*Россия, ХМАО-Югра, Тюменская область, п.Березово*

*Муниципальное бюджетное образовательное учреждение*

*Березовская средняя общеобразовательная школа*

**Исследовательская работа**

**«Анализ возможности создания игры «Тетрис»**

**на языке программирования FreePascal»**

Автор:

Баженов Дмитрий Юрьевич

учащийся 10«Б» класса

МБОУ Березовская средняя общеобразовательная школа

Руководитель:

Кулбаева Мария Михайловна

учитель информатики

МБОУ Березовская средняя общеобразовательная школа

2015г.

Оглавление

[Аннотация 3](#_Toc416992752)

[План исследования 4](#_Toc416992753)

[Описание работы 6](#_Toc416992754)

[История создания игры «Тетрис» 8](#_Toc416992755)

[*Игра Пентамино - родоначальница Тетриса* 8](#_Toc416992756)

[*Первый «Тетрис»* 9](#_Toc416992757)

[Правила игры "Тетрис" 10](#_Toc416992758)

[Реализация "Тетриса" на разных платформах 11](#_Toc416992759)

[Результаты социологических опросов и анализ информации в сети Интернет 12](#_Toc416992760)

[Анализ возможности создания игры «Тетрис» на языке программирования FreePascal 13](#_Toc416992761)

[Заключение 15](#_Toc416992762)

[Список использованой литературы: 16](#_Toc416992763)

[Приложение 17](#_Toc416992765)

*Исследовательская работа «Анализ возможности создания игры «Тетрис»*

*на языке программирования FreePascal»*

*Баженов Дмитрий Юрьевич,*

*Россия, ХМАО-Югра Тюменская область, п.Березово*

*Муниципальное бюджетное образовательное учреждение*

*Березовская средняя общеобразовательная школа 9 класс*

# Аннотация

Игра – это не только любимое и главное занятие для каждого ребенка, но и постоянный вид деятельности, в котором дети проводят большую часть времени. Роль игры, как формирующую составляющую в поведении ребенка подтверждают многие ученые, психологи и педагоги нашего времени. Самые [развивающие игры](http://345-games.ru/), которые помогают подготовиться ребенку к новой взрослой школьной жизни – это игры с собиранием разных частей, т.е. [дидактические](http://vospitauka.ru/%d0%be%d1%80%d0%b3%d0%b0%d0%bd%d0%b8%d0%b7%d0%b0%d1%86%d0%b8%d1%8f_%d0%b4%d0%b8%d0%b4%d0%b0%d0%ba%d1%82%d0%b8%d1%87%d0%b5%d1%81%d0%ba%d0%be%d0%b9_%d0%b8%d0%b3%d1%80%d1%8b/). Каждому ребенку полезно поиграть в головоломки, собирать конструктор, [пазлы](http://345-games.ru/category/pazly/) и другие игры, одной из таких является игра «Тетрис».

Исследователь поставил перед собой цель: создание игры на языке программирования FreePascal с учётом анализа игровой механики «Тетриса».

Для достижения цели и доказательства гипотезы исследования необходимо решить следующие задачи:

* изучить и проанализировать и научную, учебную, методическую литературу по истории создания игры «Тетрис»;
* опросить респондентов для изучения необходимости и актуальности создания игры;
* провести сравнительный анализ объектов реальной игры «Тетрис» с возможными графическими примитивами языка программирования FreePascal;
* разработать программный код игры для использования его любым пользователем;
* провести оценку полученных результатов.

В ходе решения поставленных задач применялись следующие **методы исследования**: анализ методической литературы по проблеме исследования; анализ и синтез; тестирование; интервьюирование.

Исследовательская работа проводилась один год. В данной исследовательской работе изучена возможность создания игры «Тетрис» на языке программирования FreePascal. В ходе работы была изучена история создания игры «Тетрис» и её разновидности. Исследователь изучил алгоритм и правила работы игры «Тетрис», проанализировал возможности языка программирования FreePascal, что стало основой для разработки программного кода. Таким образом, гипотеза исследования: если исследовать правила игры «Тетрис» и возможности языка программирования FreePascal, то можно создать игру «Тетрис», так как начинающему программисту необходимо начинать свою профессиональную деятельность с создания игр подтвердилась. Цель исследовательской работы автором достигнута.

*Исследовательская работа «Анализ возможности создания игры «Тетрис»*

*на языке программирования FreePascal»*

*Баженов Дмитрий Юрьевич,*

*Россия, ХМАО-Югра Тюменская область, п.Березово*

*Муниципальное бюджетное образовательное учреждение*

*Березовская средняя общеобразовательная школа 9 класс*

# План исследования

Проблема исследования нашей работы заключается в том, что необходимо рассмотреть возможность использования языка программирования FreePascal для создания игр.

Мы вывели гипотезу: если исследовать правила игры «Тетрис» и возможности языка программирования FreePascal, то можно создать игру «Тетрис», так как начинающему программисту необходимо начинать свою профессиональную деятельность с создания игр.

Объектом исследования стала игра «Тетрис», а предметом – четырёх клеточные полимино, их прорисовка и движение по экрану в игре «Тетрис».

Наш план работы над исследованием представлен в виде таблицы.

|  |  |  |  |
| --- | --- | --- | --- |
| **Этапы** | **Виды деятельности** | **Сроки исполнения** | **Результат исследования** |
| **1** | **Работа с теорией:**   1. Выявление теоретического обоснования и актуальности темы | 1 неделя октября 2013 г. | Первой программой любого программиста является игра, так как необходимо разобраться в структуре языка и попробовать создать что-то стоящее, поэтому было решено попытаться написать игру «Тетрис», например на языке программирования FreePascal |
| 1. Работа с Интернет-ресурсами по вопросу «История игры «Тетрис»» | Октябрь-ноябрь 2013г. | Изучение материала по вопросу «История игры «Тетрис» показало, что она была впервые написана Алексеем Пажитновым в июне 1984 года на компьютере Электроника-60.Работая в ВЦ Академии наук СССР, Пажитнов занимался проблемами искусственного интеллекта и распознавания речи, а для обкатки идей применял головоломки, в том числе и классическое пентамино. Пажитнов пытался автоматизировать укладку пентамино в заданные фигурки. |
| **2.** | **Выбор респондентов** | Ноябрь 2013г. | Респонденты выбраны:   1. ученики школы; 2. учителя школы. |
| **3.** | **Подготовка к исследованию**   1. Проведение опроса «Необходимость создания головоломок» | Декабрь 2013г. | Опрошено 22 старшеклассников, большая часть из них играют в игру-головоломку «Тетрис», которая развивает логическое мышление. Почти 50% опрошенных считают, что начинать программировать необходимо с игр. Это подтверждается информацией, размещенной в сети Интернет. |
| 1. Выявление функциональных возможностей языка программирования FreePascal | Декабрь 2013г. | Знакомство с функциональными возможностями языка программирования FreePascal позволило рассмотреть операторы и их назначение. В результате проделанной работы была написана программа реализующая игру «Тетрис». |
| **4.** | **Проведение исследования**  Выявление возможности:  1.движения созданных геометрических фигур; | Январь 2014г. | В приложении представлены процедуры движения геометрических фигур и организации счета очков. |
| 2. организации счёта очков. |
| **5.** | **Обобщение теории и практики**   1. Создание игры «Тетрис» | Февраль 2014г. | Игра создана |
|  | 1. Оформление исследовательской работы | Февраль 2014г. | Работа оформлена |

Библиография:

1. Методическое пособие «Уроки программирования» ОМУ города Томск
2. Ушаков Д.М., Т.А. Юркова. Паскаль для школьников.-СПБ.: Питер, 2008. – 256 с.:ил
3. Википедия

*Исследовательская работа «Анализ возможности создания игры «Тетрис»*

*на языке программирования FreePascal»*

*Баженов Дмитрий Юрьевич,*

*Россия, ХМАО-Югра Тюменская область, п.Березово*

*Муниципальное бюджетное образовательное учреждение*

*Березовская средняя общеобразовательная школа 9 класс*

# Описание работы

**Актуальность**

Игра – это не только любимое и главное занятие для каждого ребенка, но и постоянный вид деятельности, в котором дети проводят большую часть времени. Роль игры, как формирующую составляющую в поведении ребенка подтверждают многие ученые, психологи и педагоги нашего времени. Любая игра будь то компьютерная, настольная является самой первой школой ребенка, как самостоятельного человека. Именно в игре, у него проявляется способность самостоятельно и добровольно подчиняться различным правилам и требованиям, которые расписаны в описании. Самые [развивающие игры](http://345-games.ru/), которые помогают подготовиться ребенку к новой взрослой школьной жизни – это игры с собиранием разных частей, то есть [дидактические](http://vospitauka.ru/%d0%be%d1%80%d0%b3%d0%b0%d0%bd%d0%b8%d0%b7%d0%b0%d1%86%d0%b8%d1%8f_%d0%b4%d0%b8%d0%b4%d0%b0%d0%ba%d1%82%d0%b8%d1%87%d0%b5%d1%81%d0%ba%d0%be%d0%b9_%d0%b8%d0%b3%d1%80%d1%8b/). Каждому ребенку полезно поиграть в головоломки, собирать конструктор, [пазлы](http://345-games.ru/category/pazly/) и другие игры, одной из таких является игра «Тетрис».

