

**Tugas Besar Teori Bahasa dan Automata:  
Lexical Analyzer dan Parser Sederhana  
untuk Teks Bahasa Alami dalam Bahasa Melayu**



Oleh:  
Risma Amaliyah Mahmudah (1301204087)  
Vania Amadea (1301204365)  
IF4408

Program Studi S1 Informatika  
Fakultas Informatika  
Universitas Telkom  
Bandung

## DAFTAR ISI

<b>DAFTAR ISI</b>	<b>2</b>
<b>BAB 1</b>	<b>3</b>
<b>BAB 2</b>	<b>4</b>
2.1 Finite Automata	4
2.2 Context Free Grammar	4
2.3 Lexical Analysis	4
2.4 Parser	4
<b>BAB 3</b>	<b>5</b>
3.1 Context Free Grammar	5
3.2 Finite Automata	5
3.3 Parser Table LL (1)	6
<b>BAB 4</b>	<b>7</b>
4.1 Lexical Analysis	7
4.2 Parser	13
4.3 Program Gabungan (JavaScript)	20
4.4 HTML	22
4.5 CSS	23
<b>REFERENSI</b>	<b>24</b>

## **BAB 1**

### **PENDAHULUAN**

Tata Bahasa Bebas Konteks (*Context Free Grammar* / CFG) adalah sebuah cara membuat string dalam suatu bahasa. Pada saat menurunkan suatu string, simbol-simbol variabel akan mewakili bagian-bagian yang belum yang belum diturunkan dari string tersebut. Bila pada tata bahasa regular, bagian yang belum terturunkan tersebut selalu terjadi pada suatu ujung. *Context Free Grammar* memungkinkan terdapat lebih banyak bagian yang belum diturunkan dan bisa terjadi di mana saja. Jika penurunan sudah lengkap, semua bagian yang belum diturunkan telah diganti oleh string-string yang mungkin saja kosong dari himpunan simbol terminal. Hal ini menjadi dasar dalam pembentukan suatu parser. *Context Free Grammar* sederhana yang dibuat ini menggunakan representasi aturan atau sintaks kalimat dalam sebuah bahasa, bahasa Melayu.

## **BAB 2**

### **KAJIAN PUSTAKA**

#### **2.1 Finite Automata**

Finite automata adalah mesin abstrak berupa sistem model matematika dengan masukan dan keluaran diskrit yang dapat mengenali bahasa paling sederhana (bahasa reguler) dan dapat diimplementasikan secara nyata di mana sistem dapat berada di salah satu dari sejumlah berhingga konfigurasi internal disebut state.

#### **2.2 Context Free Grammar**

CFG atau *Context Free Grammar* adalah tata bahasa formal di mana setiap aturan produksi adalah dalam bentuk  $A \rightarrow B$  di mana A adalah pemproduksi, dan B adalah hasil produksi. Batasannya hanyalah ruas kiri adalah sebuah simbol variabel. Dan pada ruas kanan bisa berupa terminal, symbol, variable ataupun  $\epsilon$ .

#### **2.3 Lexical Analysis**

*Lexical analyzer* adalah tahapan pertama yang dilakukan pada compiler. Proses yang dilakukan pada tahapan ini adalah membaca program sumber karakter per karakter. Satu atau lebih (deretan) karakter karakter ini dikelompokkan menjadi suatu kesatuan mengikuti pola kesatuan kelompok karakter (token) yang ditentukan dalam bahasa sumber dan disimpan dalam table simbol, sedangkan karakter yang tidak mengikuti pola akan dilaporkan sebagai token tak dikenal.

#### **2.4 Parser**

*Parsing* adalah suatu cara memecah-mecah suatu rangkaian masukan (misalnya dari berkas atau keyboard) yang akan menghasilkan suatu pohon uraian (parse tree) yang akan digunakan pada tahap kompilasi berikutnya yaitu analisis semantik. Di dalam komputasi, parser adalah salah satu komponen dalam sebuah interpreter atau compiler yang bertugas memeriksa sintaks secara benar serta membangun struktur data yang tersirat dalam token masuka

