Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Вятский государственный университет»

Колледж ВятГУ

**ОТЧЕТ**

**ПО ДОМАШНЕЙ КОНТРОЛЬНОЙ РАБОТЕ №1**

**«ВЫЧИСЛЕНИЕ ЗНАЧЕНИЯ ФУНКЦИИ»**

**ПО ДИСЦИПЛИНЕ**

**«ОСНОВЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ»**

Выполнила: студент учебной группы

ИСПк-202-52-00

Долгополов Ярослав Максимович

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2024

**Цель работы**: изучить базовую структуру организации программы и основным конструкция языка программирования Pascal.

1. **Формулировка задания 1. Вариант 8**

**Постановка задачи**

Написать программу, вычисляющую значение функции:

*(x ^ (0,1 \* x) / cos(x)) \* (tg(x) / x ^ 3)*, если x < -9;

*x ^ (0,1 \* x) + -x*, если -9 <= x < 0;

*x ^ 3 / tg(x) + cos(x)*, если 0 <= x < 2;

*ln(x) \* cos(x)*, если 2 <= x.

1. **Описание алгоритма и ответы на вопросы**

***1. Что такое алгоритм?***

**Алгоритм** – это набор инструкций, описывающих порядок действий исполнителя для достижения результата решения задачи за конечное число действий.

***2. Перечислите виды алгоритмов.***

* Линейный
* Разветвляющийся
* Циклический

***3. Дайте определение условному алгоритму.***

**Условный алгоритм** — это алгоритм, содержащий хотя бы одну проверку условия, в результате которой обеспечивается переход на один из возможных вариантов решения;

***4. Виды условных алгоритмов. Какие использовали для решения задачи?***

Существует две реализации структуры ветвления – полная и неполная:

**Полная структура** ветвления означает, что осуществляется выбор между двумя действиями. Если проверка условия даёт результат «да», то выбирается первое действие, иначе — второе действие.

**Неполная структура** ветвления предполагает, что если условие истинно, то выполняется первая команда, иначе — никакие действия не выполняются.

При выполнении работы была использована полная форма ветвления для обеих задач.

***5. Дайте определение циклическому алгоритму.***

**Циклический** — алгоритм, предусматривающий многократное повторение одного и того же действия (одних и тех же операций) над новыми исходными данными. К циклическим алгоритмам сводится большинство методов вычислений и перебора вариантов.

***6. Виды циклических алгоритмов. Какие использовали для решения задачи?***

**Циклы с предусловием** – цикл, который выполняется, пока истинно некоторое условие, указанное перед его началом. Может ни разу не выполнятся, если условие ни разу не будет удовлетворено.

**Циклы с постусловием** – цикл, в котором условие проверяется после выполнения тела цикла. Цикл выполняется до тех пор, пока условие не будет удовлетворено. В цикле с постусловием команды в теле цикла выполняются как минимум один раз (даже если начальное значение параметра цикла удовлетворяет условию выхода из него).

**Цикл с параметром** – цикл, который позволяет выполнить набор команд фиксированное число раз, то есть число итераций должно быть известно до начала выполнения цикла.

В программе по решению задачи №2 был использован цикл с параметром.

***7. Описание алгоритма***

1. Мы вводим значение переменной **x**.
2. Далее мы переходим к первому условию программы, которое имеет вид x < -9, то используется функция *(x ^ (0,1 \* x) / cos(x)) \* (tg(x) / x ^ 3).* Если введённое число удовлетворяет условию, то мы производим вычисления по функции, в ином же случае мы переходим к следующему условию.
3. Второе условие имеет вид x < 0, то вычисления производятся по функции *x ^ (0,1 \* x) + -x*. Если введённое число удовлетворяет условию, то мы производим вычисления по функции, в ином же случае мы переходим к следующему условию, третьему условию.
4. Третье условие имеет вид x < 2, то мы производим вычисления по формуле функции *x ^ 3 / tg(x) + cos(x)*.
5. Если число не удовлетворило ни одному условию, то мы производим вычисления по формуле *ln(x) \* cos(x).*
6. После прохождения всех условий программы мы выводим значение **x**, которое было введено, и значение функции, которое мы получили в итоге проведенной работы.
7. **Схема алгоритма с комментариями**

