**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Парадигмы и конструкции языков программирования»

Отчет по лабораторной работе №2

(Лабораторная работа №3 F#)

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-34Б |  | преподаватель каф. ИУ5 |
| Байдаков Владислав |  | Нардид Анатолий Николаевич |
|  |  |  |
|  |  |  |

2024 г.

**Задание**

1. Разработайте функцию, которая принимает три параметра обобщенных

типов и возвращает их в виде кортежа. Модифицируйте функцию: не

указывая явно типы параметров, задавая выражения в теле функции,

сделайте так, чтобы параметры были типов int, float, string.

2. С использованием двухэтапного создания обобщенных функций

реализуйте функции, которые осуществляют сложение:

11

• трех аргументов типа int;

• трех аргументов типа float;

• трех аргументов типа string.

3. С использованием list comprehension для четных элементов списка [1..10]

верните список кортежей. Каждый кортеж содержит элемент списка, его

квадрат и куб.

4. Напишите два варианта функции, которая принимает на вход список и

возвращает квадраты его значений. Необходимо использовать свойства

списка Head и Tail. Первый вариант функции использует оператор if, второй

вариант использует сопоставление с образцом на уровне функции.

5. Последовательно примените к списку функции map, sort, filter, fold, zip,

функции агрегирования. Функции применяются в любом порядке и

произвольно используются в трех комбинациях.

• Первая комбинация заканчивается функцией агрегирования (например,

сумма элементов списка). Cписок предварительно может быть

отсортирован, отфильтрован и т.д.

• Вторая комбинация заканчивается функцией fold, которая

осуществляет свертку списка. Вторая комбинация выполняет те же

действия, что и первая комбинация и должна возвращать такой же

результат.

• Третья комбинация заканчивается функцией zip, которая соединяет два

списка.

6. Реализуйте предыдущий пункт с использованием оператора потока « |> ».

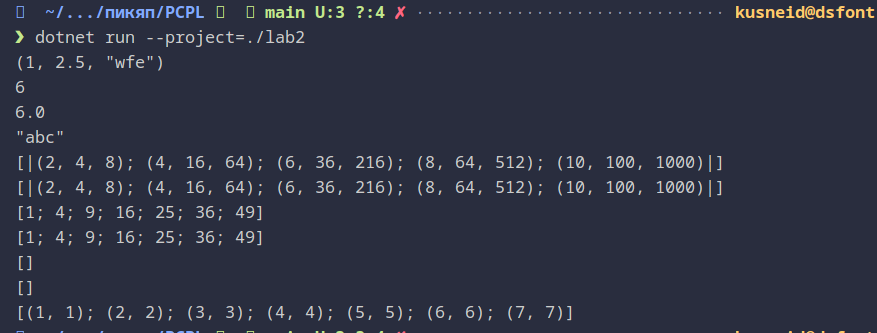
7. Реализуйте предыдущий пункт с использованием оператора композиции

функций « >> ».

**Текст программы**

см. https://github.com/kusneid/PCPL/tree/main/lab2

**Результаты программы:**

****