# Домашнее задание к занятию «3.1. Исключения»

В качестве результата пришлите ссылки на ваши GitHub-проекты в личном кабинете студента на сайте [netology.ru](https://netology.ru).

**Важно**: ознакомьтесь со ссылками на главной странице [репозитория с домашними заданиями](https://github.com/netology-code/kt-homeworks/blob/master/README.md).

Если у вас что-то не получилось, то оформляйте Issue [по установленным правилам](https://github.com/netology-code/kt-homeworks/blob/master/report-requirements.md).

Нужно сделать все задачи в **одном** репозитории.

## Как сдавать задачи

1. Создайте на вашем компьютере Gradle-проект.
2. Инициализируйте в нём пустой Git-репозиторий.
3. Добавьте в него готовый файл [.gitignore](https://github.com/netology-code/kt-homeworks/blob/master/.gitignore).
4. Добавьте в этот же каталог остальные необходимые файлы.
5. Сделайте необходимые коммиты.
6. Создайте публичный репозиторий на GitHub и свяжите свой локальный репозиторий с удалённым.
7. Сделайте пуш и удостоверьтесь, что ваш код появился на GitHub.
8. Ссылку на ваш проект отправьте из личного кабинета на сайте [netology.ru](https://netology.ru).
9. Необязательные задачи можно не сдавать — это не повлияет на получение зачёта. В этом ДЗ все задачи обязательные.

## Задача №1 - PostNotFoundException

Мы продолжаем развивать наш сервис с постами и решили внедрить в него возможность комментирования постов: [createComment](https://vk.com/dev/wall.createComment)\*.

Примечание\*: по традиции есть [сохранённая версия](https://github.com/netology-code/kt-homeworks/blob/master/07_exceptions/assets/wall.createComment.pdf)

В чём суть? Вы создаёте member function внутри WallService:

class WallService {

private var posts = emptyArray<Post>()

private var comments = emptyArray<Comment>()

fun createComment(comment: Comment) {

TODO()

}

}

Как работает эта функция: она сначала проверяет, существует ли в массиве posts пост с id равным тому, что в комментарии в свойстве postId. Если существует, то добавляет комментарий в массив comments.

А вот если не существует - выкидывает исключение PostNotFoundException.

Спроектировать класс для комментариев вам нужно самостоятельно на базе описания типа [Comment](https://vk.com/dev/objects/comment). По традиции, есть [сохранённая копия](https://github.com/netology-code/kt-homeworks/blob/master/07_exceptions/assets/comment.pdf).

Итог: у вас должен быть репозиторий на GitHub, в котором расположен ваш Gradle-проект.

## Задача №2 - JUnit

Как вы уже знаете - без тестов никуда. А это значит, что нам нужно написать целых два теста на созданную нами функцию:

1. Функция отрабатывает правильно, если мы добавляем комментарий к правильному посту.
2. Функция выкидывает исключение, если мы пытаемся добавить комментарий к несуществующему посту.

Тестировать, что какой-то код выкидывает исключение, достаточно просто:

@Test(expected = PostNotFoundException::class)

fun shouldThrow() {

// здесь код с вызовом функции, которая должна выкинуть PostNotFoundException

}

**Важно**: после ваших обновлений, функциональность WallService должна оставаться работоспособной (т.е. автотесты должны проходить).

Итог: у вас должен быть репозиторий на GitHub, в котором расположен ваш Gradle-проект. Автотесты также должны храниться в репозитории.

## Задача №3 - Жалобы\*

**Важно**: это не обязательная задача. Её выполнение не влияет на получение зачёта по ДЗ.

В социальных сетях есть разные люди, которые оставляют разные комментарии.

Неплохо бы внедрить [функцию для уведомления о нехороших комментариях](https://vk.com/dev/wall.reportComment). По традиции, есть [сохранённая копия](https://github.com/netology-code/kt-homeworks/blob/master/07_exceptions/assets/wall.reportComment.pdf).

Что нужно сделать:

1. Реализовать саму функцию (для простоты храните эти репорты в отдельном массиве).
2. Подумать в каких случаях и какие нужно выкидывать исключения\*.
3. Написать автотесты.

Подсказка\*

**Важно**: после ваших обновлений, функциональность WallService должна оставаться работоспособной (т.е. автотесты должны проходить).

Итог: у вас должен быть репозиторий на GitHub, в котором расположен ваш Gradle-проект. Автотесты также должны храниться в репозитории.