Laporan Teori Bahasa Automata
Kelompok 7 (Bahasa Makassar)

Disusun Oleh :
Muhammad Nur Iqbal Wariesky (1301204044)
Muh Thoriq Akhdan (1301204031)

Program Studi S1 Informatika – Fakultas Informatika Universitas Telkom
Jalan Telekomunikasi Terusan Buah Batu, Bandung Indonesia

A. Context Free Grammar

S → N V N

N → Nakke **|** Katte **|** Amma **|** Je'ne **|** Snggara **|** Balla

V → anganre **|** anginung **|** Nangai **|** A'jappa

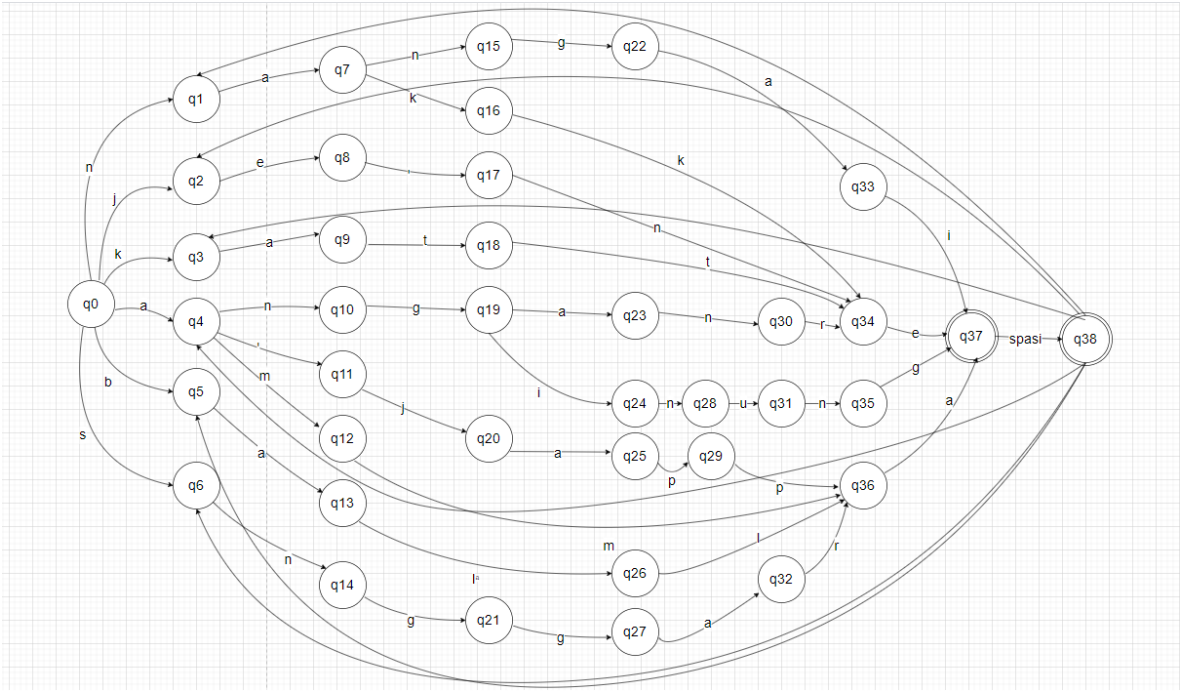
Simbol non - terminal : S (*starting symbol*), N (*Noun*), V (*Verb*)

Simbol terminal : nakke, katte, amma, je'ne, snggara, balla, anganre, anginung, nangai, a'jappa

Berikut ini adalah pengelompokan simbol non-terminal dengan simbol terminal serta arti dari Bahasa Makassar ke Bahasa Indonesia.