Сложно встретить человека, ни разу не игравшего в эту игру. Поколение девяностых с азартом играло в эту игру, развивающую логическое мышление, скорость реакции и пространственное воображение. Тетрис за это время обошел весь мир. Выпущено множество его вариантов и модификаций. Несложная игра, в которой падающие фигуры различной конфигурации складываются на плоскости, заполняя игровое пространство. Игра имеет множество уровней. С каждым уровнем увеличивается скорость, с которой падают фигурки. Однако не все знают, что изобретена она в СССР, программистом Алексеем Пажитновым в июне 1984 года на компьютере Электроника-60. Нам стало интересно возможно ли написать программу для игры «Тетрис» на одном из языков программирования высокого уровня.

Таким образом, **проблема исследования** нашей работы заключается в том, что необходимо рассмотреть возможность использования языка программирования FreePascal для создания игр. Поэтому **актуальность** данного исследования состоит в анализе использования графического модуля «Graph» и процедур языка программирования FreePascal для создания графических элементов и анимации, применяемых в программном коде игры «Тетрис».

**Цель исследования –** создание игры на языке программирования FreePascal с учётом анализа игровой механики «Тетриса».

**Объект исследования** - игра «Тетрис».

**Предмет исследования**–четырёх клеточные полимино, их прорисовка и движение по экрану в игре «Тетрис».

(возможности реализации игры «Тетрис» на языке программирования FreePascal.)

**Гипотеза исследования:** если исследовать правила игры «Тетрис» и возможности языка программирования FreePascal, то можно создать игру «Тетрис», так как начинающему программисту необходимо начинать свою профессиональную деятельность с создания игр.

Для достижения цели и доказательства гипотезы исследования необходимо решить следующие **задачи**:

* изучить и проанализировать и научную, учебную, методическую литературу по истории создания игры «Тетрис»;
* опросить респондентов для изучения необходимости и актуальности создания игры;
* провести сравнительный анализ объектов реальной игры «Тетрис» с возможными графическими примитивами языка программирования FreePascal;
* разработать программный код игры для использования его любым пользователем;
* провести оценку полученных результатов.

В ходе решения поставленных задач применялись следующие **методы исследования**:

* анализ методической литературы по проблеме исследования;
* анализ и синтез;
* тестирование;
* интервьюирование.

**Теоретическая значимость** данной работы определяется тем, что в результате проведенного исследования выявлены назначение и применение операторов, процедур языка программирования FreePascal.

**Практическая значимость** предлагаемого исследования состоит в том, чтобы выявить возможность графического модуля «Graph» и процедур языка программирования FreePascal для создания графических элементов и анимации, применяемых в программных кодах различных однотипных игр.

# История создания игры «Тетрис»

Интерес к фигурам домино, тримино, тетрамино и пентамино в СССР возник благодаря книге С. В. Голомба «Полимино» (издательство «Мир», 1975 год). В частности, пентамино было настолько популярно, что в «Науке и жизни» начиная с 1960-х годов был постоянный раздел, посвящённый составлению фигурок из набора пентамино, а пластмассовые наборы пентамино иногда продавались в магазинах.

## *Игра Пентамино - родоначальница Тетриса*

|  |
| --- |
|  |
| *Рисунок 1. Пентамино, предложенная Соломоном Голомбом* |
|  |
| *Рисунок2. Пентамино, предложенная Джоном Конвеем* |

Пентамино́ (от др.-греч. πέντα пять, и домино) — пятиклеточные полимино, то есть плоские фигуры, каждая из которых состоит из пяти одинаковых квадратов, соединённых между собой сторонами («ходом ладьи»). Этим же словом иногда называют головоломку, в которой такие фигуры требуется укладывать в прямоугольник или другие формы

|  |
| --- |
|  |
| *Рисунок 3. Возможные прямоугольники* |

Самая распространённая задача в пентамино — сложить из всех фигурок, без перекрытий и зазоров, прямоугольник. Поскольку каждая из 12 фигур включает в себя 5 квадратов, то прямоугольник должен быть площадью 60 единичных квадратов. Возможны прямоугольники 6×10, 5×12, 4×15 и 3×20. Каждую из этих головоломок можно решить вручную, но более сложной задачей является подсчёт общего числа возможных решений в каждом случае. (Очевидно, прямоугольники 2×30 и 1×60 составить из пентамино невозможно, поскольку многие фигуры в них просто не помещаются по ширине.)

Для случая 6×10 эту задачу впервые решил в 1965 году Джон Флетчер. Существует ровно 2339 различных укладок пентамино в прямоугольник 6×10, не считая поворотов и отражений целого прямоугольника, но считая повороты и отражения его частей (иногда внутри прямоугольника образуется симметричная комбинация фигур, поворачивая которую можно получить дополнительные решения; для прямоугольника 3×20, приведённого на рисунке, второе решение можно получить поворотом блока из 7 фигур, или, иначе говоря, если поменять местами четыре фигуры, крайние слева, и одну крайнюю справа).

Для прямоугольника 5×12 существует 1010 решений, 4×15 — 368 решений, 3×20 — всего 2 решения (отличающихся вышеописанным поворотом). В частности, существует 16 способов сложить два прямоугольника 5×6, из которых можно составить как прямоугольник 6×10, так и 5×12.[1]

## *Первый «Тетрис»*

|  |
| --- |
|  |
| *Рисунок 4. Алексей Пажитнов* |

|  |
| --- |
|  |
| *Рисунок5. МикроЭВМ «Электроника - 60»* |

«Тетрис» был впервые написан Алексеем Пажитновым в июне 1984 года на компьютере «Электроника-60», производимом в СССР. Работая в ВЦ Академии наук СССР, Пажитнов занимался проблемами искусственного интеллекта и распознавания речи, а для обкатки идей применял головоломки, в том числе и классическое пентамино. Пажитнов пытался автоматизировать укладку пентамино в заданные фигурки. Однако вычислительных мощностей тогдашнего оборудования для вращения пентамино не хватало, приходилось отлаживать на тетрамино, что и определило название игры. В тех опытах и родилась основная идея «Тетриса» — чтобы фигурки падали, а заполненные ряды исчезали.

Машины серии «Электроника-60» предназначались для использования в составе управляющих комплексов систем дискретной автоматики либо для отладки программ встраиваемых специализированных микроЭВМ с интерфейсом МПИ по ОСТ 11.305.903—80. Система команд соответствует ОСТ 11.305.909—80 и совместима с семейством PDP-11 фирмы DEC. Некоторые из машин серии были прямыми аналогами PDP-11 на отечественной элементной базе.

|  |
| --- |
|  |
| *Рисунок6. «Кирпичики-тетрамино» тетриса: I, J, L, O, S, T, Z.* |

МикроЭВМ серии «Электроника-60» имеют модульный принцип построения, то есть все функциональные модули ЭВМ выполнены в виде конструктивно законченных устройств (модулей), связь между которыми осуществляется через системный канал микроЭВМ.

Игра быстро распространилась по Москве и далее по всему миру. Несколько месяцев спустя про игру узнал импортёр программного обеспечения из Венгрии — Роберт Стейн. Стейн поехал в Москву, где он встретился с Алексеем Пажитновым, и договорился о лицензии на выпуск игры. По какой-то неизвестной для Стейна причине Пажитнов подарил ему тетрис Стейн продал права на «Тетрис» компании Microsoft (и её дочерней компании SpectrumHoloByte), принадлежащей британскому медиа-магнату Роберту Максвеллу. Стейн приехал договариваться о покупке прав у реальных правообладателей спустя несколько месяцев после заключения сделки. Русские отказались продавать Стейну права на «Тетрис» на его условиях. Тем временем две компании Максвелла — британская Microsoft и американская SpectrumHolobyte — выпускают свой вариант «Тетриса». У игры появляются качественные по меркам того времени графика и звук, а также «русский колорит» — в фоновых заставках программы появляются Юрий Гагарин, Матиас Руст, незадолго до этого совершивший посадку своего спортивного самолёта на Красной площади, и другие подобающие случаю персонажи. На глазах рождается сенсация — первая игра из-за «железного занавеса».

Вполне возможно, что о Пажитнове так бы никто и не узнал, если б не пронырливость журналистов CBS, представивших всему миру настоящего автора популярной игры. После показа интервью с Пажитновым позиции Стейна пошатнулись, свои поездки в Москву и безуспешные попытки договориться с советскими организациями он не афишировал.

В 1989 году в Nintendo полным ходом идёт разработка карманного игрового компьютера GameBoy, в этот же год до полудюжины различных компаний заявили о своих правах на версии «Тетриса» для разных компьютеров, игровых консолей и карманных игровых систем «ЭЛОРГ»

В 1996 году Алексей Пажитнов устроился на работу в Microsoft, где под его руководством был выпущен набор головоломок Pandora’s Box. Сейчас непосредственно программированием Алексей Пажитнов уже давно не занимается, он работал в подразделении Microsoft разработчиком компьютерных игр с 1996 по 2005 год.

Однако, 29 июня 2010 года, в интервью журналистам одного из геймерских порталов, Алексей Пажитнов сказал, что последние десять лет он работал над многопользовательским режимом для своего детища. Но самое интересное, что он ещё не закончил.

# Правила игры "Тетрис"

Случайные фигурки тетрамино падают сверху в прямоугольный стакан шириной 10 и высотой 20 клеток. В полёте игрок может поворачивать фигурку и двигать её по горизонтали. Также можно «сбрасывать» фигурку, то есть ускорять её падение, когда уже решено, куда фигурка должна упасть. Фигурка летит, пока не наткнётся на другую фигурку либо на дно стакана. Если при этом заполнился горизонтальный ряд из 10 клеток, он пропадает и всё, что выше него, опускается на одну клетку. В специальном поле игрок видит фигурку, которая будет следовать после текущей - эта подсказка позволяет планировать свои действия. Темп игры постепенно увеличивается. Название игры происходит от количества клеток, из которых состоит каждая фигура. Игра заканчивается, когда новая фигурка не может поместиться в стакан. Игрок получает очки за каждый заполненный ряд, поэтому его задача — заполнять ряды, не заполняя сам стакан (по вертикали) как можно дольше, чтобы таким образом получить как можно больше очков.

