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»**

Выполнил: студент учебной группы

ИСПк-204-52-00

Ураков Глеб Олегович

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2024

**Цель работы**:

изучить базовую структуру организации программы и основные конструкции языка программирования Pascal.

**Формулировка задания.**

Вариант: 23  
1. Написать программу, вычисляющую значение функции:  
cos(2 \* x) \* x ^ 2 + lg(x) / x ^ (1 / 3), если x < -8;  
x ^ (1 / 3) - cos(2 \* x), если -8 <= x < 1;  
(ln(x) / -x) \* (lg(x) / x ^ (1 / 3)), если 1 <= x < 3;  
e ^ x / e ^ x - -x / 85, если 3 <= x.  
2. Вычислить значение функции на интервале [-10; 5] с шагом 0,3.

**Описание алгоритма.**

**Алгоритм программы выглядит следующим образом:**

**1. Инициализация переменных:**

\* Объявляются вещественные переменные x, f, i.

\* Переменной x присваивается начальное значение -10.

2. Цикл while:

\* Цикл while повторяется до тех пор, пока значение x не станет больше 5.

\* Внутри цикла выполняются следующие действия:

3. Условные операторы if:

\* Программа проверяет значение x и выполняет соответствующие действия:

\* Если x меньше -8, то вычисляется значение функции f по формуле: f = Cos(2\*x)\*Sqr(x) + (Ln(x)/Ln(10))/Power(x,1/3).

\* Если x больше или равно -8, но меньше 1, то вычисляется значение функции f по формуле: f = Power(x,1/3) - Cos(2\*x).

\* Если x больше или равно 1, но меньше 3, то вычисляется значение функции f по формуле: f = (Ln(x)/-x)\*(Ln(10)/Power(x,1/3)).

\* Если x больше или равно 3, то вычисляется значение функции f по формуле: f = 1 + x/85.

4. Изменение значения x:

\* К x прибавляется значение 0.3.

\* Проверяется, что x - 0.3 больше нуля.

5. Вывод результата:

\* Если x - 0.3 больше нуля, то выводится значение функции f и x в заданном формате.

\* Если x - 0.3 меньше или равно нулю (т.е. x меньше 0.3), то выводится сообщение "Невозможно посчитать логарифм из отрицательного числа".

6. Повтор цикла:

\* Цикл while повторяется с новой итерацией.

1. Алгоритм – это система последовательных операций, для решения какой-либо задачи.

2. Линейные алгоритмы, ветвящиеся алгоритмы, циклические алгоритмы, рекурсивные алгоритмы.

3. Условный алгоритм – это алгоритм, в зависимости от значений выводится различный результат.

4. Условные алгоритмы могут быть с полным выражением if – else

C неполным выражением if

C оператором case

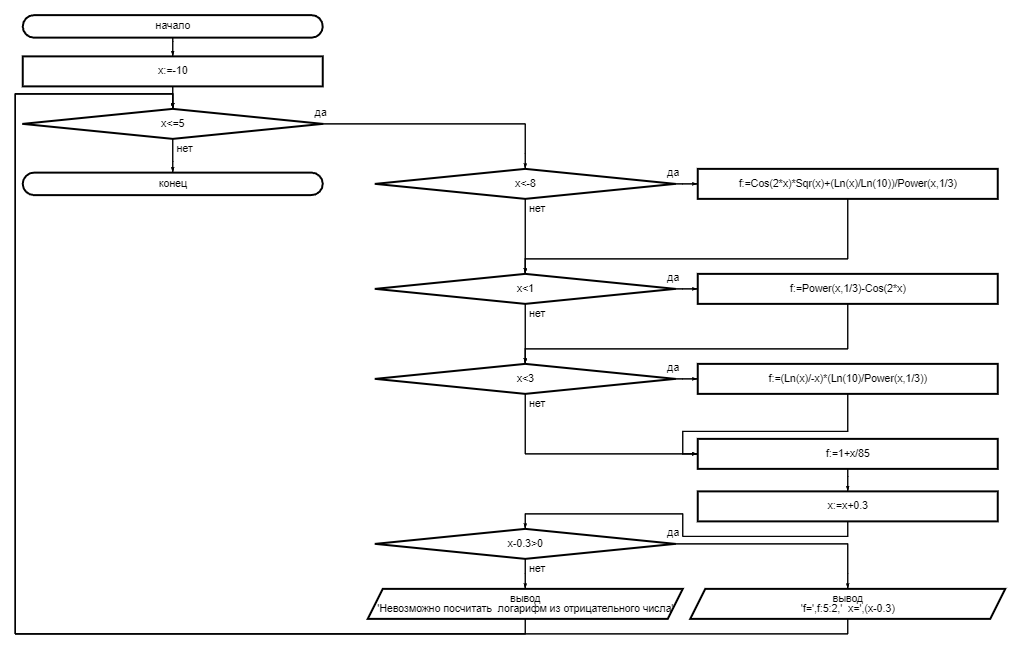
Для решения задачи было использовано: Условный алгоритм if – else

5. Циклический алгоритм – это алгоритм, в котором условие выполняется по несколько раз или больше.

6. Циклический алгоритм с постусловием. Циклический алгоритм с предусловием. Циклический алгоритм со счетчиком. Циклический алгоритм с выходом из середины.

Для решения задачи было использовано: Циклический алгоритм с предусловием и с постусловием.

**Схема алгоритма.**



**Код программы**

**var** x,f,i:real;

**begin**

x:=-10;

**while** (x<=5) **do**

**begin**

**if** x<-8 **then** f:=Cos(2\*x)\*Sqr(x)+(Ln(x)/Ln(10))/Power(x,1/3)

**else if** (x<1) **then** f:=Power(x,1/3)-Cos(2\*x)

**else if** (x<3) **then** f:=(Ln(x)/-x)\*(Ln(10)/Power(x,1/3))

**else** f:=1+x/85;

x:=x+0.3;

**if** x-0.3>0 **then** Writeln('f=',f:5:2,' x=',(x-0.3):3:1)

**else** writeln('Невозможно посчитать логарифм из отрицательного числа');

**end**;

**end**.

**Результат выполнения программы**

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

Невозможно посчитать логарифм из отрицательного числа

f=-0.34 x=0.2

f= 0.25 x=0.5

f= 0.96 x=0.8

f=-0.19 x=1.1

f=-0.49 x=1.4

f=-0.60 x=1.7

f=-0.63 x=2.0

f=-0.63 x=2.3

f=-0.62 x=2.6

f=-0.59 x=2.9

f= 1.04 x=3.2

f= 1.04 x=3.5

f= 1.04 x=3.8

f= 1.05 x=4.1

f= 1.05 x=4.4

f= 1.06 x=4.7

f= 1.06 x=5.0

**Вывод**

Вычисление значения функции – это мощный инструмент, который помогает нам понять и применять математические модели в разнообразных задачах программирования.

В ходе работы мы изучили основы организации программ и разобрались с ключевыми конструкциями языка программирования Pascal. Мы решили задачу с помощью PascalABC, которая включала циклический алгоритм с постусловием. Также, мы построили блок-схему для решения этой задачи.

Теперь мы можем самостоятельно описывать алгоритмы, проводить их проверку и получать результаты. Кроме того, мы закрепили знания об условных и циклических алгоритмах, их видах и принципах работы. Мы освоили новые алгоритмы и методы взаимодействия с ними, а также повторили операции возведения в степень и использование цикла while.