Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития

Кафедра инфокоммуникаций

**ОТЧЕТ**

**ПО ЛАБОРАТОРНОЙ РАБОТЕ №2.9**

**дисциплины «Программирование на Python»**

**Вариант\_\_\_**

|  |  |  |  |
| --- | --- | --- | --- |
|  | | Выполнил:  Иващенко Олег Андреевич  2 курс, группа ИВТ-б-о-22-1,  09.03.02 «Информационные и вычислительные машины», направленность (профиль) «Программное обеспечение средств вычислительной техники и автоматизированных систем»  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) | |
|  | | Руководитель практики:  Воронкин Роман Александрович, доцент кафедры инфокоммуникаций  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) | |
|  | |  | |

Отчет защищен с оценкой \_\_\_\_\_\_\_\_\_\_\_ Дата защиты\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ставрополь, 2023 г.

**Тема**: «Рекурсия в языке Python»

**Цель**: Приобретение навыков по работе с рекурсивными функциями при написании программ с помощью языка программирования Python версии 3.x.

Порядок выполнения работы

Таблица 1 – Код программы example.py

|  |
| --- |
| #!/usr/bin/env python3  # -\*- coding: utf-8 -\*-  def recursion(n):  if n == 1:  return 1    return n + recursion(n - 1)  if \_\_name\_\_ == "\_\_main\_\_":  print(recursion(int(input("Введите числовое значение: ")))) |



Рисунок 1 – Результат работы программы example.py

Индивидуальное задание. Написать программу вычисления функции Аккермана для всех неотрицательных целых аргументов m и n.

Таблица 2 – Код программы individual.py

|  |
| --- |
| #!/usr/bin/env python3  # -\*- coding: utf-8 -\*-  def accerman(m, n):  """  Функция принимает значения n и m, нужные для  рассчёта значения функции Аккермана с помощью  рекурсивных вызовов данного метода  ПРЕДУПРЕЖДЕНИЕ: вводить значения больше 3 не рекомендуется  """    if m > 0 and n > 0:  return accerman(m - 1, accerman(m, n - 1))  elif m > 0 and n == 0:  return accerman(m - 1, 1)  elif m == 0:  return n + 1  if \_\_name\_\_ == "\_\_main\_\_":  n = input("Введите целочисленное число n: ")  m = input("Введите целочисленное число m: ")  result = 0  if (n.isdigit() and m.isdigit()):  n = int(n)  m = int(m)  result = accerman(m, n)  print(f"Для n = {n} и m = {m} "  f"значение функции Аккермана = {result}")  else:  print("Ошибка: одно из значений не является"  "целочисленным и/или положительным числом") |



Рисунок 2 – Результат работы программы individual.py

Контрольные вопросы

1. Для чего нужна рекурсия?

Рекурсия – это метод в программировании, при котором функция вызывает саму себя. Рекурсия используется для решения задач, которые могут быть разбиты на подзадачи того же типа, она может сделать код более понятным, лаконичным и легко поддерживаемым.

1. Что называется базой рекурсии?

База рекурсии – это условие, при котором рекурсивная функция прекращает вызывать саму себя. Она предотвращает бесконечное выполнение и определяет конечный случай задачи.

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

Стек программы – это структура данных, которая хранит информацию о вызовах функций в программе. Каждый раз, когда функция вызывается, информация о состоянии вызова помещается в стек. Когда функция завершается, эта информация удаляется из стека, и управление передаётся обратно вызывающей функции.

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

Текущее значение максимальной глубины рекурсии в Python можно получить с помощью функции «sys.getrecursionlimit()» из модуля «sys». Пример использования:

import sys

recursion\_limit = sys.getrecursionlimit()

1. Что произойдёт, если число рекурсивных вызовов превысит максимальную глубину рекурсии в языке Python?

В случае превышения максимального количества вызова рекурсии произойдёт исключение «RecursionError», указывающее на превышение максимальной глубины рекурсии.

1. Как изменить максимальную глубину рекурсии в языке Python?

Максимальную глубину рекурсии можно изменить с помощью функции «sys.setrecursionlimit(значение)». Но изменение этого значения может повлиять на стабильность работы программы, поэтому следует быть осторожным и изменять его только в случае крайней необходимости.

1. Каково назначение декоратора lre\_cache?

Декоратор «lru\_cache» используется для кеширования результатов вызова функции с определённым набором аргументов. Это позволяет избежать повторных вычислений для одних и тех же входных данных и улучшить производительность функции.

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

Хвостовая рекурсия – это форма рекурсии, при которой рекурсивный вызов является последней операцией в функции. Оптимизация хвостовых вызовов, называемая «хвостовой рекурсией», позволяет некоторым интерпретаторам языков программирования оптимизировать исползование стека для хвостовых рекурсивных вызовов, избегая переполнения стека. В Python, к сожалению, интерпретатор CPython не поддерживает оптимизацию хвостовых вызовов, хотя некоторые другие языки или реализации Python могут предоставлять такую оптимизацию.

**Выводы**: В процессе выполнения лабораторной работы были приобретены навыки по работе с рекурсивными функциями при написании программ с помощью языка программирования Python версии 3.x, был проработан пример, код которого представлен в таблице 1, а так же в файле example.py в данном репозитории, а так же было выполнено индивидуальное задание, код которого представлен в таблице 2 и файле individual.py данного репозитория.