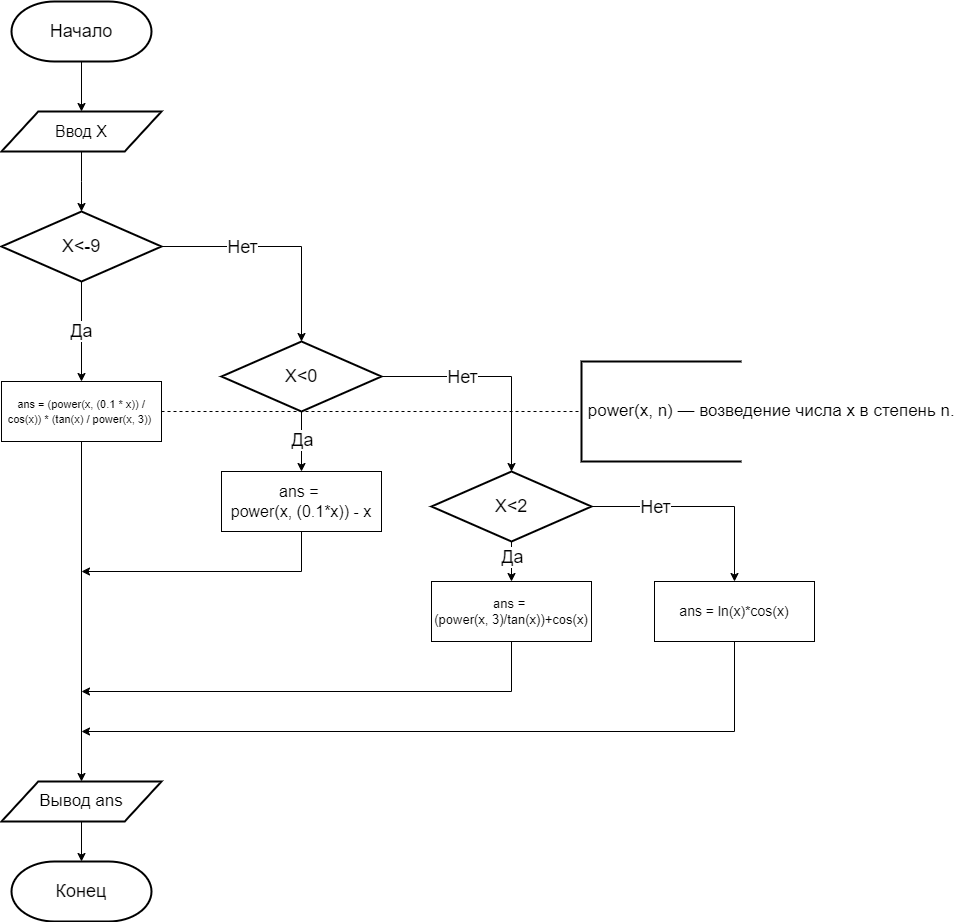
****

Рисунок 1- Алгоритм решения

1. **Код программы**

**var**

x, ans: real;

**begin**

writeln('Введите x:');

readln(x);

**if** (x < (-9)) **then**

ans := (power(x, (0.1 \* x)) / cos(x)) \* (tan(x) / power(x, 3))

**else**

**if** (x < 0) **then**

ans := power(x, (0.1 \* x)) - x

**else**

**if** (x < 2) **then**

ans := (power(x, 3) / tan(x)) + cos(x)

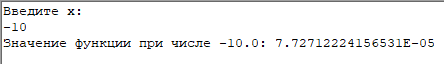
**else**

ans := ln(x) \* cos(x);

writeln('Значение функции при числе ', x:5:1, ': ', ans)

**end**.

1. **Результат выполнения программы**

****

****

****

****

****

****

****

Рисунок 2- Результаты выполнения программы

1. **Формулировка задания 2. Вариант 8**

**Постановка задачи**

Написать программу, вычисляющую значение функции на отрезке [-11; 4] с шагом 0.2:

*(x ^ (0,1 \* x) / cos(x)) \* (tg(x) / x ^ 3)*, если x < -9;

*x ^ (0,1 \* x) + -x*, если -9 <= x < 0;

*x ^ 3 / tg(x) + cos(x)*, если 0 <= x < 2;

*ln(x) \* cos(x)*, если 2 <= x.

