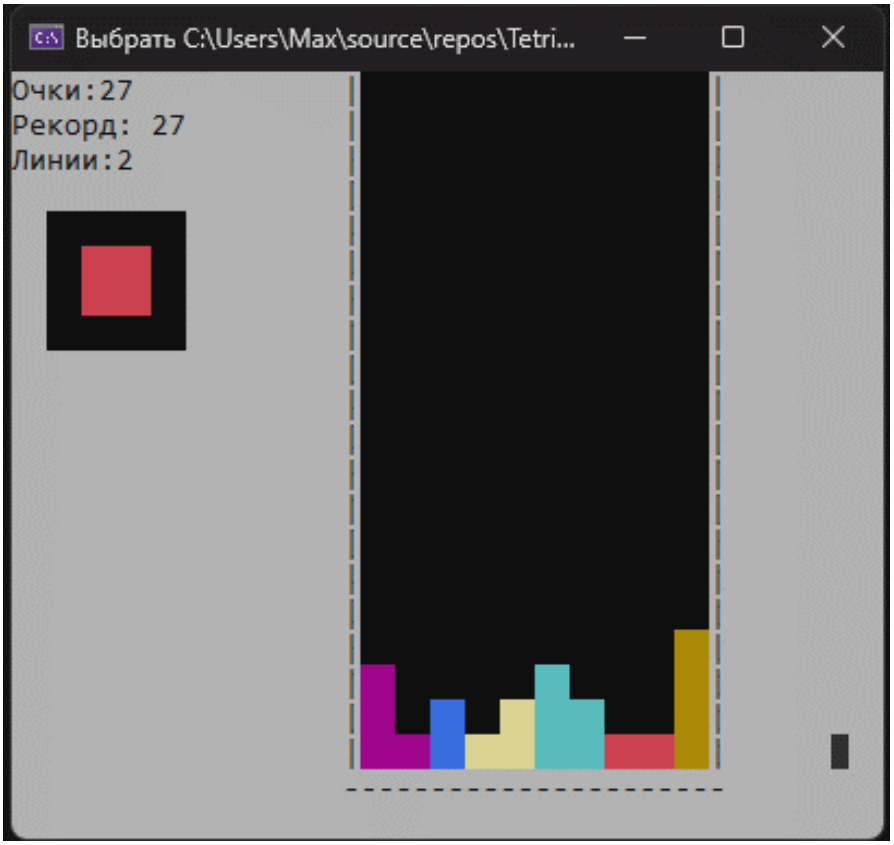
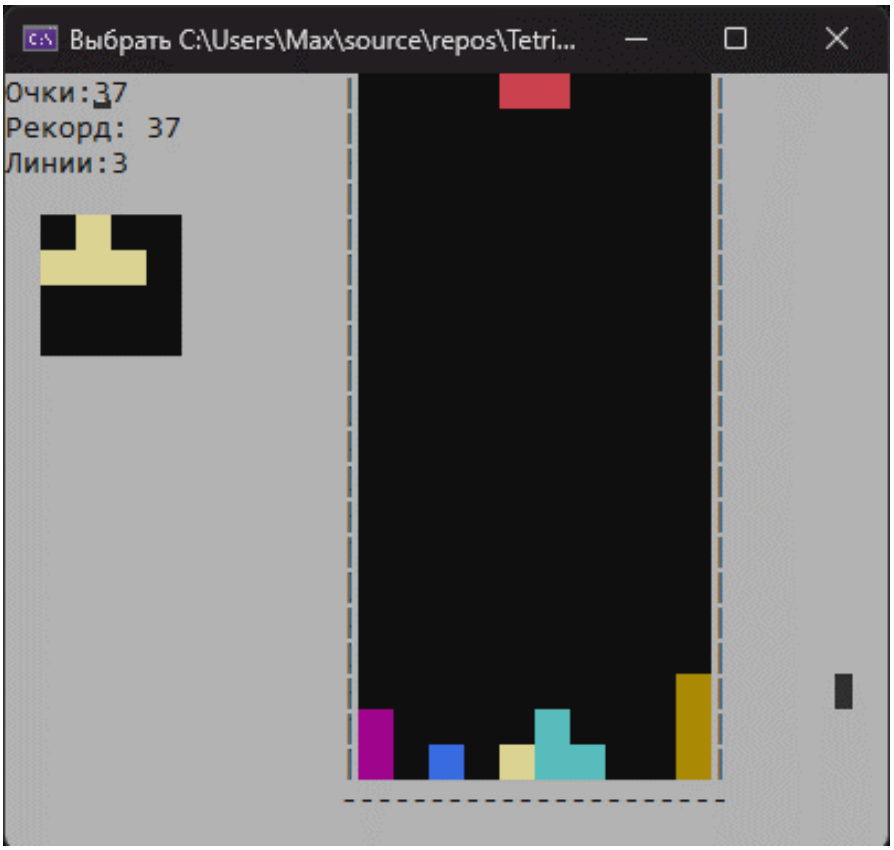


