|  |  |  |
| --- | --- | --- |
| 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 №4  по дисциплине «Языки программирования»  Вариант 5 | | |
|  | | Работу выполнил  студент группы ИТ-11,12-2023 2 курса  Простолупов Р. Г.  «4» марта 2025 г. |
| Работу проверил  Ракина В. Д.  «\_\_\_» \_\_\_\_\_\_\_\_\_ 2025 г. |
| Пермь 2025 | | |

СОДЕРЖАНИЕ

[Задание 1 (1.1) 3](#_Toc192116041)

[Текст задания 3](#_Toc192116042)

[Алгоритм решения 3](#_Toc192116043)

[Тестирование 3](#_Toc192116044)

[Задание 2 (2.1) 4](#_Toc192116045)

[Текст задания 4](#_Toc192116046)

[Алгоритм решения 4](#_Toc192116047)

[Тестирование 4](#_Toc192116048)

[Задание 3 (3.4) 5](#_Toc192116049)

[Текст задания 5](#_Toc192116050)

[Алгоритм решения 5](#_Toc192116051)

[Тестирование 5](#_Toc192116052)

[Задание 4 (4.4) 5](#_Toc192116053)

[Текст задания 5](#_Toc192116054)

[Алгоритм решения 6](#_Toc192116055)

[Тестирование 6](#_Toc192116056)

[Задание 6 (3.3) 7](#_Toc192116057)

[Текст задания 7](#_Toc192116058)

[Алгоритм решения 7](#_Toc192116059)

[Тестирование 7](#_Toc192116060)

[Задание 7 (3.4) 8](#_Toc192116061)

[Текст задания 8](#_Toc192116062)

[Алгоритм решения 8](#_Toc192116063)

[Тестирование 8](#_Toc192116064)

[Код программы 8](#_Toc192116065)

# Задание 1 (1.1)

## 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

Создан класс Дробь, у которого есть две переменные: числитель и знаменатель; и следующие методы:

* nod – приватный метод поиска наибольшего общего делителя;
* simplify – метод, приводящий дробь к наименьшему кратному виду и решающий проблему отрицательного знаменателя;
* getNumerator, getDenominator
* setNumerator, setDenominator – установка значений переменных и выполнение simplify;
* getValue – получение вещественного значения дроби;
* equals – переопределение метода из класса Object;
* toString

Конструктор класса также предполагает выполнение simplify.

Создан интерфейс FractionInterface со следующими методами:

* setNumerator
* setDenominator
* getValue

Создан класс CacheFraction c двумя переменными: объектом класса Дробь и вещественным кэшированным значением. В классе переопределены все три метода вышеупомянутого интерфейса.

Создан класс InitiateFraction, в котором реализован метод создания экземпляра дроби с кэшированием вычисления вещественного значения.

## Тестирование

Задание 1.1: Класс дробь.

Введите числитель и знаменатель дроби (два целых числа через пробел): 4 16

Строковое представление дроби: 1/4

Десятичное значение дроби: 0.25

Введите новые числитель и знаменатель дроби (два целых числа через пробел): 16 4

Строковое представление дроби: 4/1

Десятичное значение дроби: 4.0

Задание 1.1: Класс дробь.

Введите числитель и знаменатель дроби (два целых числа через пробел): -3 7

Строковое представление дроби: -3/7

Десятичное значение дроби: -0.42857142857142855

Введите новые числитель и знаменатель дроби (два целых числа через пробел): 14 -10

Строковое представление дроби: -7/5

Десятичное значение дроби: -1.4

Задание 1.1: Класс дробь.

Введите числитель и знаменатель дроби (два целых числа через пробел): 0 -1000

Строковое представление дроби: 0/1

Десятичное значение дроби: 0.0

Введите новые числитель и знаменатель дроби (два целых числа через пробел): 2 0

Получена ошибка: java.lang.IllegalArgumentException: Знаменатель не может быть равным нулю

# Задание 2 (2.1)

## 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.

1. 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

Создан интерфейс Meowable с одним методом meow.

Создан класс Кот с переменной имени и методом meow, который однократно выводит мяуканье в консоль.

Создан класс MeowCount с переменной интерфейса Meowable и счётчиком. В нём переопределён метод meow таким образом, чтобы помимо вызова этого метода у экземпляра Meowable счётчик инкрементировался.