1. **Описание алгоритма**
2. Задаём переменной **i** начальное значение = 0.
3. Устанавливаем ограничение для выполнения цикла <=75.
4. Присваиваем x значение (i\*0.2 – 11).
5. Далее мы переходим к первому условию программы, которое имеет вид x < -9, то используется функция *(x ^ (0,1 \* x) / cos(x)) \* (tg(x) / x ^ 3).* Если введённое число удовлетворяет условию, то мы производим вычисления по функции, в ином же случае мы переходим к следующему условию.
6. Второе условие имеет вид x < 0, то вычисления производятся по функции *x ^ (0,1 \* x) + -x*. Если введённое число удовлетворяет условию, то мы производим вычисления по функции, в ином же случае мы переходим к следующему условию, третьему условию.
7. Третье условие имеет вид x < 2, то мы производим вычисления по формуле функции *x ^ 3 / tg(x) + cos(x)*.
8. Если число не удовлетворило ни одному условию, то мы производим вычисления по формуле *ln(x) \* cos(x).*
9. После прохождения всех условий программы мы выводим значение **х**, которое было использовано, и значение функции, которое мы получили в итоге проведенной работы.
10. После выполнения тела цикла мы прибавляем к значению **i** 1.
11. **Схема алгоритма с комментариями**