## BAB 3

### ANALISIS DAN PERANCANGAN

#### 3.1 Context Free Grammar

Deskripsi CFG untuk Bahasa Melayu

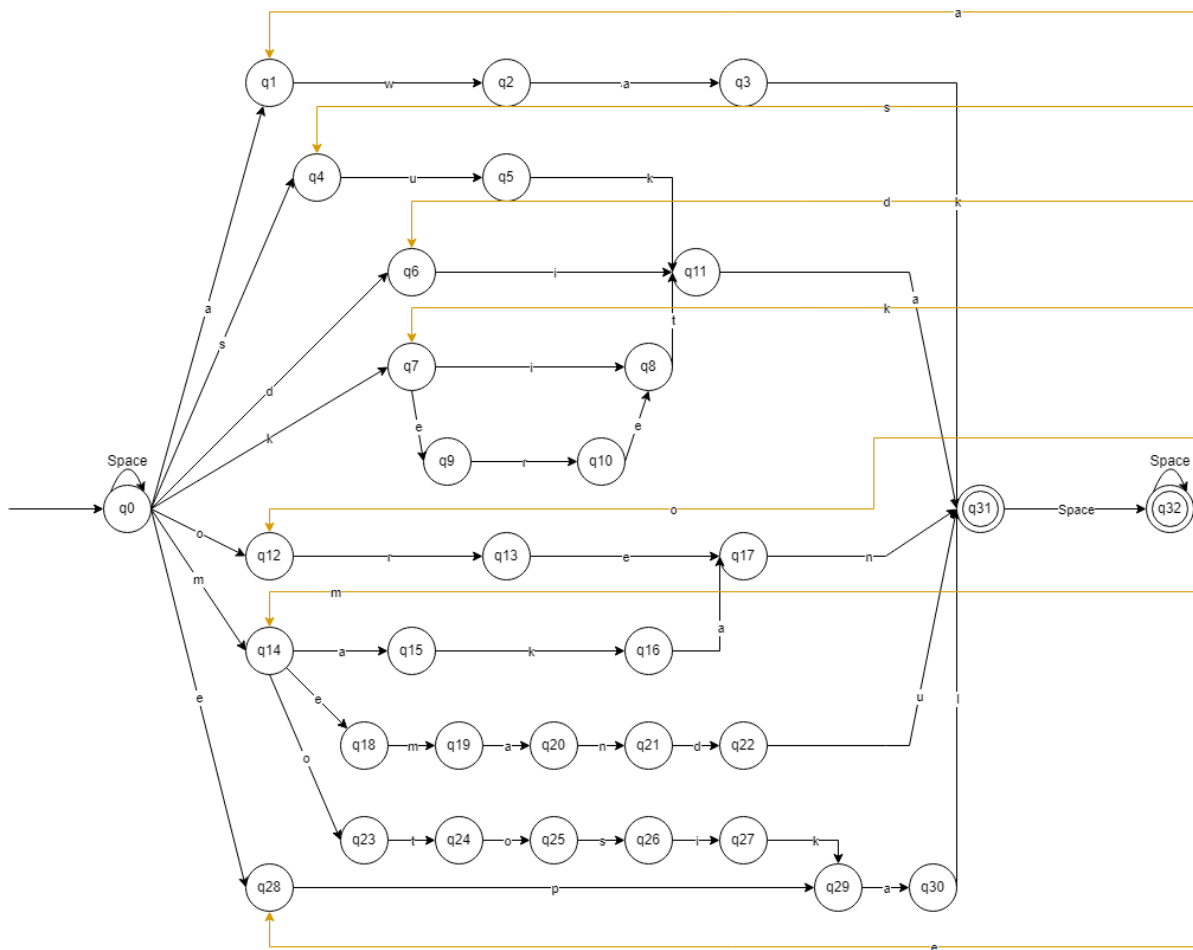
$S = \{SB, VB, OB\}$

$SB \rightarrow awak \mid dia \mid kita$

$VB \rightarrow makan \mid memandu \mid suka$

$OB \rightarrow epal \mid kereta \mid motosikal \mid oren$

#### 3.2 Finite Automata



### 3.3 Parser Table LL (1)

	awak	dia	kita	makan	memandu	suka	epal	kereta	motosikal	oren	EOS
<b>S</b>	SB	SB	SB	error	error	error	SB	SB	SB	SB	error
	VB	VB	VB				VB	VB	VB	VB	
	OB	OB	OB				OB	OB	OB	OB	
<b>SB</b>	awak	dia	kita	error	error	error	error	error	error	oren	error
<b>VB</b>	error	error	error	makan	memandu	suka	error	error	error	oren	error
<b>OB</b>	error	error	error	error	error	error	epal	kereta	motosikal	oren	error

