
КАФЕДРА

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ
РУКОВОДИТЕЛЬ

должность, уч. степень, звание

подпись, дата

инициалы, фамилия

Отчет о лабораторной работе №1

Элементы теории рекурсивных функций

По дисциплине: Теория вычислительных процессов

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. №

подпись, дата

инициалы, фамилия

Санкт-Петербург 2024

Цель работы:

Целью данной работы является преобразование заданной арифметической функции в рекурсивную с использованием примитивной рекурсии, а также реализация программы, вычисляющей эту функцию как итеративным, так и рекурсивным способами.

Основные сведения из теории:

Рекурсия (от лат. *recurso* – бегу назад, возвращаюсь) – есть такой способ задания вычислимой функции, когда каждое ее значение задается через значение этой (или другой) вычислимой функции для меньших (или ранее определенных) значений аргументов, а функции задаваемые таким образом называются рекурсивными. Наиболее ярким примером является ряд Фибоначчи.

Постановка задачи:

Необходимо преобразовать заданную арифметическую функцию в рекурсивную, используя операторы примитивной рекурсии.

Во второй части лабораторной работы требуется создать программу на языке высокого уровня содержащую две функции. Первая функция должна вычислять заданную арифметическую функцию итеративным способом. Вторая функция должна вычислять заданную арифметическую функцию рекурсивным способом.

Аргументы арифметической функции должны задаваться пользователем в процессе выполнения программы.

Выполнение задачи:

Вариант:

38.

$x_1 \times x_2 - y$

Для преобразования функции $x_1 * x_2 - y$ в рекурсивную, используя примитивную рекурсию, можно рассмотреть следующие шаги:

1. Произведение:

Выражение $x_1 * x_2$ можно преобразовать в рекурсивную форму, где произведение двух чисел сводится к последовательному сложению. Примитивная рекурсия для умножения:

$$P(x_1, x_2) = \begin{cases} 0, & \text{если } x_2 = 0 \\ \{x_1 + P(x_1, x_2 - 1)\}, & \text{если } x_2 > 0 \\ \{-x + P(x_1, x_2 + 1)\}, & \text{если } x_2 < 0 \end{cases}$$

То есть, для положительного x_2 , мы складываем x_1 рекурсивно x_2 раз. Для отрицательного x_2 , результат складывается с обратным знаком.

2. Окончательная функция:

После получения произведения $P(x_1, x_2)$, вычитаем y , что не требует рекурсии:

$$F(x_1, x_2, y) = P(x_1, x_2) - y$$

Таким образом, рекурсия используется для вычисления произведения, а результат просто корректируется вычитанием y .

Разложение функции на простые шаги

Функция $f(x_1, x_2, y) = x_1 * x_2 - y$ состоит из двух частей: умножения и вычитания. Мы можем разложить её на две отдельные функции:

$g(x_1, x_2) = x_1 * x_2$ — функция умножения.

$h(z, y) = z - y$ — функция вычитания.

Теперь докажем, что каждая из этих функций рекурсивна.

Функция умножения является примитивно-рекурсивной

Здесь, если $z > 0$, то на каждом шаге мы уменьшаем z на единицу, пока не достигнем значения y , либо пока z не станет равным нулю.

Функция умножения $g(x_1, x_2) = x_1 * x_2$ уже была доказана как примитивно-рекурсивная:

$$g(x_1, 0) = 0$$

$$g(x_1, x_2 + 1) = g(x_1, x_2) + x_1$$

Это определение через примитивную рекурсию, как мы показывали ранее.

Функция вычитания

Функция вычитания $h(z, y) = z - y$ также может быть определена с помощью примитивной рекурсии. Однако, нужно учитывать, что вычитание, в отличие от сложения, ограничено нулём: если $z < y$, то результат должен быть равен нулю.

Функцию вычитания можно определить с помощью другой рекурсивной функции — функции "обратного преемника", которая вычисляет разность двух чисел, но при этом не опускается ниже нуля:

$$h(z, 0) = z$$

$$h(z, y + 1) =$$

0, если $z = 0$ - функция возвращает 0.

$h(z - 1, y)$, если $z > 0$ функция рекурсивно вызывает сама себя с уменьшенным z на единицу и тем же y . То есть $h(z - 1, y)$

Здесь, если $z > 0$, то на каждом шаге мы уменьшаем z на единицу, пока не достигнем значения y , либо пока z не станет равным нулю.