Создан класс CatCountInit, с единственным методом, который позволяет создать экземпляр интерфейса Meowable с экземпляром класса MeowCount с экземпляромм класса Cat.

## Тестирование

Задание 2.1: Количество мяуканий.

Meow...

Meow...

Meow...

Meow...

Кот: Миша, мяукал 4 раз

# Задание 3 (3.4)

## Текст задания

Составить программу, которая вставляет в список L за первым вхождением элемента E все элементы списка L, если E входит в L.

## Алгоритм решения

Ввод в программу исходного списка, реализованный с помощью Stream API; ввод в программу искомого значения.

Определение индекса первого вхождения искомого значения в список.

Если вхождение есть, создаётся копия исходного списка и вставляется в исходный, начиная со следующего индекса.

## Тестирование

Задание 3.1: Список. Вставить все элементы список после первого вхождения заданного значения.

Введите исходный список целых чисел через пробел (не меньше одного элемента):

1 2 3 4 5 6 7 8 9 10

Введите искомое значение:

3

Обновлённый список: [1, 2, 3, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 4, 5, 6, 7, 8, 9, 10]

Задание 3.1: Список. Вставить все элементы список после первого вхождения заданного значения.

Введите исходный список целых чисел через пробел (не меньше одного элемента):

-4 -30 -200 1000 -200 0

Введите искомое значение:

-200

Обновлённый список: [-4, -30, -200, -4, -30, -200, 1000, -200, 0, 1000, -200, 0]

Задание 3.1: Список. Вставить все элементы список после первого вхождения заданного значения.

Введите исходный список целых чисел через пробел (не меньше одного элемента):

10

Введите искомое значение:

0

В заданном списке нет ни одного вхождения указанного значения.

# Задание 4 (4.4)

## Текст задания

В некотором вузе абитуриенты проходят предварительное тестирование, по результатам которого могут быть допущены к сдаче вступительных экзаменов в первом потоке. Тестирование проводится по двум предметам, по каждому предмету абитуриент может набрать от 0 до 100 баллов. При этом к сдаче экзаменов в первом потоке допускаются абитуриенты, набравшие по результатам тестирования не менее 30 баллов по каждому из двух предметов. На вход программы подаются сведения о результатах предварительного тестирования. Известно, что общее количество участников тестирования не превосходит 500.

В первой строке вводится количество абитуриентов, принимавших участие в тестировании, N.

Далее следуют N строк, имеющих следующий формат:

**<Фамилия><Имя><Баллы>**

Здесь – строка, состоящая не более чем из 20 символов; – строка, состоящая не более чем из 15 символов; – строка, содержащая два целых числа, разделенных пробелом, соответствующих баллам, полученным на тестировании по каждому из двух предметов. При этом и , и разделены одним пробелом.

Примеры входных строк:

**Ветров Роман 68 59**

**Анисимова Екатерина 64 88**

Напишите программу, которая будет выводить на экран фамилии и имена абитуриентов, потерпевших неудачу, то есть не допущенных к сдаче экзаменов в первом потоке. При этом имена и фамилии можно выводить в произвольном порядке.

## Алгоритм решения

Создан класс StudentList, содержащий два поля:

* studentList - списки слов в списке строк исходного файла;
* studentMap – словарь соответствия студентов факту их успеха или неуспеха.

В классе определены два метода: getStudentList и getStudentMap.

Создан интерфейс StudentListInterface с двумя вышеупомянутыми методами.

Создан класс InitiateStudentList с единственным статичным методом, который создаёт экземпляр StudentList под интерфейсом, принимая в качестве параметра путь к исходному файлу.

## Тестирование

Задание 4.4: Мап. Вывести имена всех студентов, набравших не менее 30 баллов по каждому из экзаменов.

