**Санкт-Петербургский государственный университет**

**Факультет прикладной математики и процессов управления**

**ОТЧЁТ**

**по дисциплине “Функциональное программирование”**

**об использовании функций, действующих над коллекциями**

**1. Введение**

В данном отчете будет рассмотрено решение трех задач, связанных с обработкой данных с использованием языка программирования Python. Каждая задача будет подробно описана, а также предоставлен код решения для каждой из них.

**2. Цель работы**

Цель работы состоит в решении трех задач:

1. Вычисление статистики успеваемости студентов.
2. Расчет общей суммы расходов для пользователей с заданными критериями.
3. Работа с большой базой данных заказов и клиентов.

**3. Задачи**

Для достижения цели работы необходимо решить следующие задачи:

Задача 1: Вычисление статистики успеваемости студентов.

* Фильтрация данных: Отфильтровать студентов определенного возраста и/или с определенным списком предметов.
* Преобразование данных: Вычислить средний балл для каждого студента и общий средний балл по всем студентам.
* Агрегация данных: Найти студента (или студентов) с самым высоким средним баллом.

Задача 2: Расчет общей суммы расходов для пользователей с заданными критериями.

* Отфильтровать пользователей по заданным критериям.
* Для каждого пользователя рассчитать общую сумму его расходов.
* Получить общую сумму расходов всех отфильтрованных пользователей.

Задача 3: Работа с большой базой данных заказов и клиентов.

* Фильтрация заказов: Отфильтровать заказы только для определенного клиента с заданным идентификатором клиента.
* Подсчет суммы заказов: Подсчитать общую сумму всех заказов для данного клиента.
* Подсчет средней стоимости заказов: Найти среднюю стоимость заказов для данного клиента.

**4. Техническое описание**

**4.1 Используемые инструменты и библиотеки**

Для выполнения задач был использован язык программирования Python. Для обработки данных использовались встроенные функции Python и библиотека functools.

**4.2 Структура программы**

Задача 1: Вычисление статистики успеваемости студентов

1. students: Это список словарей, представляющих данные о студентах и их оценках. Каждый словарь содержит информацию о имени студента ("name"), его возрасте ("age") и списке оценок ("grades").
2. filtered\_students: Этот список содержит результат фильтрации данных. Он создается с использованием функции filter, которая отфильтровывает студентов в соответствии с заданными критериями (в данном случае, возрастом и оценками).
3. average\_grades\_per\_student: Этот список содержит результат преобразования данных. Он создается с использованием функции map, которая преобразует каждого студента в словарь с именем студента и его средней оценкой.
4. total\_average\_grade: Эта переменная хранит общий средний балл по всем студентам. Это значение вычисляется с использованием функции reduce, которая суммирует средние оценки всех студентов и делит на количество студентов.
5. student\_with\_highest\_average\_grade: Эта переменная содержит информацию о студенте с самым высоким средним баллом. Она находится с использованием функции max, которая определяет студента с наибольшей средней оценкой.

Задача 2: Расчет общей суммы расходов для пользователей

1. users: Это список словарей, представляющих данные о пользователях и их расходах. Каждый словарь содержит информацию о имени пользователя ("name") и списке расходов ("expenses").
2. filtered\_users: Этот список содержит результат фильтрации данных. Он создается с использованием спискового включения (list comprehension), который отфильтровывает пользователей в соответствии с заданными критериями (в данном случае, суммой расходов).
3. total\_expenses\_per\_user: Это словарь, который хранит информацию о каждом пользователе и общей сумме его расходов. Словарь создается с использованием словарного включения (dictionary comprehension).
4. total\_expenses\_filtered: Эта переменная содержит общую сумму расходов всех отфильтрованных пользователей. Она вычисляется с использованием встроенной функции sum, которая суммирует значения из списка.

Задача 3: Работа с большой базой данных заказов и клиентов

1. orders: Это список словарей, представляющих данные о заказах и клиентах. Каждый словарь содержит информацию о номере заказа ("order\_id"), идентификаторе клиента ("customer\_id") и стоимости заказа ("amount").
2. filtered\_orders: Этот список содержит результат фильтрации данных. Он создается с использованием спискового включения, которое отфильтровывает заказы только для определенного клиента с заданным идентификатором клиента.
3. total\_amount: Эта переменная хранит общую сумму заказов для данного клиента. Она вычисляется с использованием встроенной функции sum, которая суммирует стоимость заказов из отфильтрованных заказов.
4. average\_amount: Эта переменная содержит среднюю стоимость заказов для данного клиента. Она вычисляется путем деления общей суммы заказов на количество заказов.

**5** **Описание программы**

**5.1 Описание ключевых компонентов программы**

Задача 1: Вычисление статистики успеваемости студентов

* students: Список словарей, представляющих данные о студентах и их оценках.
* filtered\_students: Результат фильтрации студентов по заданным критериям.
* average\_grades\_per\_student: Список словарей с именами студентов и их средними оценками.
* total\_average\_grade: Общий средний балл по всем студентам.
* student\_with\_highest\_average\_grade: Студент с наивысшим средним баллом.

Задача 2: Расчет общей суммы расходов для пользователей

* users: Список пользователей с информацией о расходах.
* filtered\_users: Результат фильтрации пользователей по заданным критериям.
* total\_expenses\_per\_user: Словарь с именами пользователей и общими суммами их расходов.
* total\_expenses\_filtered: Общая сумма расходов всех отфильтрованных пользователей.

