|  |  |  |
| --- | --- | --- |
| Министерство науки и высшего образования РФ  Федеральное государственное автономное образовательное учреждение высшего образования  «Пермский государственный национальный исследовательский университет» | | |
|  | Институт компьютерных наук и технологий | |
| **ОТЧЁТ**  по лабораторной работе №5  по дисциплине «Языки программирования»  Вариант 8 | | |
|  | | Работу выполнил  студент группы ПМИ-1,2-2023 2 курса  Хисматов Е.С.  «16» декабря 2024 г. |
| Работу проверил  Ракина В. Д.  «\_\_\_» \_\_\_\_\_\_\_\_\_ 2024 г. |
| Пермь 2024 | | |

СОДЕРЖАНИЕ

[Задание 1 4](#_Toc185336405)

[Текст задания 4](#_Toc185336406)

[Алгоритм решения 4](#_Toc185336407)

[Тестирование 5](#_Toc185336408)

[Код программы 5](#_Toc185336409)

[Задание 2 6](#_Toc185336410)

[Текст задания 6](#_Toc185336411)

[Алгоритм решения 6](#_Toc185336412)

[Тестирование 7](#_Toc185336413)

[Код программы 7](#_Toc185336414)

[Задание 3 8](#_Toc185336415)

[Текст задания 8](#_Toc185336416)

[Алгоритм решения 8](#_Toc185336417)

[Тестирование 8](#_Toc185336418)

[Код программы 8](#_Toc185336419)

[Задание 4 9](#_Toc185336420)

[Текст задания 9](#_Toc185336421)

[Алгоритм решения 10](#_Toc185336422)

[Тестирование 10](#_Toc185336423)

[Код программы 10](#_Toc185336424)

[Задание 5 11](#_Toc185336425)

[Текст задания 11](#_Toc185336426)

[Алгоритм решения 11](#_Toc185336427)

[Тестирование 11](#_Toc185336428)

[Код программы 11](#_Toc185336429)

[Задание 6 12](#_Toc185336430)

[Текст задания 12](#_Toc185336431)

[Алгоритм решения 12](#_Toc185336432)

[Тестирование 12](#_Toc185336433)

[Код программы 12](#_Toc185336434)

[Задание 7 13](#_Toc185336435)

[Текст задания 13](#_Toc185336436)

[Алгоритм решения 13](#_Toc185336437)

[Тестирование 14](#_Toc185336438)

[Код программы 14](#_Toc185336439)

[Задание 8 15](#_Toc185336440)

[Текст задания 15](#_Toc185336441)

[Алгоритм решения 15](#_Toc185336442)

[Тестирование 15](#_Toc185336443)

[Код программы 16](#_Toc185336444)

# Задание 1

## Текст задания

В класс Дробь, добавить интерфейс на два метода: получение вещественного значения, установка числителя и установка знаменателя. Сгенерировать такую версию дроби, которая будет кэшировать вычисление вещественного значения. Если раннее в вашем варианте не было Дроби, то создайте сущность Дробь со следующими особенностями:

• Имеет числитель: целое число

• Имеет знаменатель: целое число

• Дробь может быть создана с указанием числителя и знаменателя

• Может вернуть строковое представление вида “числитель/знаменатель”

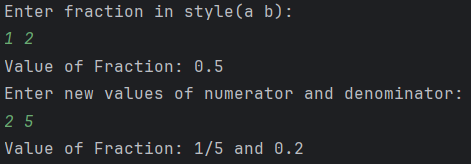
• Необходимо корректно обрабатывать отрицательные значения. Учтите, что знаменатель не может быть отрицательным.

• Переопределите метод сравнения объектов по состоянию таким образом, чтобы две дроби считались одинаковыми тогда, когда у них одинаковые значения числителя и знаменателя.

## Алгоритм решения

Создать класс Fraction, реализующий интерфейс с методами getValue, setNumerator и setDenominator. Класс содержит числитель и знаменатель, гарантируя, что знаменатель всегда положительный. Создать класс CacheFraction с полем кэша вещественного значения дроби для оптимизации повторных вычислений. Реализуем методы интерфейса getValue, setNumerator и setDenominator, которые изменяют данные дроби и переопределяют кэш. Реализовать строковое представление в формате "числитель/знаменатель" и метод сравнения, определяющий равенство дробей на основе их числителя и знаменателя.

## Тестирование



## Код программы

[https://github.com/NTHrdd/Lab5](https://github.com/NTHrdd/Lab3)

# Задание 2

## Текст задания

Количество мяуканий. Необходимо воспользоваться классом Кот и методом принимающим всех мяукающих из задачи 2.5.4. Необходимо таким образом передать кота в указанный метод, что бы после окончания его работы узнать сколько раз мяукал кот за время его работы. На рисунке показан пример работы. Перед вызовом метода создаем кота, отправляем ссылку на кота в метод, после окончания его работы выводим количество мяуканий на экран. Кота изменять нельзя. Если раннее в вашем варианте не было Кота, то создайте

1. сущность Кот, которая описывается следующим образом:

• Имеет Имя (строка)

• Для создания необходимо указать имя кота.