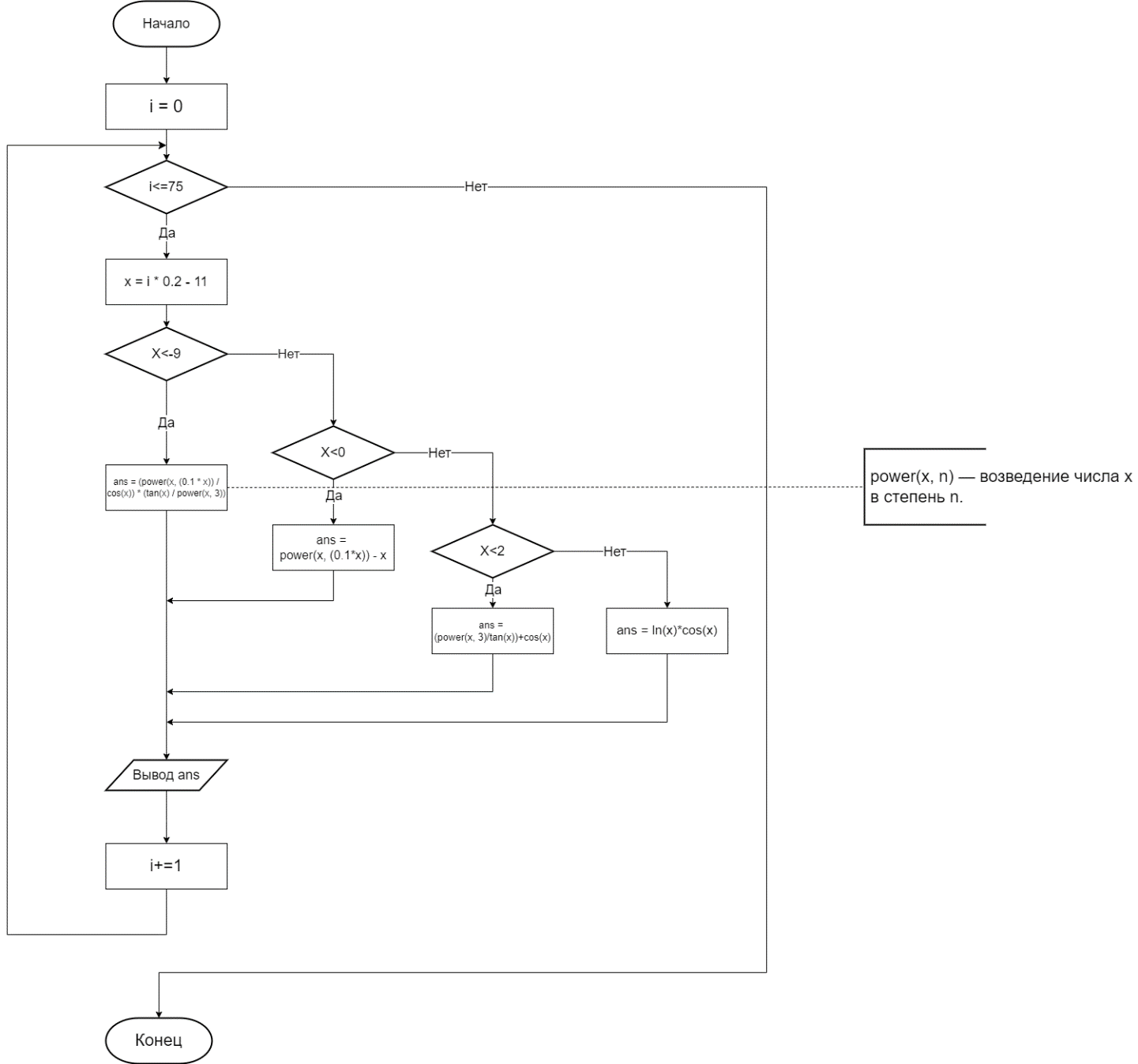


Рисунок 3- Алгоритм решения

1. **Код программы**

**var**

i: integer;

x, ans: real;

**begin**

**for** i:=0 **to** 75 **do**

**begin**

x:=i\*0.2-11;

**if** (x < (-9)) **then**

ans := (power(x, (0.1 \* x)) / cos(x)) \* (tan(x) / power(x, 3))

**else**

**if** (x < 0) **then**

ans := power(x, (0.1 \* x)) - x

**else**

**if** (x < 2) **then**

ans := (power(x, 3) / tan(x)) + cos(x)

**else**

ans := ln(x) \* cos(x);

writeln(x:5:1, ': ', ans)

**end**

**end**.

1. **Результат выполнения программы**

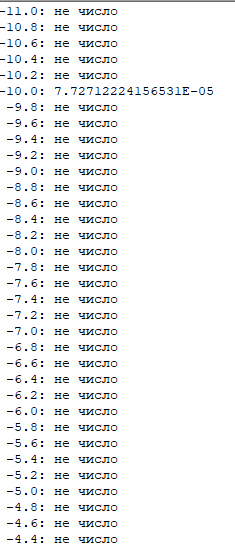
****

Рисунок 4 - Результаты выполнения программы

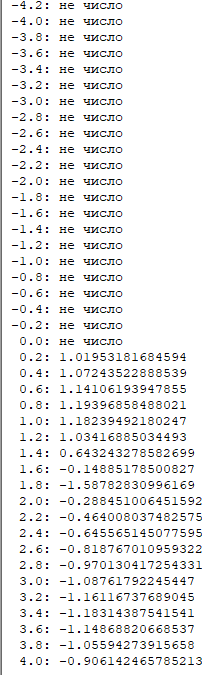


Рисунок 4 - Результаты выполнения программы

1. **Вывод**

В процессе работы над домашней контрольной мы изучили основные принципы организации программы и ключевые конструкции языка программирования Pascal. Мы углубились в понимание понятий алгоритмов, включая условные и циклические алгоритмы, а также рассмотрели различные их виды.

Для создания алгоритмических схем мы освоили правильное построение письменных алгоритмов, а также научились представлять их визуально с помощью сервиса draw.io — бесплатного онлайн-инструмента для создания и совместного редактирования диаграмм и схем.

В ходе работы мы столкнулись с некоторыми трудностями. Например, когда мы запускали программу, вместо ожидаемого результата выводилось значение «Не число», что обозначает неопределённое или несуществующее число. Это происходило в двух случаях: когда по уравнению отрицательное число возводилось в нецелую степень, либо когда происходило деление на ноль. Также происходили ситуации, когда для одних и тех же чисел были разные значения в 1 и во 2 заданиях. Это было исправлено заменой цикла с постусловием на цикл с параметром. Для этого было вычислено количество итераций для выведения всех чисел путём вычитания из конца диапазона начала и разделение разности на размер шага. Затем прибавили 1, чтобы начало диапазона также вычислялось. X присваивалось значение разности произведения параметра на размер шага и конца диапазона.

В итоге, благодаря полученным знаниям и устранённым ошибкам, мы смогли успешно решить задачу, применив код с условными и циклическими конструкциями, а также расширили наши знания о языке программирования Pascal.