Bahasa Makassar	Bahasa Indonesia	Simbol Non - Terminal
N	Nakke	Saya
N	Katte	Kamu
N	Amma	Ibu
N	Je'ne	Air
N	Snggara	Pisang
N	Balla	Rumah
V	Anganre	Makan
V	Anginung	Minum
V	Nangai	Suka
V	A'jappa	Jalan

B. Finite Automata



C. Parser Table

	nakke	katte	amma	je'ne	sngg ara	balla	anga nre	angi nung	nangai	a'jappa	EOS
S	N V N	N V N	N V N	N V N	N V N	N V N	error	error	error	error	error
N	Nakke	Katte	Amma	Je'ne	Sngg ara	Balla	error	error	error	error	error
V	error	error	error	error	error	error	anga nre	angi nung	nangai	a'jappa	error

D. Lexical Analyzer

```
import string

# input
sentence = input("Ketik Kalimat : ")
input_string = sentence.lower() + '#'

# initialization
alphabet_list = list(string.ascii_lowercase)
state_list = ['q0', 'q1','q2', 'q3', 'q4', 'q5', 'q6', 'q7', 'q8', 'q9', 'q10', 'q11',
'q12', 'q13', 'q14', 'q15', 'q16', 'q17', 'q18', 'q19', 'q20', 'q21', 'q22', 'q23',
'q24', 'q25', 'q26', 'q27', 'q28', 'q29', 'q30', 'q31', 'q32', 'q33', 'q34', 'q35',
'q36', 'q37', 'q38']

transition_table = {}

for state in state_list:
  for alphabet in alphabet_list:
    transition_table[(state, alphabet)] = 'error'
  transition_table[(state, '#')] = 'error'
  transition_table[(state, ' ' )] = 'error'

#space before input string
transition_table['q0', ' '] = 'q0'

#update the transition table for the following token : nakke
transition_table[('q0','n')] = 'q1'
transition_table[('q1','a')] = 'q7'
transition_table[('q7','k')] = 'q16'
transition_table[('q16','k')] = 'q34'
transition_table[('q34','e')] = 'q37'
transition_table[('q37','#')] = 'accept'
transition_table[('q37',' ')] = 'q38'
transition_table[('q38','#')] = 'accept'

#update the transition table for the following token : amma
transition_table[('q0','a')] = 'q4'
transition_table[('q4','m')] = 'q12'
transition_table[('q12','m')] = 'q36'
transition_table[('q36','a')] = 'q37'
transition_table[('q37','#')] = 'accept'
transition_table[('q37',' ')] = 'q38'
transition_table[('q38','#')] = 'accept'
```

```
#update the transition table for the following token : katte
transition_table[('q0','k')] = 'q3'
transition_table[('q3','a')] = 'q9'
transition_table[('q9','t')] = 'q18'
transition_table[('q18','t')] = 'q34'
transition_table[('q34','e')] = 'q37'
transition_table[('q37','#')] = 'accept'
transition_table[('q37',' ')] = 'q38'
transition_table[('q38','#')] = 'accept'

#update the transition table for the following token : je'ne
transition_table[('q0','j')] = 'q2'
transition_table[('q2','e')] = 'q8'
transition_table[('q8', "'")] = 'q17'
transition_table[('q17','n')] = 'q34'
transition_table[('q34','e')] = 'q37'
transition_table[('q37','#')] = 'accept'
transition_table[('q37',' ')] = 'q38'
transition_table[('q38','#')] = 'accept'

#update the transition table for the following token : snggara
transition_table[('q0','s')] = 'q6'
transition_table[('q6','n')] = 'q14'
transition_table[('q14','g')] = 'q21'
transition_table[('q21','g')] = 'q27'
transition_table[('q27','a')] = 'q32'
transition_table[('q32','r')] = 'q36'
transition_table[('q36','a')] = 'q37'