## BAB 4

### KODE PROGRAM DAN HASIL

#### 4.1 Lexical Analysis

Berikut adalah kode program untuk *lexical analyzer* untuk menguji kata yang dimasukkan user. Adapun kata yang valid dalam *lexical analyzer* ini, yaitu awak, dia, kita, makan, memandu, suka, epal, kereta, motosikal, dan oren. Untuk kata awak, dia, dan kita merupakan kata *subject*. Untuk kata makan, memandu, dan suka merupakan kata kerja (*verb*). Untuk kata epal, kereta, motosikal, dan oren merupakan kata *object*.

```
JS lexical_analyzer.js > ...
1 var masukkan = document.getElementById('masukkan_kalimat');
2 var submit = document.getElementById('btn-analyze');
3 var hasil = document.getElementById('result');
4 var clear = document.getElementById('btn-clear');
5 var loading = document.getElementById('loading');
6
7 //input example
8 //var input_string;
9 //var sentence;
10 //sentence = 'awak memandu motosikal';
11 //input_string = sentence.lower () + '#';
12
13 //initialization
14 var state_list, transition_table;
15 var alphabet_list = [];
16 for(var i = 32; i <= 126; i++) {
17     alphabet_list.push(String.fromCharCode( i ))
18 }
19 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",
20 transition_table = {};
21
22
23 for(var state in state_list) {
24     for(alphabet in alphabet_list) {
25         transition_table[[state_list[state], alphabet_list[alphabet]]] = 'error'
26     }
27     transition_table[[state_list[state], '#']] = 'error'
28     transition_table[[state_list[state], ' ']] = 'error'
29 }
30
31 //spaces before input string
32 transition_table [['q0', ' ']] = 'q0'
33 //final state
34 transition_table[['q31', ' ']] = 'q32'
35 transition_table[['q31', '#']] = 'accept'
36
37 transition_table[['q32', ' ']] = 'q32'
```

```
JS lexical_analyzer.js > ...
38 transition_table[['q32', '#']] = 'accept'
39
40 //update the transition table for the following token: awak
41 transition_table[['q0', 'a']] = 'q1'
42 transition_table[['q1', 'w']] = 'q2'
43 transition_table[['q2', 'a']] = 'q3'
44 transition_table[['q3', 'k']] = 'q31'
45 transition_table[['q31', ' ']] = 'q32'
46
47 //update the transition table for the following token: suka
48 transition_table[['q0', 's']] = 'q4'
49 transition_table[['q4', 'u']] = 'q5'
50 transition_table[['q5', 'k']] = 'q11'
51 transition_table[['q11', 'a']] = 'q31'
52 transition_table[['q31', ' ']] = 'q32'
53
54 //update the transition table for the following token: dia
55 transition_table[['q0', 'd']] = 'q6'
56 transition_table[['q6', 'i']] = 'q11'
57 transition_table[['q11', 'a']] = 'q31'
58 transition_table[['q31', ' ']] = 'q32'
59
60 //update the transition table for the following token: kita
61 transition_table[['q0', 'k']] = 'q7'
62 transition_table[['q7', 'i']] = 'q8'
63 transition_table[['q8', 't']] = 'q11'
64 transition_table[['q11', 'a']] = 'q31'
65 transition_table[['q31', ' ']] = 'q32'
66
67 //update the transition table for the following token: kereta
68 transition_table[['q0', 'k']] = 'q7'
69 transition_table[['q7', 'e']] = 'q9'
70 transition_table[['q9', 'r']] = 'q10'
71 transition_table[['q10', 'e']] = 'q8'
72 transition_table[['q8', 't']] = 'q11'
73 transition_table[['q11', 'a']] = 'q31'
74 transition_table[['q31', ' ']] = 'q32'
```

```

JS lexical_analyzer.js > ...
75
76 //update the transition table for the following token: oren
77 transition_table[['q0', 'o']] = 'q12'
78 transition_table[['q12', 'r']] = 'q13'
79 transition_table[['q13', 'e']] = 'q17'
80 transition_table[['q17', 'n']] = 'q31'
81 transition_table[['q31', ' ']] = 'q32'
82
83 //update the transition table for the following token: makan
84 transition_table[['q0', 'm']] = 'q14'
85 transition_table[['q14', 'a']] = 'q15'
86 transition_table[['q15', 'k']] = 'q16'
87 transition_table[['q16', 'a']] = 'q17'
88 transition_table[['q17', 'n']] = 'q31'
89 transition_table[['q31', ' ']] = 'q32'
90
91 //update the transition table for the following token: memandu
92 transition_table[['q0', 'm']] = 'q14'
93 transition_table[['q14', 'e']] = 'q18'
94 transition_table[['q18', 'm']] = 'q19'
95 transition_table[['q19', 'a']] = 'q20'
96 transition_table[['q20', 'n']] = 'q21'
97 transition_table[['q21', 'd']] = 'q22'
98 transition_table[['q22', 'u']] = 'q31'
99 transition_table[['q31', ' ']] = 'q32'
100
101 //update the transition table for the following token: motosikal
102 transition_table[['q0', 'm']] = 'q14'
103 transition_table[['q14', 'o']] = 'q23'
104 transition_table[['q23', 't']] = 'q24'
105 transition_table[['q24', 'o']] = 'q25'
106 transition_table[['q25', 's']] = 'q26'
107 transition_table[['q26', 'i']] = 'q27'
108 transition_table[['q27', 'k']] = 'q29'
109 transition_table[['q29', 'a']] = 'q30'
110 transition_table[['q30', 'l']] = 'q31'
111 transition_table[['q31', ' ']] = 'q32'