Рисунок 3 – Ход игры

6.4 Удаление линии

Когда линия заполняется, она удаляется.





Рисунки 4,5 – Удаление линии

6.5 Проигрыш

При пересечении верхнего уровня игрок проигрывает.

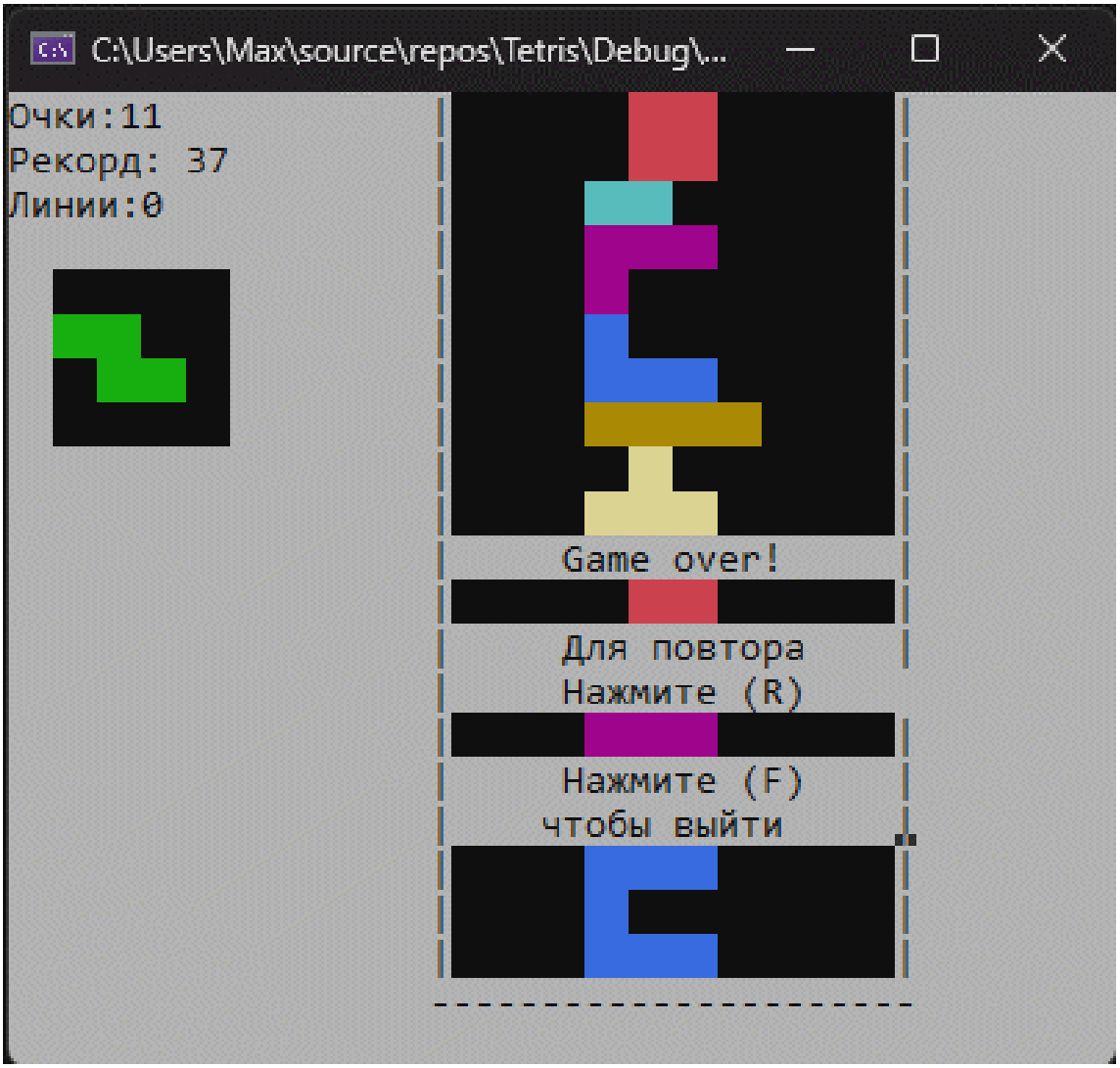


Рисунок 6 - Проигрыш

6.6 Сохранение в файл

Сохранения рекорда игрока в файл.

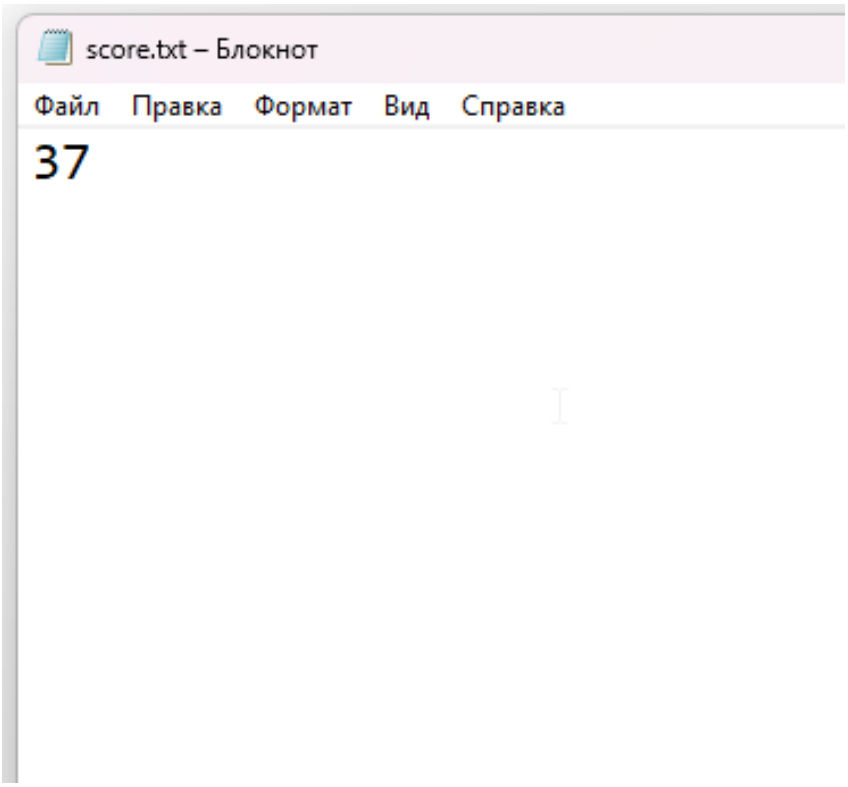


Рисунок 7 – Файл score.txt

Заключение

При выполнении данной курсовой работы были получены навыки разработки многомодульных программ. Были освоены способы обработки событий от клавиатуры. Усвоены принципы реализации интерфейса. Получены базовые навыки программирования на языке С и Ассемблер. Изучены возможность среды программирования Visual Studio 2022. Получены навыки отладки и тестирования программ.

