Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Вятский государственный университет»

Колледж ВятГУ

**ОТЧЕТ**

**ПО ДОМАШНЕЙ КОНТРОЛЬНОЙ РАБОТЕ №3**

**«ИЗУЧЕНИЕ БАЗОВЫХ ПРИНЦИПОВ ОРГАНИЗАЦИИ ПРОЦЕДУР И ФУНКЦИЙ»**

**ПО ДИСЦИПЛИНЕ**

**«ОСНОВЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ»**

Выполнила: студентка учебной группы

ИСПк-202-52-00

Широнина Анна Андреевна

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2024

**Цель работы:** освоить синтаксис построения процедур и функций, изучить способы передачи данных в подпрограммы, получить навыки организации минимального пользовательского интерфейса.

1. **Формулировка заданий. Вариант 23**

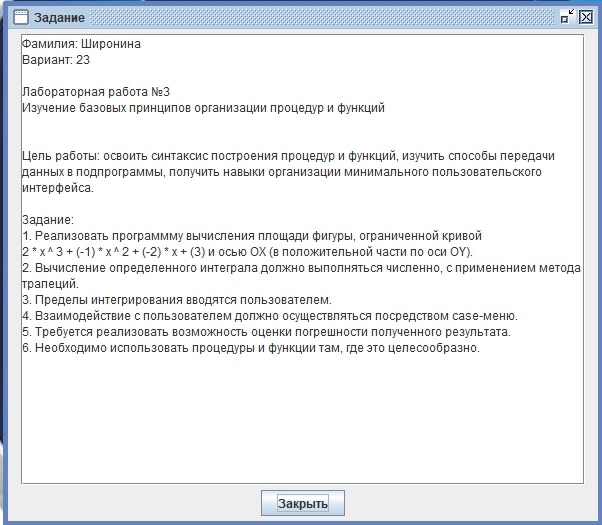
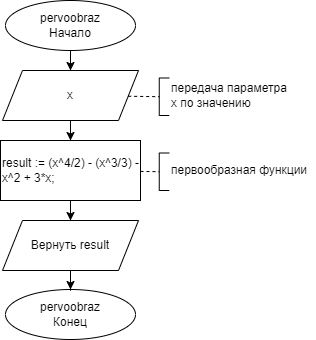
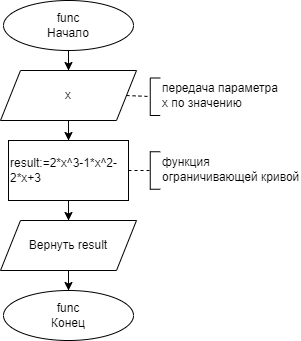


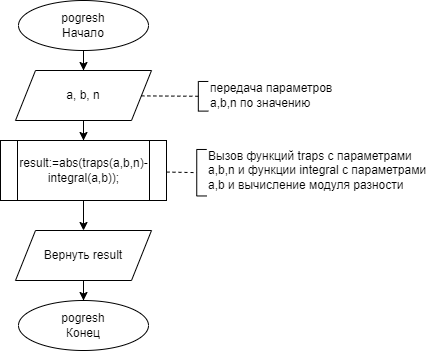
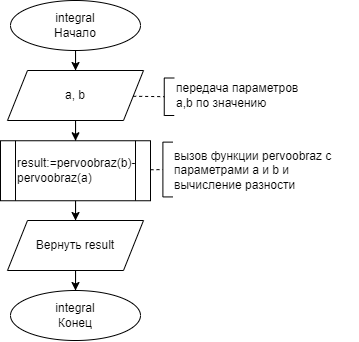
Рисунок 1 – Постановка задачки

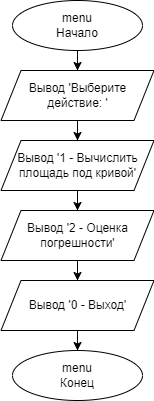
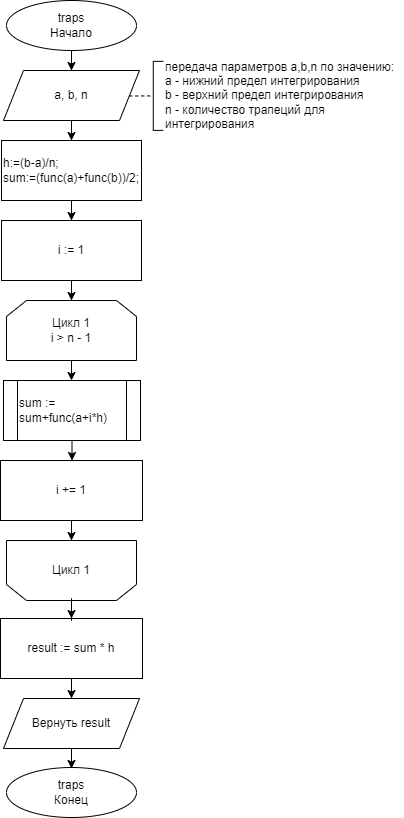
1. **Описание алгоритма**

