Что сдавать?

Архив содержащий следующее:

1. Полный проект с тестами и необходимыми зависимостями (допускаются указанные в сборочном скрипте, но они должны присутствовать);

2. Файл содержащий список обнаруженных несоответствий (если были обнаружены);

3. Отчет о покрытии как минимум класса AccountManager.

Задание

Провести анализ и разработать набор тестов для класса `AccountManager` согласно приложенной спецификации с использованием заглушек при взаимодействии с `IServer` и `IPasswordEncoder`. При этом требуется обеспечить проверки как отдельных методов класса согласно спецификации и допустимых вариантов ответа сервера, так и сценариев работы, описанных ниже (в виде отдельных тестов):

1. Сценарий 1. Пользователь (user) проводит попытку авторизации в системе управления аккаунтами с указанием некорректного логина. После этого он проводит вторую попытку авторизации с указанием корректного логина и неправильного пароля. С третьей попытки пользователь авторизуется и делает запрос баланса. После получения значения проводится попытка внести на счет 100 единиц. Полученное значение сравнивается с ожидаемым.

2. Сценарий 2. Пользователь (user) проводит успешную авторизацию. Проводится попытка снятия 50 единиц (неудачная). Делается запрос на количество средств на счету. Проводится внесение 100 единиц. Проводится снятие 50 единиц с указанием некорректного номера сессии (неудачное). Проводится снятие 50 единиц с правильным номером сессии. Проводится выход из системы (logout).

PS. Стоит проверить как наличие обработки тех ответов что должны быть обработаны, так и то что при наличии некорректного ответа будет возвращено определенное значение.

Приложение 1. Тестирование (с заглушками) класса `AccountManager`

`AccountManager`

Класс `AccountManager` представляет собой класс для работы с пользовательскими аккаунтами. В классе содержится экземпляр `IServer`, используемый для связи с сервером а также `IPasswordEncoder` используемый при шифровании пароля. При обмене данными с сервером используется экземпляры `ServerResponse`.

Все методы обрабатывающие ответы сервера могут возвращать в ответ INCORRECT_RESPONSE в ситуации когда ответ сервера не попадает в список допустимых для соответствующего метода интерфейса.

Для доступа к функциям изменения баланса используются методы `withdraw` и `deposit`. Получение текущего баланса производится при помощи метода `getBalance`. При этом для работы с этими методами требуется авторизация на сервере. При вызове методов с отсутствующей авторизацией будет возвращаться `NOT_LOGGED`.

Авторизация производится при помощи метода `callLogin`, после завершения операций требуется вызвать `callLogout` для разлогинивания.

Большая часть методов возвращает объекты класса `AccountManagerResponse`, в которых содержится код успешности завершения запроса и передаваемые данные.

Описание методов:
`init(IServer s, IPasswordEncoder encoder)` — проводит инициализацию объекта для связи с сервером и шифратора пароля. Принимает объект `IServer` в качестве первого аргумента и `IpasswordEncoder` в качестве второго.

`callLogin(String login, String password)` — используется при авторизации пользователя на сервере, возвращает номер сессии при успешном завершении (`SUCCEED`). Пароль передается в метод открытым образом, но перед отправкой на сервер проводится шифрование с использованием passEncoder. При возникновении ошибки при шифровании возвращается код ENCODING_ERROR. Считается что сервер хранит зашифрованную версию пароля и сравнивает переданное значение с ней.
PS ремарка. Вне рамок данной работы не стоит хранить ваши пароли в виде строк — по сути все имеющиеся в программе строки достаточно просто могут быть просмотрены в профайлерах и потому потенциально пароль может немного утечь на сторону. Насколько понимаю, при небходимости поработать с паролями их хранят в других представлениях — например, byte массивах.
Возвращаемые коды: `ALREADY_LOGGED`, `ENCODING_ERROR`, `NO_USER_INCORRECT_PASSWORD`, `UNDEFINED_ERROR`, `SUCCESS`

`callLogout(String user, long session)` — метод для окончания рабочей сессии. Возвращаемые коды: `NOT_LOGGED`, `INCORRECT_SESSION`, `UNDEFINED_ERROR`, `SUCCESS`

`withdraw(String login, long session, double amount)` — метод для списания средств со счета. При успешном снятии, а также при ответе `NO_MONEY` также содержит значение текущей суммы на счету. Коды возвращения: `NOT_LOGGED`, `INCORRECT_SESSION`, `UNDEFINED_ERROR`, `NO_MONEY`, `SUCCESS`

`deposit(String login, long session, double amount)` — метод для внесения средств на счет. При успешном завершении также возвращает количество средств на счету. Возможные коды: `NOT_LOGGED`, `INCORRECT_SESSION`, `UNDEFINED_ERROR`, `SUCCESS`