В рамках выполнения курсовой работы была написана программа (игра) «Тетрис». Программа не требует большого количества системных ресурсов, обеспечивает быструю обработку данных и мгновенный показ результата пользователю.

Программу можно улучшить, оптимизировав код для более быстрого выполнения функций и разнообразив игру.

Список используемых материалов

1. Дэвид Гриффитс, Дон Гриффитс. Изучаем программирование на C. 2013 г.
2. https://habr.com/ru/articles/464075
3. https://learn.microsoft.com/ru-ru/cpp/c-language/?view=msvc-170
3. https://learnc.info/c/functions.html
5. Брайан Керниган. Деннис Ритчи. Язык программирования C. 2009 г.
6. Википедия – свободная энциклопедия

Приложение А. Листинг программы.

Файл Tetris.cpp

```
#include "Data.h"

bool figure[4][4] = { }, _next_fig[4][4] = { };
unsigned char map[10][20] = { }; // карта
struct Cur {// координаты фигуры
    int  x = 3;
    int  y = -3;
};
struct Cur CurFig;

int  CurFigType = 0, NextFigType = 0, NextFigID, score = 0, lines = 0;
const bool // фигуры
__Fig0[4][4] = {
  {0, 0, 0, 0},
  {0, 1, 1, 0},
  {0, 1, 1, 0},
  {0, 0, 0, 0}
},
__Fig1[4][4] = {
  {0, 1, 0, 0},
  {1, 1, 0, 0},
  {0, 1, 0, 0},
  {0, 0, 0, 0}
},
__Fig2[4][4] = {
  {0, 1, 1, 0},
  {0, 1, 0, 0},
  {0, 1, 0, 0},
  {0, 0, 0, 0}
},
__Fig3[4][4] = {
  {0, 1, 1, 0},
  {0, 0, 1, 0},
  {0, 0, 1, 0},
  {0, 0, 0, 0}
},
__Fig4[4][4] = {
  {0, 0, 1, 0},
  {0, 1, 1, 0},
  {0, 1, 0, 0},
  {0, 0, 0, 0}
},
__Fig5[4][4] = {
  {0, 1, 0, 0},
  {0, 1, 1, 0},
  {0, 0, 1, 0},
  {0, 0, 0, 0}
},
__Fig6[4][4] = {
  {0, 1, 0, 0},
  {0, 1, 0, 0},
  {0, 1, 0, 0},
  {0, 1, 0, 0}
};
```

```
void record(int a) { // записывает рекорд
    FILE* f = fopen("score.txt", "w");
    fprintf(f, "%d\n", a);
    fclose(f);
}
int recordmax() { // Считывает рекорд
    int r = 0;
    FILE* f = fopen("score.txt", "r");
    if (!f)
        record(0);
    else
        fscanf(f,  "%d", &r);
    fclose(f);
    return r;
}

void setpos(int x, int y) { //положение вывода
    COORD C = { x, y };
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), C);
}

bool Game_Over() { //проверка на проигрыш
    int x, y;
    for (x = 3; x < 7; x++)
        if (map[x][0] != 0)
           // lines = 0;
            return true;
    return false;
}

void circle() { // перевот фигуры
    if (CurFigType == (BACKGROUND_INTENSITY | BACKGROUND_RED)) return;
    int x, y, emptylines = 0;
    bool is3x3 = true, buffer[4][4] = { };
    for (y = 0; y < 4; y++)
        if (figure[3][y])
            is3x3 = false;
    for (x = 0; x < 4; x++)
        if (figure[x][3])
            is3x3 = false;
    if (is3x3) {
        for (x = 0; x < 3; x++)
            for (y = 0; y < 3; y++)
                buffer[x][y] = figure[y][2 - x];
    }
    else {
        for (x = 0; x < 4; x++)
            for (y = 0; y < 4; y++)
                buffer[x][y] = figure[y][3 - x];
    }
    //проверка на переворот
    for (x = 0; x < 4; x++) {
        for (y = 0; y < 4; y++) {
            if (buffer[x][y] && map[CurFig.x + x][CurFig.y + y])
                return;
        }
    }
```

```
    for (x = 0; x < 4; x++) {
        for (y = 0; y < 4; y++) {
            if (buffer[x][y] && (CurFig.x + x > 9 || CurFig.y + y > 19))
                return;