• Может быть приведен к текстовой форме вида: “кот: Имя”

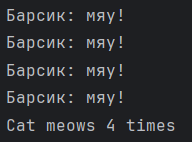
• Может помяукать, что приводит к выводу на экран следующего текста: “Имя: мяу!”, вызвать мяуканье можно без параметров.

2. интерфейс Мяуканье: разработайте метод, который принимает набор объектов способных мяукать и вызывает мяуканье у каждого объекта. Мяукающие объекты должны иметь метод со следующей сигнатурой: public void meow();

## Алгоритм решения

Создаем интерфейс Meowable с методом void meow(). Создаем сущность Cat с полем имени, конструктором для его инициализации, методом toString, возвращающим строку "кот: Имя", и методом meow, выводящим на экран "Имя: мяу!". Для отслеживания количества мяуканий создать декоратор) вокруг объекта Кот, который будет перехватывать вызовы метода meow и увеличивать счетчик мяуканий. Создаем статический метод, который создает объект кота и его декоратор.

## Тестирование



## Код программы

[https://github.com/NTHrdd/Lab5](https://github.com/NTHrdd/Lab3)

# Задание 3

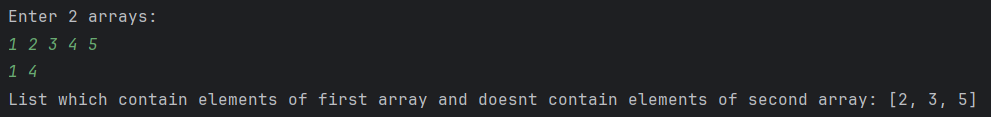
## Текст задания

Составить программу, которая формирует список L, включив в него по одному разу элементы, которые входят в список L1, но не входят в список L2.

## Алгоритм решения

Создать метод formList, который принимает два списка list1 и list2 и возвращает результирующий список. Инициализировать пустой список result для хранения элементов. Пройти по каждому элементу t из list1 и добавить его в result, если он отсутствует в list2 и еще не был добавлен в result. Проверку наличия элемента выполнять с помощью методов contains. После завершения цикла вернуть сформированный список result.

## Тестирование



## Код программы

[https://github.com/NTHrdd/Lab5](https://github.com/NTHrdd/Lab3)

# Задание 4

## Текст задания

Имеется список учеников разных школ, сдававших экзамен по информатике, с указанием их фамилии, имени, школы и набранного балла. Напишите программу, которая будет определять двух учеников школы № 50, которые лучше всех сдали информатику, и выводить на экран их фамилии и имена.

Если наибольший балл набрали более двух человек, нужно вывести только их количество. Если наибольший балл набрал один человек, а следующий балл набрало несколько человек, нужно вывести только фамилию и имя лучшего. Известно, что информатику сдавали не менее 5 учеников школы № 50.

На вход программе в первой строке подается количество учеников списке N. В каждой из последующих N строк находится информация в следующем формате:

<Фамилия><Имя><Школа><Балл>

где <Фамилия> – строка, состоящая не более, чем из 20 символов без пробелов, <Имя> – строка, состоящая не более, чем из 20 символов без пробелов, <Школа> – целое число от 1 до 99, <Балл> – целое число от 1 до 100.

Пример входной строки:

Иванов Сергей 50 87

Пример выходных данных, когда найдено два лучших:

Иванов Сергей

Сергеев Иван

Если больше двух учеников набрали высший балл, то программа должна вывести их количество.

Пример вывода в этом случае:

8

Если высший балл набрал один человек, а следующий балл набрало несколько человек, то программа должна вывести только фамилию и имя лучшего.

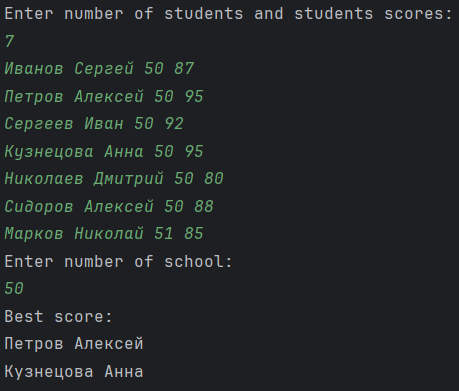
Пример вывода в этом случае:

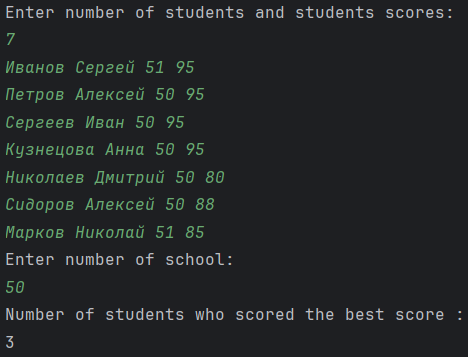
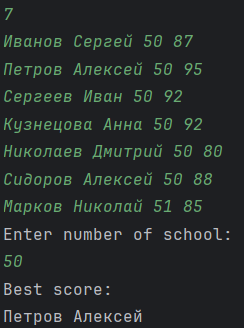
Иванов Сергей

## Алгоритм решения

Создать сущность студенты, которая имеет поля Фамилии, Имени, Номера школы и Количества набранных баллов. Создать сущность Топа студентов методами сканирования данных, который возвращает список студентов по номерам школ, фильтрации, который возвращает список студентов по номеру школы, и получение лучших результатов, выводящим на экран фамилии и имена студентов с лучшими результаты.