Начисление очков в разных версиях «Тетриса» довольно разнообразное. Очки могут начисляться за убранные линии, за сброшенные фигурки, за переход на новую скорость и тому подобное.

При начислении очков за линии количество очков обычно зависит от того, сколько линий убрано за один раз. Например, в китайских «Тетрисах», популярных в СНГ в 1990-х годах, начисление очков обычно было таким: 1 линия — 100 очков, 2 линии — 300 очков, 3 линии — 700 очков, 4 линии (то есть, сделать Тетрис) — 1500 очков. То есть, чем больше линий убирается за один раз, тем больше отношение количества очков к количеству линий. Любопытно, что тетрисом во многих версиях игры также называется действие, после которого исчезает сразу 4 линии. Это можно сделать только одним способом — сбросить «палку» (фигурку, в которой все клетки расположены на одной линии) в «шахту» ширины 1 и глубины как минимум 4.

При начислении очков за сброшенные фигурки могут учитываться высота, на которой остановилась фигурка (например, чем ниже, тем лучше), расстояние, которое пролетела фигурка после «сбрасывания» (ускорения падения). Хотя обычно приоритетом являются линии, а за фигурки начисляется относительно небольшое количество очков.[2]

# Реализация "Тетриса" на разных платформах

Игра реализована практически на всех современных компьютерах, включая КПК, мобильные телефоны, игровые видеоприставки, телевизоры (как доп. функция), множество карманных игровых устройств. Есть варианты игры для всех сколько-нибудь распространённых ОС, а также для Java. Есть порт даже для осциллографа. Трудно, если вообще возможно, назвать такую вычислительную платформу, где бы не было этой игры.

Пожалуй, наибольшую популярность приобрела реализация тетриса для игровой консоли GameBoy и видеоприставки NES (и её многочисленных клонов).

Были написаны трёх- (Blockout фирмы CaliforniaGames, 1989 год), четырёх- (HyperTetris, 1996 год) и даже n-мерный (PolytopeTetris, 2003 год) варианты тетриса, а также модификации для двух и более игроков[7].

Существуют также версии игры, в которых игра ведётся не на очки, а на открытие спрятанного изображения. Для облегчения игры есть варианты без сложных S- и Z-образных фигур и без увеличения скорости игры.

Кроме того, очень популярной разновидностью Тетриса является TetColor[8], написанный в 1991 году Сергеем Сотниковым (Тула), где исчезают линии (горизонтальные, вертикальные и диагональные), собранные из одного цвета. Аналогична ей AcidDrop разработанная программистом Дэннисом М. Киссом.

Существует и более сложная версия игры, основанная на пентамино, с фигурками из пяти элементов — Pentix («Пентикс»), в которой количество возможных фигур увеличено до 12-18.

Также есть реализация игры, где элементами фигур являются треугольники — CrazyTetris от AstatixSoftware, AmazingTetris.

Один из новейших примеров — игра Dwice, разработанная в 2006 году самим изобретателем Тетриса, Алексеем Пажитновым.

Также Тетрис вложен в качестве пасхального яйца в текстовый редактор emacs (открывается после нажатия Esc + X и ввода команды tetris).

Кроме того, Тетрис встроен в клиент μTorrent. Для этого необходимо открыть «Справка» — «О программе» и нажать клавишу Т.

PhysTris — тетрис с реалистичным физическим поведением блоков фигур. В игре можно поменять режим игры, количество кубиков фигур (4,5 и более), музыку и фон. Игру разработал PeterPetrov в 2009 году.[3]

# Результаты социологических опросов и анализ информации в сети Интернет

|  |
| --- |
|  |
| *Рисунок7.Анализ ответов на вопрос «Необходимо ли играть в головоломки?»* |

Мы провели опрос среди учащихся 10 и 11 классов с целью выявления необходимости играть в головоломки и создания игр программистами, результаты представлены в диаграмме.

Из диаграммы, представленной на рисунке 7, видно, что большинство респондентов уверены в необходимости играть в головоломки, аргументируя тем, что головоломки развивают мышление, логику, память, терпение, внимание. Каждый из учащихся может назвать самые любимые из них (кубик-рубик, пятнашки, пазлы),но все из них подтвердили, что они играют в Тетрис.

На вопрос «Считаете ли вы необходимым программистам начинать свою деятельность с создания игр» мнения разделись: «да» ответили 11 человек, а «нет» - 12.

На сайтах в сети Интернет представлена следующая информация о программистах-игроделах:

* типовой портрет начинающего разработчика игр: «Начинающий "игродел" - это чаще всего учащийся школы в возрасте 12-18 лет, внезапно загоревшийся идеей создания игр. Идея приходит спонтанно либо после какого-то толчка к вдохновению, вроде нового фильма или популярной игры. Как правило, сам он ничего не умеет в силу возраста, недостатка опыта и иных причин, поэтому хочет выступать в качестве "руководителя" и/или "сценариста", "автора идей"».
* игра «Тетрис», созданная Алексеем Пажитновым, была переписана на TurboPascal 16-летним школьником Вадимом Герасимовым.[4]

Исходя из опроса, можно сделать вывод: большая часть старшеклассников играют в игру-головолоку «Тетрис», которая развивает логическое мышление. Почти 50% опрошенных считают, что начинать программировать необходимо с игр. Это подтверждается информацией, размещенной в сети Интернет.

# Анализ возможности создания игры «Тетрис» на языке программирования FreePascal

При разработки сложного алгоритма целесообразно стараться выделить в нем последовательность действий, в котором выполняют решение каких-либо подзадач и могут многократно вызываться из основного алгоритма. Такие алгоритмы называются вспомогательными и в алгоритмических языках программирования реализуются в форме подпрограмм, процедур или функций, которые вызываются из основной программы. При создании программы в основном использовались процедуры:

procedure <имя процедуры> (<описание параметров-значений>);

var: <описание параметров-переменных>);

begin

<операторы>

end.

В работе использованы условные операторы двух видов: полный, который имеет вид:

if<условие>then<оператор1>else<оператор2>

и не полный: if<условие>then<оператор>

Слова if, then, else с английского на русский переводятся как если, то, иначе.

После then и else можно поставить лишь по одному оператору. Если нужно выполнить последовательность операторов, то их объединяют в один составной оператор вида:

begin<последовательность операторов> end.

В игре «Тетрис» фигурки появляются случайным образом, а очки начисляются за убранные линии до тех пор, пока игрок не допустит ошибку такую, что следующая фигурка не поместятся в прямоугольную область. Чтобы написать программу для реализации такой игры необходимо использовать циклы. Всего существует несколько типов циклов: цикл со счетчиком, цикл с предусловием и цикл с постусловием.[6]

Цикл со счетчиком используется, когда заранее известно, какое число повторений тела цикла необходимо выполнить. Количество повторений задается счетчиком.

Когда заранее известно, какое число повторений тела цикла необходимо выполнить, можно воспользоваться оператором цикла со счетчиком For. В заголовке цикла устанавливается начальное значение переменной счетчик, определяется величина его коечного значения и величина изменения значения за один шаг. Затем располагаются многократно выполняющиеся операторы, являются телом цикла.

Цикл с условием используется, когда заранее неизвестно, какое количество раз должно повторятся тело цикла. В таких случаях количество повторений зависит от некоторого условия. Цикл называется цикл с предусловием, если условие выхода из цикла стоит вначале, перед телом цикла. Цикл с предусловием не выполняется даже один раз в случае ложности условия.

Цикл с предусловием реализуется с помощью инструкций While. Проверка условия выхода из цикла проводится до начала цикла с помощью ключевого слова While. Ключевое слово While обеспечивает выполнение цикла, пока истинно условие. Как только условие примет значение «ложь», выполнение цикла закончится.

Цикл называется цикл с постусловием, если условие выхода из цикла стоит в конце, после тела цикла. Цикл с постусловием выполняется обязательно, как минимум, один раз, независимо от того, истинно условие или нет. Ключевое слово Until обеспечивает выполнение цикла до тех пор, пока условие не станет истинным, т. е. пока условие имеет значение «ложь». Как только условие примет значение «истинна», выполнение цикла закончится.

