МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Вятский государственный университет»**

Факультет автоматики и вычислительной техники

Кафедра электронных вычислительных машин

РЕАЛИЗАЦИЯ АЛГОРИТМОВ РАСТРОВОЙ ГРАФИКИ ДЛЯ ЗАПОЛНЕНИЯ СПЛОШНЫХ ОБЛАСТЕЙ

Отчет по лабораторной работе №7

по дисциплине «Компьютерная графика»

Выполнил студент группы ИВТ-21 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Седов М.Д./

Проверил преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Клюкин В.Л/

Киров 2018

1 Цель работы

Закрепить лекционный материал по изучению базовых алгоритмов компьютерной графики алгоритмов закраски.

2 Постановка задачи

Написать на языке PASCAL программу, реализующую алгоритм заливки с затравкой многоугольника любой формы.

Проверить правильность работы программы, нарисовав с помощью функций модуля GRAPH невыпуклый многоугольник (с дырами внутри), закрасьте его заданным цветом, указав координаты принадлежащей многоугольнику точки.

3 Алгоритм работы программы

8

Рисунок 1- Алгоритм закраски

4 Экранная форма работы программы

Экранная форма работы программы представлена на рисунке 2.

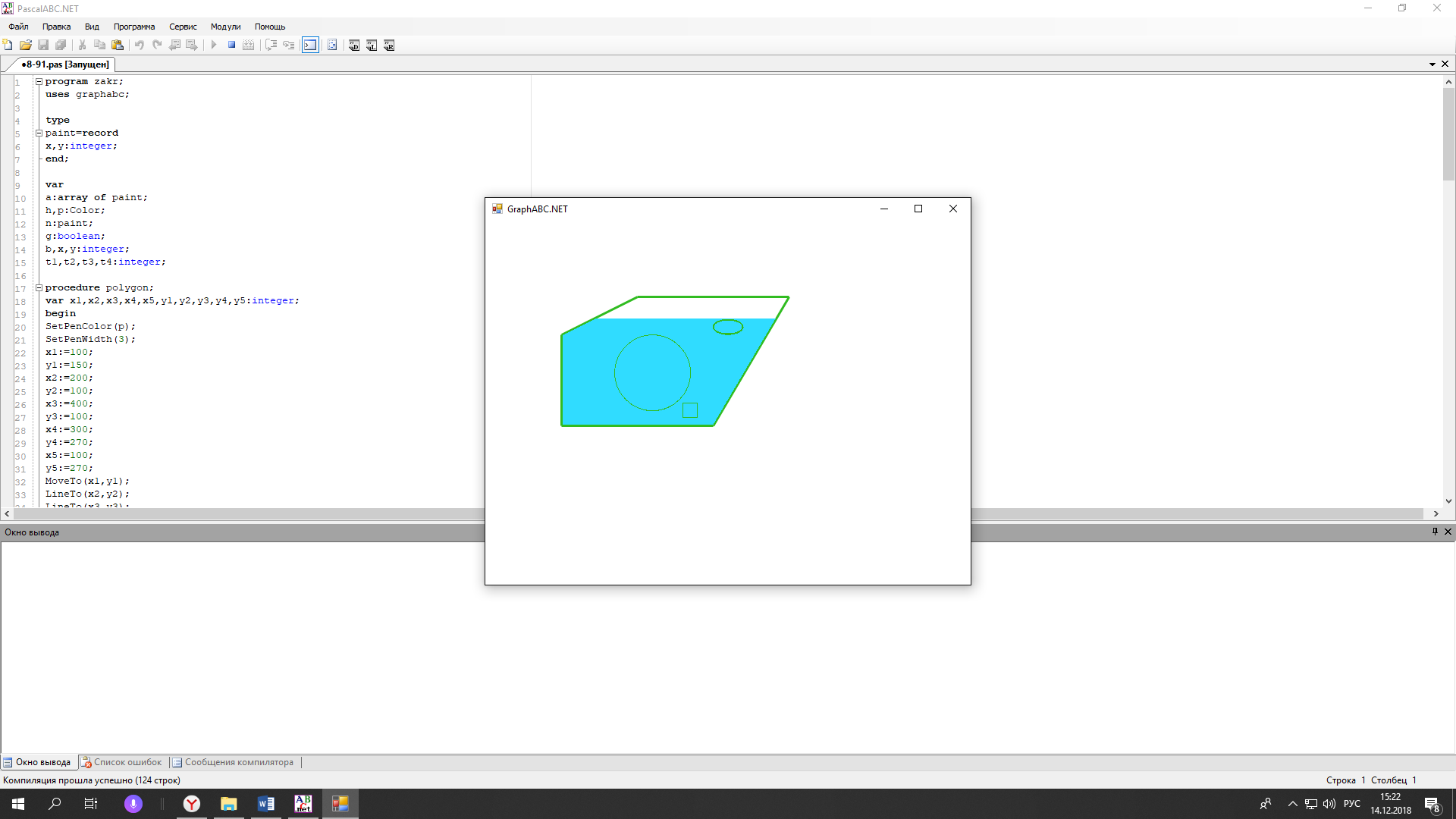


Рисунок 2 - Экранная форма работы программы

5 Код программы

Код программы представлен в приложении А.

