|  |  |  |
| --- | --- | --- |
| 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 №3  по дисциплине «Языки программирования»  Вариант 2 | | |
|  | | Работу выполнил  студент группы ДВБ7-ПР3-2023 НБ 2 курса  Румянцев В.И.  «11» ноября 2024 г. |
| Работу проверил  Ракина В.Д.  «\_\_\_» \_\_\_\_\_\_\_\_\_ 2024 г. |
| Пермь 2024 | | |

СОДЕРЖАНИЕ

[Задание 1.4. 5](#_Toc182270721)

[Текст задания 5](#_Toc182270722)

[Алгоритм решения 5](#_Toc182270723)

[Тестирование 5](#_Toc182270724)

[Код программы 5](#_Toc182270725)

[Задание 1.10. 6](#_Toc182270726)

[Текст задания 6](#_Toc182270727)

[Алгоритм решения 6](#_Toc182270728)

[Тестирование 6](#_Toc182270729)

[Код программы 6](#_Toc182270730)

[Задание 2.5. 7](#_Toc182270731)

[Текст задания 7](#_Toc182270732)

[Алгоритм решения 7](#_Toc182270733)

[Тестирование 8](#_Toc182270734)

[Код программы 8](#_Toc182270735)

[Задание 3.1. 9](#_Toc182270736)

[Текст задания 9](#_Toc182270737)

[Алгоритм решения 9](#_Toc182270738)

[Тестирование 9](#_Toc182270739)

[Код программы 9](#_Toc182270740)

[Задание 4.2. 10](#_Toc182270741)

[Текст задания 10](#_Toc182270742)

[Алгоритм решения 10](#_Toc182270743)

[Тестирование 10](#_Toc182270744)

[Код программы 10](#_Toc182270745)

[Задание 5.1. 11](#_Toc182270746)

[Текст задания 11](#_Toc182270747)

[Алгоритм решения 11](#_Toc182270748)

[Тестирование 11](#_Toc182270749)

[Код программы 11](#_Toc182270750)

[Задание 6.5. 12](#_Toc182270751)

[Текст задания 12](#_Toc182270752)

[Алгоритм решения 12](#_Toc182270753)

[Тестирование 12](#_Toc182270754)

[Код программы 12](#_Toc182270755)

[Задание 7.1. 13](#_Toc182270756)

[Текст задания 13](#_Toc182270757)

[Алгоритм решения 13](#_Toc182270758)

[Тестирование 13](#_Toc182270759)

[Код программы 14](#_Toc182270760)

[Задание 7.2. 15](#_Toc182270761)

[Текст задания 15](#_Toc182270762)

[Алгоритм решения 15](#_Toc182270763)

[Тестирование 15](#_Toc182270764)

[Код программы 15](#_Toc182270765)

[Задание 7.3. 16](#_Toc182270766)

[Текст задания 16](#_Toc182270767)

[Алгоритм решения 16](#_Toc182270768)

[Тестирование 16](#_Toc182270769)

[Код программы 16](#_Toc182270770)

[Задание 8.3. 17](#_Toc182270771)

[Текст задания 17](#_Toc182270772)

[Алгоритм решения 17](#_Toc182270773)

[Тестирование 17](#_Toc182270774)

[Код программы 17](#_Toc182270775)

# Задание 1.4.

## Текст задания

Измените сущность Дробь из задачи 1.5.5. Реализуйте следующие требования:

Дробь не может быть изменена после создания

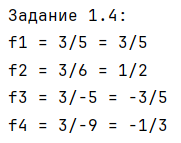
Необходимо корректно обрабатывать отрицательные значения. Учтите, что знаменатель не может быть отрицательным.

Продемонстрируйте работоспособность решения на примерах.

## Алгоритм решения

1. Сделал поля сущности Дробь финальными, чтобы исключить возможность их изменения после создания
2. Организовал корректное отображение отрицательных значений. В случае указания отрицательного знаменателя, числитель и знаменатель умножаются на -1 в конструкторе
3. Продемонстрировал работоспособность решения на примерах в классе Main

## Тестирование



## Код программы

https://github.com/Vladimir-Rumyantsev/JavaLabs/tree/master/src/LabWork3

# Задание 1.10.

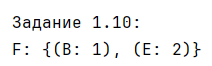
## Текст задания

Измените сущности из задачи 1.3.3. Гарантируйте, что между двумя городами может быть только одна прямая дорога (другой путь может быть проложен только транзитом через другие города). Города можно создавать с указанием заранее заданных путей, в любой момент времени можно добавить новую дорогу в любой город и удалить имеющуюся дорогу.

