Федеральное государственное автономное

образовательное учреждение

высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт Космических и информационных технологий

институт

Кафедра «Информатика»

кафедра

**ОТЧЕТ О ПРАКТИЧЕСКОЙ РАБОТЕ №7**

Исчисления и абстрактная интерпретация

тема

Преподаватель \_\_\_\_\_\_\_\_\_ А. С. Кузнецов

подпись, дата инициалы, фамилия

Студент КИ18-17/1б 031830504 \_\_\_\_\_\_\_\_\_ Е.В. Железкин

номер группы, зачетной книжки подпись, дата инициалы, фамилия

Красноярск 2021

# Цель работы

Исследование проблем вычислимости без использования абстрактной машины Тьюринга.

# Задача работы

В части 1 необходимо произвести программную реализацию вычислителя

заданной математической функции для заданных аргументов, причем исключительно средствами примитивной и частичной рекурсии, или формально доказать невозможность этого. Привести примеры выполнения вычислений. В части 2 необходимо, используя метод абстрактной интерпретации, для произвольной программной процедуры определить знаки всех переменных.

*Вариант (15)*

## Инструкция по запуску

Необходимо установить *python*, желательно версии 3 и выше (выполнено на версии 3.9.4):

* Страница загрузки для Windows: <https://www.python.org/downloads/>
* Для Linux есть несколько способов, один из них инструмент apt-get:

*$ sudo apt-get update*

*$ sudo apt-get install python3.8*

* Или загрузить, распаковать и установить образ:

*$ wget* [*https://www.python.org/ftp/python/3.8.2/Python-3.8.2.tgz*](https://www.python.org/ftp/python/3.8.2/Python-3.8.2.tgz)

*$ tar -xvf Python-3.8.2.tgz*

Для следующего шага понадобится компилятор gcc, но, думаю, это не проблема. Переходим в распакованную папку и собираем+устанавливаем:

*$ cd Python-3.8.2*

*$ ./configure*

*$ make*

*$ sudo make install*

Далее на любой из двух систем выполнить:

*$ python var15.py <argv>*

# Ход работы

*Часть 1*

Произведена программная реализация вычислителя математической функции :

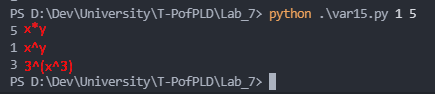


Рисунок 1 – Результаты работы программы с параметрами «1 5»

При вызове моей функции для аргумента |x| > 1 происходит переполнение рекурсивного стека(Рисунок 2). При увеличении глубины функция отрабатывает некорректно - ничего не выводится (Рисунок 3):

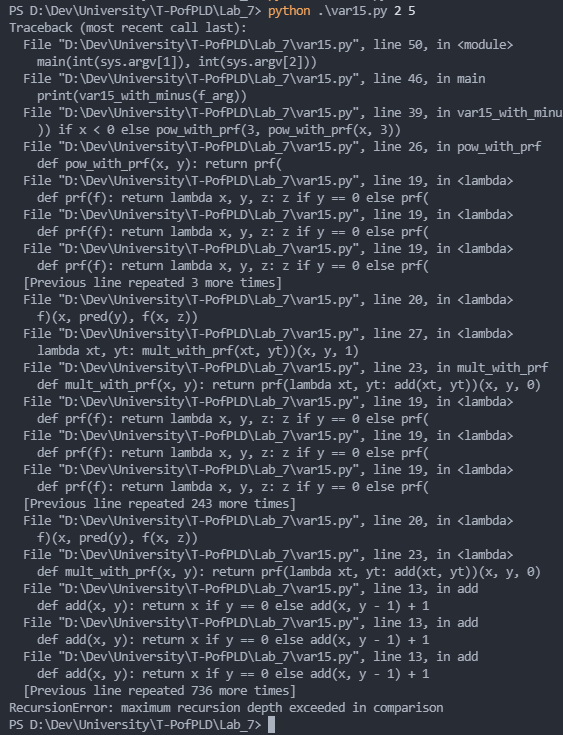


Рисунок 2 – Результаты работы программы с параметрами «2 5»