Алгоритмическая структура «цикл» может быть зафиксирована графически, с помощью блок-схемы.

|  |  |  |
| --- | --- | --- |
| **Цикл со счетчиком** | **Цикл с предусловием** | **Цикл с постусловием** |
|  |  |  |
| **For** i:=начзн**to**конзн**do**  **begin**  телоцикла  **end** | **While**условие**do**  **begin**  телоцикла  **end** | **Repeat**  тело цикла  **until**условие |

Для отображения процесса игры использован модуль Graph. Модуль Graph - это расширение для среды программирования Pascal, позволяющее создать графическое окно, в котором, с помощью дополнительных команд, пользователь может рисовать геометрические примитивы. Графическое окно имеет разрешение 640x480. Модуль добавляет следующие команды:

* putpixel(x,y,c) - раскрашивает пиксель в координатах x,y в цвет c;
* setcolor(c) - устанавливает цвет для некоторых команд;
* line(x1,y1,x2,y2) - рисует прямую линии из точки с координатами x1, y1 в точку x2, y2, цвет устанавливается командой setcolor;
* circle(x,y,r) - рисует окружность с центром в точке (x, y) радиуса r, цвет устанавливается командой setcolor;
* rectangle(x1,y1,x2,y2) - рисует незакрашенный прямоугольник по двум противоположным углам с координатами x1, y1 и x2, y2, цвет устанавливается командой setcolor;
* ellipse(x1, y1, angle1, angle2,rx,ry) - рисует дугу с центром в точке (x, y), радиусом по вертикали ry, радиусом по горизонтали rx, от угла angle1 до угла angle2, цвет устанавливается командой setcolor;
* setfiilstyle(t, c) - устанавливает тип t и цвет c заливки некоторыми командами;
* bar(x1, y1, x2, y2) - рисует закрашенный прямоугольник по двум противоположным углая с координатами x1, y1 и x2, y2, цвет и тип заливки устанавливается командой setfillstyle;
* floodfill(x, y, c) - заливает область, включающую координаты x, y и имеющую границу цвета c, тип и цвет заливки устанавливается командой setfillstyle.

Данные команды позволяют создать геометрические фигуры и оболочку, используемые в игре «Тетрис».

Таким образом, знакомство с функциональными возможностями языка программирования FreePascal позволило рассмотреть операторы и их назначение. В результате проделанной работы была написана программа реализующая игру «Тетрис». (Приложение 1)[7]

# Заключение

В данной исследовательской работе изучена возможность создания игры «Тетрис» на языке программирования FreePascal. Выявлено, что данная игра очень популярна, начиная с 1989 года, и реализована на разных платформах. Это очень удобная и полезная игра, развивающая логическое мышление. В ходе работы выполнены поставленные задачи:

* Изучение литературы по теме исследования предоставило возможность проанализировать историю создания игры «Тетрис». Родоначальницей данной игры является игра «Пентамино» -пятиклеточные полимино, то есть плоские фигуры, каждая из которых состоит из пяти одинаковых квадратов, соединённых между собой сторонами. Первый «Тетрис» был написан в июне 1984 года на компьютере «Электроника-60» Алексеем Пажитновым, который занимался проблемами искусственного интеллекта и распознавания речи. Пажитнов пытался автоматизировать укладку пентамино в заданные фигурки. Игра быстро распространилась по Москве и далее по всему миру. Она была реализована практически на всех современных компьютерах, включая КПК, мобильные телефоны, игровые видеоприставки, телевизоры (как доп. функция), множество карманных игровых устройств. Есть варианты игры для всех распространённых ОС, а также для Java. В настоящее время трудно назвать такую вычислительную платформу, где бы не было этой игры.
* При опросе респондентов для изучения необходимости и актуальности создания игры выявлено, что большая часть из них играют в «Тетрис», которая, по их мнению, развивает логическое мышление. Почти 50% опрошенных считают, что начинать программировать необходимо с игр. В результате анализа информации в сети Интернет был выявлен портрет начинающего игродела: «Он учащийся школы, внезапно загоревшийся идеей создания игр, которому 12-18 лет», именно в этом возрасте школьник Вадим Герасимов переписал на TurboPascal игру «Тетрис», созданную Алексеем Пажитновым.
* В ходе работы был проведен анализ возможности реализации игры на одном из языков программирования. Сравнительный анализ объектов реальной игры «Тетрис» и операторов, команд, модулей языка программирования FreePascal позволил разработать программный код игры.
* Программа протестирована и может быть использована пользователями операционной системы Windows.

Таким образом, в данной работе исследована возможность реализации игры «Тетрис» на языке программирования FreePascal, значит **гипотеза исследования подтвердилась**.

Использованные методы исследования можно применять для дальнейшей разработки подобных игр. **Цель** исследовательской работы нами достигнута.

# Список использованной литературы:

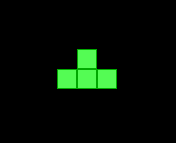
1. <http://www.kakprosto.ru/kak-848066-v-chem-zaklyuchaetsya-rabota-tehnika-programmista>
2. <http://3dg.me/ru/gamedev/basics/sozdanie-igr-dlya-nachinayushchih>
3. <http://345-games.ru/vliyanie-igr-na-razvitie-i-psixiku-rebenka/>
4. <https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D1%82%D1%80%D0%B8%D1%81>
5. Методическое пособие «Уроки программирования» ОМУ города Томск
6. Ушаков Д.М., Т.А. Юркова. Паскаль для школьников.-СПБ.: Питер, 2008. – 256 с.:ил
7. Информатика и ИКТ: учебник 9 класс: в 2 ч. Ч.1/ Л.Л.Босова, А.Ю.Босова.-.3 изд.-М.: БИНОМ. Лаборатория знаний, 2014г. \_ 248с.:ил.

# Приложения

## Приложение 1

**Процедура рисования фигуры**

procedurefigura(f,p:byte);



begin

ifcolor then begin setfillstyle(1,c);setcolor(c-8) end

else begin setfillstyle(1,0);setcolor(0) end;

case f of

1: begin

bar(x-20,y-20,x+19,y+19);

rectangle(x-20,y-20,x-1,y-1);

rectangle(x,y-20,x+19,y-1);

rectangle(x-20,y,x-1,y+19);

rectangle(x,y,x+19,y+19);

end;

2: case p of

1,3: begin

bar(x-40,y-20,x-1,y-1);bar(x-20,y,x+19,y+19);

rectangle(x-40,y-20,x-21,y-1);

rectangle(x-20,y-20,x-1,y-1);

rectangle(x-20,y,x-1,y+19);

rectangle(x,y,x+19,y+19);

end;

2,4: begin

bar(x,y-40,x+19,y-1);bar(x-20,y-20,x-1,y+19);

rectangle(x,y-40,x+19,y-21);

rectangle(x,y-20,x+19,y-1);

rectangle(x-20,y-20,x-1,y-1);

rectangle(x-20,y,x-1,y+19);

end;

end;

3: case p of

1,3: begin

bar(x-40,y,x-1,y+19); bar(x-20,y-20,x+19,y-1);

rectangle(x,y-20,x+19,y-1);

rectangle(x-20,y-20,x-1,y-1);

rectangle(x-40,y,x-21,y+19);

rectangle(x-20,y,x-1,y+19);

end;

2,4: begin

bar(x-20,y-40,x-1,y-1); bar(x,y-20,x+19,y+19);

rectangle(x-20,y-40,x-1,y-21);

rectangle(x-20,y-20,x-1,y-1);

rectangle(x,y-20,x+19,y-1);

rectangle(x,y,x+19,y+19);

end;

end;

4: case p of

1,3: begin

bar(x,y-40,x+19,y+39);

rectangle(x,y-40,x+19,y-21);

rectangle(x,y-20,x+19,y-1);

rectangle(x,y,x+19,y+19);

rectangle(x,y+20,x+19,y+39);

end;

2,4: begin

bar(x-40,y,x+39,y+19);

rectangle(x-40,y,x-21,y+19);

rectangle(x-20,y,x-1,y+19);

rectangle(x,y,x+19,y+19);

rectangle(x+20,y,x+39,y+19);

end;

end;

5: case p of

1: begin

bar(x-20,y-20,x-1,y-1); bar(x-20,y,x+39,y+19);

rectangle(x-20,y-20,x-1,y-1);

rectangle(x-20,y,x-1,y+19);

rectangle(x,y,x+19,y+19);

rectangle(x+20,y,x+39,y+19);

end;

2: begin

bar(x,y-20,x+39,y-1); bar(x,y,x+19,y+39);

rectangle(x,y-20,x+19,y-1);

rectangle(x+20,y-20,x+39,y-1);

rectangle(x,y,x+19,y+19);

rectangle(x,y+20,x+19,y+39);

end;

3: begin

bar(x-20,y,x+39,y+19); bar(x+20,y+20,x+39,y+39);

rectangle(x-20,y,x-1,y+19);

rectangle(x,y,x+19,y+19);

rectangle(x+20,y,x+39,y+19);

rectangle(x+20,y+20,x+39,y+39);

end;

4: begin

bar(x,y-20,x+19,y+39); bar(x-20,y+20,x-1,y+39);

rectangle(x,y-20,x+19,y-1);

rectangle(x,y,x+19,y+19);

rectangle(x,y+20,x+19,y+39);

rectangle(x-20,y+20,x-1,y+39);

end;

end;

6: case p of

1: begin

bar(x-20,y,x+39,y+19); bar(x+20,y-20,x+39,y-1);

rectangle(x-20,y,x-1,y+19);

rectangle(x,y,x+19,y+19);

rectangle(x+20,y,x+39,y+19);

rectangle(x+20,y-20,x+39,y-1);

end;

2: begin

bar(x,y-20,x+19,y+39); bar(x+20,y+20,x+39,y+39);

rectangle(x,y-20,x+19,y-1);

rectangle(x,y,x+19,y+19);

rectangle(x,y+20,x+19,y+39);

rectangle(x+20,y+20,x+39,y+39);

end;

3: begin

bar(x-20,y,x+39,y+19); bar(x-20,y+20,x-1,y+39);

rectangle(x-20,y+20,x-1,y+39);

rectangle(x-20,y,x-1,y+19);

rectangle(x,y,x+19,y+19);

rectangle(x+20,y,x+39,y+19);

end;

