|  |  |  |
| --- | --- | --- |
| 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 №6  по дисциплине «Языки программирования»  Вариант 6 | | |
|  | | Работу выполнил  студент группы ПМИ-2/2023 2 курса  Волков А.Н.  «\_\_\_» \_\_\_\_\_\_\_\_2025 г. |
| Работу проверил  Ракина В.Д.  «\_\_\_» \_\_\_\_\_\_\_\_\_ 2025 г. |
| Пермь 2024 | | |

Оглавление

[Задание 1.3. Сравнимое 3](#_Toc186087340)

[Текст задания 3](#_Toc186087341)

[Алгоритм решения 3](#_Toc186087342)

[Тестирование 3](#_Toc186087343)

[Задание 2.4. Заполнение списка 4](#_Toc186087344)

[Текст задания 4](#_Toc186087345)

[Алгоритм решения 4](#_Toc186087346)

[Тестирование 4](#_Toc186087347)

[Задание 3.1. Функция 5](#_Toc186087348)

[Текст задания 5](#_Toc186087349)

[Алгоритм 5](#_Toc186087350)

[Тестирование 5](#_Toc186087351)

[Задание 3.2. Фильтр 6](#_Toc186087352)

[Текст задания 6](#_Toc186087353)

[Алгоритм 6](#_Toc186087354)

[Тестирование 6](#_Toc186087355)

[Задание 3.3. Сокращение 7](#_Toc186087356)

[Текст задания 7](#_Toc186087357)

[Алгоритм 7](#_Toc186087358)

[Тестирование 7](#_Toc186087359)

[Задание 3.4. Очередь 8](#_Toc186087360)

[Текст задания 8](#_Toc186087361)

[Алгоритм 8](#_Toc186087362)

[Тестирование 8](#_Toc186087363)

[Ссылка на GitHub 9](#_Toc186087364)

Задание 1.2. Валидация

## Текст задания

Разработайте утилитный метод validate, позволяющий проверять объекты на корректность их состояния. На вход данный метод принимает два объекта: один типа Object и второй типа Class. В рамках задачи:

• Первый объект представляет собой тестируемую сущность. Необходимо проверить, удовлетворяет ли она определенным критериям.

• Второй объект представляет собой набор тестов, реализованных как методы, принимающих тестируемый объект.

• Если тест пройден успешно – метод заканчивает работу без дополнительной информации.

• Если объект не прошел тест – должно быть выброшено исключение с информацией о проблеме.

## Алгоритм решения

1. Создать утилитарный метод validate, принимающий объект (Object) и класс (Class<?>).
2. В указанном классе найти все методы, принимающие объект для тестирования.
3. Вызывать найденные методы и проверять результат.
4. Если тест не пройден, выбрасывать исключение.

## Тестирование

 

# Задание 2.1. @Invoke

## Текст задания

Разработайте аннотацию @Invoke, со следующими характеристиками:

• Целью может быть только МЕТОД

• Доступна во время исполнения программы

• Не имеет свойств

Проаннотируйте какой-либо метод данной аннотацией.

## Алгоритм решения

1. Создать аннотацию с @Target(ElementType.METHOD) и @Retention(RetentionPolicy.RUNTIME).
2. Проаннотировать тестовый метод.

## Тестирование



# Задание 2.2. @Default

## Текст задания

Разработайте аннотацию @Default, со следующими характеристиками:

• Целью может быть ТИП или ПОЛЕ

• Доступна во время исполнения программы

• Имеет обязательное свойство value типа Class Проаннотируйте какой-либо класс данной аннотацией.

## Алгоритм

1. Создать аннотацию с @Target({ElementType.TYPE, ElementType.FIELD}), @Retention(RetentionPolicy.RUNTIME).
2. Добавить обязательное свойство value типа Class <?>.
3. Проаннотировать тестовый класс

## Тестирование



# Задание 2.3. @ToString

## Текст задания

Разработайте аннотацию @ToString, со следующими характеристиками:

• Целью может быть ТИП или ПОЛЕ

• Доступна во время исполнения программы

• Имеет необязательное свойство valuec двумя вариантами значений: YES или NO

• Значение свойства по умолчанию: YES Проаннотируйте какой-либо класс данной аннотацией.

## Алгоритм

1. Создать аннотацию с @Target({ElementType.TYPE, ElementType.FIELD}), @Retention(RetentionPolicy.RUNTIME).
2. Добавить необязательное свойство value с возможными значениями YES и NO (по умолчанию YES).
3. Проаннотировать тестовый класс

## Тестирование



# Задание 2.4. @Validate

## Текст задания

Разработайте аннотацию @Validate, со следующими характеристиками:

• Целью может быть ТИП или АННОТАЦИЯ

• Доступна во время исполнения программы

• Имеет обязательное свойство value, типа Class[] Проаннотируйте какой-либо класс данной аннотацией.

## Алгоритм

1. Создать аннотацию с @Target({ElementType.TYPE, ElementType.ANNOTATION\_TYPE}), @Retention(RetentionPolicy.RUNTIME).
2. Добавить обязательное свойство value типа Class <?> [].
3. Проаннотировать тестовый класс

## Тестирование



# Задание 2.5. @Two

## Текст задания

Разработайте аннотацию @Two, со следующими характеристиками:

• Целью может быть ТИП

• Доступна во время исполнения программы

• Имеет два обязательных свойства: first типа String и second типа int Проаннотируйте какой-либо класс данной аннотацией.

## Алгоритм

1. Создать аннотацию с @Target(ElementType.TYPE), @Retention(RetentionPolicy.RUNTIME).
2. Добавить два обязательных свойства: first (String) и second (int).
3. Проаннотировать тестовый класс

## Тестирование



# Задание 2.6. @Cache

## Текст задания

Разработайте аннотацию @Cache, со следующими характеристиками:

• Целью может быть ТИП

• Доступна во время исполнения программы

• Имеет необязательное свойство value, типа String[]

• Значение свойства по умолчанию: пустой массив Проаннотируйте какой-либо класс данной аннотацией.

## Алгоритм

1. Создать аннотацию с @Target(ElementType.TYPE), @Retention(RetentionPolicy.RUNTIME).
2. Добавить необязательное свойство value типа String[] (по умолчанию пустой массив).
3. Проаннотировать тестовый класс

## Тестирование



# Задание 3.2. Валидация

## Текст задания

Измените метод validate из задачи 1.2. Теперь метод должен принимать произвольный набор объектов по ссылкам типа Object. В рамках задачи:

• Классы объектов должны быть проаннотированы с помощью @Validate (из предыдущей задачи).

• Аннотация должна описывать набор тестов, которые должен пройти данный объект.

• Если тест пройден успешно – метод заканчивает работу без дополнительной информации.

• Если объект не прошел тест – должен быть выброшен Exception с информацией о проблеме.

## Алгоритм

1. Принимать массив Object... objects.
2. Проверять, аннотирован ли класс @Validate.
3. Из аннотации получать классы-тесты и вызывать их методы.
4. Выбрасывать исключение при некорректных данных

## Тестирование

 

# Ссылка на GitHub

<https://github.com/kuv4lda/PSU_Java/blob/main/lab6/>