Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение   
высшего образования

«МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ,

(национальный исследовательский университет)» (МАИ)

КУРСОВОЙ ПРОЕКТ

по дисциплине: «Вычислительные системы»

на тему «Разработка вычислительной системы  
«Псевдографический редактор»

**Выполнил:**  студент группы: М8О-112Б-17

Горшков Дмитрий Сергеевич

(Фамилия, имя, отчество)

(подпись)

**Принял**: доцент кафедры 813

Мокряков Алексей Викторович

(Фамилия, имя, отчество)

(подпись)

**Оценка: Дата:**

МОСКВА 2017

Содержание

[Задачи проекта 3](#_Toc501804912)

[Цель и задачи проекта 3](#_Toc501804913)

[Предлагаемые методы решения 3](#_Toc501804914)

[Работа программы 3](#_Toc501804914)

Работа с инструментами 3

Интерфейс 4

Подключение дополнительных библиотек 4

Обработка ошибок ввода 5

[Используемые инструменты 5](#_Toc501804915)

Среда разработки Lasarus IDE 5

Встроенные библиотеки языка 5

Hex Workshop 5

[Реализация проекта 5](#_Toc501804916)

[Алгоритмы 5](#_Toc501804917)

Алгоритм рисования элементов графического интерфейса 5

Алгоритмы инструментов для рисования 7

[Тестирование проекта 9](#_Toc501804918)

[Описание разработанной системы 10](#_Toc501804919)

[Руководство пользователя 10](#_Toc501804920)

[Список заявленных особенностей проекта и выявленных ошибок 14](#_Toc501804921)

[Список литературы 15](#_Toc501804922)

[Код программы 16](#_Toc501804923)

## Задачи проекта

## Цель и задачи проекта

1. **Цель проекта:**

Реализовать программу, которая должна «рисовать изображения» с помощью использования символов из таблицы ASCII (т.е. с использованием т.н. «псевдографики»).

1. **Задачи проекта**.
2. Программа должна сохранять и загружать файлы собственного формата и с собственной структурой, в которых содержатся следующие данные: данные с графическим интерфейсом (ГИ), с данными о местоположении (сдвиге относительно начала) данных о ГИ, файлы с настройками программы.
3. «Рисовать» с помощью различных инструментов «псевдографические изображения.
4. Обрабатывать ввод команд в «командной строке».
5. Обрабатывать ввод данных с клавиатуры.

## Предлагаемые методы решения

1. **Работа программы.**

В ходе разработки приложения возникала потребность быстро и оперативно переключаться с одного «цикла работы» программы на другой. При этом была возможность использовать несколько методов работы программы: 1) с использованием процедур вызывать поочередно конкретный цикл программы. 2) реализовать работу программы с помощью условных операторов if (усл. название – скрипты). В ходе разработки программы было выяснено, что первый метод недостаточно удовлетворял условиям программы, был сложным для реализации и неэффективным в некоторых ситуациях работы программы. Так, например, в ходе разработки было выяснено, что без определенных трудностей было невозможно реализовать переход от создания нового файла к редактированию изображения напрямую. Так же данный метод содержал в себе достаточно много повторяющихся элементов. Например, в ходе работы, по оценкам разработчика, код создания файла мог повториться до 6-7 раз. Второй же метод является «громоздким», но более многозадачным и эффективным, а так же он полностью удовлетворял поставленным задачам (например, с использованием этого метода была решена основная проблема метода 1), в результате чего данный метод и был выбран.

1. **Работа с инструментами.**

Большинство инструментов работает исходя из разных интерпретаций алгоритма Брезенхэма[1]. Были рассмотрены также другие варианты рисования треугольников, прямоугольников и линий. В частности, вместо использования алгоритма Брезенхэма предполагался алгоритм с реализацией через цикл for и деления прямой на равные отрезки. Однако он оказался не точным, громоздким с точки зрения объема кода и, при некоторых углах наклона работал некорректно: появлялись изгибы, а линия могла «разорваться» на две и более. Алгоритм Брезенхема же показал достаточно точные результаты, а проблем с рисованием линий в виде разрывов не возникало, в результате чего он был выбран в качестве основного. Для рисования четырехугольников также был применен алгоритм Брезенхэма из-за более высокой скорости рисования, однако это влекло за собой эффект, который называют «аффинным искажением» (или ошибочным жаргонным названием – «излом перспективы»), который представлен на скриншоте.



С треугольниками же была иная ситуация: алгоритм Брезенхэма показал низкую точность при окрашивании треугольника: возникали уже вышеупомянутый эффект «аффинного искажения» и т.н. «дыры» внутри треугольника. Та же проблема возникала, если закрашивать треугольник, при этом рисуя отрезки не из вершины в точку на противолежащей стороне, а соединяя точки на смежных сторонах. Несмотря на то, что такой метод точнее, результат все же получился не слишком впечатляющим: возникала та же проблема, если один из углов треугольника был бы либо слишком острым (< 8°-10°), либо слишком тупым (>170°). Был опробован метод закрашивания с использованием барицентрических координат треугольника. Однако данный метод сложен в реализации, достаточно медленен, и с его помощью достаточно проблематично было реализовать плавный цветовой переход (в связи с ограниченностью палитры). Проблема с плавным переходом более-менее решена при использовании алгоритма Брезенхэма, однако из-за использования целочисленных координат он оказался достаточно неточным. Остальные алгоритмы рисования не отличаются особой сложностью и информативностью – они были реализованы с помощью цикла for и поэтапной отрисовкой, а так же некоторые из них так или иначе используют в своей работе алгоритм Брезенхэма.

1. Изначально, интерфейс программы состоял не из цельных изображений, а из «окон», которые обладали своими особыми свойствами. Например, изначально существовало окно-«прародитель» main\_window\_frame, которое содержало в себе окна с помощью, с рабочей областью и т.д. Каждое из них обладало собственными свойствами (например, окно win\_newfile\_frame, содержащее надпись «Новый», при определенных условиях (например, при активации скрипта) могло окраситься в красный или зеленый цвета). Однако такой подход был признан нецелесообразным: обработка двух файлов была чересчур медленной, а общее количество «проходов» по файлу было равно трем, а обработку различных свойств приходилось реализовывать с применением достаточно «нестандартных» способов. В результате был разработан новый метод рисования ГИ: на экран сразу выводились изображения, а обработка их особых свойств выполнялась в самой обработке скриптов. Такой способ реализации ГИ оказался достаточно быстрым (два файла приходилось считывать всего два раза – по одному «проходу» на файл), легким в реализации, но от того не менее эффективным с точки зрения поставленных задач в проекте.
2. **Подключение дополнительных библиотек.**

Без подключения некоторых библиотек пришлось бы самостоятельно реализовывать множество «стандартных» функций и процедур. Например, процедуры GotoXY, TextColor, TextBackground и т.д.

1. **Функции обработки ошибок и ввода.**

В ходе разработки и тестирования проекта стало ясно, что пользователи бывают разные – если одни вникнут в суть работы программы сразу и не будут совершать ошибок в процессе ее работы, то существуют и другие, которые не сразу поймут, как работает данная программа, и может работать с ней так, что она может аварийно завершить свою работу. Для предотвращения таких ситуаций приходилось следить за «каждым шагом» пользователя – следить, что он нажимает, каким образом он это нажимает, и указывать ему, что он «не прав», и что не нужно так делать, что это ошибка и прочее.

## Используемые инструменты

1. **Среда разработки IDE Lasarus**

Для написания программы использовалась среда разработки на языке Pascal – Lasarus. Изначально, выбор состоял из 4 различных сред разработки: Turbo Pascal 7, Pascal ABC, Lasarus и Visual Studio. Несмотря на то, что в Visual Studio крайне удобные интерфейс и отладчик, язык Паскаль в нем официально не поддерживался, однако существуют сторонние средства для работы с этим языком, предназначенные для данной среды. Понятно, что гарантировать корректность и стабильность работы программы нельзя, поэтому данный вариант сразу же был признан хоть и интересным, но крайне как нестабильным, так и просто нецелесообразным и эксцентричным. Pascal ABC так же обладал приветливым интерфейсом, однако из-за того, что .ABC – диалект языка Паскаль, и многие функции могут работать не так, как следует того ожидать с точки зрения «оригинального» языка Паскаль, он так же был признан «не отвечающим поставленным целям» (в качестве грубого и утрированного сравнения можно сравнить разницу между Pascal и Pascal ABC с разницей между языками С и С++). Turbo Pascal, в сравнении с Lasarus’ом, оказался менее дружественным к пользователю, в нем менее удобна отладка программы.

1. **Встроенные библиотеки языка**

CRT – основная библиотека, без которой невозможна была бы работа с консолью в том виде, в котором она представлена в программе.

SysUtils – используется, в основном, для создания лог-файлов, а именно для получения информации о дате и времени.

1. **Hex Workshop**

Достаточно удобный инструмент для работы с файлами, а именно с его шестнадцатеричным кодом. Использовался для создания всего графического интерфейса программы и файла с данными.

## Реализация проекта

## Алгоритмы

В ходе работы над проектом были использованы следующие алгоритмы:

1. **Стандартные алгоритмы работы с файлами.**
2. **Алгоритм рисования элементов графического интерфейса:**

…

{Закрытие и открытие файлов для чтения их с самого начала}

…

{Посимвольное чтение файла с данными для выяснения, на сколько сдвинуты данные о ГИ}

for i := 0 to 1018 + id \* 4 do

begin

Read(fdata, ch);

for i := 1 to 4 do

begin {перевод из 16-ой системы в 10-ую}

Read(fdata, ch); strr := strr + ch;

end;

size := Perevod(strr);

{Посимвольное чтение файла с графикой и получение информации об изображении}

for i := 0 to size + 16 do Read(fgraph, ch); sizeX := Perevod(ch);

Read(fgraph, ch); sizeY := Perevod(ch);

x\_pos := 0; y\_pos := 0;

Colorer := True; gotoxy(posx, posy);

for i := 1 to (sizeX \* sizeY) \* 2 do

begin

Read(fgraph, ch);

if x\_pos = sizeX then begin

{Отрисовка изображения из файла}

if (y\_pos <> sizeY) and (WhereX <> 80) then

gotoxy(WhereX - x\_pos, WhereY + 1);

Inc(y\_pos); x\_pos := 0;

end;

if Colorer then begin {Если считываем байт, отвечающий за цвет}

TextColor((Ord(ch) and 15));

TextBackground((Ord(ch) and 240) shr 4);

Colorer := False;

end else begin

Write(ch); Inc(x\_pos); Colorer := True;

end; end;

…

Текстовое описание алгоритма:

1. Закрываем и открываем файлы с данными и графикой для считывания файла с начала.
2. Посимвольное считывание файла с данными: читаем до 1018 символа без обработки данных. С 1018 до 1018+номер изображения\*4 (число, на которое произведен сдвиг кода окна относительно начала закодирован 4 байтами). В результате «считыватель» доходит до начала числа сдвига.
3. Считываем 4 байта (число).
4. Переводим ASCII­-коды 4-х символов в число в 10-ой системе счисления. (фактически – перевод из 16-ой системы счисления в 10-ую).
5. Считываем посимвольно файл с графикой. Начинаем считывать с того места, на которое указывали данные из файла с данными. Считываем первые 16 байт информации без обработки (в них содержится заголовок с данными об изображении: 1 байт – номер изображения, а остальные 15 – название изображения). Далее обрабатываем 2 байта – размер по горизонтали и размер по вертикали. Исходя из полученных данных, считываем Размер\_по\_Х \*

\* Размер\_по\_У байтов. Поочередно считываем: нечетные байты (Если Colorer = = true) – цвет символа: 4 бита на цвет фона и 4 бита на цвет символа, четные ( при Colorer = false) – символ, который будет отрисован в ходе рисования интерфейса.