4: begin

bar(x-20,y-20,x+19,y-1); bar(x,y,x+19,y+39);

rectangle(x-20,y-20,x-1,y-1);

rectangle(x,y-20,x+19,y-1);

rectangle(x,y,x+19,y+19);

rectangle(x,y+20,x+19,y+39);

end;

end;

7: case p of

1: begin

bar(x-20,y,x+39,y+19); bar(x,y-20,x+19,y-1);

rectangle(x-20,y,x-1,y+19);

rectangle(x,y,x+19,y+19);

rectangle(x,y-20,x+19,y-1);

rectangle(x+20,y,x+39,y+19);

end;

2: begin

bar(x,y-20,x+19,y+39); bar(x+20,y,x+39,y+19);

rectangle(x,y-20,x+19,y-1);

rectangle(x,y,x+19,y+19);

rectangle(x+20,y,x+39,y+19);

rectangle(x,y+20,x+19,y+39);

end;

3: begin

bar(x-20,y,x+39,y+19); bar(x,y+20,x+19,y+39);

rectangle(x-20,y,x-1,y+19);

rectangle(x,y,x+19,y+19);

rectangle(x,y+20,x+19,y+39);

rectangle(x+20,y,x+39,y+19);

end;

4: begin

bar(x,y-20,x+19,y+39); bar(x-20,y,x-1,y+19);

rectangle(x,y-20,x+19,y-1);

rectangle(x,y,x+19,y+19);

rectangle(x-20,y,x-1,y+19);

rectangle(x,y+20,x+19,y+39);

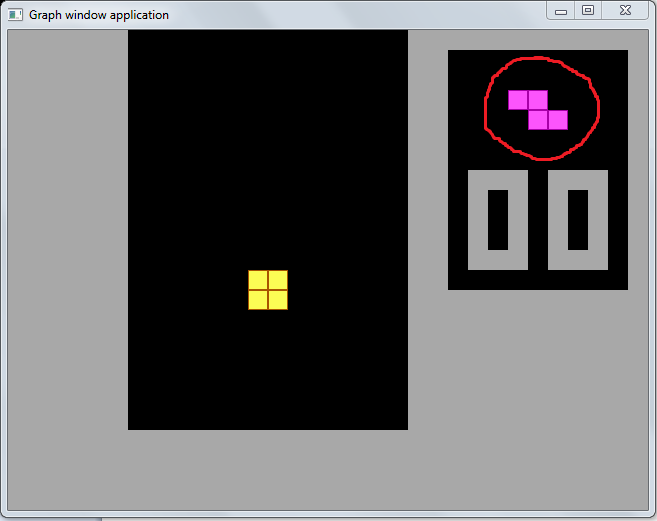
end;

end;

end;

end;

**процедура показа следующей фигуры**



procedurenfigura;

begin

setfillstyle(1,nc);

setcolor(nc-8);

casenf of

1: begin

bar(500,60,539,99);

rectangle(500,60,519,79);

rectangle(520,60,539,79);

rectangle(500,80,519,99);

rectangle(520,80,539,99);

end;

2: begin

bar(500,60,539,79);bar(520,80,559,99);

rectangle(500,60,519,79);

rectangle(520,60,539,79);

rectangle(520,80,539,99);

rectangle(540,80,559,99);

end;

3: begin

bar(500,80,539,99); bar(520,60,559,79);

rectangle(540,60,559,79);

rectangle(520,60,539,79);

rectangle(500,80,519,99);

rectangle(520,80,539,99);

end;

4: begin

bar(520,40,539,119);

rectangle(520,40,539,59);

rectangle(520,60,539,79);

rectangle(520,80,539,99);

rectangle(520,100,539,119);

end;

5: begin

bar(500,60,519,79); bar(500,80,559,99);

rectangle(500,60,519,79);

rectangle(500,80,519,99);

rectangle(520,80,539,99);

rectangle(540,80,559,99);

end;

6: begin

bar(500,80,559,99); bar(540,60,559,79);

rectangle(500,80,519,99);

rectangle(520,80,539,99);

rectangle(540,80,559,99);

rectangle(540,60,559,79);

end;

7: begin

bar(500,80,559,99); bar(520,60,539,79);

rectangle(500,80,519,99);

rectangle(520,80,539,99);

rectangle(520,60,539,79);

rectangle(540,80,559,99);

end;

end;

end;

**процедура проверки "приземления" фигуры**

procedurecheckend(f:byte);

begin

case f of

1: begin if getpixel(x-11,y+25)>0 then cend:=true else if getpixel(x+9,y+25)>0 then cend:=true else cend:=false end;

2: case p of

1,3: begin if getpixel(x-31,y+5)>0 then cend:=true else if getpixel(x-11,y+25)>0 then cend:=true else if getpixel(x+14,y+25)>0 then cend:=true else cend:=false end;

2,4: begin if getpixel(x-11,y+25)>0 then cend:=true else if getpixel(x+9,y+4)>0 then cend:=true else cend:=false end;

end;

3: case p of

1,3: begin if getpixel(x-31,y+29)>0 then cend:=true else if getpixel(x-11,y+29)>0 then cend:=true else if getpixel(x+9,y+9)>0 then cend:=true else cend:=false end;

2,4: begin if getpixel(x-11,y+9)>0 then cend:=true else if getpixel(x+9,y+29)>0 then cend:=true else cend:=false end;

end;

4: case p of

1,3: begin if getpixel(x+9,y+49)>0 then cend:=true else cend:=false end;

2,4: begin if getpixel(x-31,y+29)>0 then cend:=true else if getpixel(x-11,y+29)>0 then cend:=true else if getpixel(x+9,y+29)>0 then cend:=true else if getpixel(x+29,y+29)>0 then cend:=true else cend:=false end;

end;

5: case p of

1: begin if getpixel(x-11,y+29)>0 then cend:=true else if getpixel(x+9,y+29)>0 then cend:=true else if getpixel(x+29,y+29)>0 then cend:=true else cend:=false end;

2: begin if getpixel(x+9,y+49)>0 then cend:=true else if getpixel(x+29,y+9)>0 then cend:=true else cend:=false end;

3: begin if getpixel(x-11,y+29)>0 then cend:=true else if getpixel(x+9,y+29)>0 then cend:=true else if getpixel(x+29,y+49)>0 then cend:=true else cend:=false end;

4: begin if getpixel(x-11,y+49)>0 then cend:=true else if getpixel(x+9,y+49)>0 then cend:=true else cend:=false end;

end;

6: case p of

1: begin if getpixel(x-11,y+29)>0 then cend:=true else if getpixel(x+9,y+29)>0 then cend:=true else if getpixel(x+29,y+29)>0 then cend:=true else cend:=false end;

2: begin if getpixel(x+9,y+49)>0 then cend:=true else if getpixel(x+29,y+49)>0 then cend:=true else cend:=false end;

3: begin if getpixel(x-11,y+49)>0 then cend:=true else if getpixel(x+9,y+29)>0 then cend:=true else if getpixel(x+29,y+29)>0 then cend:=true else cend:=false end;

4: begin if getpixel(x+9,y+49)>0 then cend:=true else if getpixel(x-11,y+9)>0 then cend:=true else cend:=false end;

end;

7: case p of

1: begin if getpixel(x-11,y+29)>0 then cend:=true else if getpixel(x+9,y+29)>0 then cend:=true else if getpixel(x+29,y+29)>0 then cend:=true else cend:=false end;

2: begin if getpixel(x+9,y+49)>0 then cend:=true else if getpixel(x+29,y+29)>0 then cend:=true else cend:=false end;

3: begin if getpixel(x-11,y+29)>0 then cend:=true else if getpixel(x+9,y+49)>0 then cend:=true else if getpixel(x+29,y+29)>0 then cend:=true else cend:=false end;

4: begin if getpixel(x-11,y+29)>0 then cend:=true else if getpixel(x+9,y+49)>0 then cend:=true else cend:=false end;

end;

end;

end;

**процедура проверки возможности сдвига фигуры влево**

procedurecheckleft(f:byte);

begin

case f of

1: begin if getpixel(x-26,y-11)>0 then cl:=true else if getpixel(x-26,y+9)>0 then cl:=true else cl:=false end;

2: case p of

1,3: begin if getpixel(x-46,y-11)>0 then cl:=true else if getpixel(x-26,y+9)>0 then cl:=true else cl:=false end;

2,4: begin if getpixel(x-6,y-31)>0 then cl:=true else if getpixel(x-26,y-11)>0 then cl:=true else if getpixel(x-26,y+9)>0 then cl:=true else cl:=false end;

end;

3: case p of

1,3: begin if getpixel(x-31,y-11)>0 then cl:=true else if getpixel(x-51,y+9)>0 then cl:=true else cl:=false end;

2,4: begin if getpixel(x-31,y-31)>0 then cl:=true else if getpixel(x-31,y-11)>0 then cl:=true else if getpixel(x-11,y+9)>0 then cl:=true else cl:=false end;

end;

4: case p of

1,3: begin if getpixel(x-11,y-31)>0 then cl:=true else if getpixel(x-11,y-11)>0 then cl:=true else if getpixel(x-11,y+9)>0 then cl:=true else if getpixel(x-11,y+29)>0 then cl:=true else cl:=false end;

2,4: begin if getpixel(x-51,y+9)>0 then cl:=true else cl:=false end;

end;

5: case p of

1: begin if getpixel(x-31,y-11)>0 then cl:=true else if getpixel(x-31,y+9)>0 then cl:=true else cl:=false end;

2: begin if getpixel(x-11,y-11)>0 then cl:=true else if getpixel(x-11,y+9)>0 then cl:=true else if getpixel(x-11,y+29)>0 then cl:=true else cl:=false end;