```

```

JS lexical_analyzer.js > ...
112
113 //update the transition table for the following token: epal
114 transition_table[['q0', 'e']] = 'q28'
115 transition_table[['q28', 'p']] = 'q29'
116 transition_table[['q29', 'a']] = 'q30'
117 transition_table[['q30', 'l']] = 'q31'
118 transition_table[['q31', ' ']] = 'q32'
119
120 //update the transition table for the new token
121 transition_table[['q32', 'a']] = 'q1'
122 transition_table[['q32', 's']] = 'q4'
123 transition_table[['q32', 'd']] = 'q6'
124 transition_table[['q32', 'k']] = 'q7'
125 transition_table[['q32', 'o']] = 'q12'
126 transition_table[['q32', 'm']] = 'q14'
127 transition_table[['q32', 'e']] = 'q28'
128
129 clear.onclick = (event) => {
130     masukan.value = "";
131     hasil.value = "";
132 }
133
134 submit.onclick = (event) => {
135
136     loading.style = 'display: inline-block'
137
138     // lexical analysis
139     var indexChar = 0;
140     var state = 'q0';
141     var currentToken = '';
142     var validation = '';
143     var inputChar = masukan.value + '#';
144     console.log(inputChar);
145     while (state != 'accept') {
146         var currentChar = inputChar.charAt(indexChar)
147         currentToken += currentChar
148         state = transition_table[[state, currentChar]]

```

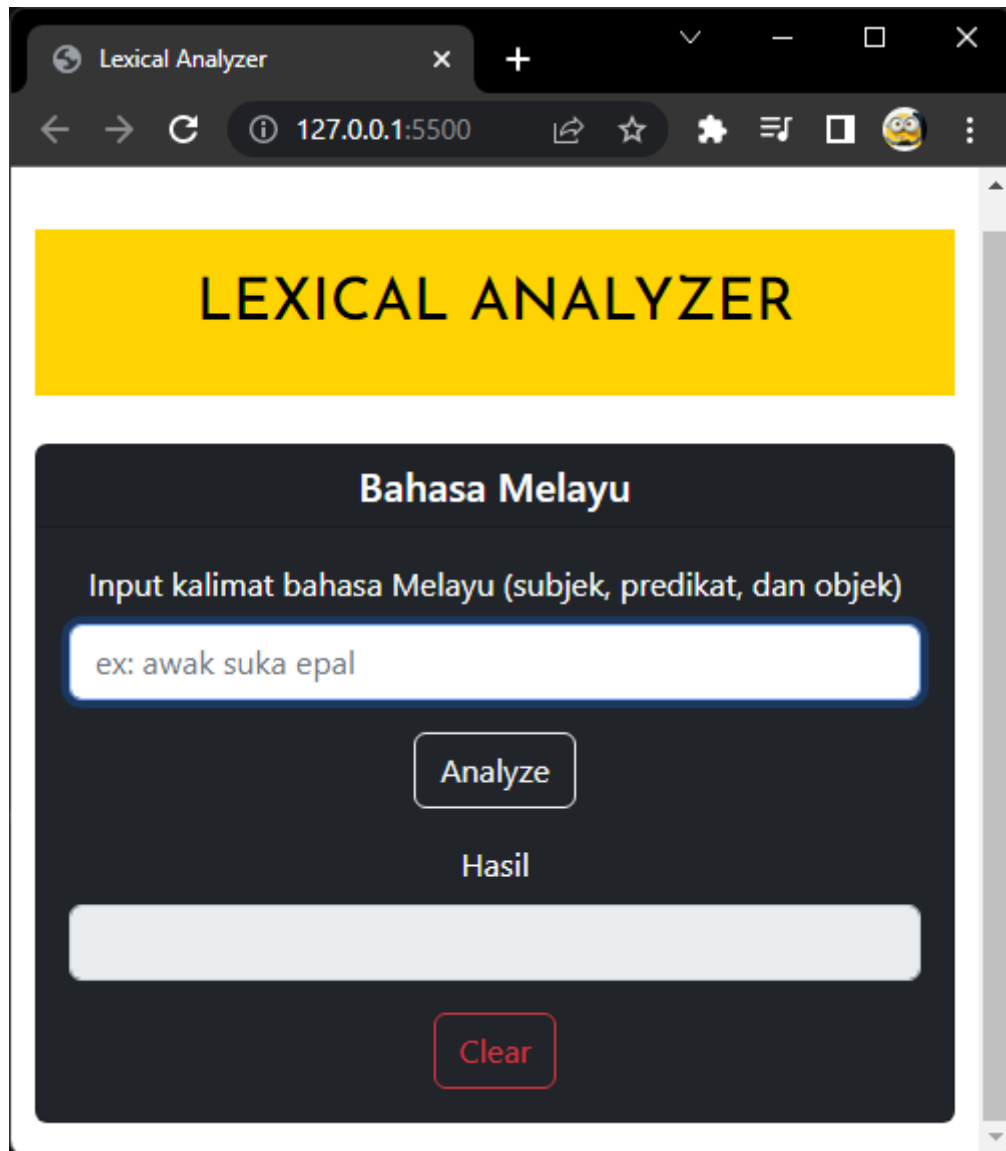
```

JS lexical_analyzer.js > ...
149         if(state == 'q31') {
150             console.log("valid")
151             validation += "valid "
152             currentToken = ''
153         }
154         if(state == 'error') {
155             console.log("error")
156             validation += "error "
157             break;
158         }
159         indexChar += 1
160     }
161
162     console.log(validation);
163     hasil.value = validation.trim();
164
165     loading.style = 'display: none'
166 }

