Санкт-Петербургский государственный политехнический университет Институт компьютерных наук и кибербезопасности

Кафедра «Информационные и управляющие системы»

**Курсовая работа**

По дисциплине «Объектно-ориентированное программирование»

Выполнил студент гр. 20030 Богданов В.В.

Руководитель

Старший преподаватель Маслаков А.П.

Санкт-Петербург

2024

**Введение**

Данный проект представляет собой приложение с графическим интерфейсом, позволяющий запускать лабораторные работы из единого меню. Приложение написано на Java с использованием Swing.

Перечень выполненных работ включает следующее:

1. Создание класса App, который наследуется от JFrame и представляет главное окно приложения.

2. В конструкторе класса App инициализируются элементы пользовательского интерфейса: кнопки для каждой лабораторной работы и кнопка "Выход".

3. Для каждой кнопки лабораторной работы добавлен (ActionListener), который при нажатии кнопки создает экземпляр соответствующего класса лабораторной работы и отображает его на экране.

4. Для кнопки "Выход" добавлен listener, который завершает выполнение программы при ее нажатии.

5. В методе main создается экземпляр класса App и отображается на экране.

Данный код создает простой GUI-интерфейс для запуска различных лабораторных работ, каждая из которых представлена своим собственным классом.

**Описание задания**

**Часть 1.** В компьютерной игре герой (класс Hero) может перемещаться между двумя точками (метод move) различными способами: идти пешком, ехать на лошади, лететь и т. п. Реализовать классы, позволяющие пользователю выбирать и менять в ходе выполнения программы способ перемещения героя, используя паттерн “стратегия” (strategy). Продемонстрировать работу реализованных классов.

**Часть 2.** Написать консольное приложение, которое:

a. Считывает из текстового файл размерность матрицы N\*N.

b. Создаёт и заполняет матрицу случайными числами от -N до N.

c. Последовательно поворачивает матрицу на 90, 180 и 270 градусов

против часовой стрелки и делит каждый элемент на сумму

соседних.

d. Каждую из трёх получившихся матриц вывести в общий файл

Требования к обработке исключительных ситуаций:

a. контролировать состояние потоков ввода/вывода (отсутствие

записи в файле, недопустимые значения, etc);

b. генерировать и обрабатывать исключение при некорректных

математических операциях;

c. логировать исключение при нехватке памяти;

d. реализовать собственные классы исключений для случаев

● деление на 0

● файл не существует/нет доступа к файлу

● N > 1\_000\_000

**Часть 3.** Реализовать программу-переводчик.

• При запуске программы выполняется чтение словаря в

следующем формате:

o слово или выражение | перевод

• Затем читается входной файл в формате \*.txt

o файл может быть пустым

o файл может содержать текст в любом формате

• Выполняется перевод

• Перевод осуществляется по следующим правилам:

o регистр букв игнорируется

o если искомого слова нет в словаре – выводится без перевода

o если есть несколько подходящих вариантов, выбирается

вариант максимальной длиной левой части. Например:▪ словарь:

• look | смотреть

• look forward | ожидать

▪ текст:

• dog look to the window, dog look forward

▪ перевод:

• dog смотреть to the window, dog ожидать

• Результат перевода выводится в консоль

**Часть 4.** Написать аннотацию с целочисленным параметром. Создать класс, содержащий методы со всеми возможными модификаторами доступа (2–3 каждого вида), аннотировать любые из них. Вызвать из другого класса все аннотированные методы, которые имеют модификатор private или protected столько раз, сколько указано в параметре аннотации.