3: begin if getpixel(x-31,y+9)>0 then cl:=true else if getpixel(x+9,y+29)>0 then cl:=true else cl:=false end;

4: begin if getpixel(x-11,y-11)>0 then cl:=true else if getpixel(x-11,y+9)>0 then cl:=true else if getpixel(x-31,y+29)>0 then cl:=true else cl:=false end;

end;

6: case p of

1: begin if getpixel(x-31,y+9)>0 then cl:=true else if getpixel(x+9,y-11)>0 then cl:=true else cl:=false end;

2: begin if getpixel(x-11,y-11)>0 then cl:=true else if getpixel(x-11,y+9)>0 then cl:=true else if getpixel(x-11,y+29)>0 then cl:=true else cl:=false end;

3: begin if getpixel(x-31,y+9)>0 then cl:=true else if getpixel(x-31,y+29)>0 then cl:=true else cl:=false end;

4: begin if getpixel(x-31,y-11)>0 then cl:=true else if getpixel(x-11,y+9)>0 then cl:=true else if getpixel(x-11,y+29)>0 then cl:=true else cl:=false end;

end;

7: case p of

1: begin if getpixel(x-11,y-11)>0 then cl:=true else if getpixel(x-31,y+9)>0 then cl:=true else cl:=false end;

2: begin if getpixel(x-11,y-11)>0 then cl:=true else if getpixel(x-11,y+9)>0 then cl:=true else if getpixel(x-11,y+29)>0 then cl:=true else cl:=false end;

3: begin if getpixel(x-31,y+9)>0 then cl:=true else if getpixel(x-11,y+29)>0 then cl:=true else cl:=false end;

4: begin if getpixel(x-11,y-11)>0 then cl:=true else if getpixel(x-31,y+9)>0 then cl:=true else if getpixel(x-11,y+29)>0 then cl:=true else cl:=false end;

end;

end;

end;

**процедура проверки возможности сдвига фигуры вправо**

procedurecheckright(f:byte);

begin

case f of

1: begin if getpixel(x+24,y-11)>0 then cr:=true else if getpixel(x+24,y+9)>0 then cr:=true else cr:=false end;

2: case p of

1,3: begin if getpixel(x+4,y-11)>0 then cr:=true else if getpixel(x+24,y+9)>0 then cr:=true else cr:=false end;

2,4: begin if getpixel(x+24,y-31)>0 then cr:=true else if getpixel(x+24,y-11)>0 then cr:=true else if getpixel(x+4,y+9)>0 then cr:=true else cr:=false end;

end;

3: case p of

1,3: begin if getpixel(x+29,y-11)>0 then cr:=true else if getpixel(x+9,y+9)>0 then cr:=true else cr:=false end;

2,4: begin if getpixel(x+9,y-31)>0 then cr:=true else if getpixel(x+29,y-11)>0 then cr:=true else if getpixel(x+29,y+9)>0 then cr:=true else cr:=false end;

end;

4: case p of

1,3: begin if getpixel(x+29,y-31)>0 then cr:=true else if getpixel(x+29,y-11)>0 then cr:=true else if getpixel(x+29,y+9)>0 then cr:=true else if getpixel(x+29,y+29)>0 then cr:=true else cr:=false end;

2,4: begin if getpixel(x+49,y+9)>0 then cr:=true else cr:=false end;

end;

5: case p of

1: begin if getpixel(x+9,y-11)>0 then cr:=true else if getpixel(x+49,y+9)>0 then cr:=true else cr:=false end;

2: begin if getpixel(x+49,y-11)>0 then cr:=true else if getpixel(x+29,y+9)>0 then cr:=true else if getpixel(x+29,y+29)>0 then cr:=true else cr:=false end;

3: begin if getpixel(x+49,y+9)>0 then cr:=true else if getpixel(x+49,y+29)>0 then cr:=true else cr:=false end;

4: begin if getpixel(x+29,y-11)>0 then cr:=true else if getpixel(x+29,y+9)>0 then cr:=true else if getpixel(x+29,y+29)>0 then cr:=true else cr:=false end;

end;

6: case p of

1: begin if getpixel(x+49,y-11)>0 then cr:=true else if getpixel(x+49,y+9)>0 then cr:=true else cr:=false end;

2: begin if getpixel(x+29,y-11)>0 then cr:=true else if getpixel(x+29,y+9)>0 then cr:=true else if getpixel(x+49,y+29)>0 then cr:=true else cr:=false end;

3: begin if getpixel(x+49,y+9)>0 then cr:=true else if getpixel(x+9,y+29)>0 then cr:=true else cr:=false end;

4: begin if getpixel(x+29,y-11)>0 then cr:=true else if getpixel(x+29,y+9)>0 then cr:=true else if getpixel(x+29,y+29)>0 then cr:=true else cr:=false end;

end;

7: case p of

1: begin if getpixel(x+29,y-11)>0 then cr:=true else if getpixel(x+49,y+9)>0 then cr:=true else cr:=false end;

2: begin if getpixel(x+29,y-11)>0 then cr:=true else if getpixel(x+49,y+9)>0 then cr:=true else if getpixel(x+29,y+29)>0 then cr:=true else cr:=false end;

3: begin if getpixel(x+49,y+9)>0 then cr:=true else if getpixel(x+29,y+29)>0 then cr:=true else cr:=false end;

4: begin if getpixel(x+29,y-11)>0 then cr:=true else if getpixel(x+29,y+9)>0 then cr:=true else if getpixel(x+29,y+29)>0 then cr:=true else cr:=false end;

end;

end;

end;

**процедура проверки возможности повернуть фигуру**

procedurecheckpov;

begin

case f of

2: case p of

1,3: if (getpixel(x+9,y-11)=0) and (getpixel(x+9,y-31)=0) then cp:=true else cp:=false;

2,4: if (getpixel(x-36,y-11)=0) and (getpixel(x+9,y+9)=0) then cp:=true else cp:=false;

end;

3: case p of

1,3: if (getpixel(x-31,y-31)=0) and (getpixel(x-31,y-11)=0) then cp:=true else cp:=false;

2,4: if (getpixel(x-11,y+9)=0) and (getpixel(x-31,y+9)=0) then cp:=true else cp:=false;

end;

4: case p of

1,3: if (getpixel(x-31,y+9)=0) and (getpixel(x-11,y+9)=0) and (getpixel(x+29,y+9)=0) then cp:=true else cp:=false;

2,4: if (getpixel(x+9,y-31)=0) and (getpixel(x+9,y-11)=0) and (getpixel(x+9,y+29)=0) then cp:=true else cp:=false;

end;

5: case p of

1: if (getpixel(x+9,y-11)=0) and (getpixel(x+29,y-11)=0) and (getpixel(x+9,y+29)=0) then cp:=true else cp:=false;

2: if (getpixel(x+29,y+9)=0) and (getpixel(x+29,y+29)=0) and (getpixel(x-11,y+9)=0) then cp:=true else cp:=false;

3: if (getpixel(x+9,y-11)=0) and (getpixel(x+9,y+29)=0) and (getpixel(x-11,y+29)=0) then cp:=true else cp:=false;

4: if (getpixel(x-11,y-11)=0) and (getpixel(x-11,y+9)=0) and (getpixel(x+29,y+9)=0) then cp:=true else cp:=false;

end;

6: case p of

1: if (getpixel(x+9,y-11)=0) and (getpixel(x+9,y+29)=0) and (getpixel(x+29,y+29)=0) then cp:=true else cp:=false;

2: if (getpixel(x-11,y+9)=0) and (getpixel(x-11,y+29)=0) and (getpixel(x+29,y+9)=0) then cp:=true else cp:=false;

3: if (getpixel(x-11,y-11)=0) and (getpixel(x+9,y-11)=0) and (getpixel(x+9,y+29)=0) then cp:=true else cp:=false;

4: if (getpixel(x-11,y+9)=0) and (getpixel(x+29,y-11)=0) and (getpixel(x+29,y+9)=0) then cp:=true else cp:=false;

end;

7: case p of

1: if getpixel(x+9,y+29)=0 then cp:=true else cp:=false;

2: if getpixel(x-11,y+9)=0 then cp:=true else cp:=false;

3: if getpixel(x+9,y-11)=0 then cp:=true else cp:=false;

4: if getpixel(x+29,y+9)=0 then cp:=true else cp:=false;

end;

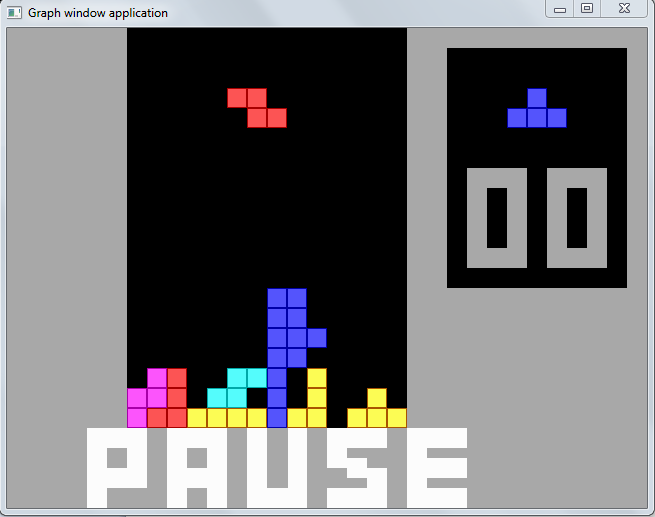
end;

end;

