Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Вятский государственный университет»

Колледж ВятГУ

**ОТЧЕТ**

**ПО ДОМАШНЕЙ КОНТРОЛЬНОЙ РАБОТЕ №5**

**«Исследование алгоритмов сортировки»**

**ПО ДИСЦИПЛИНЕ «ОСНОВЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ»**

Выполнил: студент учебной группы ИСПк- 204-52-00

Кривошеин Дмитрий Васильевич

Преподаватель:

Сергеева Елизавета Григорьевна

г. Киров

2023

1. **Цель работы**: получить базовые сведения о наиболее известных алгоритмах сортировки, изучить принципы работы с текстовыми файлами
2. **Задание**:
3. Реализовать сортировку данных с помощью алгоритма выбором
4. Реализовать сортировку данных с помощью пирамидального алгоритма
5. В обоих случаях необходимо предусмотреть возможность изменения компаратора (реализация компараторов в виде передаваемой в подпрограмму функции)
6. Считывание и ввод данных необходимо производить из текстового файла.
7. Для демонстрации работы программных реализаций самостоятельно подготовить варианты входных данных (при этом объем тестовых файлов должен позволять оценить скорость работы программ)
8. **Описание алгоритма**:

Алгоритм позволяет ввести необходимые для сортировки данные в текстовый файл и произвести сортировку одним из двух предложенных способов

1. **Схема алгоритма**:

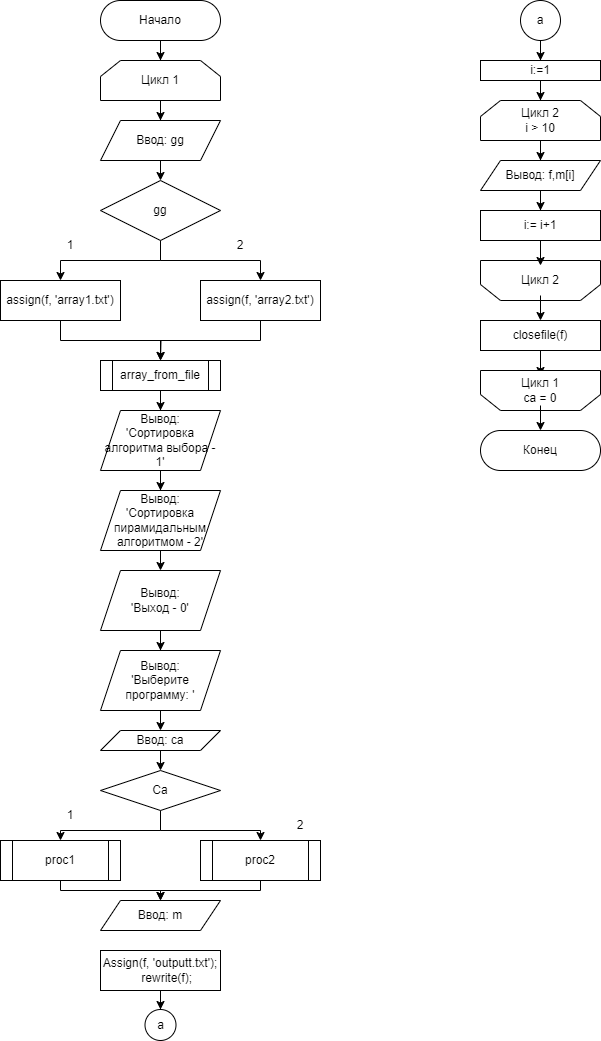


Рисунок 1.1 – Тело программы

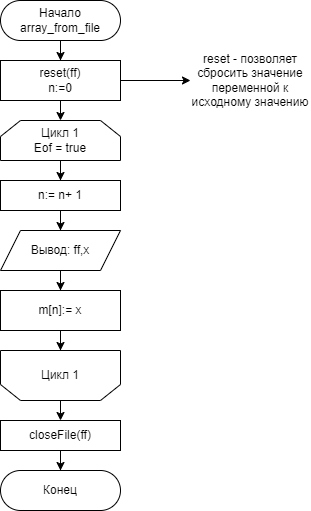


Рисунок 1.2 – процедура array\_from\_file



Рисунок 1.3 – Процедура continuee

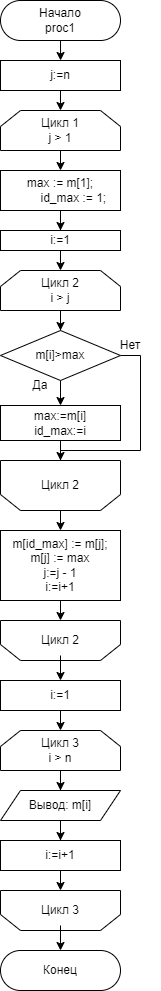


Рисунок 1.4 – Процедура proc1



Рисунок 1.5 – Процедура proc2

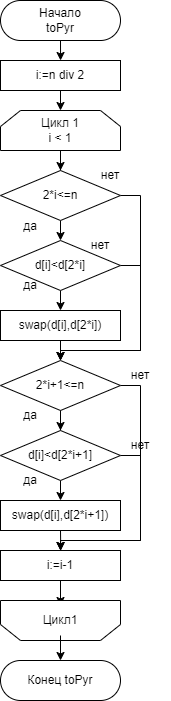


Рисунок 1.6 – процедура toPyr

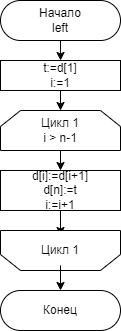


Рисунок 1.7 – процедура Left

1. **Код программы**:

**Uses** Crt, GraphABC;

**var** c,d,h,S1,S,pog: real;

a,b,x:integer;

ca:Byte;