1. Во время считывания данных из файла с графикой, мы также отрисовываем изображение на экран.
2. **Алгоритмы инструментов для рисования**

Большинство алгоритмов представляют из себя производные алгоритмы от т.н. алгоритма Брезенхэма[2] для рисования линии. Код будет представлен лишь для алгоритма Брезенхэма.

1. Карандаш

С помощью двойного цикла for происходит отрисовка квадрата на экран. В основном цикле мы передвигаем координату по оси ОУ, а с помощью внутреннего – рисуем символы в линию.

1. Кисть

Алгоритм схож с тем, что используется при рисовании карандашом, однако при этом во внутреннем цикле мы меняем прозрачность с помощью выбора нового символа и оператора case.

1. Линия

procedure DrawLine(x1, x2, y1, y2: integer);

var delta\_xpos, delta\_ypos, delta2, delta3, mask\_op, t1, t2, Careter, smb: integer;

begin

delta\_xpos:= abs(x2-x1); {Находим число, на которое изменяется координата х}

delta\_ypos:= abs(y2-y1); {Находим число, на которое изменяется координата у}

delta2:=delta\_xpos-delta\_ypos; {Разница, по которой мы смотрим, переносить ли каретку на новую строку (ошибка)}

delta3:=delta2\*2; {Проверка на значение ошибки}

if delta\_xpos > delta\_ypos then mask\_op:= delta\_xpos div (opacity+1) else mask\_op:= delta\_ypos div (opacity+1); {Какое расстояние необходимо пройти, чтобы изменить интенсивность}

if x1 < x2 then t1:=1 else t1:=-1; {Если мы рисуем слева направо, или справа налево}

if y1 < y2 then t2:=1 else t2:=-1; {Если мы рисуем сверху вниз, или снизу вверх}

Careter:=0; {Переменная, проверяющая расстояние, которая прошла каретка (для изменения интенсивности}

smb:=opacity; {Переменная, отвечающая за прозрачность}

gotoXY(18+x1, 4 + y1);

while (x1 <> x2) or (y1 <> y2) do begin

gotoXY(18+x1, 3 + y1);

delta3:=delta2\*2;

{Выбираем, согласно текущей интенсивности, каким символом рисовать}

case smb of

0: symb\_paint:=' ';

1: symb\_paint:=#176;

2: symb\_paint:=#177;

3: symb\_paint:=#178;

4: symb\_paint:=#219;

end;

write(symb\_paint);

gotoXY(WhereX-1, WhereY);

{Записываем и рисуем символ в соответствии с текущей интенсивностью}

drawed\_color[(y1-1)\*61+x1] := color\_symb + (color\_bg shl 4);

drawed\_char[(y1-1)\*61+x1] := symb\_paint;

{Проверяем, нужно ли сдвигать каретку по оси х}

if delta3 > delta\_ypos \* (-1) then begin

delta2 := delta2 - delta\_ypos;

x1 := x1 + t1;

end;

{Проверяем, нужно ли сдвигать каретку по оси у}

if delta3 < delta\_xpos then begin

delta2 := delta2 + delta\_xpos;

y1 := y1 + t2;

end;

{Если пройдено расстояние, необходимое для понижения интенсивности}

if (Careter = mask\_op) and (pbucket\_method = 2) then begin

if (smb > 0) then dec(smb);

Careter:=0;

end else inc(Careter);

end;

{Если рисовали фигуру – обнуляем координаты вершин.}

n1\_pressed := false;

n2\_pressed := false;

{Отрисовываем последний символ у линии}

case smb of

0: symb\_paint:=' ';

1: symb\_paint:=#176;

2: symb\_paint:=#177;

3: symb\_paint:=#178;

4: symb\_paint:=#219;

end;

gotoXY(18+x2, 4 + y2 - 1);

write(symb\_paint);

gotoXY(WhereX-1, WhereY);

drawed\_color[(y1-1)\*61+x1] := color\_symb + (color\_bg shl 4);

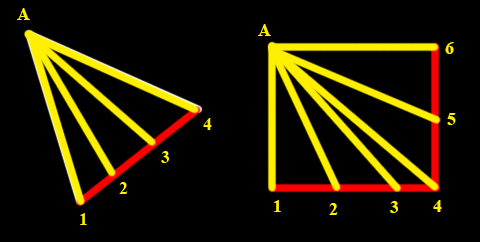
drawed\_char[(y1-1)\*61+x1] := symb\_paint;

end;

Текстовое описание алгоритма.

1. Находим число, на которое изменяется координата по оси ОХ (проекцию пройденного кареткой расстояния на ось ОХ). Для этого из конечной координаты по х вычтем начальную по х. Так как конечная координата может быть меньше начальной, то берем модуль от полученного числа.
2. Находим число, на которое изменяется координата по оси ОУ (проекцию пройденного кареткой расстояния на ось ОУ). Для этого из конечной координаты по у вычтем начальную по у. Так как конечная координата может быть меньше начальной, то берем модуль от полученного числа.
3. Вычисляем значение ошибки, которое означает — вертикальное расстояние между текущим значением y и точным значением y для текущего x.
4. Пока мы не нарисуем линию, в цикле мы проверяем и делаем следующее: всякий раз, когда мы увеличиваем (уменьшаем) x, мы увеличиваем (уменьшаем) значение ошибки на величину наклона. Если ошибка превысила 0.5 (от дроби можно избавится, сверяя значение ошибкой с удвоенным значением той же самой ошибки), линия стала ближе к следующему y, поэтому мы уменьшаем (увеличиваем) y на единицу, одновременно уменьшая (увеличиваем) значение ошибки на 1. Увеличение или уменьшение зависит от того, в какую сторону мы чертим линию.
5. Из-за выбранной системы отсчета, дорисовываем последний символ, завершающий нашу линию.
6. Фигура

Действие основано на работе алгоритма Брезенхэма: сначала задается основная вершина, из которой будет исходить линия. Потом задаются оставшиеся вершины, и при получении координат вершин, происходит их обработка, после чего, по точкам на мнимых линиях будут отрисованы линии.

 Здесь:

А – вершина, из которой рисуем линии, желтые отрезки – рисуемые линии (порядок указан цифрами для наглядности), красные отрезки – мнимая линия, на которой лежат точки, являющиеся конечными координатами желтых отрезков.

Все остальные алгоритмы – производные от перечисленных.

## Тестирование проекта

1. **Главный экран.**

|  |  |  |
| --- | --- | --- |
| Клавиша | Эффект | Что должно быть. |
| Случайные наборы клавиш, которые не перечислены ниже | Ничего не происходит. | Ничего не происходит. |
| Alt+I | Открытие окна с информацией. | Открытие окна с информацией. |
| Alt+S | Неудачная попытка записать несуществующий файл. Вывод надписи посередине экрана «Нечего записывать» | Вывод надписи посередине экрана «Нечего записывать» |
| Alt+L | Открытие окна с загрузкой файла. | Открытие окна с загрузкой файла. |
| Alt+N | Открытие окна с созданием файла. | Открытие окна с созданием файла. |
| Alt+C | Открытие окна с настройками. | Открытие окна с настройками. |
| / | Ввод команды. Появление символа «/». | Ввод команды. Появление символа «/». |

1. **Ввод команды**

При вводе несуществующих команд (подробнее в «Руководстве пользователя») происходит сброс ввода. При вводе любых некорректных символов (не латинских больших и маленьких букв, символов: + - \_ , = и пробел) происходит сброс ввода. При вводе служебных символов и комбинаций происходит ввод некорректных символов.

1. **Окно создания файла.**

Ввод названия файла: см. ввод команды.

Помимо сброса ввода происходит еще и выход из окна создания файла без предупреждений об ошибке.

1. **Окно загрузки файла.**

Нажатие клавиш: см. Окно создания файла.

1. **Работа с файлами**

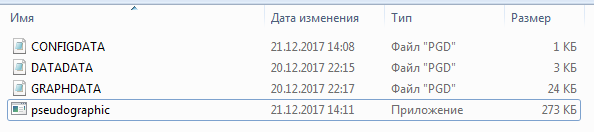
При некоторых манипуляциях с открытием, созданием и сохранением файла происходят фатальные ошибки. Подробнее об этом написано в «Особенностях проекта», п. 9.

## Описание разработанной системы

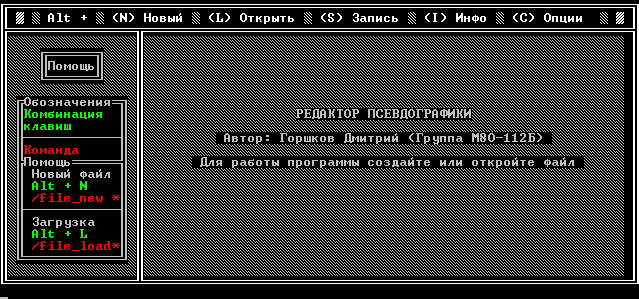
## Руководство пользователя

1. Перед началом работы удостоверьтесь, что в директории с файлом pseudographic.exe присутствуют файлы DATADATA.PGD, GRAPHDATA.PGD и CONFIGDATA.PGD.
2. Запустите файл pseudographic.exe.

И0 – Запуск программы и директория с программой



1. *Основное меню.*



Основные действия, которые доступны в основном меню, заключаются в создании и открытии файлов, просмотра информации, открытии меню с настройками и ввод команд в импровизированную командную строку.