**процедура, включающая и выключающая паузу**

procedure pause;

begin



setfillstyle(1,15);

bar(80,400,139,479);

bar(160,400,219,479);

bar(240,400,299,479);

bar(320,400,379,479);

bar(400,400,459,479);

setfillstyle(1,7);

bar(100,420,119,439);

bar(100,460,139,479);

bar(180,420,199,439);

bar(180,460,199,479);

bar(260,400,280,459);

bar(320,440,339,459);

bar(340,450,359,459);

bar(340,420,359,429);

bar(360,420,379,439);

bar(420,420,459,429);

bar(420,450,459,459);

delay(10);

repeat

k:=readkey;

if k=#27 then begin quit:=true; y:=460; k:=#32; end;

until k=#32;

bar(80,400,459,479);

end;

**процедура, считывающая нажатия клавиш клавиатуры и выполняющая соответствующие нажатым клавишам действия**

procedureupr;

begin

for i:=1 to 25 do

begin

ifkeypressed then

begin

k:=readkey;

case k of

#77: begin checkright(f); if not cr then begin color:=false;figura(f,p);x:=x+20;color:=true;figura(f,p);end;end;

#75: begin checkleft(f); if not cl then begin color:=false;figura(f,p);x:=x-20;color:=true;figura(f,p);end;end;

#72: begin checkpov; if cp then begin color:=false; figura(f,p); if p=4 then p:=1 else inc(p); color:=true; figura(f,p); end;end;

#80: begin checkend(f); if not cend then begin color:=false; figura(f,p); y:=y+20; color:=true; figura(f,p) end;end;

#32: pause;

end;

end;

delay(20-(score div 10));

ifkeypressed then

begin

k:=readkey;

case k of

#77: begin checkright(f); if not cr then begin color:=false;figura(f,p);x:=x+20;;color:=true;figura(f,p);end;end;

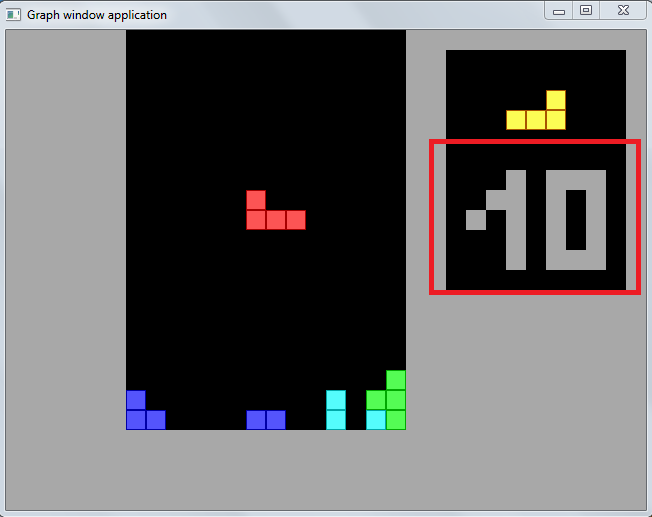
#75: begin checkleft(f); if not cl then begin color:=false;figura(f,p);x:=x-20;color:=true;figura(f,p);end;end;

#72: begin checkpov; if cp then begin color:=false; figura(f,p); if p=4 then p:=1 else inc(p); color:=true; figura(f,p); end;end;

#80: begin checkend(f); if not cend then begin color:=false; figura(f,p); y:=y+20; color:=true; figura(f,p) end;end;

#32: pause;

end; end; end; end;



**процедура, используемая в процедуре стирания линии**

procedurekvad;

begin

ll:=ll-1;

c:=getpixel(110+hh\*20,ll\*20-10);

ll:=ll+1;

setfillstyle(1,c);

if c=0 then setcolor(0) else setcolor(c-8);

bar((110+hh\*20)-10,(ll\*20-10)-10,(110+hh\*20)+9,(ll\*20-10)+9);

rectangle((110+hh\*20)-10,(ll\*20-10)-10,(110+hh\*20)+9,(ll\*20-10)+9);

end;

**процедура, показывающая количество убранных линий, или очков**

procedurescoreshow;

begin

escore:=score mod 10;

dscore:=score div 10;

setfillstyle(1,0);

bar(460,140,599,239);

caseescore of

0: begin setfillstyle(1,7); bar(540,140,599,239); setfillstyle(1,0); bar(560,160,579,219); end;

1: begin setfillstyle(1,7); bar(580,140,599,239); bar(560,160,579,179); bar(540,180,559,199); end;

2: begin setfillstyle(1,7); bar(540,140,599,239); setfillstyle(1,0); bar(540,160,579,179); bar(560,200,599,219); end;

3: begin setfillstyle(1,7); bar(540,140,599,239); setfillstyle(1,0); bar(540,160,579,179); bar(540,200,579,219); end;

4: begin setfillstyle(1,7); bar(540,140,559,199); bar(560,180,579,199); bar(580,140,599,239); end;

5: begin setfillstyle(1,7); bar(540,140,599,239); setfillstyle(1,0); bar(560,160,599,179); bar(540,200,579,219); end;

6: begin setfillstyle(1,7); bar(540,140,599,239); setfillstyle(1,0); bar(560,160,599,179); bar(560,200,579,219); end;

7: begin setfillstyle(1,7); bar(540,140,599,159); bar(580,160,599,179); bar(560,180,579,199); bar(540,200,559,239); end;

8: begin setfillstyle(1,7); bar(540,140,599,239); setfillstyle(1,0); bar(560,160,579,179); bar(560,200,579,219); end;

9: begin setfillstyle(1,7); bar(540,140,599,239); setfillstyle(1,0); bar(560,160,579,179); bar(560,200,579,219); end;

end;

casedscore of

0: begin setfillstyle(1,7); bar(460,140,519,239); setfillstyle(1,0); bar(480,160,499,219); end;

1: begin setfillstyle(1,7); bar(500,140,519,239); bar(480,160,499,179); bar(460,180,479,199); end;

2: begin setfillstyle(1,7); bar(460,140,519,239); setfillstyle(1,0); bar(460,160,499,179); bar(480,200,519,219); end;

3: begin setfillstyle(1,7); bar(460,140,519,239); setfillstyle(1,0); bar(460,160,499,179); bar(460,200,499,219); end;

4: begin setfillstyle(1,7); bar(460,140,479,199); bar(480,180,499,199); bar(500,140,519,239); end;

5: begin setfillstyle(1,7); bar(460,140,519,239); setfillstyle(1,0); bar(480,160,519,179); bar(460,200,499,219); end;

6: begin setfillstyle(1,7); bar(460,140,519,239); setfillstyle(1,0); bar(480,160,519,179); bar(480,200,499,219); end;

7: begin setfillstyle(1,7); bar(460,140,519,159); bar(500,160,519,179); bar(480,180,499,199); bar(460,200,479,239); end;

8: begin setfillstyle(1,7); bar(460,140,519,239); setfillstyle(1,0); bar(480,160,499,179); bar(480,200,499,219); end;

9: begin setfillstyle(1,7); bar(460,140,519,239); setfillstyle(1,0); bar(480,160,499,179); bar(460,200,499,219); end;

end;

end;

**процедура, проверяющая, нужно ли стирать линию**

procedurecheckline;

begin

for l:=1 to n do begin

for h:=1 to m do begin

lin[h,l]:=getpixel(110+h\*20,l\*20-10);

if (lin[h,l])>0 then j:=j+1; end;

if j=14 then begin

forll:=l downto 2 do

forhh:=m downto 1 do kvad;

if score<99 then score:=score+1;

sound(100);

scoreshow;

end;

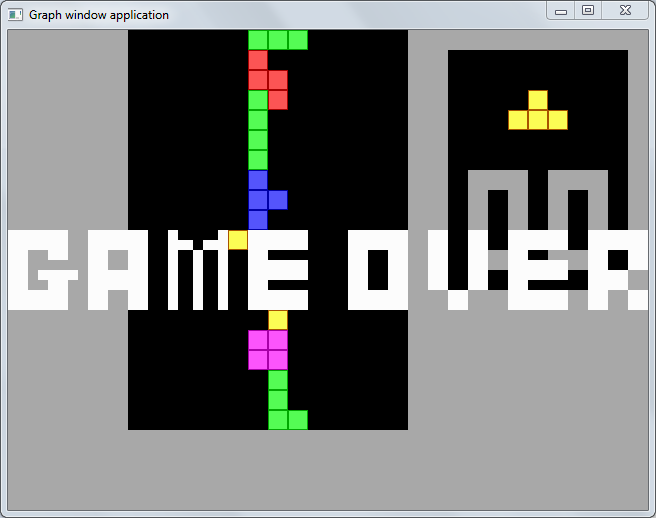
j:=0

end;

end;

**процедура проверки проигрыша**

procedurecheckgameover;



begin

j:=0;

for l:=1 to n do

for h:=1 to m do

ifgetpixel(110+h\*20,-10+20\*l)>0 then if j=l-1 then j:=j+1;

if j=n then begin

setfillstyle(1,15);

bar(40,200,59,229); bar(0,200,39,219); bar(0,220,19,279); bar(20,260,59,279); bar(40,250,59,259); bar(30,240,69,249);

bar(80,200,99,279); bar(100,200,119,219); bar(100,240,119,259); bar(120,200,139,279);

bar(160,200,169,279); bar(170,210,184,219); bar(185,220,194,279); bar(195,210,209,219); bar(210,200,219,279);

bar(240,200,299,219); bar(240,220,259,229); bar(240,230,299,249); bar(240,250,259,259); bar(240,260,299,279);

bar(340,200,399,219); bar(340,220,359,259); bar(340,260,399,279); bar(380,220,399,259);

bar(420,200,439,259); bar(440,260,459,279); bar(460,200,479,259);

bar(500,200,559,219); bar(500,220,519,229); bar(500,230,559,249); bar(500,250,519,259); bar(500,260,559,279);

bar(580,200,639,219); bar(580,220,599,279); bar(600,240,619,259); bar(620,220,640,239); bar(620,260,639,279);

sound(1000);

delay(500);

sound(1000);

delay(125);

sound(1000);

delay(125);

sound(1000);

delay(125);

sound(1000);

delay(250);

sound(1000);

delay(375);

sound(1000);

delay(3500);

write(#7);

quit:=true;

end;