Задача 3: Работа с большой базой данных заказов и клиентов

* orders: Список заказов с информацией о заказах и клиентах.
* filtered\_orders: Результат фильтрации заказов по заданному клиенту.
* total\_amount: Общая сумма заказов для данного клиента.
* average\_amount: Средняя стоимость заказов для данного клиента.

**5.2 Общий ход программы**

Задача 1: Вычисление статистики успеваемости студентов

1. **Фильтрация данных (**filtered\_students**):** В данной задаче данные о студентах фильтруются в соответствии с заданными критериями (в данном случае, возрастом и оценками). Функция filter используется для создания нового списка filtered\_students, в котором будут только те студенты, которые соответствуют заданным условиям (моложе 22 лет и имеют оценку выше 90 в первом и последнем предмете).
2. **Преобразование данных (**average\_grades\_per\_student**):** После фильтрации данных, создается список average\_grades\_per\_student, который содержит информацию о каждом студенте и его средней оценке. Это выполняется с помощью функции map, которая применяет функцию вычисления средней оценки к каждому студенту из filtered\_students.
3. **Агрегация данных (**total\_average\_grade **и** student\_with\_highest\_average\_grade**):** После преобразования данных вычисляется общий средний балл по всем студентам и находится студент с самым высоким средним баллом. Общий средний балл вычисляется с использованием функции reduce, которая суммирует средние баллы всех студентов и делит на количество студентов. Студент с самым высоким средним баллом определяется с помощью функции max, которая находит максимальное значение среди всех студентов.

Задача 2: Расчет общей суммы расходов для пользователей

1. **Фильтрация данных (**filtered\_users**):** В данной задаче пользователи фильтруются в соответствии с заданными критериями (суммой расходов). Списковое включение (list comprehension) используется для создания нового списка filtered\_users, который содержит только тех пользователей, чьи расходы соответствуют заданному условию (расходы менее 400).
2. **Расчет общей суммы расходов для каждого пользователя (**total\_expenses\_per\_user**):** Создается словарь total\_expenses\_per\_user, который содержит информацию о каждом пользователе и общей сумме его расходов. Для этого используется словарное включение (dictionary comprehension).
3. **Получение общей суммы расходов всех отфильтрованных пользователей (**total\_expenses\_filtered**):** Общая сумма расходов всех отфильтрованных пользователей вычисляется с использованием встроенной функции sum, которая суммирует значения расходов из списка filtered\_users.

Задача 3: Работа с большой базой данных заказов и клиентов

1. **Фильтрация заказов (**filtered\_orders**):** Заказы фильтруются в соответствии с заданным идентификатором клиента. Создается новый список filtered\_orders, который содержит только заказы для данного клиента. Это выполняется с использованием спискового включения.
2. **Подсчет суммы заказов (**total\_amount**):** Общая сумма заказов для данного клиента вычисляется с использованием встроенной функции sum, которая суммирует стоимость заказов из списка filtered\_orders.
3. **Подсчет средней стоимости заказов (**average\_amount**):** Средняя стоимость заказов для данного клиента вычисляется путем деления общей суммы заказов (total\_amount) на количество заказов (длина списка filtered\_orders).

На рис. 1 приведено окно пользователя.

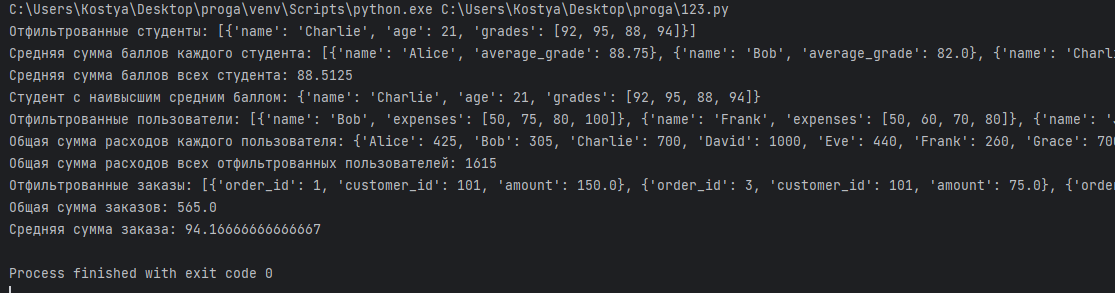


Рис. 1 Окно пользователя

**6. Рекомендации пользователю**

* Запустите программу используя интерпретатор python
* Если при работе программа выдает ошибку, обратитесь к программисту

**7. Рекомендации программисту**

* **Убедитесь в совместимости версии Python:** Проверьте, что ваш код будет работать на конкретной версии Python, которую вы используете. В приведенных вами кодах, используются функции filter, map, reduce, sum, и другие. Убедитесь, что эти функции доступны и работают правильно в вашей версии Python.
* **Проверьте структуру данных:** Убедитесь, что структура данных, с которой вы работаете (например, список студентов, пользователей или заказов), соответствует ожидаемой структуре, определенной в коде.

**8. Вывод**

В ходе выполнения задач были успешно решены следующие задачи:

1. Вычисление статистики успеваемости студентов.
2. Расчет общей суммы расходов для пользователей с заданными критериями.
3. Работа с большой базой данных заказов и клиентов.

Код решения каждой задачи представлен выше, и он может быть использован как пример для аналогичных задач обработки данных.

**Библиография**

Python.org - Официальный сайт Python:

* Ссылка: <https://www.python.org/>

Python Documentation - Официальная документация Python:

* Ссылка: <https://docs.python.org/>

Документация по модулю functools:

* Ссылка: <https://docs.python.org/3/library/functools.html>

GitHub - https://github.com/Gorbunov-Konstantin/Chess.git