`getBalance(String login, long session)` — метод для получения текущего баланса на счету. При успешном завершении возвращает величину баланса в ответе. Коды ответов: `NOT_LOGGED`, `INCORRECT_SESSION`, `UNDEFINED_ERROR`, `SUCCESS`

`IServer`

Интерфейс доступа к серверу. Предоставляет методы для авторизации и запросов, связанных с балансом. Методы возвращают объекты класса `ServerResponse`.
Доступны следующие методы:

`login(String userName, String mdPass)` - авторизация пользователя. При успешном завершении возвращает в ответе номер сессии. Номер сессии генерируется случайным образом. При получении зашифрованного пароля mdPass сравнивает его с хранимым на сервере зашифрованным паролем. Может возвращать следующие коды: `ALREADY_LOGGED`, `NO_USER_INCORRECT_PASSWORD`, `UNDEFINED_ERROR`, `SUCCESS`

`logout(long id)` — метод для окончания рабочей сессии. Принимает идентификатор сессии. Может возвращать следующие коды: `NOT_LOGGED`, `UNDEFINED_ERROR`, `SUCCESS`

`withdraw(long id, double balance)` — метод для снятия средств со счета. При отсутствии соответствующей суммы возвращает `NO_MONEY`. При успешном завершении, а также при возвращении `NO_MONEY` также возвращает текущий баланс в данных ответа. Возвращаемые коды: `NOT_LOGGED`, `UNDEFINED_ERROR`, `NO_MONEY`, `SUCCESS`.

`deposit(long id, double balance)` — метод для внесения средств на счет. При успешном завершении возвращает текущий баланс в данных ответа. Возвращаемые коды: `NOT_LOGGED`, `UNDEFINED_ERROR`, `SUCCESS`.

`getBalance(long id)` — метод для получения количества средств на счету. Сумма возвращается в данных ответа при успешном завершении запроса. Возвращаемые коды: `NOT_LOGGED`, `UNDEFINED_ERROR`, `SUCCESS`.

ServerResponse

Класс используется для хранения и передачи данных о результатах запроса на сервер и включает код успешного завершения\ошибки и дополнительные данные.
Среди возможных результатов запросов

- `SUCCESS = 0` — успешное завершение запроса
- `UNDEFINED_ERROR = 1` — не распознанная ошибка
- `ALREADY_LOGGED = 2` — попытка зайти на сервер в то время как с этим логином уже была успешная попытка залогиниться
- `NOT_LOGGED = 3` — возвращается при попытке выполнить операцию при отсутствии активной сессии не сервере
- `NO_USER_INCORRECT_PASSWORD = 4` — возвращается при попытке начать рабочую сессию с некорректным паролем или при отсутствии пользователя с соответствующим именем
- `NO_MONEY = 5` — возвращается при попытке снять деньги при отсутствии соответствующей суммы на счету.

AccountManagerResponse

Класс используется для хранения и передачи данных о результатах запроса извне в менеджер пользовательских аккаунтов.
Содержит код успешности завершения запроса\ошибки, а также может содержать дополнительные данные.

Возможные коды:

- `SUCCEED = 0` — успешное завершение запроса
- `ALREADY_LOGGED = 1` — возвращается при попытке залогиниться при уже активной сессии
- `NOT_LOGGED = 2` — возвращается при попытке вызова методов в незалогиненном состоянии
- `NO_USER_INCORRECT_PASSWORD = 3` — возвращается при отсутствующем логине или некорректном пароле
- `INCORRECT_RESPONSE = 4` — возвращается, когда от сервера пришел некорректный ответ
- `UNDEFINED_ERROR = 5` — возвращается при возникновении непредусмотренной ошибки
- `INCORRECT_SESSION = 6` — возвращается при попытке запроса с указанием некорректного номера сессии
- `NO_MONEY = 7` — возвращается при попытке снять средства при отсутствии нужного количества на счету
- `ENCODING_ERROR = 8` — возвращается при возникновении ошибки шифрования

`IPasswordEncoder`

Интерфейс используется для получения шифрованной версии пароля.

`makeSecure(String password)` — метод для шифрования переданного открытым образом пароля. При успешном завершении возвращает зашифрованную версию пароля которая удовлетворяет следующим правилам:

- возвращаемое значение не должно соответствовать исходной строке;
- для двух разных переданных паролей должен в большинстве случаев возвращаться разный результат шифрования (PS не стоит воспринимать этот пункт слишком строго, достаточно что для многих случаев он будет выполняться);
- при последовательных запусках на одном и том же аргументе будет возвращаться аналогичное значение.

При неудаче шифрования или передаче null в качестве аргумента выбрасывает исключение `java.lang.NullPointerException.`