transition_table[('q37','#')] = 'accept'
transition_table[('q37',' ')] = 'q38'
transition_table[('q38','#')] = 'accept'

#update the transition table for the following token : balla
transition_table[('q0','b')] = 'q5'
transition_table[('q5','a')] = 'q13'
transition_table[('q13','l')] = 'q26'
transition_table[('q26','l')] = 'q36'
transition_table[('q36','a')] = 'q37'
transition_table[('q37','#')] = 'accept'
transition_table[('q37',' ')] = 'q38'
transition_table[('q38','#')] = 'accept'



#update the transition table for the following token : a'jappa
```

```
transition_table[('q0','a')] = 'q4'
transition_table[('q4',"'")] = 'q11'
transition_table[('q11','j')] = 'q20'
transition_table[('q20','a')] = 'q25'
transition_table[('q25','p')] = 'q29'
transition_table[('q29','p')] = 'q36'
transition_table[('q36','a')] = 'q37'
transition_table[('q37','#')] = 'accept'
transition_table[('q37',' ')] = 'q38'
transition_table[('q38','#')] = 'accept'

#update the transition table for the following token : nangai
transition_table[('q0','n')] = 'q1'
transition_table[('q1','a')] = 'q7'
transition_table[('q7','n')] = 'q15'
transition_table[('q15','g')] = 'q22'
transition_table[('q22','a')] = 'q33'
transition_table[('q33','i')] = 'q37'
transition_table[('q37','#')] = 'accept'
transition_table[('q37',' ')] = 'q38'
transition_table[('q38','#')] = 'accept'

#update the transition table for the following token : anginung
transition_table[('q0','a')] = 'q4'
transition_table[('q4','n')] = 'q10'
transition_table[('q10','g')] = 'q19'
transition_table[('q19','i')] = 'q24'
transition_table[('q24','n')] = 'q28'
transition_table[('q28','u')] = 'q31'
transition_table[('q31','n')] = 'q35'
transition_table[('q35','g')] = 'q37'
transition_table[('q37','#')] = 'accept'
transition_table[('q37',' ')] = 'q38'
transition_table[('q38','#')] = 'accept'

#update the transition table for the following token : anganre
transition_table[('q0','a')] = 'q4'
transition_table[('q4','n')] = 'q10'
transition_table[('q10','g')] = 'q19'
transition_table[('q19','a')] = 'q23'
transition_table[('q23','n')] = 'q30'
transition_table[('q30','r')] = 'q34'
transition_table[('q34','e')] = 'q37'
transition_table[('q37','#')] = 'accept'
transition_table[('q37',' ')] = 'q38'
```

```
transition_table[('q38','#')] = 'accept'

#transition for new token
transition_table[('q38','n')] = 'q1'
transition_table[('q38','j')] = 'q2'
transition_table[('q38','k')] = 'q3'
transition_table[('q38','a')] = 'q4'
transition_table[('q38','b')] = 'q5'
transition_table[('q38','s')] = 'q6'

# lexical analysis
idx_char = 0
state = 'q0'
current_token = ''
while state != 'accept':
  current_char = input_string[idx_char]
  current_token += current_char
  state = transition_table[(state, current_char)]
  #if state == '37':
  print('current token: ', current_token, ', valid')
  #current_token = ''
  if state == 'error':
    print('error, token tidak valid')
    break;
  idx_char = idx_char + 1

# conclusion
if state == 'accept':
  print('semua token di input: ', sentence, ', valid')
```

