# Ореховский Антон P3317

## Домашняя работа №4, вариант 10

Задание на лабораторную работу «Распознаватель для грамматики простого предшествования». По грамматике (из файла SPG.docx), выбранной в соответствии с вариантом:

* Удалить ε-правила из грамматики;
* Построить (используя программу grammar) бесконфликтную матрицу отношений предшествования, возможно модифицировать правила грамматики;

Для бесконфликтной матрицы реализовать программу-распознаватель.

|  |  |  |
| --- | --- | --- |
| 1.10 | S → A  A → A + T | T  T → F \* T | F  F → a | (A) |  |

* Удаление ε-правила не требуется
* Приведение к форме предшествования

S → A

A → A + T | T

T → F \* T | F

F → a | (X)

X → S

* Бесконфликтная матрица, приведенная к форме предшествования:

<S>::=<A>

<A>::=<A>"+"<T>

<A>::=<T>

<T>::=<F>"\*"<T>

<T>::=<F>

<F>::="a"

<X>::=<S>

<F>::="("<X>")"

* Матрица предшествования

; ; ; ; ; ; ; ; ;>; S

; ; ; ; ;=; ; ; ;>; A

; ; ; ; ;>; ; ; ;>; T

; ; ; ; ;>;=; ; ;>; F

; ; ; ; ; ; ; ; ;=; X

; ;=;<; ; ; ;<;<; ; +

; ;=;<; ; ; ;<;<; ; \*

; ; ; ; ;>;>; ; ;>; a

<;<;<;<;=; ; ;<;<; ; (

; ; ; ; ;>;>; ; ;>; )

* Программа распознаватель:

def checkString(string):

if (string == ''):

return True

in\_st = list(string)

in\_st.append('#')

op\_st = list('#')

while in\_st[0] != '#' or (len(op\_st) != 2 or op\_st[0] != '#' or op\_st[1] != 'S'):

try:

prev = op\_st[len(op\_st)-1] #последний элемент преобразуемой строки

cur = in\_st[0] #первый элемент входной строки

relation = matrix[prev][cur] #отношение между предыдущим и текущим элементами

if (relation == '<' or relation == '='):

op\_st.append(in\_st.pop(0)) #сдвиг

elif (relation == '>'): #свёртка

last = op\_st.pop()

reducing\_str = last

red\_prev = op\_st[len(op\_st)-1] #последний элемент преобразуемой строки

while(matrix[red\_prev][last] != '<'):

last = op\_st.pop()

reducing\_str += last #накапливаем преобразуемую подстроку в обратном порядке

red\_prev = op\_st[len(op\_st)-1] #последний элемент преобразуемой строки

op\_st.append(grammar[reducing\_str[::-1]]) #переворачиваем преобразуемую строку обратно и заменяем на сокращение из грамматики

else:

return False

except Exception:

return False

return op\_st.pop() == 'S' and op\_st.pop() == '#';

tests = ['', 'abacb', 'aabcbbaabacccc', 'aaaaa']

for test in tests:

if (checkString(test)):

print('word ' + test + ' is ok to this grammar')

else:

print('word ' + test + ' is not ok to this grammar')