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 №11
дисциплины «Анализ данных»

Выполнила:
Кубанова Ксения Олеговна
2 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика и вычислительная техника», очная форма обучения

_____
(подпись)

Руководитель практики:
Воронкин Р. А.

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2024 г.

Тема: управление процессами

Цель: приобрести навыки управления процессами

Порядок выполнения работы

Индивидуальное задание.

Необходимо реализовать вычисление значений в двух функций в отдельных процессах для кода лабораторной работы 2.23.

```
# Создаем процессы для вычисления числителя и знаменателя
chis_process = Process(target=calc_chis, args=(x, i, result_queue))
znam_process = Process(target=calc_znam, args=(i, result_queue))

# Запускаем процессы
chis_process.start()
znam_process.start()

# Добавляем процессы в список для последующего ожидания их завершения
processes.append(chis_process)
processes.append(znam_process)

# Получаем результаты из очереди
chis = result_queue.get()
znam = result_queue.get()
```

Рисунок 1 – реализация управления процессами

```
x = -0.7
Ожидаемое значение y = 0.6126263941844161
Подсчитанное знанчение = 0.51
```

Рисунок 2 – результат выполнения кода

Для вычисления значений числителя и знаменателя в отдельных процессах был использован модуль multiprocessing. Результаты каждого процесса помещаются в очередь result_queue, из которой они затем извлекаются и используются для вычисления следующего элемента ряда.

Готовый код располагается в файле ind.py.

Контрольные вопросы

1 Как создаются и завершаются процессы в Python?

Для создания процессов в Python используется модуль multiprocessing. Процесс создается путем создания экземпляра класса Process, который принимает функцию, которая будет выполняться в процессе, в качестве

аргумента. Пример создания процесса: my_process = Process(target=my_function).

Для запуска процесса вызывается метод start(): my_process.start().

Завершение процесса происходит путем вызова метода join(), который ожидает завершения процесса: my_process.join().

2 В чем особенность создания классов-наследников от Process?

При создании классов-наследников от Process необходимо переопределить метод run(), который будет содержать код, выполняемый в процессе.

В отличие от обычных функций, которые могут быть переданы в качестве аргумента в конструктор Process, класс-наследник должен быть создан и передан экземпляром Process для выполнения.

3 Как выполнить принудительное завершение процесса?

В стандартной библиотеке Python нет способа принудительного завершения процесса. Однако вы можете использовать флаг или переменную для обозначения того, что процесс должен завершиться, и проверять эту переменную внутри цикла выполнения процесса.

Другой вариант - это использование метода terminate(), который завершает процесс, но при этом не гарантирует, что все ресурсы будут правильно освобождены.

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

Процессы-демоны (daemon processes) - это процессы, которые работают в фоновом режиме и не препятствуют завершению программы.

Для запуска процесса-демона в Python используется метод daemon класса Process. Установка этого атрибута в True перед запуском процесса указывает на то, что процесс является демоном.

Вывод: в ходе выполнения лабораторной работы были приобретены навыки управления процессами.