Задача 1

Скачайте архив https://assets.codeforces.com/files/943c97ce9320d0ea9/hw3.zip. Откройте содержимое как проект в IDEA, выбрав pom.xml при открытии проекта. (Использовать IDEA не обязательно, но крайне рекомендуется). Запустите результат в Tomcat по порту 8080. Убедитесь, что страница http://localhost:8080/index.html отображается.

В настоящий момент, если делать правки в статике прям в IDEA, то по F5 ресурсы не обновляются. Модифицируйте StaticServlet таким образом, чтобы он сначала смотрел в нужную подпапку src вашего проекта, и если файл найден, то возвращал его. В getServletContext().getRealPath("/static" + uri) пусть сервлет смотрит во вторую очередь, если не получилось найти файл в соответствующей подпапке в src.

Убедитесь что проходят все три теста из TestsA.

Задача 2

Скачайте архив https://assets.codeforces.com/files/943c97ce9320d0ea9/hw3.zip. Откройте содержимое как проект в IDEA, выбрав pom.xml при открытии проекта. Запустите результат в Tomcat по порту 8080. Убедитесь, что страница http://localhost:8080/index.html отображается.

Часто стараются сократить количество ресурсов, подгружаемых с HTML-страниц. Обратите внимание, что index.html пытается загрузить странный ресурс с путём css/r.css+css/g.css+css/b.css. Пока у него не получается это сделать. Ваша задача добавить в StaticServlet функциональность, чтобы можно было в рамках одного запроса загружать множество статических файлов. Файлы в пути будут соединены символом плюс. Определяйте MIME-тип ответа на основании первого из запрошенных файлов. В ответе вы должны просто конкатенировать запрошенные файлы.

После правильной реализации на странице http://localhost:8080/index.html будут отображены три цветных блока. Убедитесь что проходят все тесты из TestsB.

Задача 3

Скачайте архив https://assets.codeforces.com/files/943c97ce9320d0ea9/hw3.zip. Откройте содержимое как проект в IDEA, выбрав pom.xml при открытии проекта. Запустите результат

в Tomcat по порту 8080. Убедитесь, что страница http://localhost:8080/index.html отображается.

Ваша задача добиться работоспособности страницы http://localhost:8080/messages.html

Для этой страницы написан фронтенд, всё что осталось реализовать — один сервлет для обработки запросов. Должны корректно обрабатываться следующие запросы (все запросы передаются методом POST, возвращают результат в JSON):

- /message/auth: ожидается опциональный параметр user, если он задан, то сохранить в сессию имя текущего пользователя (то, что пришло в параметре user), в любом случае вернуть имя пользователя (нового, если был только что установлен) или пустую строку, если не авторизован;
- /message/findAll: вернуть все сообщения в виде массива пар с ключами user и text (например, «`[{"user":"mike","text":"test"},{"user":"lena","text":"hi"}]`»). Сообщения надо возвращать в правильном порядке.
- /message/add: ожидается параметр text, добавить сообщение от имени текущего пользователя (взять из сессии) с текстом text.

Для вывода результата всюду используйте JSON. Не забудьте установить у ответа Content-Type в значение "application/json". Для простого форматирования объекта в JSON рекомендуется использовать библиотеку Gson. Для этого добавьте зависимость в pom.xml от артефакта groupId=com.google.code.gson, artifactId=gson, version=2.9.1 и используйте код типа такого:

```
String json = new Gson().toJson(objectToConvert);

response.getWriter().print(json);

response.getWriter().flush();
```

Задача 4

Скачайте архив https://assets.codeforces.com/files/943c97ce9320d0ea9/hw3.zip. Откройте содержимое как проект в IDEA, выбрав pom.xml при открытии проекта. Запустите результат в Tomcat по порту 8080. Убедитесь, что страница http://localhost:8080/index.html отображается.

Ваша задача написать фильтр CaptchaFilter, который будет перехватывать все GET-запросы, если в текущей сессии не отмечено, что пройдена каптча.

В таком случае фильтр должен загадать случайное число от 100 до 999, записать это в сессию (ожидаемый ответ) и простейшую форму с картинкой и полем для ввода ответа. Картинку по тексту можно сгененировать с помощью кода, который приложен в проекте (ImageUtils.java). Пользователь отправляет форму, фильтр ловит ответ и если ответ совпал с ожидаемым, то помечает в сессии, что каптча пройдена и больше не перехватывает запросы. Иначе перезагадывает число и опять показывает форму.

Таким образом, вы должны написать один фильтр CaptchaFilter и добавить его в web.xml для перехвата всех запросов.