Белорусский государственный технологический университет

Факультет информационных технологий

Кафедра программной инженерии

 Лабораторная работа №13

По дисциплине «Конструирование Программного Обеспечения»

На тему «Подготовка к разработке лексического распознавателя»

Выполнил:

Студент 1 курса 10 группы

Жамойдо Артём Игоревич

Преподаватель: А. С. Север

Минск, 2024

**Вариант №4**

proc(□)+(start(cos(y\*)+sin(x+)))\*□+end

**Пусть:**

proc – a

□ – b

start – c

cos – d

sin – e

end – f

**Тогда выражение примет вид:**

a(b)\*(c(d + e)\*b+f

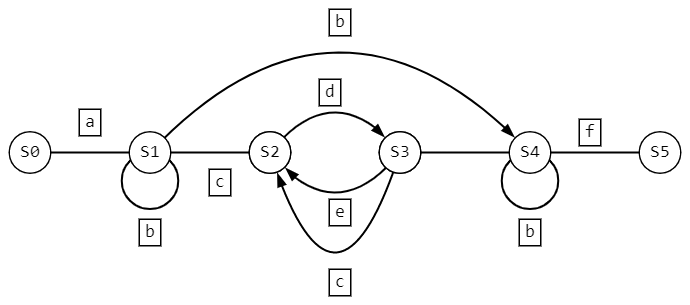
**Примеры цепочек:**

1. abbf
2. abcdbf
3. abbbcebbf
4. abcecebbbbf
5. abbcdcdbbbf
6. abbbbcdcdcdcdbbbbbbf
7. abcebf

**Диаграмма мгновенных состояний:**

abcdbf –> bcdbf –>cdbf–>dbf–>bf–>f–> λ

**Граф конечного автомата:**



1) Он определяет набор символов, из которых состоят допустимые строки или цепочки языка.

2) λ обозначает пустую строку, не содержащую ни одного символа.

I\* все цепочки, которые можно образовать, используя символы I.

I+ все НЕПУСТЫЕ цепочки, которые можно образовать, используя I.

3) Язык L(I) над алфавитом I представляет собой множество всех возможных цепочек или слов, которые могут быть образованы из символов алфавита I.

4) Формальная грамматика G представляет собой набор правил, которые определяют структуру формального языка.

5) описывают процесс замены одной цепочки символов на другую в грамматике.

6) представляет собой множество всех цепочек или слов, которые можно породить или сгенерировать с помощью правил продукции данной грамматики G.

7) Система для описания синтаксических категорий через последовательное описание других синтаксических категорий

8) Регулярная грамматика является формальной грамматикой, определяющей регулярный язык. Она состоит из четырех компонентов: алфавита (терминальные символы), нетерминальных символов, стартового символа и правил продукции.

9) Регулярное множество представляет собой множество строк или цепочек символов, которые могут быть порождены или распознаны с использованием регулярного выражения или конечного автомата.

10) Регулярный язык — это язык, который может быть описан с помощью регулярного выражения, регулярной грамматики или распознан с использованием конечного автомата.

11) Лексический анализ является первым этапом в компиляции или интерпретации программного кода. Он отвечает за разбиение исходного кода на лексемы или токены, которые являются минимальными значимыми элементами языка программирования.

12) Лексический анализатор является компонентом компилятора или интерпретатора, отвечающим за выполнение лексического анализа.

13) Вход – программный код, выход – токены

14) Параллельный – многопоточный, последовательный – однопоточный

15) Регулярное выражение над алфавитом I — это формальное выражение, используемое для описания и поиска шаблонов в строках, составленных из символов алфавита I.

16) S – множество состояний, I – множество входных символов, Σ – множество выходных символов, s0 – начальное состояние, F – множество конечных состояний.

17) В детерминированном – один переход для каждого состояния

18) Мгновенное состояние конечного автомата — это пара ((s, w)), где (s) — текущее состояние автомата, а (w) — оставшаяся часть входного слова.

19) - ((s, aw) > (s', w)): обозначает, что автомат ( M ) переходит из состояния (s) в состояние ( s' ) при чтении символа ( a ) из входного слова ( w ).

- ((si, wi) >\* (sk, wk)): обозначает, что автомат ( M ) переходит из состояния ( s ) в состояние ( s' ) после чтения всей строки ( w ).

20) Все эти понятия взаимосвязаны и описывают одно и то же множество строк — регулярный язык:

- Регулярная грамматика описывает регулярный язык.

- Регулярное выражение задает регулярный язык.

- Конечный автомат распознает регулярный язык.

- Граф состояний конечного автомата визуально представляет переходы между состояниями автомата для данного регулярного языка.