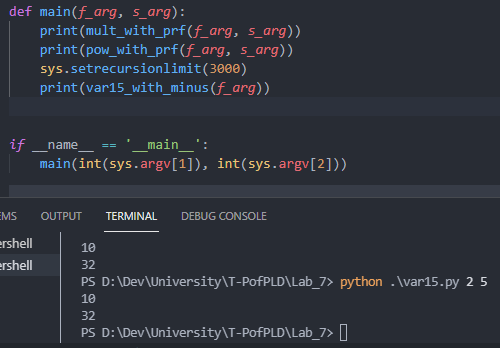


Рисунок 3 – Результаты работы программы с параметрами «2 5»

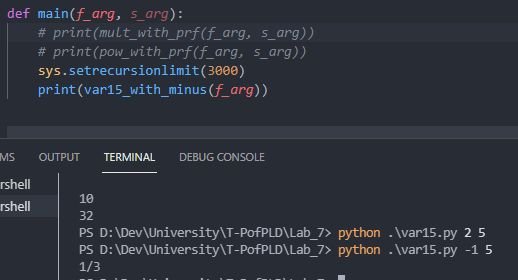


Рисунок 4 – Результаты работы программы с параметрами «-1 5»

*Часть 2*

Имеем домен S = {*minus, zero, plus, unknown, NaN*}, где *minus* – отрицательные числа; *zero* – нуль, *plus* – положительные, *unknown* – неизвестно, *infinity –* числовой эквивалент бесконечности.

Если , то

R = A / B:

* *zero*, если A = *zero* и B = *zero*;
* *plus*, если A = *plus* и B = *plus* или A = *minus* и B = *minus*;
* *minus*, если A = *minus* и B = *plus* или A = *plus* и B = *minus*;
* *unknown*, если A = *unknown* и B = *unknown*;
* *infinity*, если B = zero;

Произвольная программная процедура:

a = 3 (*plus*)

b = -1 (*minus*)

c = a / b (*minus*)

d = 0 / 1 (*zero*)

e = 0 / 0 (*infinity*)

f = 0 + z (*unknown*)

# Вывод

В ходе данной лабораторной работы были исследованы проблемы вычислимости без использования абстрактной машины Тьюринга.

**ПРИЛОЖЕНИЕ А**

Листинг 1 – файл var15.py

import sys

def z(x): return 0

def pred(x): return 0 if x == 0 else x - 1

def subtr(x, y): return x if y == 0 else pred(subtr(x, y - 1))

def add(x, y): return x if y == 0 else add(x, y - 1) + 1

def mult(x, y): return add(x, mult(x, y - 1))

def prf(f): return lambda x, y, z: z if y == 0 else prf(

    f)(x, pred(y), f(x, z))

def mult\_with\_prf(x, y): return prf(lambda xt, yt: add(xt, yt))(x, y, 0)

def pow\_with\_prf(x, y): return prf(

    lambda xt, yt: mult\_with\_prf(xt, yt))(x, y, 1)

# RecursionError: maximum recursion depth exceeded in comparison if |x| > 1

# but it should work right

# for all power functions it is indicated that the exponent is greater than zero, I will assume that for mine too

# prf - primitive recursive function

def var15(x): return pow\_with\_prf(3, pow\_with\_prf(x, 3))

# crutch

def var15\_with\_minus(x): return '1/' + str(pow\_with\_prf(3, pow\_with\_prf(-x, 3)

                                                        )) if x < 0 else pow\_with\_prf(3, pow\_with\_prf(x, 3))

def main(f\_arg, s\_arg):

    # print(mult\_with\_prf(f\_arg, s\_arg))

    # print(pow\_with\_prf(f\_arg, s\_arg))

    print(var15\_with\_minus(f\_arg))

if \_\_name\_\_ == '\_\_main\_\_':

    main(int(sys.argv[1]), int(sys.argv[2]))