МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ» ИНСТИТУТ ЦИФРОВОГО РАЗВИТИЯ

Отчет о лабораторной работе №12

по дисциплине основы программной инженерии

Выполнил: Духно Михаил Александрович,

2 курс, группа ПИЖ-б-о-20-1,

Проверил: Доцент кафедры инфокоммуникаций, Воронкин Р.А.

Ставрополь, 2021 г

```python
n = 0
for i in range(1, n+1):
    n += i


def recursion(n):
    if n == 1:
        return 1
    return n + recursion(n - 1)
```

Рисунок 12.1 – Сущность рекурсии

```python
def factorial(n):
    if n == 0:
        return 1
    else:
        return n * factorial(n - 1)


def factorial(n):
    if n == 0:
        return 1
    elif n == 1:
        return 1
    else:
        return n * factorial(n - 1)


def fib(n):
    if n == 0 or n == 1:
        return n
    else:
        return fib(n - 2) + fib(n - 1)
```

Рисунок 12.2 – Как и когда происходит рекурсия (1)

```python
def factorial(n):
    product = 1
    while n > 1:
        product *= n
        n -= 1
    return product

def fib(n):
    a, b = 0, 1
    while n > 0:
        a, b =b, a + b
        n -= 1
    return a
```

Рисунок 12.3 – Как и когда происходит рекурсия (2)

```python
from functools import lru_cache

#.....

@lru_cache
def fib(n):
    if n == 0 or n == 1:
        return n
    else:
        return fib(n - 2) + fib(n - 1)


def fib(n):
    if n <= 1:
        return (n, 0)
    else:
        (a, b) = fib(n - 1)
        return (a + b, a)
```

Рисунок 12.4 – Как и когда происходит рекурсия (3)

```python
def cursing(depth):
    try:
        cursing(depth + 1)
    except RuntimeError as RE:
        print('I recursed {} times!'.format(depth))


if __name__ == '__main__':
    cursing(0)
```

Рисунок 12.5 – Увеличение максимальной глубины рекурсии

```python
def countdown(n):
    if n == 0:
        print("Blastoff")
    else:
        print(n)
        countdown(n - 1)


def find_max(seq, max_so_far):
    if not seq:
        return max_so_far
    if max_so_far < seq[0]:
        return find_max(seq[1:], seq[0])
    else:
        return find_max(seq[1:], max_so_far)
```

Рисунок 12.6 – Хвостовая рекурсия

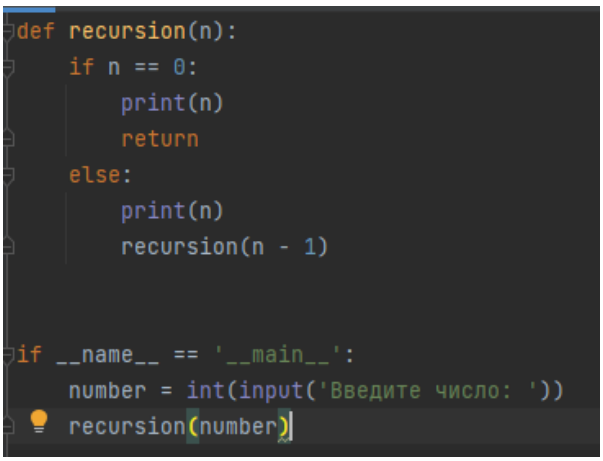

Рисунок 12.7 – Код программы индивидуального задания

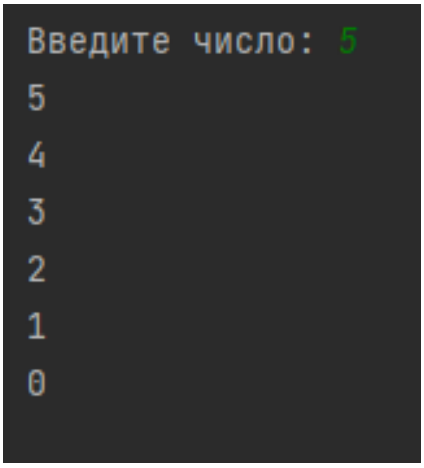

Рисунок 12.8 – Результат работы программы

Контрольные вопросы:

1. Для чего нужна рекурсия?

В некоторых случаях лучше использовать рекурсию (например, путешествие по дереву), в таких случаях более естественно использовать "think recursively". Однако, если использование циклов не сложнее и намного сложнее, чем рекурсия, я предпочитаю их.

2. Что называется базой рекурсии?

База рекурсии – это такие аргументы функции, которые делают задачу настолько простой, что решение не требует дальнейших вложенных вызовов. Рекурсивно определяемая структура данных – это структура данных, которая может быть определена с использованием самой себя

3. Самостоятельно изучите что является стеком программы. Как используется стек программы при вызове функций?

Стек вызовов — в теории вычислительных систем, LIFO-стек, хранящий информацию для возврата управления из подпрограмм (процедур, функций) в программу (или подпрограмму,

при вложенных или рекурсивных вызовах) и/или для возврата в программу
из обработчика прерывания

4. Как получить текущее значение максимальной глубины рекурсии в языке Python?

Вывести на печать значение изменяемого параметра функции в условии, которое определяет конец рекурсии

5. Что произойдет если число рекурсивных вызовов превысит максимальную глубину рекурсии

в языке Python?

Произойдёт переполнение памяти и функция упадёт в ошибку

6. Как изменить максимальную глубину рекурсии в языке Python?

Sys.setrecurionlimit(limit)

7. Каково назначение декоратора lru_cache ?

Декоратор @lru_cache() модуля functools оборачивает функцию с переданными в нее аргументами и запоминает возвращаемый результат соответствующий этим аргументам.

8. Что такое хвостовая рекурсия? Как проводится оптимизация хвостовых вызовов?

Хвостовая рекурсия — частный случай рекурсии, при котором любой рекурсивный вызов является последней операцией перед возвратом из функции

Оптимизация хвостовой рекурсии путём преобразования её в плоскую итерацию реализована во многих оптимизирующих компиляторах.