        }
    }
    for (x = 0; x < 4; x++)
        for (y = 0; y < 4; y++)
            figure[x][y] = buffer[x][y];
}

void Shift_Right() { // сдвиг вправо
    int x, y, lastline = 3;
    bool empty = true, possible = true;
    for (x = 3; x > -1; x--) {
        for (y = 0; y < 4; y++)
            if (figure[x][y])
                empty = false;
        if (empty) lastline--;
    }
    if (CurFig.x + 3 != 9)
        for (x = 0; x < 4; x++)
            for (y = CurFig.y; y < CurFig.y + 4; y++)
                if (figure[x][y - CurFig.y] && map[CurFig.x + x + 1][y])
                    possible = false;

    if (CurFig.x < 9 - lastline && possible)
        CurFig.x++;
}

void Shift_Left() { // сдвиг влево
    int x, y, lastline = 0;
    bool empty = true, possible = true;
    for (x = 0; x < 4; x++) {
        for (y = 0; y < 4; y++)
            if (figure[x][y])
                empty = false;
        if (empty) lastline++;
    }
    if (CurFig.x > 0)
        for (x = 0; x < 4; x++)
            for (y = CurFig.y; y < CurFig.y + 4; y++)
                if (figure[x][y - CurFig.y] && map[CurFig.x - 1 + x][y])
                    possible = false;

    if (CurFig.x > 0 - lastline && possible)
        CurFig.x--;
}

bool Shift_Down() { // сдвиг вниз
    bool empty = true, possible = true;
    int x, y, emptylines = 0;
    for (y = 3; y > -1; y--) {
        for (x = 0; x < 4; x++) {
            if (figure[x][y]) {
                empty = false;
                break;
            }
        }
```

```
        if (!empty) break;
        emptylines++;
    }
    for (y = 0; y < 4; y++)
        for (x = CurFig.x; x < CurFig.x + 4; x++)
            if (CurFig.y >= -1)
                if (figure[x - CurFig.x][y] && map[x][CurFig.y + y + 1]) {
                    possible = false;
                    break;
                }
    if (CurFig.y < 16 + emptylines && possible)
        CurFig.y++;
    else return false;
    return true;
}


int main()
{
    srand(time(NULL));
    setlocale(LC_ALL, "Russian");

    int i, x, y, points;
    char key;
    bool quick = false;
    NextFigID = rand() % 7 + 1;
    New_Fig();

    printf("            Курсовая работа\n     По дисциплине 'Программирование' \n    На тему: 'Игра Тетрис'\n\n  Выполнил студент группы 24ВВВ4: Чернышевский Е.И.\n  Проверил: к.т.н. доцент Генералова А.А.");
    Sleep(3000);
    system("cls");
    system("mode con cols=50 lines=22");

    printf("                    TETRIS");
    setpos(1, 4);
    printf("            Нажмите (R) чтобы начать");
    setpos(1, 6);
    printf("          Нажмите (F) кнопку чтобы выйти");

    for ( ; ; )
    {
        key = getch();
        if (key == 'r') break;
        if (key == 'f') return 0;
    }

start:
    score = lines = 0;
    for (x = 0; x < 10; x++)
        for (y = 0; y < 20; y++)
            map[x][y] = 0;

    for ( ; ; ) {
        key = 0;
        for (i = 0; i < (quick ? 10000 : 25000); i++)
            if (kbhit())
            {
```

```
            key = getch();
        }
        quick = false;
        if (key == 27) break; else
            if (key == 'w') circle();
                if (key == 'a') Shift_Left();
                    if (key == 'd') Shift_Right();