Для начала мы запрашиваем нижний предел интегрирования (a), верхний предел интегрирования (b) и количество трапеций для интегрирования (n). Для поиска площади фигуры, ограниченной кривой 2\*x^3+(-1)\*x^2+(-2)\*x+(3) и осью OX, мы используем численный метод интегрирования – метод трапеции. Для оценки погрешности мы находим модуль разности получившейся по методу трапеции площади фигуры и значения точного интеграла (разность первообразной от функции f:=2\*x^3+(-1)\*x^2+(-2)\*x+(3) с параметром b и первообразной от функции f:=2\*x^3+(-1)\*x^2+(-2)\*x+(3) с параметром a).

1. **Схема алгоритма**

****

****

****

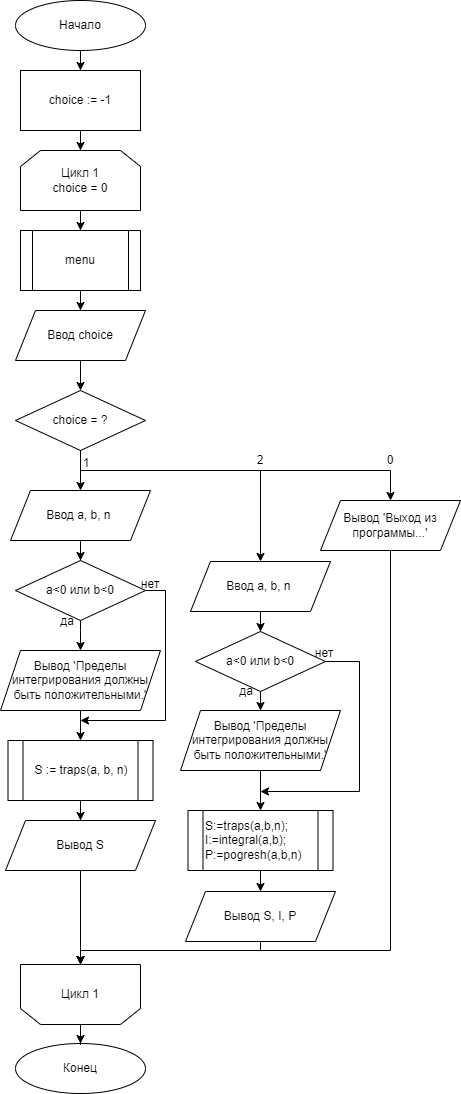
****

Рисунок 2 – Схема алгоритма задачи

1. **Код программы**

**function** func(x: real): real;

**begin**

result := 2 \* power(x, 3) + (-1) \* power(x, 2) + (-2) \* x + (3);

**end**;

**function** traps(a, b: real; n: integer): real;

**var**

h, sum: real;

i: integer;

**begin**

h := (b - a) / n;

sum := (func(a) + func(b)) / 2;

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

sum := sum + func(a + i \* h);

result := sum \* h;

**end**;

**function** pervoobraz(x:real):real;

**begin**

result := (power(x,4) / 2 ) - (power(x,3) / 3) - power(x,2) + 3\*x;

**end**;

**function** integral(a, b: real): real;

**begin**

result := pervoobraz(b) - pervoobraz(a);

**end**;

**function** pogresh(a, b: real; n: integer): real;

**begin**

result := abs(traps(a,b,n) - integral(a,b));

**end**;

**procedure** menu;

**begin**

writeln('Выберите действие:');

writeln('1 - Вычислить площадь под кривой');

writeln('2 - Оценка погрешности');

writeln('0 - Выход');

**end**;

**var**

a, b, S, P, I: real;

choice, n: integer;

**begin**

**repeat**

menu;

readln(choice);

