\b \a \z ^ $ - simple assertions просто проверяют условия при этом не двигая позицию совпадения

Основные операторы:

Конкатенация

Альтернативы

Квантификация – {2,4} , {3,} , {,4} , {2}; ? – 0 или 1, \* - 0+, + - 1+;

Скобки

1. Влияют на приоритет
2. Задают захватывающие подмаски

(?: ) – задают только приоритет, без захвата подмаски

\1 – back reference – она будет совпадать с тем, что совпадает с подмаской под указанным номером

(?1) – вызов подмаски (может быть рекурсивной)

Complex assertions

X(?=Y ) – найти x при условии что после идет Y

X(?!Y ) - найти x при условии что после не идет Y

(?<=Y)X - найти x при условии что до него идет Y

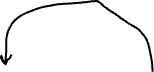
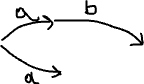
(?<!Y)X - найти x при условии что до него не идет Y

Конечные автоматы – состоит из набора состояний и переходов между ними, в которых описаны условия

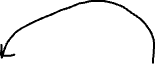
Недетерминированные (НКА)

Детерминированные – из любого узла, по любому одному символу может быть не более одного перехода (ДКА)

x(ab|acd)+y



Любой недетерминированный аппарат можно преобразовать в детерминированный



Трансляция

Компиляция - процесс перевода из одного языка на другой(например на машинный код)

Интерпретация – транслятор, разобравшись с программой выполняет ее

Трансляция – обобщающий термин для компиляции и интерпретации

Целевой язык – байт-код java

Процесс компиляции состоит из 4х стадий

1. Лексический анализ (lexer)

Задача разбить программу на массив лексем(токенов)

На входе текст – на выходе поток лексем(линейный массив). Лексемы имеют тип.

1. Синтаксический анализ (parser)

На выходе абстрактное синтаксическое дерево

1. Семантический анализ (обработка дерева)

Поиск ошибок, составление таблиц, дополнение и преобразование дерева

1. Генерация кода(выполнение программы) – исполняемый файл