## Алгоритм решения

1. Переопределил методы добавления новых путей из города и конструктор создания города для гарантирования, что между двумя городами может быть лишь одна прямая дорога

## Тестирование



## Код программы

https://github.com/Vladimir-Rumyantsev/JavaLabs/tree/master/src/LabWork3

# Задание 2.5.

## Текст задания

Данная задача предполагает разработку новой сущности на основе той, что была получена в задаче 1.4.8 (Город). Сущность Город может быть доработана по своему усмотрению для более удобного использования.

Основная идея задачи в разработке такой сущности, которая будет представлять собой маршрут между двумя городами. Данный маршрут в любой момент времени можно получить как массив для дальнейшего использования. Сущность Маршрут имеет следующие характеристики:

• Имеет Город начала и Город конца пути.

• Инициализация Маршрута может быть выполнена только если указана точка начала и конца пути. Если указано null значение - то ошибка.

• Точку начала и конца можно изменить в любой момент времени, но они всегда должны существовать.

• Может вернуть массив Городов, представляющий собой маршрут из начала в конец. Массив содержит все Города (в порядке очереди) через которые надо пройти что бы попасть из Города начала в Город конца, причем и начало и конец также содержатся в этом массиве. Алгоритм формирования пути в данном случае не существенен, можно выбрать вариант со случайным путем, путем проходящим через наименьшее число городов, или самым дешевым путем. Если путь найти невозможно – возвращается пустой массив.

• Маршрут может быть приведен к строке, которая будет возвращать название всех городов маршрута в порядке очередности.

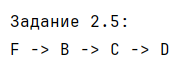
• Создание объекта и изменение точек начала и конца выполняется за константное время O(1).

Воспользуйтесь картой городов из задачи 1.3.3 и выведите маршрут из Города F в Город D

## Алгоритм решения

1. Создание класса Route с полями startCity и endCity.
2. Реализация конструктора для инициализации полей startCity и endCity с проверкой на null.
3. Реализация методов setStartCity и setEndCity для изменения начала и конца маршрута.
4. Реализация метода getRoute для получения массива городов, представляющего маршрут.
5. Реализация метода findRoute для поиска маршрута между городами.
6. Реализация метода toString для возврата строкового представления маршрута.
7. Создание объекта Route и вывод информации о маршруте на экран.

## Тестирование



## Код программы

https://github.com/Vladimir-Rumyantsev/JavaLabs/tree/master/src/LabWork3

# Задание 3.1.

## Текст задания

Измените сущность Дробь, полученную в задаче 2.1.4. Гарантируйте, что невозможно создать такой подвид дроби, который позволял бы создавать Дроби с изменяемым состоянием.

## Алгоритм решения

1. Обновление класса Fraction для гарантии, невозможности создания подвида дроби, который создавал бы объекты с изменяемым состоянием.

## Тестирование

## Код программы

https://github.com/Vladimir-Rumyantsev/JavaLabs/tree/master/src/LabWork3

# Задание 4.2.

## Текст задания

Измените сущность Дробь, полученную в задаче 2.3.1. Дробь должна быть подтипом класса Number. Данный класс входит в стандартную редакцию языка Java

## Алгоритм решения

1. Сделал сущность Дробь подтипом класса Number.

## Тестирование

## Код программы

https://github.com/Vladimir-Rumyantsev/JavaLabs/tree/master/src/LabWork3

# Задание 5.1.

## Текст задания

Разработайте метод, который принимает набор числовых значений и возвращает их сумму в вещественной форме. С использованием данного метода выполните следующие сложения:

• 2 + (3/5) + 2.3

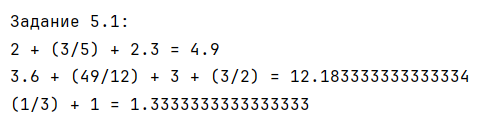
• 3.6 + (49/12) + 3 + (3/2)

• (1/3) + 1

## Алгоритм решения

1. Реализация метода sumNumbers для приема массива числовых значений и возврата их суммы в вещественной форме.
2. Создание массивов числовых значений.
3. Вызов метода sumNumbers для каждого массива и вывод результатов на экран.