Управление:

|  |  |
| --- | --- |
| Клавиши | Результат |
| Alt + N | Открытие окна создания файла. |
| Alt + L | Открытие окна загрузки файла. |
| Alt + S | Открытие окна записи файла. (В связи с тем, что программа еще не начала полноценную работу, на экране высветится ошибка). |
| Alt + I | Открытие окна с информацией. |
| Alt + C | Открытие окна с настройками. |
| / | Начало ввода команды. |

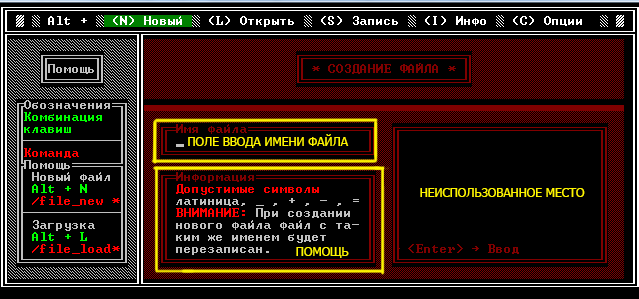
Ввод команды:

После нажатия соответствующей клавиши, станет доступен ввод команды. (Опознается по нарисованному символу / в левом нижнем углу).

Список допустимых команд:

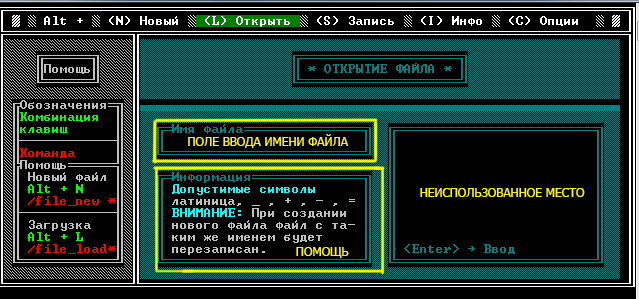
|  |  |
| --- | --- |
| file\_new | Аналог нажатия клавиши |
| file\_load | Аналог нажатия клавиши |
| file\_save | Аналог нажатия клавиши |
| config | Аналог нажатия клавиши |
| Info | Аналог нажатия клавиши |
| Exit | Выход из программы без сохранения файлов. |

1. *Окно создания файлов.*



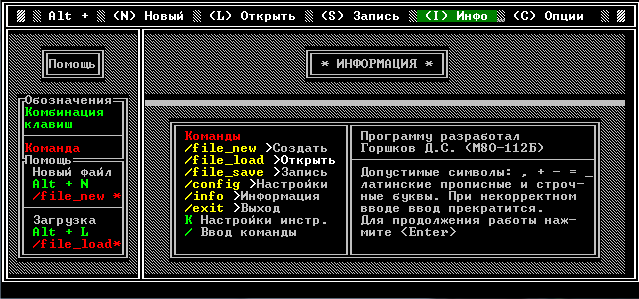
При открытии окна создания файла автоматически активируется ввод имени файла. Если пользователь не хочет создавать файл, рекомендуется очистить поле ввода и попытаться написать любой недопустимый символ. Например, цифру. К допустимым символам относятся любые буквы из латинского алфавита, а также знаки: + - \_ , =. После успешного создания файла Вас автоматически перенаправят на «поле для рисования»

1. *Окно загрузки файла.*



При открытии окна открытия файла автоматически активируется ввод имени файла. Если пользователь не хочет создавать файл, рекомендуется очистить поле ввода и попытаться написать любой недопустимый символ. Например, цифру. К допустимым символам относятся любые буквы из латинского алфавита, а также знаки: + - \_ , =. После успешного открытия файла Вас автоматически перенаправят на «поле для рисования»

1. *Окно информации*



Зеленым цветом указаны клавиши, желтым – команды. Для выхода из окна необходимо нажать клавишу Enter.

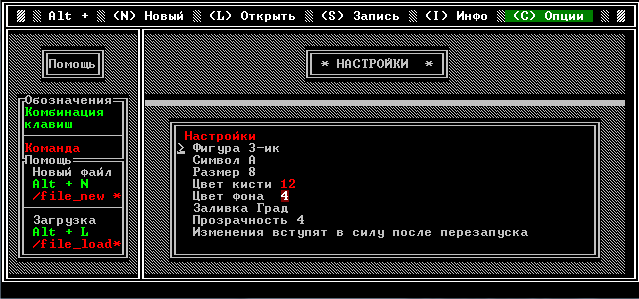
1. *Окно с настройками.*

В этом окне вы можете менять изначальные параметры кистей.

Управление:

Стрелки вверх и вниз – выбор опции

Стрелки влево и вправо – изменение параметра.

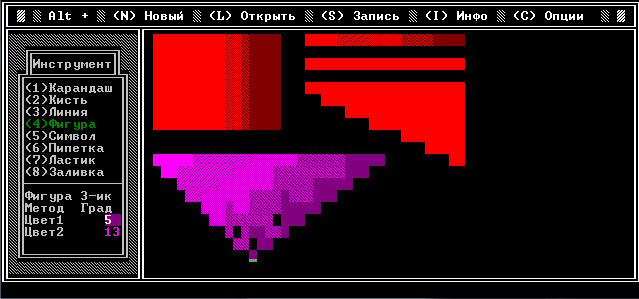


Допустимые значения параметров:

|  |  |  |
| --- | --- | --- |
| Параметр | Мин. значение | Макс. значение |
| Фигура | Четырехугольник | Треугольник |
| Символ | #32 | #254 |
| Размер | 1 | 8 |
| Цвет кисти | Черный (0) | Белый (15) |
| Цвет фона | Черный (0) | Серый (7(15)) |
| Заливка | Полная | Градиентная |
| Прозрачность | 0 | 4 |

Для выхода из окна и сохранения настроек необходимо нажать на клавишу Enter.

1. *Рабочий экран*



Рабочий экран – основной экран, который содержит: рабочую область (расположена справа) и окно с инструментарием (расположено слева).

Управление (Помимо клавиш из Основного окна)

|  |  |
| --- | --- |
| Клавиша | Эффект |
| Стрелки | Перемещение курсора |
| Enter | Печать символа |
| 1, P | Выбор инструмента Карандаш |
| 2, B | Выбор инструмента Кисть |
| 3, S | Выбор инструмента Линия |
| 4, L | Выбор инструмента Фигура |
| 5, U | Выбор инструмента Символ |
| 6, F | Выбор инструмента Пипетка |
| 7, A | Выбор инструмента Ластик |
| 8, E | Выбор инструмента Заливка |
| K | Настройки инструмента. |

**Инструменты:**

1. *Карандаш.*

Рисует закрашенный символ с прозрачностью и размером, установленными пользователем.

C:\Users\Дима\Desktop\Скриншоты для доклада\И2.bmp

1. *Кисть.*

Рисует квадрат (согласно системе координат), который при этом плавно закрашен. При закраске слева направо – от цвета кисти к фону. Интенсивность закрашивания и размеры задаются пользователем.



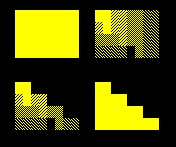
1. *Линия.*

Рисует прямую линию. При рисовании линии сначала задаются координаты начала линии, а затем ее конца. Режим закрашивания линии – плавный (от цвета кисти к цвету фона) или монотонный, а так же интенсивность закрашивания (можно выбрать в меню настроек для карандаша и кисти (вкладка Прозрачность)) задаются пользователем.



1. *Фигура.*

Рисует простую геометрическую фигуру: треугольник и прямоугольник. При рисовании сначала задаются координаты вершин (3 для треугольника и 2 для прямоугольника). Режим закрашивания фигуры – плавный (от цвета кисти к цвету фона) или монотонный, а так же интенсивность закрашивания (можно выбрать в меню настроек для карандаша и кисти (вкладка Прозрачность)) задаются пользователем.



1. *Символ.*

Отрисовывает символ из таблицы ASCII. Символ и его цвет задается пользователем.

1. *Пипетка.*

Собирает информацию о символе, его цвете и цвете фона с того места, где сейчас находится курсор.

1. *Ластик.*

Закрашивает место, где находится курсор, цветом фона.

1. *Заливка.*

Плавно закрашивает всю рабочую область. Цвет закрашивания и интенсивность (в меню параметров карандаша или кисти (вкладка Прозрачность)).

Работа в окне параметров инструментов аналогична работе в окне настроек.

Таблица параметров (З – Зависит, но в списке в программе не указан)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Инструмент | Параметр | | | | | | |
| Размер | Прозрачность | Фигура | Символ | Цвет | Фон | Заливка |
| Карандаш | **+** | **+** | **-** | **-** | **+** | **+** | **-** |
| Кисть | **+** | **+** | **-** | **-** | **+** | **+** | **-** |
| Линия | **-** | **З** | **-** | **-** | **+** | **+** | **+** |
| Фигура | **-** | **З** | **+** | **-** | **+** | **+** | **+** |
| Символ | **-** | **-** | **-** | **+** | **+** | **+** | **-** |
| Пипетка | **-** | **-** | **-** | **-** | **-** | **-** | **-** |
| Ластик | **+** | **-** | **-** | **-** | **-** | **+** | **-** |
| Заливка | **-** | **З** | **-** | **-** | **+** | **+** | **-** |

## Список заявленных особенностей проекта и выявленных ошибок

1. При загрузке пустого файла все рабочее пространство заполняется различными символами. Чаще – символами «стрелка влево», или иными символами. На дальнейшую работу программы влияет слабо.
2. При открытии окон: создания, открытия файлов, информации и настроек, окно закрывает не всю рабочую область (точнее – не заполняет предпоследний столбец справа).
3. При попытке открыть несуществующий файл, программа выдает сообщение о том, что загружаемого файла не существует, **и моментально закрывает** окно открытия файла. На дальнейшую работу программы влияет слабо.
4. «Скачущий курсор» - постоянно перемещающийся курсор при открытии окон, отрисовывании новых изображений связан с методом вывода символов самой консолью.
5. «Графические артефакты» при отрисовывании прямоугольников и треугольников, которые описаны выше.
6. Незаполненное место в файле DATADATA. Возникло из-за полной переработки метода отрисовывания изображений.
7. При нажатии на некоторые символы (в частности, **почти** все служебные (т. е. с скан-кодом = #0, #1..#255)) в поле ввода могут появляться различные символы, которые там быть не должны, однако на работу программы они влияют слабо. (Не решено в связи с тем, что при вводе должны обрабатываться нажатия на служебные клавиши и служебные комбинации клавиш). Как итог – при использовании таких символов при работе с файлами и при вводе команд, ввод может быть обработан некорректно, в результате чего программа, например, не сможет открыть файл, содержащий символы, которые были напечатаны в результате данной ошибки, при вводе допустимых символов.
8. При нажатии стрелки вниз или вверх в окне параметров инструмента, происходит выход за пределы допустимых параметров. При этом происходит неудачная попытка изменить несуществующий параметр (ничего не происходит). Что бы избавиться от данной проблемы, достаточно «поднять» указатель на ту позицию, где параметр существует. Так же, некоторые параметры не указываются в окне с параметрами инструментов, но при этом от них зависят (В основном – интенсивность закрашивания, зависящая от прозрачности кисти).
9. При попытке открыть файл, с которым программа уже работает (после создания, но не после сохранения), происходит неудачная обработка файла, и программа выдает сообщение об ошибке «Нечего открывать!». Связана ошибка с методом проверки имени файла, которая сопряжена с проверкой на то, открываем ли мы файл на чтение из основного окна, так как без нее программа часто аварийно прекращала работу из-за попытки открыть файл на чтение, который уже открыт на запись. Также, при попытке открыть файл, созданный в период работы программы между созданиями файлов (пример: запустили – создали файл – попытались открыть (неудачно) файл - создали другой файл – попытались открыть первый файл (успешно), и второй (неудачно)), будет выводиться ошибка «Нечего открывать!». При попытке создать файл с таким же именем, какое было у загружаемого перед созданием файла, происходит фатальная ошибка и аварийное завершение работы программы.