**Часть 5.** С использованием только Stream API реализовать следующие методы:

● метод, возвращающий среднее значение списка целых чисел;

● метод, приводящий все строки в списке в верхний регистр и

добавляющий к ним префикс «\_new\_»;

● метод, возвращающий список квадратов всех встречающихся

только один раз элементов списка;

● метод, принимающий на вход коллекцию строк и возвращающий

все строки, начинающиеся с заданной буквы, отсортированные по

алфавиту;

● метод, принимающий на вход коллекцию и возвращающий её

последний элемент или кидающий исключение, если коллекция

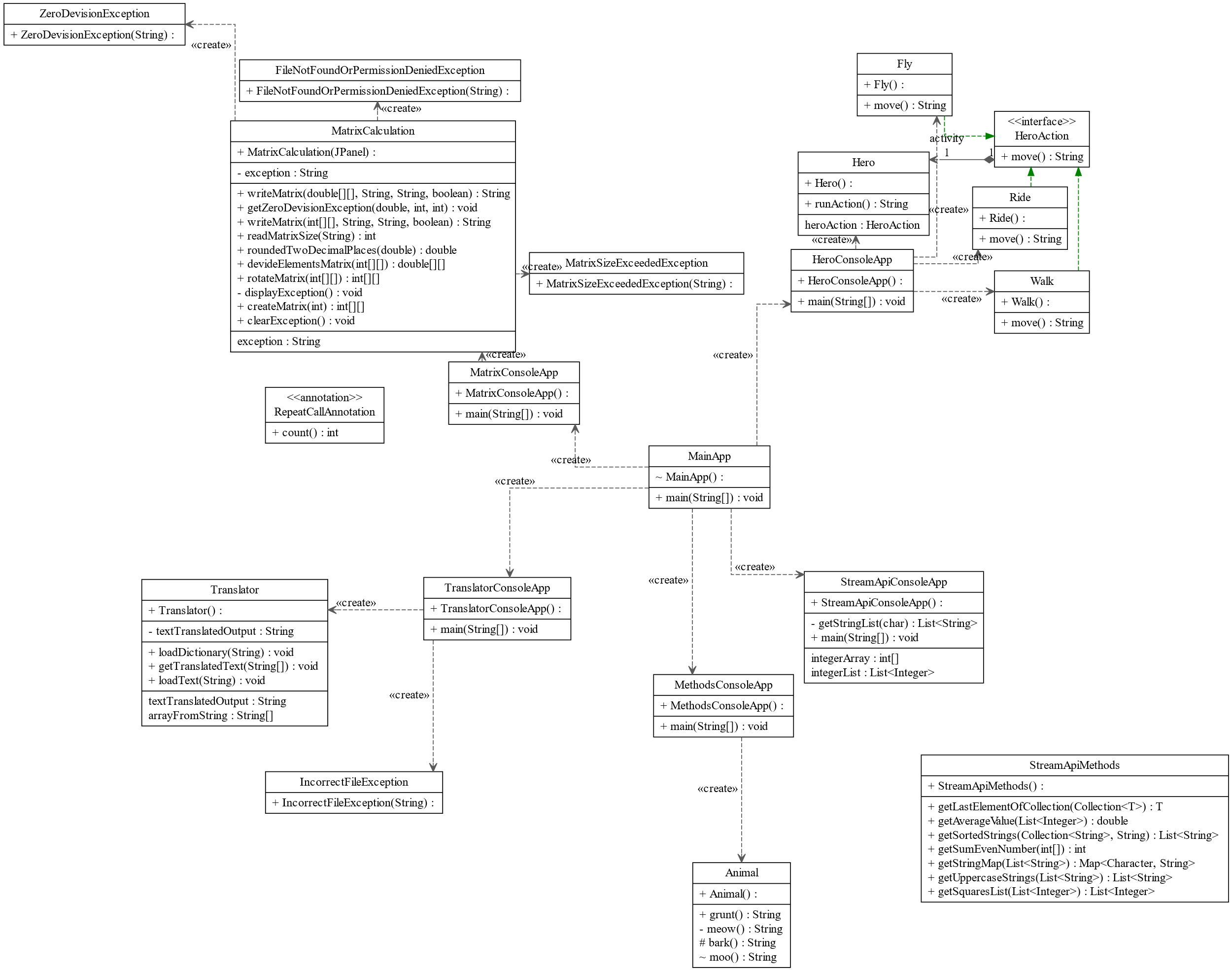
пуста;

● метод, принимающий на вход массив целых чисел, возвращающий

сумму чётных чисел или 0, если чётных чисел нет;

● метод, преобразовывающий все строки в списке в Map, где первый

символ – ключ, оставшиеся – значение;

**Диаграммы классов:**