                        if (key == 's') quick = true;
                            if (key == '\r') while (Shift_Down());

        if (!Shift_Down()) {
            Draw_Fig_Map();
            CurFig.x = 3;
            CurFig.y = -3;
            lines += (points = CheckLine());
            if (points)
                score = SumScore(score);
            else score++;
            New_Fig();
        }

        Draw();
        if (Game_Over()) {
            setpos(20, 10);
            printf("     Game over!     ");
            setpos(20, 12);
            printf("     Для повтора    ");
            setpos(20, 13);
            printf("     Нажмите (R)    ");
            setpos(20, 15);
            printf("     Нажмите (F)    ");
            setpos(20, 16);
            printf("    чтобы выйти     ");
            for ( ; ; )
            {
                key = getch();
                if (key == 'r') break;
                if (key == 'f') return 0;
            }

            goto start;
        }
    }
}

void FigCpy(bool a[4][4], const bool b[4][4]) {
    int x, y;
    for (y = 0; y < 4; y++)
        for (x = 0; x < 4; x++)
            a[x][y] = b[x][y];
}

void New_Fig() { //новая фигуры
    switch (NextFigID) {
    case 1: FigCpy(figure, __Fig0); CurFigType = (BACKGROUND_INTENSITY |
BACKGROUND_RED); break;
    case 2: FigCpy(figure, __Fig1); CurFigType = (BACKGROUND_INTENSITY |
BACKGROUND_RED | BACKGROUND_GREEN); break;
```

```
    case 3: FigCpy(figure, __Fig2); CurFigType = (BACKGROUND_INTENSITY |
BACKGROUND_BLUE | BACKGROUND_RED); break;
    case 4: FigCpy(figure, __Fig3); CurFigType = (BACKGROUND_INTENSITY |
BACKGROUND_BLUE); break;
    case 5: FigCpy(figure, __Fig4); CurFigType = (BACKGROUND_INTENSITY |
BACKGROUND_BLUE | BACKGROUND_GREEN); break;
    case 6: FigCpy(figure, __Fig5); CurFigType = (BACKGROUND_INTENSITY |
BACKGROUND_GREEN); break;
    case 7: FigCpy(figure, __Fig6); CurFigType = (BACKGROUND_RED | BACKGROUND_GREEN);
    }
    NextFigID = rand()%7+1;
    switch (NextFigID) {
    case 1: FigCpy(_next_fig, __Fig0); NextFigType = (BACKGROUND_INTENSITY |
BACKGROUND_RED); break;
    case 2: FigCpy(_next_fig, __Fig1); NextFigType = (BACKGROUND_INTENSITY |
BACKGROUND_RED | BACKGROUND_GREEN); break;
    case 3: FigCpy(_next_fig, __Fig2); NextFigType = (BACKGROUND_INTENSITY |
BACKGROUND_BLUE | BACKGROUND_RED); break;
    case 4: FigCpy(_next_fig, __Fig3); NextFigType = (BACKGROUND_INTENSITY |
BACKGROUND_BLUE); break;
    case 5: FigCpy(_next_fig, __Fig4); NextFigType = (BACKGROUND_INTENSITY |
BACKGROUND_BLUE | BACKGROUND_GREEN); break;
    case 6: FigCpy(_next_fig, __Fig5); NextFigType = (BACKGROUND_INTENSITY |
BACKGROUND_GREEN); break;
    case 7: FigCpy(_next_fig, __Fig6); NextFigType = (BACKGROUND_RED |
BACKGROUND_GREEN);
    }
}

void rect(int x, int y, int color) {
    if (x < 0 || y < 0) return;
    HANDLE h = GetStdHandle(STD_OUTPUT_HANDLE);
    DWORD cWritten;
    const char fill_char = ' ';
    SetConsoleTextAttribute(h, color);
    setpos(x, y);
    WriteFile(h, &fill_char, 1, &cWritten, NULL);
    SetConsoleTextAttribute(h, 0);
}

void Vuvod(int x, int y, int color) { //вывод
    rect(2 * x, y, color);
    rect(2 * x + 1, y, color);
}

void Draw_Fig_Map() { //рисуте фигуры
