МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ Н. Э. БАУМАНА

Факультет информатики и систем управления

Кафедра теоретической информатики и компьютерных технологий

**Лабораторная работа № 1.**

**по курсу «Теория формальных языков»**

Выполнил: Иванов Г.М.

ИУ9-52

Проверил: Магазов С.C.

«19» ноября 2016 г.

Москва, 2016

Вариант 6.

1. Дан язык алфавита . Построить автоматную грамматику и автомат-распознаватель (содержащий функции переходов и диаграмму). Построить вывод заданного слова и привести такты работы распознавателя.

– множество слов, содержащих подслово . Слово: .

* Автоматная грамматика

Для построения автоматной грамматики мы определим основные его элементы:

– алфавит нетерминальных символов

– алфавит терминальных символов

– множество правил вывода, где

– начальный символ

Данная грамматика порождает язык , где

* Автомат-распознаватель

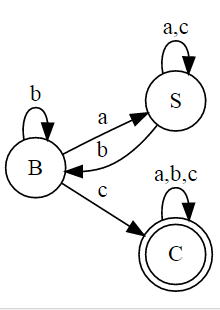
S – начальное состояние

C – конечное состояние

*Функции переходов:*

Данные функции переходов можно отобразить в виде таблицы переходов:

|  |  |  |  |
| --- | --- | --- | --- |
|  | ***a*** | ***b*** | ***c*** |
| ***S*** | *S* | *B* | *S* |
| ***B*** | *S* | *B* | *C* |
| ***C (конечное)*** | *С* | *С* | *С* |

* **Диаграмма
* Вывод слова
* Такты работы автомата распознавателя со словом :

Автомат оказался в конечном состоянии слово было распознано.

1. Дано описание языка. Построить для него регулярное выражение, автоматную грамматику и автомат распознаватель (функцию переходов и диаграмму).

L- множество паролей для следующей политики: в пароле должна быть заглавная, и маленькая буква, и цифра, длина пароля не менее 4, в нем должно отсутствовать слова MTC.

* Регулярное выражение

*Листинг 1. Задание регулярного выражения (паттерна).*

#!usr/bin/python3

# -\*- coding: utf-8 -\*-

pattern = re.compile(r"^(?=.\*[0-9])(?=.\*[a-z])(?=.\*[A-Z]).{4,}(?!^MTC)$")

*Пояснение (правил регулярного выражения):*

^ - начало строки

(?=.\*[0-9]) - как минимум 1 цифра

(?=.\*[a-z]) - как минимум 1 буква нижнего регистра (маленькая буква)

(?=.\*[A-Z]) - как минимум 1 буква верхнего регистра (заглавная буква)

{4,} - минимальная длина пароля – 4 символа

(?!^MTC) – исключать все слова MTC

$ - конец строки

Для проверки правильности нашего регулярного выражения (паттерна) составим unit-тесты:

*Листинг 2. Unit-тестирование.*

#!usr/bin/python3

# -\*- coding: utf-8 -\*-

**import** re

**import** unittest

pattern = re.compile(r"^(?=.\*[0-9])(?=.\*[a-z])(?=.\*[A-Z]).{4,}(?!^MTC)$")

**class** TestRegexPattern(unittest.TestCase):

**def** setUp(self):

self.pattern = pattern

**def** tearDown(self):

self.pattern = None

**def** test\_correct\_string(self):

correct\_string = "Ts3ys"

self.assertIsNotNone(re.match(self.pattern,correct\_string))

**def** test\_incorrect\_string(self):

incorrect\_strings = ["Tsss","dc83","TKJ2"]

**for** string **in** incorrect\_strings:

self.assertIsNone(re.match(self.pattern,string))

**def** test\_except\_string(self):

string = "MTC"

self.assertIsNone(re.match(self.pattern,string))

**if** \_\_name\_\_=="\_\_main\_\_":

unittest.main()

При выполнении программы нам выдаётся ответ в виде сообщения:

*...*

*----------------------------------------------------------------------*

*Ran 3 tests in 0.000s*

*OK*

* Автоматная грамматика

Для построения автоматной грамматики мы определим основные его элементы:

– алфавит нетерминальных символов

– алфавит терминальных символов

– множество правил вывода, где

– начальный символ

* Автомат-распознаватель

– начальное состояние

– конечное состояние

*Функции переходов:*

* Диаграмма

**