## Тестирование



## Код программы

https://github.com/Vladimir-Rumyantsev/JavaLabs/tree/master/src/LabWork3

# Задание 6.5.

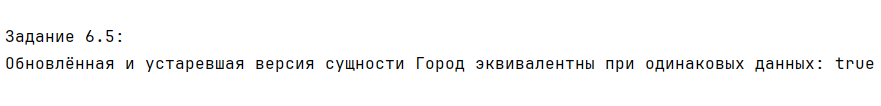
## Текст задания

Измените сущность Город, полученную в задаче 2.1.10. Переопределите метод сравнения объектов по состоянию таким образом, чтобы два Города считались одинаковыми тогда, когда у них одинаковый набор путей в другие города. Также, подвид Города из задачи 2.3.3 должен быть сравним с городом из задачи 2.1.10

## Алгоритм решения

1. Создал метод equals для сущностей Город и Путь
2. Убедился, что прежняя и обновлённая версия Города сравнимы через метод equals

## Тестирование



## Код программы

https://github.com/Vladimir-Rumyantsev/JavaLabs/tree/master/src/LabWork3

# Задание 7.1.

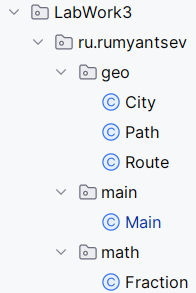
## Текст задания

Данная задача предполагает реорганизацию ранее написанных классов. Расположите все ранее написанные классы по пакетам таким образом, чтобы логически близкие классы оказались сгруппированы друг с другом. Имена пакетов должны иметь как минимум трехсоставную форму, вида: ru.surname.type. Вместо surname следует подставить свою фамилию, а вместо type подставить название логического блока. Например, классы описывающие точку, линию, ломаную линию, фигуру, квадрат, треугольник, круг и прямоугольник можно расположить в пакете ru.surname.geometry.

## Алгоритм решения

1. Создание пакетов для логически близких классов.
2. Размещение классов в соответствующих пакетах.
3. Проверка, что ни в одном другом пакете нет классов с точкой входа в исполнение программы.

## Тестирование



## Код программы

https://github.com/Vladimir-Rumyantsev/JavaLabs/tree/master/src/LabWork3

# Задание 7.2.

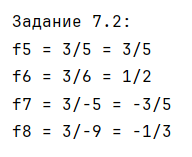
## Текст задания

Создайте пакет ru.surname.main (вместо surname необходимо подставить собственную фамилию) в котором расположить класс с точкой входа в исполнение программы (public static void main). Также следует проверить, что ни в одном другом пакете нет классов имеющих точку входа в исполнение программы. В этом же пакете необходимо расположить класс (или интерфейс) с методами из задач блока 2.5 и продемонстрировать их работоспособность.

## Алгоритм решения

1. Создание пакета ru.rumyantsev.main.
2. Размещение класса с точкой входа в исполнение программы в пакете ru.rumyantsev.main.
3. Проверка, что ни в одном другом пакете нет классов с точкой входа в исполнение программы.

## Тестирование



## Код программы

https://github.com/Vladimir-Rumyantsev/JavaLabs/tree/master/src/LabWork3

# Задание 7.3.

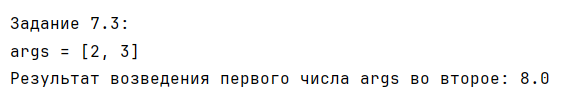
## Текст задания

Создайте метод принимающий две строки, в которых будут записаны числа X и Y. Возвращает метод результат возведения X в степень Y. Для преобразования строки в число следует использовать метод Integer.parseInt, а для возведения в степень метод Math.pow. Вызовите разработанный метод передав туда параметры командной строки полученные точкой входа в программу. Реализуйте метод так, что бы для возведения в степень и преобразования строки использовались короткие имена статических методов.

## Алгоритм решения

1. Реализация метода power для приема двух строковых значений и возврата результата возведения одного числа в степень другого.
2. Вызов метода power с параметрами командной строки и вывод результата на экран.

## Тестирование



## Код программы

https://github.com/Vladimir-Rumyantsev/JavaLabs/tree/master/src/LabWork3

# Задание 8.3.

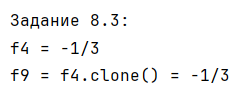
## Текст задания

Измените сущность Дробь из 2.6.1. Переопределите метод клонирования, унаследованный от класса Object, таким образом, чтобы при его вызове возвращался новый объект Дроби, значения полей которого будут копиями оригинальной Дроби.

## Алгоритм решения

1. Обновление класса Fraction для переопределения метода clone.
2. Реализация метода clone для возврата нового объекта Fraction с копиями полей оригинальной дроби.
3. Создание объекта Fraction и вызов метода clone.
4. Вывод результатов клонирования на экран.

## Тестирование



## Код программы

https://github.com/Vladimir-Rumyantsev/JavaLabs/tree/master/src/LabWork3