Программирование. Язык Python.

Лабораторная работа. Задачи. Генераторы. Менеджеры контекста.

Комплект 1: Итераторы. Генераторы.

1.1: Создайте свой класс-итератор `class RandomNumberIterator`, который, в ходе итерирования по такому итератору, генерирует случайные числа в количестве и в диапазоне, которые передаются в конструктор в виде списка параметров.

1.2: Решите задачу 1.1 уже с использованием генераторной функции, использующей ключевое слово `yield`. В качестве аргументов она должна принимать количество элементов и диапазон.

1.3: Сделайте две функции-генератора. Первый генератор создаёт ряд Фибоначчи, а второй генератор добавляет значение 10 к каждому числу. Вызовете эти генераторы так, чтобы сгенерировать некоторое количество чисел Фибоначчи с добавлением числа 10 к каждому числу.

1.4: Напишите программу, на вход к которой подается список стран и городов для каждой страны. Затем по названиям городов из ещё одного списка выводится в какой стране расположен каждый город.

Комплект 2: Менеджеры контекста.

2.1: Напишите класс менеджера контекста `Timer`, который умеет считать время в секундах, затраченное на некоторые вычисления внутри соответствующего блока `with` с помощью функции `perf_counter` модуля `time`. Используйте этот менеджер контекста для определения времени на вычисления достаточно большого количества чисел Фибоначчи (например миллиона) в цикле с помощью отдельной функции-генератора.

2.2: Напишите класс менеджера контекста `BatchCalculatorContextManager`, для вашего проекта калькулятора из предыдущих лабораторных работ. Этот менеджер контекста должен уметь открывать и закрывать

текстовый файл, в каждой строчке которого записана пара чисел в сочетании с арифметической операцией над ними в виде простого арифметического выражения без пробелов. В сочетании с дополнительной функцией генератором и вашим менеджером контекста прочитайте все строчки текстового файла и вызовите нужное число раз функцию `calculate(...)` вашего калькулятора, чтобы распечатать все результаты на экране.

2.3: Установите локально на свой компьютер объектную базу данных MongoDB. Установите с помощью менеджера пакетов `pip` или `conda`, в зависимости от того чем вы пользуетесь, пакет `pymongo` для подключения к базам данных MongoDB. Например команда для `pip`:
`pip install pymongo`. С помощью инструмента MongoDB Shell создайте нового пользователя с правами админа, к примеру. Введите в командной строке `mongosh` без аргументов и уже в командной строке внутри MongoDB Shell введите:

```
db.createUser({
    user: "myUserAdmin",
    pwd: "abc123",
    roles: [
        { role: "userAdminAnyDatabase", db: "admin" },
        "readWriteAnyDatabase"
    ]
})
```

Затем выйдите из MongoDB Shell (Введите `exit` или нажмите `Ctrl-D`). Перезайдите снова в MongoDB Shell с помощью команды `mongosh -u myUserAdmin` в командной строке и введя пароль `abc123`. Тем самым вы залогинетесь в базу MongoDB под новой учётной записью. Создайте пустую базу данных `myshinynewdb` с помощью команды `use myshinynewdb`. Добавьте коллекцию `user` в эту базу данных с одной единственной записью: `db.user.insert({name: "Ada Lovelace", age: 205})`. Коллекция будет создана автоматически. Напишите класс менеджера контекста для управляемого подключения к MondoDB и отключения от неё. Внутри блока `with` с помощью вызова метода `user_collection.find({'age': 205})` найдите вашу запись об `"Ada Lovelace"` и распечатайте её в терминале.
Пример инструкций по установке MongoDB в Windows:
`https://metanit.com/nosql/mongodb/1.2.php`.
Инструкция по созданию базы данных `myshinynewdb`:
`https://www.mongodb.com/basics/create-database`.
Пример решения приведён ниже:

```python
from pymongo import MongoClient


class MongoDBConnectionContextManager(object):
    """MongoDB Connection Context Manager"""
    def __init__(self, host='localhost', port=27017, username='admin', password='admin'):
        self.host = host; self.port = port
        self.username = username; self.password = password
        self.connection = None

    def __enter__(self):
        self.connection = MongoClient(
            self.host, self.port,
            username=self.username, password=self.password,
            authMechanism='SCRAM-SHA-1')
        return self

    def __exit__(self, exc_type, exc_val, exc_tb):
        self.connection.close()
```

[23] ✓ 0.2s

```python
mongo = MongoDBConnectionContextManager(host='localhost', port=27017, username='myUserAdmin', password='abc123')
with mongo as mongo_connection_context:
    collection = mongo_connection_context.connection['myshinynewdb']['user']
    user = collection.find({'age': 205})
    print(next(user))
```

[24] ✓ 0.6s

```
{'_id': ObjectId('6385ef3006529f7b69971ef4'), 'name': 'Ada Lovelace', 'age': 205}
```