

Проектирование программного обеспечения (практика)

Практика 1: Введение, задача про CLI

Юрий Литвинов
yurii.litvinov@gmail.com

17.01.2022

Формальности

- ▶ В конце курса оценка
 - ▶ Домашние работы
 - ▶ Дедлайны (-50% баллов за пропуск)
 - ▶ Работа в аудитории
 - ▶ Индивидуальные и групповые задачи
 - ▶ Связанные цепочки заданий
- ▶ Материалы курса, условия заданий: <https://bit.ly/sd-practice-2022>
- ▶ Коммуникации — в чате курса в Telegram
 - ▶ Ссылка на пуллреквест в собственный репозиторий на GitHub
 - ▶ Репозиторий написать мне в личку в Telegram
 - ▶ Задач будет несколько, так что выкладывать лучше так, чтобы их все было можно смердить

Критерии оценивания

- ▶ Меньше 60% заданий – 0
- ▶ От 60% до 100% — линейная шкала от 2 до 10 баллов
 - ▶ то есть, ровно 60% заданий — 2 балла
 - ▶ 80% заданий — 6 баллов
 - ▶ 100% заданий — 10 баллов
- ▶ Задачи оцениваются по 10 баллов
- ▶ Оценки за работу на паре с небольшим весом (ближайшая такая пара через две недели)
 - ▶ Оценка входит в процент выполненных заданий!
- ▶ В итоговой оценке практика учитывается с весом 0.4, экзамен — 0.6
- ▶ Округление арифметическое
- ▶ Две оценки — в конце этого и следующего модулей
 - ▶ Итоговая — оценка за 3-й модуль с весом 0.6 и оценка за 4-й модуль с весом 0.4

Краткое содержание курса по практике

- ▶ Снова лекции
 - ▶ Про архитектурную документацию
 - ▶ Про UML и другие языки проектирования
 - ▶ Про реализацию паттернов
 - ▶ Про антипаттерны
 - ▶ Различные примеры архитектур
- ▶ Небольшие задачи прямо на паре
 - ▶ Проектировать разные приложения
 - ▶ Технические вещи, типа рисования диаграмм
- ▶ Относительно большие задачи на дом

Что ожидается от решений

- ▶ Работоспособность и соответствие требованиям условия
- ▶ Наличие архитектурной документации
 - ▶ Комментарии к каждому классу, интерфейсу и public-методу
 - ▶ Краткое описание деталей реализации в README
- ▶ Следование стайлгайдам и правилам здравого смысла
- ▶ Язык программирования — любой
- ▶ Наличие юнит-тестов
- ▶ Применение индустриальных практик: логирование, Continuous Integration, обработка исключений

Ещё комментарии

- ▶ Овердизайн и активное использование знаний с лекций приветствуются
- ▶ Обоснованность принятых решений важнее, чем техника кодирования
- ▶ Некоторые требования могут показаться ненужными — это нормально
 - ▶ Мы учимся не написанию кода, а инструментам и техникам проектирования
- ▶ Комментарии вида “у вас неправильная архитектура” будут очень редки, как ни странно
- ▶ Списывать нельзя

Задача про CLI

Реализовать простой интерпретатор командной строки, поддерживающий команды:

- ▶ **cat [FILE]** — вывести на экран содержимое файла
- ▶ **echo** — вывести на экран свой аргумент (или аргументы)
- ▶ **wc [FILE]** — вывести количество строк, слов и байт в файле
- ▶ **pwd** — распечатать текущую директорию
- ▶ **exit** — выйти из интерпретатора

Задача про CLI (продолжение)

- ▶ Должны поддерживаться одинарные и двойные кавычки (full and weak quoting)
- ▶ Окружение (команды вида “имя=значение”), оператор \$
- ▶ Вызов внешней программы через Process (или его аналоги)
 - ▶ если введено что-то, чего интерпретатор не знает
- ▶ Пайплайны (оператор “|”)

Примеры

```
> echo "Hello, world!"  
Hello, world!
```

```
> FILE=example.txt  
> cat $FILE  
Some example text
```

```
> cat example.txt | wc  
1 3 18
```

```
> echo 123 | wc  
1 1 3
```

```
> x=ex  
> y=it  
> $x$y
```

Что ожидается в качестве решения

- ▶ Архитектурная документация, как умеете
 - ▶ Структурная диаграмма (классов, компонентов, квадратиков со стрелочками)
 - ▶ Словесное описание работы системы
 - ▶ Достаточно подробно, чтобы не требовалось принимать важные решения при кодировании
 - ▶ Не должно быть «ну тут мы парсим строку»
- ▶ Реализовывать проект пока не нужно
 - ▶ Через неделю будет задание это реализовать

Что делать дома

- ▶ Сделать для этого курса репозиторий
- ▶ Прислать мне ссылку в Telegram (https://t.me/yurii_litvinov)
- ▶ Одному из членов команды выложить решение в виде .md или .pdf-файла в отдельную ветку
- ▶ Сделать пуллреквест к себе в основную ветку
 - ▶ Назвать его как-то разумно, чтобы было понятно, о какой задаче речь
- ▶ Написать в чат курса, что задача готова к проверке
- ▶ Смерджить пуллреквест, когда задача зачтена
- ▶ **Дедлайн: 10:00 24.01.2022**

Что делать сейчас

Первые фазы жизненного цикла

- ▶ Выполнить анализ и определить подходы к решению
- ▶ Выявить подводные камни и способы их преодоления
- ▶ Декомпозировать задачу на подсистемы, классы и методы
- ▶ Нарисовать первое приближение структурной диаграммы
- ▶ Быть готовыми в конце пары выйти и рассказать предлагаемое решение
- ▶ Дома это надо будет уточнить, расширить и оформить

Соображения

- ▶ Проектирование сверху вниз
 - ▶ Определитесь с общей структурой системы
 - ▶ Определитесь с компонентами, их ответственностью и связями между ними
 - ▶ Только после этого переходите к проектированию компонентов
 - ▶ По такой же схеме
 - ▶ Возможно, придётся возвращаться на уровень выше
- ▶ Опасайтесь архитектурной жадности, надо вовремя остановиться

На что обратить внимание

- ▶ Как представляются команды и пайплайны?
- ▶ Как создаются команды?
- ▶ Как они исполняются? Как взаимодействуют потоки в пайплайне?
- ▶ Кто и как выполняет разбор входной строки?
 - ▶ Кто, как и когда выполняет подстановки?
- ▶ Как представляются переменные окружения?
- ▶ Что с многопоточностью?