

# Практика 1: Command Line Interface

Юрий Литвинов  
y.litvinov@spbu.ru

03.03.2022

# Задача про CLI

Спроектировать 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 как отдельного процесса
  - ▶ 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=exit
```

```
> $x
```

# Нефункциональные требования

- ▶ Легко добавлять новые команды (расширяемость)
- ▶ Наличие возможности реализовать что-то новое из того, что умеют другие шеллы (сопровождаемость)
- ▶ Архитектурное описание, как умеете (сопровождаемость)

# Что делать

## Первые фазы жизненного цикла

- ▶ Выполнить анализ и определить подходы к решению
- ▶ Выявить подводные камни и способы их преодоления
- ▶ Декомпонировать задачу на подсистемы, классы и методы
- ▶ Нарисовать диаграмму классов
- ▶ Словами описать принцип работы и основные принятые решения
  - ▶ Приложить к решению в Teams диаграмму классов в любом удобном формате
  - ▶ <https://www.diagrams.net/>
  - ▶ Если не защитите в конце пары, приложить также 1-2 страницы текстового описания

# 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?