6 Вывод

В ходе выполнения лабораторной работы были закреплены навыки по работе с графикой на языке программирования Pascal. На практике изучен один из базовых алгоритмов заполнения замкнутой области – метод затравки. Данный метод прост в реализации, но трудоемок по памяти и по времени, так как требуется использование дополнительного стека для хранения координат пикселей.

Приложение А

(обязательное)

Листинг программы с комментариями

**program** laba89;

**uses**

graphabc;

**type**

Pix = **record**

X, Y: Integer;

**end**;

**var**

Stack: **array of** Pix;

xy: Pix;

ci, cb: Color;

x, y, xw, xr, xl, xb, j: Integer;

fl: Boolean;

**procedure** Push(x, y: Integer);//Помещение в стек

**var**

i: Integer;

**begin**

i := Length(Stack);

SetLength(Stack, i + 1);

Stack[i].X := X; Stack[i].Y := Y;

**end**;

**function** Pop: Pix;//Извлечение из стека

**var**

i: Integer;

**begin**

i := Length(Stack);

Pop := Stack[i - 1];

SetLength(Stack, i - 1);

**end**;

**procedure** risb;//Прорисовка полигона

**var**

x1, x2, x3, x4, y1, y2, y3, y4: Integer;

**begin**

SetPenColor(cb);

SetPenWidth(3);

x1 := 130; y1 := 100;

x2 := 170; y2 := 300;

x3 := 450; y3 := 300;

x4 := 500; y4 := 100;

MoveTo(x1, y1);

LineTo(x2, y2);

LineTo(x3, y3);

LineTo(x4, y4);

LineTo(x1, y1);

circle(190, 130, 20);

SetPenWidth(2);

circle(290, 200, 20);

SetPenWidth(6);

circle(400, 260, 20);

**end**;

**begin**

ci := rgb(130, 45, 95);

cb := rgb(33, 33, 33);

x := 180;

y := 169;

risb;

//1.

Push(X, Y);

//2

**while** Length(Stack) > 0 **do**

**begin**

xy := Pop;

putpixel(xy.x, xy.y, ci);

//3

xw := xy.x;

inc(xy.x);

//4

**while** GetPixel(xy.x, xy.y) <> cb **do**

**begin**

putpixel(xy.x, xy.y, ci);

Inc(xy.x);

**end**;

//5

xr := xy.x - 1; xy.x := xw - 1;

//6

**while** Getpixel(xy.x, xy.y) <> cb **do** //Пока c(x,y) # cb заполнть интервал слева от затравки

**begin**

putpixel(xy.x, xy.y, ci);

Dec(xy.x);

**end**;

//7

xl := xy.x + 1;

//8

j := -1;

**while** j <= 3 **do**

**begin**

xy.x := xl; xy.y := xy.y + j;

**while** xy.x <= xr **do**

**begin**

fl := False;

**while** (Getpixel(xy.x, xy.y) <> cb) **and** (Getpixel(xy.x, xy.y) <> ci) **and** (xy.x < xr) **do**

**begin**

xy.x := xy.x + 1;

**if not** fl **then** fl := True;

**end**;

**if** fl **then**

**begin**

**if** (xy.x = xr) **and** (Getpixel(xy.x, xy.y) <> cb) **and** (Getpixel(xy.x, xy.y) <> ci) **then** Push(xy.x, xy.y) //Если ( x=xr and c(x,y)#cb and c(x,y)#ci ) то Push(x,y)

**else** Push(xy.x - 1, xy.y);

fl := False;

**end**;

xb := xy.x;

**while** (Getpixel(xy.x, xy.y) = cb) **or** (Getpixel(xy.x, xy.y) = ci) **and** (xy.x < xr) **do** xy.x := xy.x + 1; //Пока (c(x,y)=cb or c(x,y)=ci and x<xr)увеличить x на 1

**if** xy.x = xb **then** xy.x := xy.x + 1;

**end**;

j := j + 3;

**end**;

**end**;

**end**.