Hasil running program :

- **Untuk hasil yang valid :**

```
Ketik Kalimat : nakke anginung je'ne
semua token di input:  nakke anginung je'ne , valid
```

- **Untuk hasil yang tidak valid :**

```
Ketik Kalimat : amma a'jjappa riballa
error, token tidak valid
```

E. Program Parser

```
#TBA TAHAP 2 (Parser)
#input
sentence = input('input kalimat :')
tokens = sentence.lower().split()
tokens.append('EOS')

#symbol
non_terminals = ['S','N','V']
terminals                                                                    =
['nakke','katte',"je'ne",'snggara','balla','anganre','anginung','nangai',"a'jap
pa", 'amma']

#parse table
parse_table ={}

parse_table[('S', 'nakke')] = ['N','V','N']
parse_table[('S', 'katte')] = ['N','V','N']
parse_table[('S', 'amma')] = ['N','V','N']
parse_table[('S', "je'ne")] = ['N','V','N']
parse_table[('S', 'snggara')] = ['N','V','N']
parse_table[('S', 'balla')] = ['N','V','N']
parse_table[('S', 'anganre')] = ['error']
parse_table[('S', 'anginung')] = ['error']
parse_table[('S', 'nangai')] = ['error']
parse_table[('S', "a'jappa")] = ['error']
parse_table[('S', 'EOS')] = ['error']

parse_table[('N', 'nakke')] = ['nakke']
parse_table[('N', 'katte')] = ['katte']
parse_table[('N', 'amma')] = ['amma']
parse_table[('N', "je'ne")] = ["je'ne"]
parse_table[('N', 'snggara')] = ['snggara']
parse_table[('N', 'balla')] = ['balla']
parse_table[('N', 'anganre')] = ['error']
parse_table[('N', 'anginung')] = ['error']
parse_table[('N', 'nangai')] = ['error']
parse_table[('N', "a'jappa")] = ['error']
parse_table[('N', 'EOS')] = ['error']

parse_table[('V', 'nakke')] = ['error']
parse_table[('V', 'katte')] = ['error']
parse_table[('V', 'amma')] = ['error']
parse_table[('V', "je'ne")] = ['error']
parse_table[('V', 'snggara')] = ['error']
```

```
parse_table[('V', 'balla')] = ['error']
parse_table[('V', 'anganre')] = ['anganre']
parse_table[('V', 'anginung')] = ['anginung']
parse_table[('V', 'nangai')] = ['nangai']
parse_table[('V', "a'jappa")] = ["a'jappa"]
parse_table[('N', 'EOS')] = ['error']

# stack initialization
stack = []
stack.append('#')
stack.append('S')

# input reading initialization
idx_token = 0
symbol = tokens[idx_token]

# parsing process
while (len(stack) > 0):
    top = stack [len(stack) - 1]
    print('top = ', top)
    print('symbol = ', symbol)
    if top in terminals:
        print('top adalah simbol terminal')
        if top == symbol:
            stack.pop()
            idx_token = idx_token + 1
            symbol = tokens[idx_token]
            if symbol == 'EOS':
                print('isi stack: ', stack)
                stack.pop()

        else:
            print('error')
            break;
    elif top in non_terminals:
        print('top adalah simbol non-terminal')
        if parse_table[(top, symbol)][0] != 'error':
            stack.pop()
            symbols_to_be_pushed = parse_table[(top, symbol)]
            for i in range(len(symbols_to_be_pushed)-1,-1,-1):
                stack.append(symbols_to_be_pushed[i])
        else:
            print('error')
            break;
    else:
```

```
        print('error')
        break;
    print('isi stack: ', stack)
    print()

# conclusion
print()
if symbol == 'EOS' and len(stack) == 0:
    print('Input string ', sentence, ' diterima, sesuai Grammar')
else:
     print('Error, input string: ', sentence, ', tidak diterima, tidak sesuai
Grammar')
```

Hasil running program :

- **Hasil yang valid :**

```
top =  N
symbol =  amma
top adalah simbol non-terminal
isi stack:  ['#', 'N', 'V', 'amma']

top =  amma
symbol =  amma
top adalah simbol terminal
isi stack:  ['#', 'N', 'V']

top =  V
symbol =  nangai
top adalah simbol non-terminal
isi stack:  ['#', 'N', 'nangai']

top =  nangai
symbol =  nangai
top adalah simbol terminal
isi stack:  ['#', 'N']

top =  N
symbol =  snggara
top adalah simbol non-terminal
isi stack:  ['#', 'snggara']

top =  snggara
symbol =  snggara
top adalah simbol terminal
isi stack:  ['#']
isi stack:  []


Input string  amma nangai snggara  diterima, sesuai Grammar
```

- **Hasil yang tidak valid :**

```
input kalimat :nangai amma a'jappa
top =  S
symbol =  nangai
top adalah simbol non-terminal
error

Error, input string:  nangai amma a'jappa , tidak diterima, tidak sesuai Grammar
```

F. Kesimpulan

Pada tugas besar ini kami menggunakan bahasa Makassar yang memiliki struktur S-V-O (subject - verb - object) dapat dibuat context free grammar, finite automata, lexical analyzer, parse table, dan program parser.