Исходный список:

Иванов Сергей 78 45

Петров Александр 92 88

Сидоров Дмитрий 60 74

Кузнецов Алексей 34 56

Смирнов Олег 81 90

Васильев Николай 47 32

Морозов Андрей 20 65

Новиков Евгений 23 41

Федоров Виктор 77 80

Михайлов Артем 55 49

Макаров Игорь 8 91

Никитин Максим 89 78

Захаров Павел 24 37

Соловьев Юрий 35 22

Борисов Станислав 90 85

Тимофеев Глеб 79 50

Фролов Владислав 62 93

Герасимов Константин 87 77

Пантелеев Виталий 29 58

Мельников Руслан 45 72

Гаврилов Аркадий 53 14

Трофимов Валентин 95 81

Тихонов Денис 74 33

Афанасьев Кирилл 40 29

Данилов Степан 61 86

Сорокин Эдуард 97 92

Веселов Георгий 31 66

Беляев Роман 88 79

Голубев Тимур 76 48

Коновалов Василий 52 57

Список студентов, потерпевших неудачу:

Морозов Андрей

Новиков Евгений

Макаров Игорь

Захаров Павел

Соловьев Юрий

Пантелеев Виталий

Гаврилов Аркадий

Афанасьев Кирилл

# Задание 5 (5.4)

## Текст задания

Файл содержит текст на русском языке. Напечатать в алфавитном порядке все глухие согласные буквы, которые входят в каждое нечетное слово.

## Алгоритм решения

Создан класс Article со строковой переменной для текста из файла и методом searchLetters.

В метод вводится набор (set) букв, которые нужно найти в тексте. Создаётся список List<String> слов исходного текста, затем при помощи Stream API из списка удаляются чётные слова, список слов преобразуется в список букв, удаляются все не входящие в заданный набор буквы, список сортируется и выводится в формат строки.

## Тестирование

Задание 5.4: Сет. Напечатать в алфавитном порядке все глухие согласные буквы, которые входят в каждое нечетное слово текста из файла.

Исходный текст:

Съешь ещё этих мягких французских булочек, да выпей же чаю.

Извлечение указанных букв в алфавитном порядке:

Скстфххцш

Задание 5.4: Сет. Напечатать в алфавитном порядке все глухие согласные буквы, которые входят в каждое нечетное слово текста из файла.

Исходный текст:

Файл содержит текст на русском языке. Напечатать в алфавитном порядке все глухие согласные буквы, которые входят в каждое нечетное слово.

Извлечение указанных букв в алфавитном порядке:

Фкккпсссссстттттттфчч

# Задание 6 (6.4)

## Текст задания

Очередь. Проверить равенство участка очереди с i-го по j-й элемент (i < j).

## Алгоритм решения

Создан метод checkEquality, принимающий некоторую очередь, индекс первого проверяемого элемента и индекс последнего проверяемого элемента.

Метод перебирает все элементы очереди при помощи итератора и с использованием двух вспомогательных переменных проверяет равенство всех элементов заданного участка первому элементу этого участка.

## Тестирование

Задание 6.4: Очередь. Проверить равенство участка очереди с i-го по j-й элемент (i < j).

Создана очередь (1,2,2,5,5,5).

Равенство элементов с четвёртого по шестой: true

Равенство элементов со второго по пятый: false

# Задание 7 (7.1)

## 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}”.

1. 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}”

1. 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

Создано три класса по заданных описаниям.

Создаётся список List<Point> с тестовым набором данных. Создаётся объект Polyline, в который вводится обработанный при помощи Stream API список точек.

## Тестирование

Задание 7.1: Стрим для обработки Point и Polyline

Создан список точек: [{1;0}, {2;2}, {3;-17}, {-9;5}, {1;0}, {0;-1}]

Из обработанного списка точек сформирован Polyline: Линия [{-9;5}, {0;1}, {1;0}, {2;2}, {3;17}]

**Задание 8 (7.2)**

**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**

В строковую переменную записывается путь к исходному файлу.

В Map<Integer, List<String>> вводятся данные через стрим со следующими командами:

1. Чтение данных из указанного файла в формат списка строк;
2. Удаление лишних пробелов;
3. Удаление записей, не содержащих разделитель «:»;
4. Сепарация записей на две подстроки, которые были разделены символом «:»;
5. Удаление записей, которые не содержали ничего после «:»;
6. Группировка записей по целочисленному значению в них
   1. Изменение регистра в соответствии со стандартом (Только первая буква в верхнем регистре).

Результат выводится.

**Тестирование**

Задание 7.2: Стрим для обработки файла.

Исходный файл:

вася:1

ПеТя:3

рАПтОр:1

Аня:

вАНЯ:1

Результат применения стрима: {1=[Вася, Раптор, Ваня], 3=[Петя]}

# Код программы

<https://github.com/KaynehacK/Java_LAB_4>