## Список литературы

[1] https://ru.wikipedia.org/wiki/Алгоритм\_Брезенхэма

[2] https://ru.wikibooks.org/wiki/Реализации\_алгоритмов/Алгоритм\_Брезенхэма

[3] В.В.Фаронов «Turbo Pascal 7.0. Учебный курс»

[4] https://habrahabr.ru/post/248153/ - Несмотря на то, что данный материал был ориентирован на написание «собственного 3д-рендерера», а за основу был взят язык программирования Си, в нем достаточно ясно и доступно описаны многие алгоритмы, с помощью которых работает проект.

## Код программы

program pseudographic;

uses

crt,

SysUtils;

const

{Flags}

Flag\_Paused = 0;

Flag\_OpenFileWork = 2;

Flag\_Launched = 3;

Flag\_Chooser = 4;

var

fgraph, fdata, fconfig, fout, flog, fload: Text;

Config\_Info: array [1..8] of byte;

Error\_Messages: array [1..3] of string;

UsedScript: word;

Programm\_Flags: word;

Year, Month, Day: word;

YearS, MonthS, DayS, FileName: string;

Input, CommandInput, LogFile: string;

Colorer, CorrectInput, Scripted, Saved, Translate, Loaded, Created: boolean;

i, j, k: word;

Output: string;

key, unused, symb\_paint: char;

t1, t2, t3, t4, t5, t6, t7: integer;

{Переменные для работы с графикой}

cursor\_xpos, cursor\_ypos: byte; {курсор}

starter\_xpos, starter\_ypos: byte; {}

ender\_xpos, ender\_ypos: byte; {}

n1\_pressed, n2\_pressed: boolean;

n1\_xpos, n1\_ypos: byte; {}

ch: char;

selected\_instr, size, color\_symb, color\_bg, pbucket\_method, opacity, figure, used\_char: byte; {инструмент}

drawed\_char: array [1..1220] of char;

drawed\_color: array [1..1220] of byte;

procedure LogSave;

begin

close(flog);

Assign(flog, LogFile);

{$I-}

Append(flog);

{$I+}

end;

{Процедура, генерирующая сообщение о фатальной ошибке без

использования графической составляющей программы}

procedure ErrorFW(Number: integer);

begin

writeln('Fatal Error #', Number, ': ', Error\_Messages[Number]);

writeln(flog, '[', Day, '-', Month, '-', Year, ': ', TimeToStr(Time),

'] [E] Fatal Error #', Number, ': ', Error\_Messages[Number]);

LogSave;

readln;

halt;

end;

{Перевод из 16-ого кода в 10-й}

function Perevod(chr: string): longint;

var

i: byte;

begin

Perevod := 0;

for i := 1 to length(chr) do

Perevod := Perevod \* 256 + Ord(chr[i]);

end;

{посимвольное чтение}

procedure scan\_f(MaxLength: byte);

begin

Input:='';

while (key <> #10) and (key <> #13) do

begin

if (length(Input)>MaxLength) then begin

Input:='';

break;

end;

key := readkey;

if not(key in ['A'..'Z']) and not(key in ['a'..'z']) and not(key in ['\_',',','+','-', #75, #72,'=']) and not(key in [#0..#32]) then begin

Input:='';

writeln(flog, '[', Day, '-', Month, '-', Year, ': ', TimeToStr(Time),

'] [E] Incorrect data input!'); LogSave;

break;

end;

if key in [#0..#31] then begin

if key = #8 then begin

If Input <> '' then

begin

GotoXY(WhereX - 1, WhereY);

Write(' ');

GotoXY(WhereX - 1, WhereY);

Delete(Input, length(Input), 1);

end;

end;

end

else

begin

Write(key);

Input := Input + key;

end;

end;

end;

{Отрисовка изображения}

procedure Draw\_Image(id, PosX, PosY: byte);

var

i, j, size: longint;

strr: string;

ch: char;

sizeX, sizeY, X\_pos, y\_pos: byte;

Colorer: boolean;

begin

{Возврат в начало файла}

Close(fdata);

Close(fgraph);

Assign(fgraph, 'GRAPHDATA.pgd'); {файл графики}

{$I-}

Reset(fgraph);

{$I+}

if ioresult <> 0 then

ErrorFW(1);

Assign(fdata, 'DATADATA.pgd'); {файл с данными}

{$I-}

Reset(fdata);

{$I+}

if ioresult <> 0 then

ErrorFW(2);

for i := 0 to 1018 + id \* 4 do

begin

Read(fdata, ch);

end;

for i := 1 to 4 do

begin {перевод из 16-ой системы в 10-ую}

Read(fdata, ch);

strr := strr + ch;

end;

size := Perevod(strr);

for i := 0 to size + 16 do Read(fgraph, ch);

sizeX := Perevod(ch);

Read(fgraph, ch);

sizeY := Perevod(ch);

x\_pos := 0;

y\_pos := 0;

Colorer := True;

gotoxy(posx, posy);

for i := 1 to (sizeX \* sizeY) \* 2 do

begin

Read(fgraph, ch);

if x\_pos = sizeX then

begin

if (y\_pos <> sizeY) and (WhereX <> 80) then

gotoxy(WhereX - x\_pos, WhereY + 1);

Inc(y\_pos);

x\_pos := 0;

end;

if Colorer then

begin

TextColor((Ord(ch) and 15));

TextBackground((Ord(ch) and 240) shr 4);

Colorer := False;

end

else

begin

Write(ch);

Inc(x\_pos);

Colorer := True;

end;

end;

end;

{Отрисовка линии}

procedure DrawLine(x1, x2, y1, y2: integer);

var delta\_xpos, delta\_ypos, delta2, delta3, mask\_op, t1, t2, Careter, smb: integer;

begin

delta\_xpos:= abs(x2-x1);

delta\_ypos:= abs(y2-y1);

delta2:=delta\_xpos-delta\_ypos;

delta3:=delta2\*2;

if delta\_xpos > delta\_ypos then mask\_op:= delta\_xpos div (opacity+1) else mask\_op:= delta\_ypos div (opacity+1);

if x1 < x2 then t1:=1 else t1:=-1;

if y1 < y2 then t2:=1 else t2:=-1;

Careter:=0;

smb:=opacity;

gotoXY(18+x1, 4 + y1);

while (x1 <> x2) or (y1 <> y2) do begin

gotoXY(18+x1, 3 + y1);

delta3:=delta2\*2;

case smb of

0: symb\_paint:=' ';

1: symb\_paint:=#176;

2: symb\_paint:=#177;

3: symb\_paint:=#178;

4: symb\_paint:=#219;

end;

write(symb\_paint);

gotoXY(WhereX-1, WhereY);

drawed\_color[(y1-1)\*61+x1] := color\_symb + (color\_bg shl 4);

drawed\_char[(y1-1)\*61+x1] := symb\_paint;

if delta3 > delta\_ypos \* (-1) then begin

delta2 := delta2 - delta\_ypos;

x1 := x1 + t1;

end;

if delta3 < delta\_xpos then begin

delta2 := delta2 + delta\_xpos;

y1 := y1 + t2;

end;

if (Careter = mask\_op) and (pbucket\_method = 2) then begin

if (smb > 0) then dec(smb);

Careter:=0;

end else inc(Careter);

end;

n1\_pressed := false;

n2\_pressed := false;

case smb of

0: symb\_paint:=' ';

1: symb\_paint:=#176;

2: symb\_paint:=#177;

3: symb\_paint:=#178;

4: symb\_paint:=#219;

end;

gotoXY(18+x2, 4 + y2 - 1);

write(symb\_paint);

gotoXY(WhereX-1, WhereY);

drawed\_color[(y1-1)\*61+x1] := color\_symb + (color\_bg shl 4);

drawed\_char[(y1-1)\*61+x1] := symb\_paint;

end;

{отрисовка окна с интерфейсом выбранного инструмента}

procedure Select\_Instr(instr: byte);

begin

if UsedScript = 7 then begin

Draw\_Image(29+selected\_instr\*3, 4, 7+selected\_instr);

selected\_instr:=instr;

Draw\_Image(29+selected\_instr\*3+1, 4, 7+selected\_instr);

case instr of

1,2: Draw\_Image(70, 4, 17);

3: Draw\_Image(71, 4, 17);

4: Draw\_Image(72, 4, 17);

5: Draw\_Image(73, 4, 17);

6: Draw\_Image(76, 4, 17);

7: Draw\_Image(74, 4, 17);

8: Draw\_Image(75, 4, 17);

end;

writeln(flog, '[', Day, '-', Month, '-', Year, ': ', TimeToStr(Time),

'] [I] Selected instrument #', instr);

LogSave;

gotoXY(18+cursor\_xpos, 4+cursor\_ypos);

if (Programm\_Flags and (1 shl Flag\_Chooser) = 0) then

Programm\_Flags:=Programm\_Flags + (1 shl Flag\_Chooser);

end;

end;

begin

{Задание и обнуление переменных (а также заданиe начальных значений); создание лог файла}