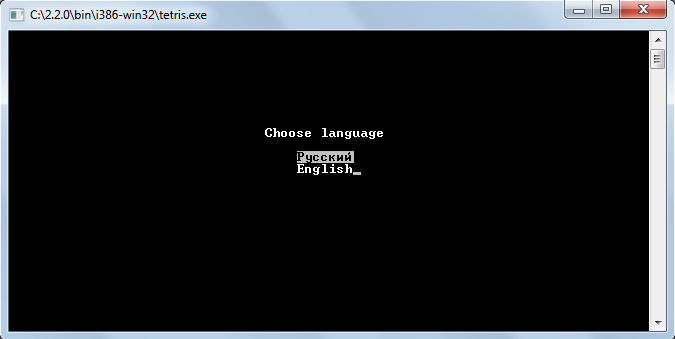
end;

**процедура, вызывающая меню выбора языка**

procedurechslng;

begin

clrscr;



lang:=0;

gotoxy(33,9);

textcolor(15);

write('Choose language');

menuopt:=1;

repeat

casemenuopt of

1: begin gotoxy(37,11); textcolor(0); textbackground(15); write('ђгббЄЁ©');

gotoxy(37,12); textcolor(15); textbackground(0); write('English'); end;

2: begin gotoxy(37,11); textcolor(15); textbackground(0); write('ђгббЄЁ©');

gotoxy(37,12); textcolor(0); textbackground(15); write('English'); end;

end;

k:=readkey;

case k of

#72: if menuopt=1 then menuopt:=2 else menuopt:=1;

#80: if menuopt=2 then menuopt:=1 else menuopt:=2;

#13: lang:=menuopt;

end;

untillang>0;

textbackground(0);

clrscr;

menuopt:=1;

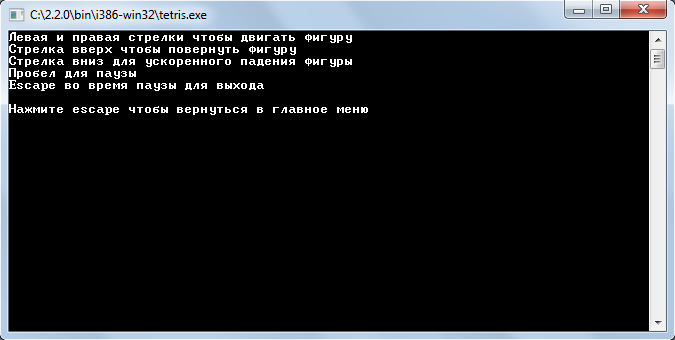
end;

**процедура, вызывающая окно помощи и управления**

procedure help;

begin

clrscr;



gotoxy(1,1);

iflang=1 then begin

writeln('левая и правая стрелки чтобы двигать фигуру');

writeln('стрелка вверх чтобы повернуть фигуру');

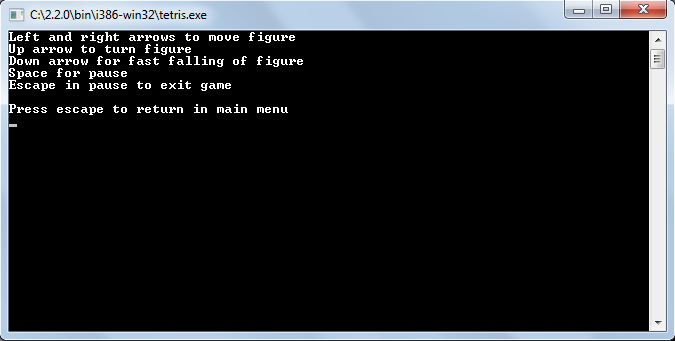
writeln('стрелка вниз для усокенного падения фигуры');

writeln('пробел для паузы');

writeln('Escape во время паузы для выхода ');

writeln();

writeln('нажмите Escape чтобы вернуться в главное меню ');



end;

iflang=2 then begin

writeln('Left and right arrows to move figure');

writeln('Up arrow to turn figure');

writeln('Down arrow for fast falling of figure');

writeln('Space for pause');

writeln('Escape in pause to exit game');

writeln();

writeln('Press escape to return in main menu');

end;

repeat

k:=readkey;

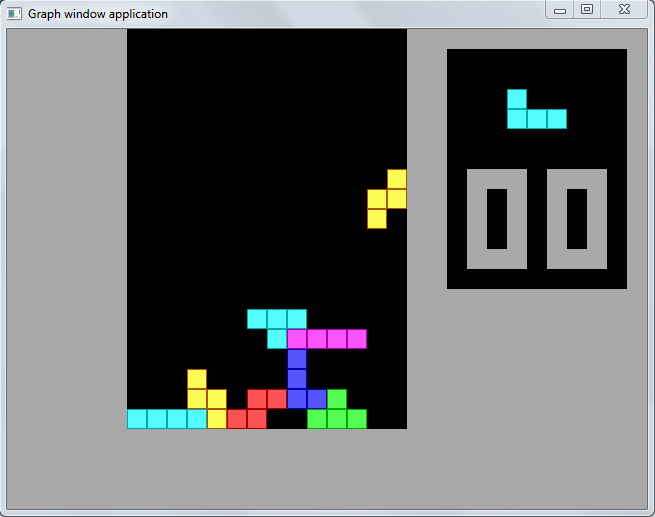
until k=#27;

textbackground(0);

clrscr;

end;

**процедура геймплея**



procedure game;

begin

clrscr;

randomize;

gd:=vga;

gm:=2;

initgraph(gd,gm,'');

score:=0;

j:=0;

setfillstyle(1,7);

bar(0,0,639,479);

setfillstyle(1,0);

bar(120,0,399,399);

bar(440,20,619,259);

nf:=random(7)+1;

nc:=random(6)+9;

scoreshow;

repeat

f:=nf;

c:=nc;

nf:=random(7)+1;

nc:=random(6)+9;

p:=1;

x:=260;

y:=-20;

setfillstyle(1,0);

bar(440,20,619,119);

nfigura;

repeat

checkend(f);

color:=false;

figura(f,p);

y:=y+20;

color:=true;

figura(f,p);

upr;

ifkeypressed then k:=readkey;

checkend(f); if cend then y:=460;

until (y=460);

checkline;

checkgameover;

until quit;

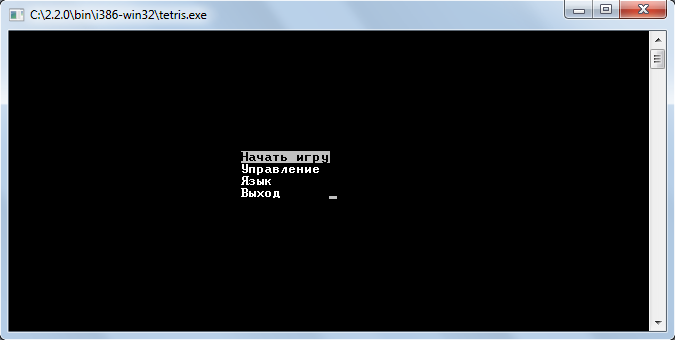
quit:=false;

closegraph;

end;

**процедура главного меню**

proceduremainmenu;



begin

textbackground(0);

clrscr;

menuopt:=1;

repeat

for i:=1 to 4 do

begin

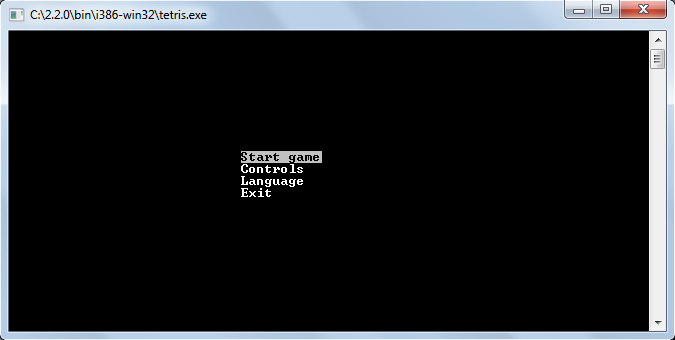
gotoxy(30,10+i);

ifi=menuopt then

begintextcolor(0); textbackground(15);

end else

begintextcolor(15); textbackground(0); end;



iflang=1 then write(rmenu[i]) else write(emenu[i]);

end;

k:=readkey;

case k of

#72: if menuopt=1 then menuopt:=4 else dec(menuopt);

#80: if menuopt=4 then menuopt:=1 else inc(menuopt);

#13: case menuopt of

1: game;

2: help;

3: chslng;

4: quit:=true;

end; end;

untilquit;

end;

**код самой программы**

begin

clrscr;

quit:=false;

chslng;

repeat

mainmenu;

until quit;

closegraph;

textbackground(0);

clrscr;

end.