**function** F(x: Real): Real;

**begin**

F := 2\*(x\*x\*x)+((-4)\*x)+13;

**end**;

**procedure** continuee;

**var** ch : char;

**begin**

**repeat**

read(ch);

**until** ch = #13;

**end**;

**function** Left(a, b: Real; n: Integer): Real;

**var**

h, x, sum,S: Real;

i: Integer;

**begin**

h := (b - a) / n;

sum := 0;

**for** i := 0 **to** n - 1 **do**

**begin**

x := a + h \* i;

sum := sum + F(x);

**end**;

Left := h \* sum;

**end**;

**var**

n, y, option, scale: Integer;

ch: char;

**procedure** draw\_plane(scale, a, b: integer);

**var**

i, x, y: integer;

**begin**

x := 30;

y := 770;

SetPenColor(clBlack);

Line(30, 770, (b + 1) \* 20 \* scale + 30, 770); // Ось X

Line(30, 30, 30, 770); // Ось Y

line((b + 1) \* 20 \* scale + 25, y - 5, (b + 1) \* 20 \* scale + 30, y);

line((b + 1) \* 20 \* scale + 25, y + 5, (b + 1) \* 20 \* scale + 30, y);

line(25, 35, 30, 30);

line(35, 35, 30, 30);

TextOut(15, 25, 'Y');

TextOut((b + 1) \* 20 \* scale + 25, 780, 'X');

TextOut(20, 775, '0');

**for** i := 1 **to** b **do**

**begin**

line(x + (20 \* scale \* i), y + 5, x + (20 \* scale \* i), y - 5);

TextOut(x + (20 \* i \* scale - 4), y + 10, i);

**end**;

**for** i := 1 **to** 36 **do**

**begin**

line(25, y - (20 \* i \* scale), 35, y - (20 \* scale \* i));

TextOut(13, y - (20 \* i \* scale) - 7, i);

**end**;

**end**;

**function** graph(a, b, n, x, y, scale: integer): boolean;

**var**

i: integer;

h: Real;

**begin**

h := (b - a) / n;

**for** i := 0 **to** n - 1 **do**

**begin**

SetPenColor(clBlack);

line(round(x + a \* 20 \* scale + h \* i \* 20 \* scale), y - round(F(a + i \* h)) \* 20 \* scale, round(x + a \* 20 \* scale + h \* i \* 20 \* scale + h \* 20 \* scale), y - round(F(a + i \* h + h)) \* 20 \* scale);

**end**;

**end**;

**function** rects(a, b, n, x, y, scale: integer): boolean;

**var**

i: integer;

h: Real;

**begin**

h := (b - a) / n;

**for** i := 0 **to** n - 1 **do**

**begin**

SetBrushColor(clPink);

rectangle(round(x + a \* 20 \* scale + h \* i \* 20 \* scale), y, round(x + a \* 20 \* scale + h \* i \* 20 \* scale + h \* 20 \* scale), y - round(F(a + i \* h + h - h / 2)) \* 20 \* scale);

**end**;

**end**;

**begin**

x := 30;

y := 770;

scale := 1;

SetConsoleIO;

**repeat**

Writeln('Выберите действие:');

Writeln('1. Рассчитать площадь фигуры');

Writeln('2. Выйти из программы');

Readln(option);

ClrScr;

**case** option **of**

1:

**begin**

Writeln('Введите пределы интегрирования a и b через пробел(в промежутке от 0 до 20):');

Readln(a, b);

ClrScr;

Writeln('Введите количество отрезков для деления:');

Readln(n);

ClrScr;

S := Left(a, b, n);

Writeln('Площадь фигуры, ограниченной кривой, равна: ', S:0:3);

Writeln('Вычисленная погрешность: ', Abs(S - Left(a, b, n \* 2)):0:3);

window.Clear;

draw\_plane(scale, a, b);

rects(a, b, n, x, y, scale);

graph(a, b, n, x, y, scale);

**repeat**

ch := ReadKey;

**begin**

ch := ReadKey;

**case** ch **of**

#40:

**begin**

window.Clear;

scale := scale - 1;

draw\_plane(scale, a, b);

rects(a, b, n, x, y, scale);

graph(a, b, n, x, y, scale);

**end**;

#38:

**begin**

window.Clear;

scale := scale + 1;

draw\_plane(scale, a, b);

rects(a, b, n, x, y, scale);

graph(a, b, n, x, y, scale);

**end**;

**end**;

**end**;

**until** ch = #27;

**end**;

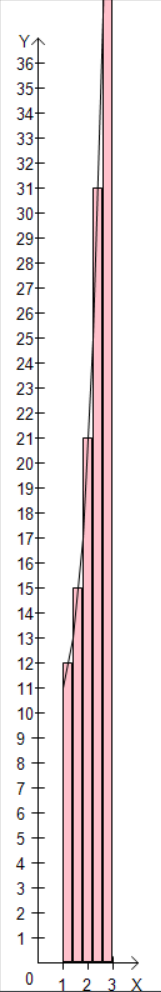
2: Writeln('Пока');

**end**;

**until** option = 2;

**end**.

**6.Результат выполнения программы**:

****

**7.Вывод**:

Эта программа использует метод левых прямоугольников для вычисления площади фигуры, ограниченной кривой, по функции из прошлой домашней контрольной работы. Она также позволяет масштабировать график и отображает результат в виде прямоугольников. Кроме того, она предоставляет возможность пользователю управлять масштабом графика. В процессе выполнения задания возникали трудности, такие как: “Неверное количество передаваемых аргументов в функцию“, “описание переменной цикла for должно производиться в заголовке цикла”, также были ошибки в построении графика функции и неправильные подписи осей координат, реализация масштабизации графика также привела к большому количеству ошибок и неточностей построения, однако все ошибки и неисправности были устранены.