Error\_Messages[1] := 'Cant open file GRAPHDATA.pgd!';

Error\_Messages[2] := 'Cant open file DATADATA.pgd!';

Error\_Messages[3] := 'Cant open file CONFIGDATA.pgd!';

n1\_pressed:=false;

n2\_pressed:=false;

Colorer := True;

FileName :='';

for i:=1 to 1220 do begin

drawed\_char[i]:=' ';

drawed\_color[i]:=15;

end;

Saved:=false;

Created:=false;

selected\_instr := 1;

{Получение значения даты}

DeCodeDate(Date, Year, Month, Day);

str(Day, DayS);

str(Month, MonthS);

str(Year, YearS);

Output := 'PG\_LOG\_' + DayS + '\_' + MonthS + '\_' + YearS + '.txt';

LogFile:= Output;

Assign(flog, Output);

Rewrite(flog);

{Открытие файлов}

Assign(fgraph, 'GRAPHDATA.pgd'); {файл графики}

{$I-}

Reset(fgraph);

{$I+}

if ioresult <> 0 then

ErrorFW(1);

writeln(flog, '[', Day, '-', Month, '-', Year, ': ', TimeToStr(Time),

'] [S] Program start!');

writeln(flog, '[', Day, '-', Month, '-', Year, ': ', TimeToStr(Time),

'] [F] Graphic file opened!');

LogSave;

Assign(fdata, 'DATADATA.pgd'); {файл с данными}

{$I-}

Reset(fdata);

{$I+}

if ioresult <> 0 then

ErrorFW(2);

writeln(flog, '[', Day, '-', Month, '-', Year, ': ', TimeToStr(Time),

'] [F] Data file opened!'); LogSave;

Assign(fconfig, 'CONFIGDATA.pgd'); {файл конфигурации}

{$I-}

Reset(fconfig);

{$I+}

if ioresult <> 0 then

ErrorFW(3);

writeln(flog, '[', Day, '-', Month, '-', Year, ': ', TimeToStr(Time),

'] [F] Config file opened!'); LogSave;

UsedScript := 1;

for k:=1 to 7 do begin

read(fconfig, ch);

Config\_Info[k]:=Perevod(ch);

end;

figure := Config\_Info[1];

used\_char:=Config\_Info[2];

size:= Config\_Info[3];

color\_symb:=Config\_Info[4];

color\_bg:=Config\_Info[5];

pbucket\_method:=Config\_Info[6];

opacity:=Config\_Info[7];

Loaded:=false;

{Основная часть программы}

while (UsedScript <> 54) do

begin {цикл с работой программы. Будет работать, пока не будет выполняться скрипт 54.}

Scripted:=false;

if UsedScript = 1 then

begin {скрипт запуска программы и отрисовки окна}

Draw\_Image(60, 1, 1);

UsedScript := 2;

Programm\_Flags := Programm\_Flags + 1 shl Flag\_Launched;

{Установка флага, указывающего на то, что программа запущена.}

end;

if UsedScript = 2 then

begin {скрипт проверки состояния программы и отрисовки интерфейса под текущее состояние}

if (Programm\_Flags and (1 shl Flag\_Launched) = (1 shl (Flag\_Launched)))

{проверка на флаг запуска} then

begin

if (Programm\_Flags and (1 shl Flag\_Paused) = (1 shl (Flag\_Paused)))

then Programm\_Flags:=Programm\_Flags - (1 shl Flag\_Paused);

Draw\_Image(61, 2, 4);

Draw\_Image(62, 19, 4);

Draw\_Image(1, 14, 2);{N}

Draw\_Image(4, 26, 2);{L}

Draw\_Image(7, 40, 2);{S}

Draw\_Image(10, 53, 2);{I}

Draw\_Image(13, 64, 2);{C}

GotoXY(1, 25);

end;

end;

if UsedScript = 4 then

begin {скрипт создания нового файла}

if (Programm\_Flags and (1 shl Flag\_OpenFileWork) = 0) or (Programm\_Flags and (1 shl Flag\_Paused) = 1 shl Flag\_Paused) then{Проверка }

begin

Programm\_Flags:=Programm\_Flags + (1 shl Flag\_OpenFileWork);

Draw\_Image(63, 19, 4);

Draw\_Image(2, 14, 2);

TextColor(7);

TextBackground(0);

{Чтение названия файла}

GotoXY(23, 12);

TextColor(7);

scan\_f(23);

{Очистка строки}

GotoXY(23, 12);

write(' ');

GotoXY(23, 12);

if Input <> '' then begin

if ((filename <> '') and not Saved and Created) or Loaded then close(fout);

FileName:=Input+'.pgf';

Assign(fout, FileName);

{$I-}

Rewrite(fout);

{$I+}

write(' ');

Draw\_Image(64, 41, 12);

GotoXY(1, 25);

writeln(flog, '[', Day, '-', Month, '-', Year, ': ', TimeToStr(Time),

'] [F] File ', filename, ' created!'); LogSave;

unused:=readkey;

for t1:=1 to 1220 do begin

drawed\_color[t1] := 0;

drawed\_char[t1] := ' ';

end;

UsedScript := 7;

Created:=true;

if (Programm\_Flags and (1 shl Flag\_Chooser) = 0) then

Programm\_Flags:=Programm\_Flags + (1 shl Flag\_Chooser);

end;

Programm\_Flags:=Programm\_Flags - (1 shl Flag\_OpenFileWork);

if UsedScript <> 7 then if (Programm\_Flags and (1 shl Flag\_Paused) = (1 shl Flag\_Paused)) and (Programm\_Flags and (1 shl Flag\_Launched) = (1 shl Flag\_Launched))

then

begin

UsedScript := 2;

end else UsedScript := 7;

Scripted:=true;

Draw\_Image(1, 14, 2);

end

else if (Programm\_Flags and (1 shl Flag\_Launched) = (1 shl Flag\_Launched))

{проверка на флаг запуска} then

UsedScript := 2;

end;

if UsedScript = 5 then

begin {скрипт открытия файла}

if (Programm\_Flags and (1 shl Flag\_OpenFileWork) = 0) or (Programm\_Flags and (1 shl Flag\_Paused) = 1 shl Flag\_Paused) then{Проверка }

begin

Programm\_Flags:=Programm\_Flags + (1 shl Flag\_OpenFileWork);

Draw\_Image(77, 19, 4);

Draw\_Image(5, 26, 2);{L}

TextColor(7);

TextBackground(0);

{Чтение названия файла}

GotoXY(23, 12);

TextColor(7);

scan\_f(23);

{Очистка строки}

GotoXY(23, 12);

write(' ');

GotoXY(23, 12);

if Input <> '' then begin

FileName:=Input+'.pgf';

Assign(fload, FileName);

{$I-}

Reset(fload);

{$I+}

if ioresult <> 0 then begin Draw\_Image(78,42,14) end else begin

Colorer:=true;

for t1:=1 to 1220 do begin

Read(fload, ch);

drawed\_color[t1] := ord(ch);

Read(fload, ch);

drawed\_char[t1] := ch;

end;

write(' ');

Draw\_Image(82, 41, 12);

GotoXY(1, 25);

unused:=readkey;

UsedScript := 7;

Loaded:=true;

close(fload);

if (Programm\_Flags and (1 shl Flag\_Launched) = (1 shl Flag\_Launched)) then begin

FileName:=Input+'.pgf';

Assign(fout, FileName);

{$I-}

Rewrite(fout);

{$I+}

end;

writeln(flog, '[', Day, '-', Month, '-', Year, ': ', TimeToStr(Time),

'] [F] File ', filename, ' opened!'); LogSave;

end;

end;

Programm\_Flags:=Programm\_Flags - (1 shl Flag\_OpenFileWork);

if UsedScript <> 7 then if (Programm\_Flags and (1 shl Flag\_Paused) = (1 shl Flag\_Paused)) and (Programm\_Flags and (1 shl Flag\_Launched) = (1 shl Flag\_Launched))

then

begin

UsedScript := 2;

end else UsedScript := 7;

Scripted:=true;

Draw\_Image(4, 26, 2);{L}

end; end;

if UsedScript = 6 then

begin {скрипт сохранения файла}

if (Programm\_Flags and (1 shl Flag\_Launched) = 0) then begin

Draw\_Image(8, 40, 2);{S}

if Saved then begin

Assign(fout, FileName);

{$I-}

Rewrite(fout);

{$I+}

end;

for i:=1 to 1220 do begin

write (fout, chr(drawed\_color[i]));

write (fout, drawed\_char[i]);

end;

close (fout);

Draw\_Image(65, 41, 12);

unused:=readkey;

UsedScript := 7;

Saved:=true;

writeln(flog, '[', Day, '-', Month, '-', Year, ': ', TimeToStr(Time),

'] [F] File saved!'); LogSave;

end else begin

Draw\_Image(68, 36, 12);

UsedScript := 2;

writeln(flog, '[', Day, '-', Month, '-', Year, ': ', TimeToStr(Time),

'] [FE] File didnt save!');

end;

Draw\_Image(7, 40, 2);{S}

end;

if UsedScript = 7 then

begin {скрипт работы с графикой}

if ((Programm\_Flags and (1 shl Flag\_Launched) = (1 shl Flag\_Launched)) or (Programm\_Flags and (1 shl Flag\_Paused) = 1 shl Flag\_Paused)) then begin

GotoXY(19, 4);

for t1:=0 to 19 do begin

for t2:=1 to 61 do begin

TextBackground((drawed\_color[t1\*61+t2] and 240) shr 4);

TextColor(drawed\_color[t1\*61+t2] and 15);

write (drawed\_char[t1\*61+t2]);

end;

GotoXY(19, WhereY+1);

end;

Draw\_Image(66, 2, 4);

for i:=1 to 8 do begin

if i <> selected\_instr then Draw\_Image(29+i\*3, 4, 7+i) else Draw\_Image(30+i\*3, 4, 7+i);

end;

GotoXY(19, 4);

if (Programm\_Flags and (1 shl Flag\_Launched) = (1 shl Flag\_Launched)) then

Programm\_Flags:=Programm\_Flags - (1 shl Flag\_Launched);

if (Programm\_Flags and (1 shl Flag\_Paused) = 1 shl Flag\_Paused) then

Programm\_Flags:=Programm\_Flags - (1 shl Flag\_Paused);

if (Programm\_Flags and (1 shl Flag\_Chooser) = 0) then

Programm\_Flags:=Programm\_Flags + (1 shl Flag\_Chooser);

cursor\_xpos:=1;

cursor\_ypos:=0;

end;

if ((Programm\_Flags and (1 shl Flag\_Chooser) = (1 shl Flag\_Chooser))) then begin

case selected\_instr of

1,2: begin

Draw\_Image(70, 4, 17);