    int x, y;
    for (y = CurFig.y; y < CurFig.y + 4; y++) {
        for (x = CurFig.x; x < CurFig.x + 4; x++) {
            if (figure[x - CurFig.x][y - CurFig.y] && x < 10 && x >= 0 && y < 20 && y >= 0)
                map[x][y] = CurFigType * (int)figure[x - CurFig.x][y - CurFig.y];
        }
    }
}

void DelLine(int Y) { // удаляет линию
    int x, y;
    for (x = 0; x < 10; x++) {
        for (y = Y; y > 0; y--) {
```

```
            map[x][y] = map[x][y - 1];
        }
    }
    for (x = 0; x < 10; x++)
        map[x][0] = 0;
}

int CheckLine() {  // количество удаленых строк
    int x, y, lines = 0;
    bool full;
    for (y = 0; y < 20; y++) {
        full = true;
        for (x = 0; x < 10; x++)
            if (!map[x][y]) {
                full = false;
                break;
            }
        if (full) {
            lines++;
            SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),BACKGROUND_BLUE |
BACKGROUND_RED | BACKGROUND_GREEN | BACKGROUND_INTENSITY);
            setpos(20, y);
            printf("                    ");
            Sleep(50);
            SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),BACKGROUND_BLUE |
BACKGROUND_RED | BACKGROUND_GREEN);

            DelLine(y);
        }
    }
    return lines;
}

void Draw() { //риссует всё
    int x, y;
    HANDLE h = GetStdHandle(STD_OUTPUT_HANDLE);
    SetConsoleTextAttribute(h, BACKGROUND_BLUE | BACKGROUND_RED |
BACKGROUND_GREEN);
    static bool done = false;
    if (!done) {
        done = true;
        system("cls");
        printf("Очки: \nРекорд:\nЛинии: ");
        for (y = 0; y < 20; y++) {
            setpos(19, y); putchar('|');
            setpos(40, y); putchar('|');
        }
        setpos(19, 20);
        for (x = 0; x < 22; x++) putchar('-');
    }
    if (recordmax() < score) record(score);
    setpos(5, 0); printf("%d", score);
    setpos(8, 1); printf("%d", recordmax());
    setpos(6, 2); printf("%d", lines);
    SetConsoleTextAttribute(h, 0);
    for (y = 0; y < 20; y++) {// рисует карту
        for (x = 0; x < 10; x++) {
            Vuvod(10 + x, y, map[x][y]);
        }
```

```
    }
    for (y = CurFig.y; y < CurFig.y + 4; y++) { // рисует фигуру
        for (x = CurFig.x; x < CurFig.x + 4; x++) {
            if (figure[x - CurFig.x][y - CurFig.y])
                Vuvod(10 + x, y, CurFigType);
        }
    }
    SetConsoleTextAttribute(h, 0);
    for (y = 0; y < 4; y++) {
        for (x = 0; x < 4; x++) {
            Vuvod(x + 1, y + 4, NextFigType * (int)_next_fig[x][y]);
        }
    }
    SetConsoleTextAttribute(h, BACKGROUND_BLUE | BACKGROUND_RED |
BACKGROUND_GREEN);
}
```

Файл Asm.cpp

```
#include "Data.h"

int SumScore(int score) //начисляет очки
{
_asm
{
mov eax, score;
add eax, 10;
mov score, eax;
}
return  score;
}
```

Файл Data.h

```
#pragma once

#ifndef DATA_H
#define DATA_H
#define  _CRT_SECURE_NO_WARNINGS

#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#include <windows.h>
#include <time.h>
#include <iostream>
#include <conio.h>
#include <iostream>

bool Game_Over();

int  CheckLine(void);
void Draw_Fig_Map(void);
void Draw(void);
void Draw2(void);
void New_Fig(void);
int SumScore(int score);
#endif //DATA.H
```