## Тестирование





## Код программы

[https://github.com/NTHrdd/Lab5](https://github.com/NTHrdd/Lab3)

# Задание 5

## Текст задания

Прочитать текст из файла и преобразовать его в строку. Создать множество всех встреченных в тексте букв русского алфавита. Циклом пройтись по всем буквам и если какая-то буква отсутствует в множестве, то счетчик инкрементируется. Результат возвращается.

## Алгоритм решения

Создать метод filter, который будет принимать: список значений типа T, объект Predicate<T>, который будет проверять каждый элемент списка. Для каждого элемента вызвать метод test объекта Predicate. Если результат true, добавить элемент в новый список. Вернуть новый список.

## Тестирование





## Код программы

[https://github.com/NTHrdd/Lab5](https://github.com/NTHrdd/Lab3)

# Задание 6

## Текст задания

Определить, есть ли в очереди L хотя бы один элемент, который равен следующему за ним (по кругу) элементу (первый элемент считать следующим для последнего).

## Алгоритм решения

Проверка на наличие в очереди хотя бы одного элемента, который равен следующему за ним элементу (с учетом кругового перехода). Для этого создаем итератор по очереди и проходим по всем элементам. При этом сравниваем текущий элемент с предыдущим, а последний элемент — с первым. Если находим равенство, возвращаем true. Если таких элементов нет, возвращаем false.

## Тестирование





## Код программы

[https://github.com/NTHrdd/Lab5](https://github.com/NTHrdd/Lab3)

# Задание 7

## Текст задания

Необходимо написать стрим: Дан набор объектов типа Point, необходимо взять все Point в разных координатах, убрать с одинаковыми X,Y, отсортировать по X, отрицательные Y сделать положительными и собрать это все в ломаную (объект типа Polyline) Если раннее в вашем варианте не было задание с классом Point и Polyline, то написать их:

1. класс Point:

• Координата Х: число.

• Координата Y: число.

• Может возвращать текстовое представление вида “{X;Y}”.

2. класс Line (Линия), расположенная на двумерной плоскости, которая описывается:

• Координата начала: Точка

• Координата конца: Точка

• Может возвращать текстовое представление вида “Линия от {X1;Y1} до {X2;Y2}”

3. класс Polyline (Ломаная), которая будет представлять собой ломаную линию. Ломаная линия представляет собой набор следующих характеристик:

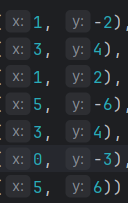
• Имеет массив Точек, через которые линия проходит.

• Может быть приведена к строковой форме вида “Линия [Т1,T2,…,TN]”, где TN – это результат приведения к строке Точки с номером N

## Алгоритм решения

Создать класс Point, содержащий координаты X и Y, с методом toString, который возвращает строку вида "{X;Y}". Затем создать класс Line, который представляет линию, используя две точки (начальную и конечную), с методом toString, возвращающим строку вида "Линия от {X1;Y1} до {X2;Y2}". Далее, создать класс Polyline, который представляет собой набор точек, с методом toString, выводящим строку в виде "Линия [Т1,T2,…,TN]". Для работы с набором объектов типа Point использовать стримы: выбрать уникальные точки с разными координатами (по X и Y), отсортировать их по X, преобразовать все отрицательные Y в положительные, и собрать результат в объект типа Polyline.

## Тестирование





## Код программы

[https://github.com/NTHrdd/Lab5](https://github.com/NTHrdd/Lab3)

# Задание 8

## Текст задания

Дан текстовый файл со строками, содержащими имя человека и его номер в следующей форме:

Вася:5

Петя:3

Аня:5

Номера людей могут повторяться. У каких-то людей может не быть номера. Необходимо написать стрим выполняющую следующее: читаются все люди из файла, все имена приводится к нижнему регистру, но с первой буквой в верхнем регистре, убираем из перечня всех людей без номеров, а имена оставшихся группируются по их номеру:

[5:[Вася, Аня], 3:[Петя]]

## Алгоритм решения

Прочитать строки из файла с помощью Files.lines. Для каждой строки разделить её на имя и номер с использованием split(":"). Отфильтровать строки, у которых отсутствует номер. Преобразовать имя, приводя его к формату, где первая буква заглавная, а остальные — строчные, используя substring(0, 1).toUpperCase() и substring(1).toLowerCase(). Собрать результаты в Map, где ключ — это номер, а значения — имена людей, сгруппированные по номерам, используя Collectors.groupingBy с дополнительным преобразованием. Использовать TreeMap для сортировки по номерам в порядке возрастания.

## Тестирование





## Код программы

[https://github.com/NTHrdd/Lab5](https://github.com/NTHrdd/Lab3)