TextBackground(0); TextColor(7);

gotoXY(14,17); Write(size);

gotoXY(14,18); Write(opacity);

TextColor(15); TextBackground(COLOR\_BG);

gotoXY(14,19); Write(color\_bg);

TextBackground(0); TextColor(COLOR\_symb);

gotoXY(14,20); Write(' ');

gotoXY(14,20); Write(color\_symb); end;

3: begin

Draw\_Image(71, 4, 17);

TextBackground(0); TextColor(7);

gotoXY(11,17);

if pbucket\_method = 1 then Write(#143#174#171#173) else Write(#131#224#160#164);

gotoXY(14,17);

TextColor(15); TextBackground(COLOR\_BG);

gotoXY(14,18); Write(' ');

gotoXY(14,18); Write(color\_bg);

TextBackground(0); TextColor(COLOR\_symb);

gotoXY(14,19); Write(' ');

gotoXY(14,19); Write(color\_symb); end;

4: begin

Draw\_Image(72, 4, 17);

TextBackground(0); TextColor(7);

gotoXY(11,17);

if figure = 1 then Write(#52#45#168#170) else Write(#51#45#168#170);

gotoXY(14,18); gotoXY(11,18);

if pbucket\_method = 1 then Write(#143#174#171#173) else Write(#131#224#160#164);

gotoXY(14,19);

TextColor(15); TextBackground(COLOR\_BG);

gotoXY(14,19); Write(' ');

gotoXY(14,19); Write(color\_bg);

TextBackground(0); TextColor(COLOR\_symb);

gotoXY(14,20); Write(' ');

gotoXY(14,20); Write(color\_symb); end;

5: begin

Draw\_Image(73, 4, 17);

TextBackground(0); TextColor(7);

gotoXY(14,17); write(chr(used\_char));

TextColor(15); TextBackground(COLOR\_BG);

gotoXY(14,19); Write(' ');

gotoXY(14,19); Write(color\_bg);

TextBackground(0); TextColor(COLOR\_symb);

gotoXY(14,20); Write(' ');

gotoXY(14,20); Write(color\_symb); end;

6: Draw\_Image(76, 4, 17);

7: begin

Draw\_Image(74, 4, 17);

TextBackground(0); TextColor(7);

gotoXY(14,17); write(size);

TextColor(15); TextBackground(COLOR\_BG);

gotoXY(14,18); Write(' ');

gotoXY(14,18); Write(color\_bg); end;

8: begin

Draw\_Image(75, 4, 17);

TextBackground(0); TextColor(COLOR\_symb);

gotoXY(14,17); Write(' ');

gotoXY(14,17); Write(color\_symb);

TextColor(15); TextBackground(COLOR\_BG);

gotoXY(14,18); Write(' ');

gotoXY(14,18); Write(color\_bg);

end;end;

Programm\_Flags:= Programm\_Flags - (1 shl Flag\_Chooser);

gotoXY(18+cursor\_xpos, 4+cursor\_ypos);

end; end;

if UsedScript = 43 then

begin {скрипт редактирования меню}

case selected\_instr of

1, 2: begin

case t1 of

1: begin

TextBackground(0); TextColor(10);

gotoXY(14,17);Write(size);

TextColor(7);gotoXY(14,18);

Write(opacity);TextColor(15);

TextBackground(COLOR\_BG);

gotoXY(14,19);Write(color\_bg);

TextBackground(0);TextColor(COLOR\_symb);

gotoXY(14,20);Write(' ');

gotoXY(14,20);Write(color\_symb);end;

2: begin

TextBackground(0);TextColor(10);

gotoXY(14,18);Write(opacity);

TextColor(7);gotoXY(14,17);

Write(size);TextColor(15);

TextBackground(COLOR\_BG);

gotoXY(14,19);Write(color\_bg);

TextBackground(0);TextColor(COLOR\_symb);

gotoXY(14,20);Write(' ');

gotoXY(14,20);Write(color\_symb);end;

3: begin

TextBackground(0);TextColor(7);

gotoXY(14,18);Write(opacity);

gotoXY(14,17);Write(size);

TextColor(15);TextBackground(COLOR\_BG);

gotoXY(14,19);Write(' ');

gotoXY(14,19);Write(color\_bg);

TextBackground(0);TextColor(COLOR\_symb);

gotoXY(14,20);Write(' ');

gotoXY(14,20);Write(color\_symb);end;

4: begin

TextBackground(0);TextColor(7);

gotoXY(14,18);Write(opacity);

gotoXY(14,17);Write(size);

TextColor(15);TextBackground(COLOR\_BG);

gotoXY(14,19);Write(' ');

gotoXY(14,19);Write(color\_bg);

TextBackground(0);TextColor(COLOR\_symb);

gotoXY(14,20);Write(' ');

gotoXY(14,20);Write(color\_symb);end;

end;end;

3: begin

case t1 of

1: begin

TextBackground(0);TextColor(10);

gotoXY(11,17);

if pbucket\_method = 1 then Write(#143#174#171#173) else Write(#131#224#160#164);

gotoXY(14,17);TextColor(15);

TextBackground(COLOR\_BG);

gotoXY(14,18);Write(' ');

gotoXY(14,18);Write(color\_bg);

TextBackground(0);TextColor(COLOR\_symb);

gotoXY(14,19);Write(' ');

gotoXY(14,19);Write(color\_symb);end;

2: begin

TextBackground(0);TextColor(7);

gotoXY(11,17);

if pbucket\_method = 1 then Write(#143#174#171#173) else Write(#131#224#160#164);

gotoXY(14,17);

TextColor(15); TextBackground(COLOR\_BG);

gotoXY(14,18);Write(' ');

gotoXY(14,18);Write(color\_bg);

TextBackground(0);TextColor(COLOR\_symb);

gotoXY(14,19);Write(' ');

gotoXY(14,19);Write(color\_symb);end;

3: begin

TextBackground(0);TextColor(7);

gotoXY(11,17);

if pbucket\_method = 1 then Write(#143#174#171#173) else Write(#131#224#160#164);

gotoXY(14,17);

TextColor(15);TextBackground(COLOR\_BG);

gotoXY(14,18);Write(' ');

gotoXY(14,18);Write(color\_bg);

TextBackground(0);TextColor(COLOR\_symb);

gotoXY(14,19);Write(' ');

gotoXY(14,19);Write(color\_symb);end;

end;end;

4: begin

case t1 of

1: begin

TextBackground(0);TextColor(10);

gotoXY(11,17);

if figure = 1 then Write(#52#45#168#170) else Write(#51#45#168#170);

gotoXY(14,18);TextColor(7);

gotoXY(11,18);

if pbucket\_method = 1 then Write(#143#174#171#173) else Write(#131#224#160#164);

gotoXY(14,19);TextColor(15);

TextBackground(COLOR\_BG);

gotoXY(14,19);Write(' ');

gotoXY(14,19);Write(color\_bg);

TextBackground(0);TextColor(COLOR\_symb);

gotoXY(14,20);Write(' ');

gotoXY(14,20);Write(color\_symb);end;

2: begin

TextBackground(0);TextColor(7);

gotoXY(11,17);

if figure = 1 then Write(#52#45#168#170) else Write(#51#45#168#170);

gotoXY(14,18);TextColor(10); gotoXY(11,18);

if pbucket\_method = 1 then Write(#143#174#171#173) else Write(#131#224#160#164);

gotoXY(14,19);TextColor(15);

TextBackground(COLOR\_BG);

gotoXY(14,19);Write(' ');

gotoXY(14,19);Write(color\_bg);

TextBackground(0);TextColor(COLOR\_symb);

gotoXY(14,20);Write(' ');

gotoXY(14,20);Write(color\_symb);end;

3, 4: begin

TextBackground(0);TextColor(7);

gotoXY(11,17);

if figure = 1 then Write(#52#45#168#170) else Write(#51#45#168#170);

gotoXY(14,18);gotoXY(11,18);

if pbucket\_method = 1 then Write(#143#174#171#173) else Write(#131#224#160#164);

gotoXY(14,19);TextColor(15);

TextBackground(COLOR\_BG);

gotoXY(14,19);Write(' ');

gotoXY(14,19);Write(color\_bg);

TextBackground(0);TextColor(COLOR\_symb);

gotoXY(14,20);Write(' ');gotoXY(14,20);

Write(color\_symb);end;

end;end;

5: begin

case t1 of

1: begin

TextBackground(0);TextColor(10);

gotoXY(14,17);write(chr(used\_char));

TextColor(15);TextBackground(COLOR\_BG);

gotoXY(14,19);Write(' ');

gotoXY(14,19);Write(color\_bg);

TextBackground(0);TextColor(COLOR\_symb);

gotoXY(14,20);Write(' ');

gotoXY(14,20);Write(color\_symb);end;

2,3: begin

TextBackground(0);TextColor(7);

gotoXY(14,17);write(chr(used\_char));

TextColor(15);TextBackground(COLOR\_BG);

gotoXY(14,19);Write(' ');

gotoXY(14,19);Write(color\_bg);

TextBackground(0);TextColor(COLOR\_symb);

gotoXY(14,20);Write(' ');

gotoXY(14,20);Write(color\_symb);end;

end;end;

7: begin

case t1 of

1: begin

TextBackground(0);TextColor(10);

gotoXY(14,17);write(size);

TextColor(15);TextBackground(COLOR\_BG);

gotoXY(14,18);Write(' ');

gotoXY(14,18);Write(color\_bg);

end;

2: begin

TextBackground(0);TextColor(7);

gotoXY(14,17);write(size);

TextColor(15);TextBackground(COLOR\_BG);

gotoXY(14,18);Write(' ');

gotoXY(14,18);Write(color\_bg);end;

end;end;

8: begin

case t1 of

1,2: begin

TextBackground(0);TextColor(COLOR\_symb);

gotoXY(14,17); Write(' ');

gotoXY(14,17); Write(color\_symb);

TextColor(15); TextBackground(COLOR\_BG);

gotoXY(14,18); Write(' ');

gotoXY(14,18); Write(color\_bg);end;

end;

end;

end;

end;

if UsedScript = 50 then

begin {скрипт окна с настройками}

if (Programm\_Flags and (1 shl Flag\_OpenFileWork) = 0) or (Programm\_Flags and (1 shl Flag\_Paused) = 1 shl Flag\_Paused) then{Проверка }

begin

if (Programm\_Flags and (1 shl Flag\_OpenFileWork) = 0) then

Programm\_Flags:=Programm\_Flags + (1 shl Flag\_OpenFileWork);

