# Вводная

Разработка через тестирование или TDD (Test-Driven Development) одна из методологий разработки программного обеспечения. Она подразумевает написание Unit-тестов прежде, собственно, основного функционала. В данной работе необходимо реализовать программную логику для уже подготовленных методов Unit-тестирования.

Сама программа - некое “процессинговым” приложение, обрабатывающее платежи между счётами пользователей. Методы

Unit-тестирования проверяют требуемый функционал приложения: то есть переводы с счёта на счёт, создание счёта, регистрация пользователя и т.п.

Для хранения данных предлагается использовать In-memory хранилище, иными словами - словари HashMap.

# Задание 1.

На основе тестов, содержащихся в проекте по пути

src/test/java/com/learning/processing

реализовать основную логику приложения по следующим функциональным требованиям:

* Пользователь должен иметь возможность регистрации и авторизации

в приложение;

* Пользователь должен иметь возможность создания одного или нескольких счетов, а также их закрытие;
* Пользователь должен иметь возможность перемещения денежных средств между своими счетами или счетами других пользователей;

Необходимо учитывать следующие требования к самой реализации:

* Все классы с логикой (то есть - не описание данных типа User или Account) должны быть реализацией соответствующего интерфейса (например UserService -> UserServiceImpl);
* Хранение данных о пользователях осуществляется с помощью

словаря HashMap<Integer, User>, где ключом будет идентификатор пользователя, а значением - объект (ссылка на объект) User;

* Хранение данных о счётах осуществляется с помощью словаря

HashMap<Integer, Account>, где ключом выступает идентификатор

счёта, а значением - объект Account;

* Для ассоциации счёта и пользователя, в структуре данных Account должно быть предусмотрено поле userId, с идентификатором пользователя - владельца счёта;
* Операции с поиском данных должны осуществляться с помощью

методов StreamAPI (например, поиск счётов по значению поля userId);

* В качестве UI может выступать как JavaFX, так и консольное окно.

После реализации основного функционала, требуемого для Unit-тестирования, произвести проверку работоспособности программы с помощью методов из тестовых классов. При необходимости дополнить классы другими методами для проведения тестирования.

Результаты тестирования записать в таблицу вида:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название метода | Описание теста | Значение аргументов для тестирования | Ожидаемый результат | Полученный результат |

Таблицу приложить к отчёту.

# Что должно быть в отчёте

В отчёте должно быть следующее:

1. Текст задания;
2. Таблица результатов тестирования;
3. После таблицы для каждого теста приложить соответствующий листинг кода.