**case** choice **of**

1: **begin**

write('Введите нижний предел интегрирования (a): ');

readln(a);

write('Введите верхний предел интегрирования (b): ');

readln(b);

write('Введите количество трапеций для интегрирования (n): ');

readln(n);

// Проверка на положительность

**if** (a < 0) **or** (b < 0) **then**

**begin**

writeln('Пределы интегрирования должны быть положительными.');

**continue**;

**end**;

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

writeln('Площадь фигуры: ', S:0:4);

**end**;

2: **begin**

write('Введите нижний предел интегрирования (a): ');

readln(a);

write('Введите верхний предел интегрирования (b): ');

readln(b);

write('Введите количество трапеций для интегрирования (n): ');

readln(n);

// Проверка на положительность

**if** (a < 0) **or** (b < 0) **then**

**begin**

writeln('Пределы интегрирования должны быть положительными.');

**continue**;

**end**;

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

I := integral(a, b);

P := pogresh(a, b, n);

writeln('Приближенное значение интеграла(вычисление по методу трапеции): ', S:0:4);

writeln('Точное значение интеграла: ', I:0:4);

writeln('Погрешность: ', P:0:4);

**end**;

0: writeln('Выход из программы...');

**else**

writeln('Некорректный выбор. Попробуйте еще раз.');

**end**;

writeln;

**until** choice = 0;

**end**.

1. **Результат выполнения программы**

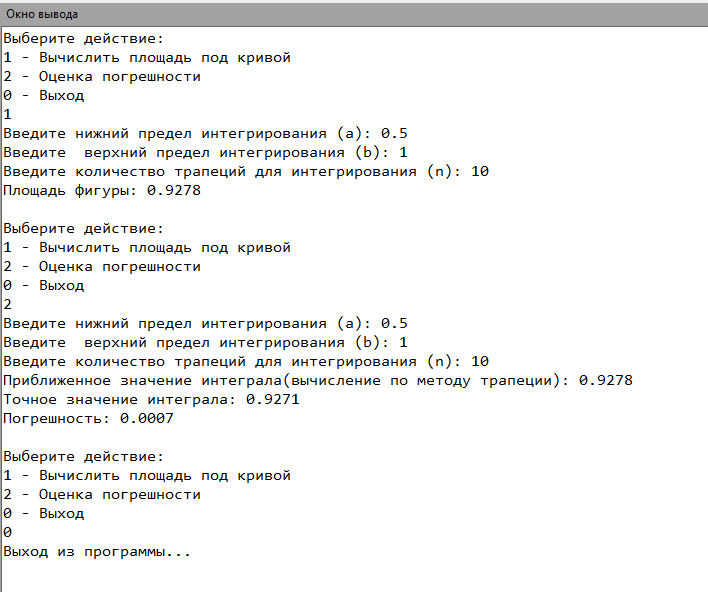
****

Рисунок 3 – Результат выполнения программы

1. **Вывод**

В данной работе мы сосредоточились на изучении процедур и функций в языке программирования Pascal. Процедуры и функции в Паскале представляют собой подпрограммы, которые используются для структурирования кода и повторного использования логики. Процедуры – это блоки кода, которые выполняют определенные действия, но не возвращают значения. Функции –это также блоки кода, но они всегда возвращают значение. Обе формы помогают организовать код, делая его более читабельным и поддерживаемым.

Для работы над схемами алгоритма необходимо четко сформулировать проблему, которую необходимо решить, собрать входные данные, необходимые для алгоритма, описать последовательность шагов, которые необходимо выполнить для решения задачи, использовать стандартные символы для построения схемы алгоритма (овалы – начало и конец, прямоугольники – действия или процессы, ромбы – условия или проверки и т.д.), проверить алгоритм на разных входных данных для выявления возможных ошибок, проанализировать алгоритм и внести изменения для повышения его эффективности, записать и оформить полученные результаты. Работа над схемами алгоритма помогает визуализировать процесс и обеспечить более легкое понимание и оптимизацию решения задачи.

В ходе выполнения работы были достигнуты поставленные цели, направленные на освоение синтаксиса построения процедур и функций. Успешно изучены основные способы передачи данных в подпрограммы, что позволило более глубоко понять их особенности и применения в программировании. Кроме того, была получена практика в организации минимального пользовательского интерфейса, что способствует улучшению взаимодействия пользователя с программным приложением.