Draw\_Image(81, 19, 4);

Draw\_Image(14, 64, 2);{I}

TextColor(7);

TextBackground(0);

unused:=#0;

t1:=1;

while (unused <>#10) and (unused <> #13) do begin

GotoXY(32,13); case Config\_Info[1] of

1: write(#52#45#168#170); 2: Write(#51#45#168#170); end;

GotoXY(32,14); write(chr(Config\_Info[2]));

GotoXY(32,15); write(Config\_Info[3]);

GotoXY(36,16); write(' ');

GotoXY(36,17); write(' ');

GotoXY(36,16); TextColor(Config\_Info[4]); write(Config\_Info[4]);

GotoXY(36,17); TextBackground(Config\_Info[5]); TextColor(15); write(Config\_Info[5]);

TextBackground(0); TextColor(7);

GotoXY(33,18); case Config\_Info[6] of

1: write(#143#174#171#173); 2: Write(#131#224#160#164); end;

GotoXY(38,19); write(Config\_Info[7]);

for t2:=1 to 7 do begin

gotoXY(23,12+t2); write(' ');

end;

gotoXY(23,12+t1); write('>');gotoXY(23,12+t1);

unused:=#0;

unused:=readkey;

if unused = #0 then begin

unused:=readkey;

case unused of

#72: begin {Bверх}

if t1 >1 then dec(t1);

end;

#80: begin {Bниз}

if t1 <7 then inc(t1);

end;

#75: begin {Влево}

case t1 of

1, 3, 6: if Config\_Info[t1] > 1 then dec(Config\_Info[t1]);

2: if Config\_Info[t1] > 32 then dec(Config\_Info[t1]);

7, 5, 4: if Config\_Info[t1] > 0 then dec(Config\_Info[t1]);

end;

end;

#77: begin

case t1 of

1, 6: if Config\_Info[t1] < 2 then inc(Config\_Info[t1]);

2: if Config\_Info[t1] < 254 then inc(Config\_Info[t1]);

3: if Config\_Info[t1] < 8 then inc(Config\_Info[t1]);

4, 5: if Config\_Info[t1] < 15 then inc(Config\_Info[t1]);

7: if Config\_Info[t1] < 4 then inc(Config\_Info[t1]);

end;

end;

end;

end;

end;

close(fconfig);

Assign(fconfig, 'CONFIGDATA.PGD');

{$I-}

Rewrite(fconfig);

{$I+}

for t2:=1 to 7 do write(fconfig, chr(Config\_Info[t2]));

close(fconfig);

Assign(fconfig, 'CONFIGDATA.PGD');

{$I-}

Reset(fconfig);

{$I+}

writeln(flog, '[', Day, '-', Month, '-', Year, ': ', TimeToStr(Time),

'] [F] Start config has been changed!'); LogSave;

Draw\_Image(13, 64, 2);{I}

if (Programm\_Flags and (1 shl Flag\_Paused) = (1 shl Flag\_Paused)) and (Programm\_Flags and (1 shl Flag\_Launched) = (1 shl Flag\_Launched))

then

begin

UsedScript := 2;

end else UsedScript := 7;

Scripted:=true;

Draw\_Image(4, 26, 2);{L}

end; end;

if UsedScript = 51 then

begin {скрипт открытия окна с информацией}

if (Programm\_Flags and (1 shl Flag\_OpenFileWork) = 0) or (Programm\_Flags and (1 shl Flag\_Paused) = 1 shl Flag\_Paused) then{Проверка }

begin

if (Programm\_Flags and (1 shl Flag\_OpenFileWork) = 0) then

Programm\_Flags:=Programm\_Flags + (1 shl Flag\_OpenFileWork);

Draw\_Image(80, 19, 4);

Draw\_Image(11, 53, 2);{I}

TextColor(7);

TextBackground(0);

{Чтение названия файла}

unused:=#0;

while (unused <>#10) and (unused <> #13) do unused:=readkey;

Draw\_Image(10, 53, 2);{I}

if (Programm\_Flags and (1 shl Flag\_Paused) = (1 shl Flag\_Paused)) and (Programm\_Flags and (1 shl Flag\_Launched) = (1 shl Flag\_Launched))

then

begin

UsedScript := 2;

end else UsedScript := 7;

Scripted:=true;

Draw\_Image(4, 26, 2);{L}

end; end;

{нажатие клавиш}

if not Scripted then begin {проверка на то, что в текущей итерации цикла не выполняется скрипт,

и что можно проверять нажатие клавиш}

key := readkey;

{writeln(ord(key);} {для отладки}

case key of

#0:

begin

key := ReadKey; {Read ScanCode}

{writeln(ord(key)); для отладки}

case key of

#75: begin {кнопка Влево}

if UsedScript = 7 then begin

if WhereX-1 <> 18 then begin gotoXY(WhereX-1,WhereY);

dec(cursor\_xpos);

end;

end;

if Programm\_Flags and (1 shl Flag\_Chooser) = (1 shl Flag\_Chooser) then begin

if UsedScript in [10..14] then

if UsedScript <> 10 then UsedScript:= UsedScript - 1;

end;

if UsedScript = 43 then begin

case selected\_instr of

1, 2: begin

case t1 of

1: if size > 1 then dec(size);

2: if opacity > 0 then dec(opacity);

3: if color\_bg > 0 then dec(color\_bg);

4: if color\_symb > 0 then dec(color\_symb);

end;

end;

3: begin

case t1 of

1: if pbucket\_method > 1 then dec(pbucket\_method);

2: if color\_bg > 0 then dec(color\_bg);

3: if color\_symb > 0 then dec(color\_symb);

end;

end;

4: begin

case t1 of

1: if figure > 1 then dec(figure);

2: if pbucket\_method > 1 then dec(pbucket\_method);

3: if color\_bg > 0 then dec(color\_bg);

4: if color\_symb > 0 then dec(color\_symb);

end;

end;

5: begin

case t1 of

1: if used\_char > 32 then dec(used\_char);

2: if color\_bg > 0 then dec(color\_bg);

3: if color\_symb > 0 then dec(color\_symb);

end;

end;

7: begin

case t1 of

1: if size > 1 then dec(size);

2: if color\_bg > 0 then dec(color\_bg);

end;

end;

8: begin

case t1 of

1: if color\_symb > 0 then dec(color\_symb);

2: if color\_bg > 0 then dec(color\_bg);

end;

end;

end; end;

end;

#77: begin {кнопка Вправо}

if UsedScript = 7 then begin

if WhereX+1 <> 80 then begin gotoXY(WhereX+1,WhereY);

inc(cursor\_xpos);

end;

end;

if UsedScript = 43 then begin

case selected\_instr of

1, 2: begin

case t1 of

1: if size < 8 then inc(size);

2: if opacity < 4 then inc(opacity);

3: if color\_bg < 15 then inc(color\_bg);

4: if color\_symb < 15 then inc(color\_symb);

end;

end;

3: begin

case t1 of

1: if pbucket\_method < 2 then inc(pbucket\_method);

2: if color\_bg < 15 then inc(color\_bg);

3: if color\_symb < 15 then inc(color\_symb);

end;

end;

4: begin

case t1 of

1: if figure < 2 then inc(figure);

2: if pbucket\_method < 2 then inc(pbucket\_method);

3: if color\_bg < 15 then inc(color\_bg);

4: if color\_symb < 15 then inc(color\_symb);

end;

end;

5: begin

case t1 of

1: if used\_char < 254 then inc(used\_char);

2: if color\_bg < 15 then inc(color\_bg);

3: if color\_symb < 15 then inc(color\_symb);

end;

end;

7: begin

case t1 of

1: if size < 8 then inc(size);

2: if color\_bg < 15 then inc(color\_bg);

end;

end;

8: begin

case t1 of

1: if color\_symb < 15 then inc(color\_symb);

2: if color\_bg < 15 then inc(color\_bg);

end;

end;

end;

end;

end;

#72: begin {кнопка Вверх}

if UsedScript = 7 then begin

if WhereY-1 <> 3 then begin gotoXY(WhereX,WhereY-1);

dec(cursor\_ypos);

end;

end;

if UsedScript = 43 then begin

if t1>1 then dec(t1);

end;

end;

#80: begin {кнопка Вниз}

if UsedScript = 7 then begin

if WhereY+1 <> 24 then begin gotoXY(WhereX,WhereY+1);

inc(cursor\_ypos);

end;

end;

if UsedScript = 43 then begin

if t1<5 then inc(t1);

end;

end;

#49: begin {N}

UsedScript := 4;

if Programm\_Flags and (1 shl Flag\_Paused) = 0 then

Programm\_Flags:=Programm\_Flags + (1 shl Flag\_Paused);

end;

#38: begin {L}

UsedScript := 5;

if Programm\_Flags and (1 shl Flag\_Paused) = 0 then

Programm\_Flags:=Programm\_Flags + (1 shl Flag\_Paused);

end;

#31: begin

UsedScript := 6;

if Programm\_Flags and (1 shl Flag\_Paused) = 0 then

Programm\_Flags:=Programm\_Flags + (1 shl Flag\_Paused);

end; {S}

#46: begin

UsedScript := 50;

if Programm\_Flags and (1 shl Flag\_Paused) = 0 then

Programm\_Flags:=Programm\_Flags + (1 shl Flag\_Paused);

end;{C}

#23: begin

UsedScript := 51;

if Programm\_Flags and (1 shl Flag\_Paused) = 0 then

Programm\_Flags:=Programm\_Flags + (1 shl Flag\_Paused);

end;{I}

end;

end;

#13: begin {кнопка Enter}

If UsedScript = 7 then begin

case selected\_instr of

1: begin

TextColor(color\_symb);

TextBackground(color\_bg);

t2:=cursor\_xpos;

t4:=cursor\_ypos;

for t1:=cursor\_ypos to cursor\_ypos + size -1 do begin

for t3:=cursor\_xpos to cursor\_xpos + size -1 do begin

if (t3 < 62) and (t3 > 0) and (t1 < 20) and (t1 >= 0) then begin

case opacity of

0: symb\_paint:=' ';

1: symb\_paint:=#176;

2: symb\_paint:=#177;

3: symb\_paint:=#178;

4: symb\_paint:=#219;

end;

write(symb\_paint);

drawed\_color[t1\*61+t3] := color\_symb + (color\_bg shl 4);

drawed\_char[t1\*61+t3] := symb\_paint;

end;

end;

if (t1 < 19) and (t1 >= 0) then

gotoXY(18+cursor\_xpos,WhereY+1)

else

gotoXY(18+cursor\_xpos,WhereY);

end;

gotoXY(18+cursor\_xpos,4+cursor\_ypos);

end;