Комбинация умножения и вычитания

Теперь, когда у нас есть примитивно-рекурсивное определение для умножения и вычитания, можем выразить полную функцию $f(x_1, x_2, y) = x_1 * x_2 - y$ через эти функции:

$$f(x_1, x_2, y) = h(g(x_1, x_2), y)$$

Где:

$g(x_1, x_2) = x_1 * x_2$ — примитивно-рекурсивная функция умножения,

$h(z, y) = z - y$ — рекурсивная функция вычитания.

Вывод

Таким образом, функция $f(x_1, x_2, y) = x_1 * x_2 - y$ является **рекурсивной**. Она может быть выражена через примитивно-рекурсивные функции умножения и вычитания, что доказывает её принадлежность к классу рекурсивных функций.

Функция вычитания

1. Итеративная версия (iterative_function):

- Вычисляет произведение $x1 * x2$ путём сложения $x1$ с самим собой $x2$ раз.
- Если $x2$ отрицательное, результат умножения инвертируется.
- После вычисления произведения из него вычитается y .

2. Рекурсивная версия (recursive_multiply):

- Вычисляет произведение $x1 * x2$ с помощью рекурсии.
- Для положительных $x2$ прибавляет $x1$ рекурсивно, пока $x2$ не станет 0.
- Для отрицательных $x2$ рекурсивно вычитает $x1$.
- Результат также корректируется на y .

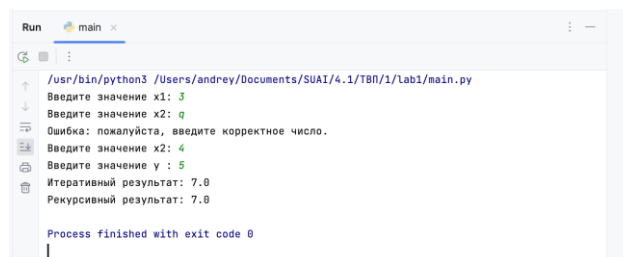
3. Функция для безопасного ввода (input_float):

- Проверяет, что пользователь вводит корректные числа. Если ввод некорректный (например, буквы), запрашивает ввод заново.

4. Главная программа (main):

- Запрашивает у пользователя значения $x1$, $x2$, и y .
- Вызывает обе версии функции для вычисления результата.
- Выводит результаты на экран.

Пример результата выполнения:



```
Run main x
/usr/bin/python3 /Users/andrey/Documents/SUA1/4.1/T8N/1/Lab1/main.py
Введите значение x1: 3
Введите значение x2: 4
Введите значение y : 5
Итеративный результат: 7.0
Рекурсивный результат: 7.0
Process finished with exit code 0
```

Рисунок 1 – результат выполнения программы



Рисунок 2 – проверка на калькуляторе

Вывод:

В ходе выполнения лабораторной работы успешно реализованы итеративная и рекурсивная версии функции, вычисляющей выражение $x1 * x2 - y$. Работа над проектом позволила углубить понимание принципов примитивной рекурсии и улучшить навыки программирования, включая обработку пользовательского ввода и рекурсивные алгоритмы.

Листинг программы:

```
# Итеративная версия функции  $x1 * x2 - y$ 
def iterative_function(x1, x2, y):
    product = 0
    abs_x2 = abs(int(x2)) # Преобразуем в целое и работаем с модулем
    for _ in range(abs_x2):
```

```

    product += x1
if x2 < 0:
    product = -product # Корректируем знак при отрицательном x2
return product - y

# Рекурсивная версия функции x1 * x2 - y
def recursive_multiply(x1, x2):
    if x2 == 0:
        return 0
    elif x2 > 0:
        return x1 + recursive_multiply(x1, x2 - 1)
    else: # Обрабатываем случай отрицательного x2
        return -x1 + recursive_multiply(x1, x2 + 1)

def recursive_function(x1, x2, y):
    return recursive_multiply(x1, int(x2)) - y

# Функции для безопасного ввода значений
def input_float(prompt):
    while True:
        try:
            return float(input(prompt))
        except ValueError:
            print("Ошибка: пожалуйста, введите корректное число.")

# Главная программа
def main():
    # Ввод каждого аргумента отдельно с проверкой
    x1 = input_float("Введите значение x1: ")
    x2 = input_float("Введите значение x2: ")
    y = input_float("Введите значение y: ")

    # Вычисление итеративным способом
    result_iterative = iterative_function(x1, x2, y)
    print(f"Итеративный результат: {result_iterative}")

    # Вычисление рекурсивным способом
    result_recursive = recursive_function(x1, x2, y)
    print(f"Рекурсивный результат: {result_recursive}")

# Запуск программы
main()

```