

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 №2.25**  
**дисциплины «Анализ данных»**  
**Вариант 13**

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, 2024 г.

**Тема:** «Управление процессами в Python»

**Цель:** Приобретение навыков написания многозадачных приложений на языке программирования Python версии 3.x.

### Порядок выполнения работы

Индивидуальное задание: Для своего индивидуального задания лабораторной работы 2.23 необходимо реализовать вычисление значений в двух функций в отдельных процессах.

### Листинг 1 – Код individual.py

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import math
import multiprocessing

"""
Задача:
Для своего индивидуального задания лабораторной работы 2.23 необходимо
реализовать вычисление значений в двух функций в отдельных процессах.
"""

e = 10e-7
stepArray = multiprocessing.Array('d', [1])

def calculateY(x):
    return 0.5 * math.log((x + 1) / (x - 1))

def calculate_step(step, index, x, n):
    step[index] = 1

    def firstStep():
        step[index] *= (2 * n - 1)

    def secondStep():
        step[index] *= x ** (2 * n - 1)

    def thirdStep():
        step[index] **= -1

    with multiprocessing.Pool(processes=3) as pool:
        pool.map(lambda f: f(), [firstStep, secondStep, thirdStep])

def main():
    x = 3
    index = 0

    while abs(stepArray[index]) > e:
        stepArray.append(0)
```

```

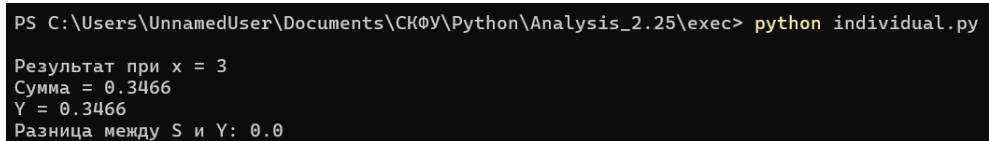
    calculate_step(stepArray, index + 1, x, index + 1)
    index += 1

S = sum(stepArray) - 1
y = calculateY(x)

print(f"\nРезультат при x = {x}")
print(f"Сумма = {round(S, 4)}")
print(f"Y = {round(y, 4)}")
print(f"Разница между S и Y: {abs(round(S - y, 4))}\n")

if __name__ == "__main__":
    main()

```



```

PS C:\Users\UnnamedUser\Documents\CKФУ\Python\Analysis_2.25\exec> python individual.py

Результат при x = 3
Сумма = 0.3466
Y = 0.3466
Разница между S и Y: 0.0

```

Рисунок 1 – Результат выполнения программы

### Контрольные вопросы

1. Как создаются и завершаются процессы в Python?

Процессы создаются в Python с использованием модуля multiprocessing путём создания экземпляра класса Process и вызова его метода start(). Завершить процесс можно вызовом метода join() для ожидания завершения процесса.

2. В чем особенность создания классов-наследников от Process?

Особенности создания классов-наследников от Process заключается в реализации метода run(), который содержит код, который будет выполняться в процессе. Также класс-наследник должен быть инициализирован через конструктор родительского класса.

3. Как выполнить принудительное завершение процесса?

Принудительное завершение процесса можно выполнить вызовом метода terminate() для объекта процесса. Этот метод прерывает выполнение процесса.

#### 4. Что такое процессы-демоны? Как запустить процесс-демон?

Процессы-демоны – это процессы, которые работают в фоновом режиме и не останавливаются автоматически при завершении основной программы. Для запуска процесса-демона необходимо установить его атрибут `daemon` в значение `True` перед вызовом метода `start()`.

**Выводы:** В процессе выполнения лабораторной работы были приобретены навыки написания многозадачных приложений на языке программирования Python, а также было выполнено индивидуальное задание.