2:begin

TextColor(color\_symb);

TextBackground(color\_bg);

t2:=cursor\_xpos;

t4:=cursor\_ypos;

t5:=size div (opacity+1);

t6:=opacity;

t7:=1;

for t1:=cursor\_ypos to cursor\_ypos + size -1 do begin

for t3:=cursor\_xpos to cursor\_xpos + size -1 do begin

if (t3 < 62) and (t3 > 0) and (t1 < 20) and (t1 >= 0) then begin

case t6 of

0: symb\_paint:=' ';

1: symb\_paint:=#176;

2: symb\_paint:=#177;

3: symb\_paint:=#178;

4: symb\_paint:=#219;

end;

write(symb\_paint);

drawed\_color[t1\*61+t3] := color\_symb + (color\_bg shl 4);

drawed\_char[t1\*61+t3] := symb\_paint;

if t7 = t5 then begin

t7:=1;

dec(t6);

end else inc(t7);

end;

end;

t6:=opacity;

if (t1 < 19) and (t1 >= 0) then

gotoXY(18+cursor\_xpos,WhereY+1)

else

gotoXY(18+cursor\_xpos,WhereY);

end;

gotoXY(18+cursor\_xpos,4+cursor\_ypos);

end;

3: begin

TextColor(color\_symb);

TextBackground(color\_bg);

if not n1\_pressed then begin

starter\_xpos:=cursor\_xpos;

starter\_ypos:=cursor\_ypos;

TextColor(0);

TextBackground(15);

write('1');

gotoXY(WhereX-1, WhereY);

n1\_pressed := true;

end else begin if (starter\_xpos <> cursor\_xpos) or (starter\_ypos <> cursor\_ypos) then begin

{Отрисовка линии по алгоритму Брезенхэма}

DrawLine(starter\_xpos, cursor\_xpos, starter\_ypos+1, cursor\_ypos+1);

end;

cursor\_xpos:=WhereX-18;

cursor\_ypos:=WhereY-4;

end;

end;

4: begin {Инструмент ФИГУРА}

if figure = 1 then begin {Фигура - ПРЯМОУГОЛЬНИК}

if not n1\_pressed then begin

starter\_xpos:=cursor\_xpos;starter\_ypos:=cursor\_ypos;

TextColor(0);TextBackground(15);

write('1');

gotoXY(WhereX-1, WhereY);

TextColor(color\_symb);TextBackground(color\_bg);

n1\_pressed := true;

end else begin if (starter\_xpos <> cursor\_xpos) or (starter\_ypos <> cursor\_ypos) then begin

t1:=cursor\_xpos; t2:= starter\_ypos+1; {Вершина В}

t3:=starter\_xpos; t4:=cursor\_ypos+1; {Вершина D}

if t1 < t3 then t5:=-1 else t5:=1;

if t2 < t4 then t6:=-1 else t6:=1;

Translate:=false;

while t3-t5 <> t1 do begin

DrawLine(starter\_xpos, t3, starter\_ypos+1, t4);

t3:=t3+t5;

end;

while t4-t6 <> t2 do begin

DrawLine(starter\_xpos, t1, starter\_ypos+1, t4);

t4:=t4+t6;

end;

cursor\_xpos:=WhereX-18;cursor\_ypos:=WhereY-4;

end;

end;

end;

if figure = 2 then begin {Фигура - ТРЕУГОЛЬНИК}

if not n1\_pressed or not n2\_pressed then begin

if not n1\_pressed then begin

starter\_xpos:=cursor\_xpos; starter\_ypos:=cursor\_ypos;

end else begin

n1\_xpos:=cursor\_xpos; n1\_ypos:=cursor\_ypos+1;

end;

TextColor(0); TextBackground(15);

if not n1\_pressed then write('1') else write('2');

gotoXY(WhereX-1, WhereY);

TextColor(color\_symb); TextBackground(color\_bg);

cursor\_xpos:=WhereX-18; cursor\_ypos:=WhereY-4;

if not n1\_pressed then n1\_pressed := true else n2\_pressed := true

end else begin if (starter\_xpos <> cursor\_xpos) or (starter\_ypos <> cursor\_ypos) then begin

ender\_xpos:=cursor\_xpos; ender\_ypos:=cursor\_ypos+1;

t1:=abs(ender\_xpos-n1\_xpos);

t2:=abs(ender\_ypos-n1\_ypos);

t3:=t1-t2;

if n1\_xpos < ender\_xpos then t6:=1 else t6:=-1;

if n1\_ypos < ender\_ypos then t7:=1 else t7:=-1;

while (n1\_xpos <> ender\_xpos) or (n1\_ypos <> ender\_ypos) do begin

DrawLine(starter\_xpos, n1\_xpos, starter\_ypos+1, n1\_ypos);

t4:=t3\*2;

if t4 > t2 \* (-1) then begin

t3 := t3 - t2;

n1\_xpos := n1\_xpos + t6;

end;

if t4 < t1 then begin

t3 := t3 + t1;

n1\_ypos := n1\_ypos + t7;

end;

end;

DrawLine(starter\_xpos, cursor\_xpos, starter\_ypos+1, cursor\_ypos+1);

cursor\_xpos:=WhereX-18;

cursor\_ypos:=WhereY-4;

end; end; end; end;

5: begin {Отрисовка символа}

TextColor(color\_symb);

TextBackground(color\_bg);

write(chr(used\_char));

gotoXY(WhereX-1,WhereY);

drawed\_color[cursor\_ypos\*61+cursor\_xpos] := color\_symb + (color\_bg shl 4);

drawed\_char[cursor\_ypos\*61+cursor\_xpos] := chr(used\_char);

cursor\_xpos:=WhereX-18;

cursor\_ypos:=WhereY-4;

end;

6: begin {Пипетка}

color\_bg := (drawed\_color[cursor\_ypos\*61+cursor\_xpos] and 240) shr 4;

color\_symb := drawed\_color[cursor\_ypos\*61+cursor\_xpos] and 15;

used\_char := ord(drawed\_char[cursor\_ypos\*61+cursor\_xpos]);

end;

7: begin {Ластик}

TextColor(color\_symb);

TextBackground(color\_bg);

t2:=cursor\_xpos;

t4:=cursor\_ypos;

for t1:=cursor\_ypos to cursor\_ypos + size -1 do begin

for t3:=cursor\_xpos to cursor\_xpos + size -1 do begin

if (t3 < 62) and (t3 > 0) and (t1 < 20) and (t1 >= 0) then begin

(write(' '));

drawed\_color[t1\*61+t3] := color\_symb + (color\_bg shl 4);

drawed\_char[t1\*61+t3] := ' ';

end;

end;

if (t1 < 19) and (t1 >= 0) then

gotoXY(18+cursor\_xpos,WhereY+1)

else

gotoXY(18+cursor\_xpos,WhereY);

end;

gotoXY(18+cursor\_xpos,4+cursor\_ypos);

end;

8: begin {Заливка}

TextColor(color\_symb);

TextBackground(color\_bg);

gotoXY(19,4);

t1:=20 div (opacity+1);

t3:=opacity;

t2:=1;

for i:=0 to 19 do begin

for j:=1 to 61 do begin

case t3 of

0: symb\_paint:=' ';

1: symb\_paint:=#176;

2: symb\_paint:=#177;

3: symb\_paint:=#178;

4: symb\_paint:=#219;

end;

write(symb\_paint);

drawed\_color[i\*61+j] := color\_symb + (color\_bg shl 4);

drawed\_char[i\*61+j] := symb\_paint;

end;

if t2 = t1 then begin

if t3 <> 0 then begin

dec(t3); t2:=1;

end;

end else inc(t2);

gotoXY(19,WhereY+1);

end;

gotoXY(19,4);

cursor\_xpos:=WhereX-18;

cursor\_ypos:=WhereY-4;

end;

end; end;

If UsedScript = 43 then begin

if (Programm\_Flags and (1 shl Flag\_Paused) = (1 shl Flag\_Paused)) then

Programm\_Flags := Programm\_Flags - (1 shl Flag\_Paused);

UsedScript:=7;

gotoXY(18+cursor\_xpos, 4+cursor\_ypos);

key:=#0;

end;end;

#47, #46: begin {кнопка /}

if Programm\_Flags and (1 shl Flag\_Paused) = 0 then begin

TextColor(15);

TextBackground(0);

GotoXY(1, 25);

write('/');

scan\_f(64);

CommandInput:=Input;

key:=#0;

t1:=UsedScript;

Programm\_Flags:=Programm\_Flags + (1 shl Flag\_Paused);

GotoXY(1, 25);

Write(' ');

writeln(flog, '[', Day, '-', Month, '-', Year, ': ', TimeToStr(Time),

'] [C] Command input: ', CommandInput); LogSave;

case CommandInput of

'file\_new': UsedScript:=4;

'file\_save': UsedScript:=6;

'file\_load': UsedScript:=5;

'exit': UsedScript:=54;

'config':UsedScript:=50;

'info':UsedScript:=51;

end;end;

end;

#43, #61: if size < 8 then inc(size); {+}

#45, #95: if size > 1 then dec(size); {-}

#97, #65, #148, #228, #55: begin {A, a, 7}

Select\_Instr(7);

end;

#101, #69, #147, #227, #56: begin {E, e, 8}

Select\_Instr(8);

end;

#70, #102, #128, #160, #54: begin {F, f, 6}

Select\_Instr(6);

end;

#117, #85, #131, #163, #53: begin {U, u, 5}

Select\_Instr(5);

end;

#108, #76, #132, #164, #52: begin {L, l, 4}

Select\_Instr(4);

end;

#83, #115, #155, #235, #51: begin {S, s, 3}

Select\_Instr(3);

end;

#66, #98, #136, #168, #50: begin {B, b, 2}

Select\_Instr(2);

end;

#112, #80, #135, #167, #49: begin {P, p, 1}

Select\_Instr(1);

end;

#171, #139, #75, #107: begin {K, k}

t1:=1;

if UsedScript = 7 then begin

Programm\_Flags := Programm\_Flags + (1 shl Flag\_Chooser);

Programm\_Flags := Programm\_Flags + (1 shl Flag\_Paused);

{t1:=2;}

key:=#0;

UsedScript := 43;

GotoXY(1, 25);

gotoXY(18+cursor\_xpos, 4+cursor\_ypos);

end;

end;

end;

end;

end;

end.