```



Berikut adalah hasil pengujian untuk *lexical analyzer* yang digunakan untuk menguji kata yang dimasukkan oleh user.



The screenshot shows a web browser window with the title "Lexical Analyzer". The address bar displays "127.0.0.1:5500". The main content area has a yellow header with the text "LEXICAL ANALYZER". Below this, there is a dark gray box titled "Bahasa Melayu". Inside this box, the text "Input kalimat bahasa Melayu (subjek, predikat, dan objek)" is followed by a text input field containing "ex: awak suka epal". Below the input field is a button labeled "Analyze". Underneath the button is the word "Hasil" above a large, empty light gray rectangular box. At the bottom of the dark gray box is a button labeled "Clear".

Berikut merupakan hasil pengujian untuk kalimat 'awak suka epal' yang ketiga kata tersebut merupakan kata yang valid.

Lexical Analyzer

127.0.0.1:5500

# LEXICAL ANALYZER

## Bahasa Melayu

Input kalimat bahasa Melayu (subjek, predikat, dan objek)

awak suka epal

Analyze

Hasil

valid valid valid

Clear

Berikut merupakan hasil pengujian untuk kalimat 'awak minum susu' dengan kata 'awak' tersebut merupakan kata yang valid, sedangkan kata 'minum' dan 'susu' merupakan kata yang tidak valid.

Lexical Analyzer

127.0.0.1:5500/index.h...

# LEXICAL ANALYZER

## Bahasa Melayu

Input kalimat bahasa Melayu (subjek, predikat, dan objek)

awak minum susu

Analyze

Hasil

valid error

Clear

Berikut merupakan hasil pengujian untuk kalimat 'risma awak oren' dengan kata 'risma' tersebut merupakan kata yang tidak valid, maka program akan langsung menyatakan bahwa kalimat tersebut error.

Lexical Analyzer

127.0.0.1:5500/index.h...

# LEXICAL ANALYZER

## Bahasa Melayu

Input kalimat bahasa Melayu (subjek, predikat, dan objek)

risma awak oren

Analyze

Hasil

error

Clear

## 4.2 Parser

Program ini akan memeriksa kalimat yang sudah diinputkan dan diterima oleh fungsi *lexical analyzer* sepanjang tiga kata sesuai *grammar* SB-VB-OB dalam bahasa Melayu. Berikut adalah program *parser*

```
169 function parser(sentence){
170     sentence = sentence.replace(/\s+/g, ' ').trim()
171     var tokens = sentence.toLowerCase().split(' ');
172     tokens.push('EOS')
173     console.log('tokens: ', tokens)
174     //Symbols definition
175     non_terminals = ['S', 'SB', 'VB', 'OB'];
176     terminals = ['awak', 'dia', 'kita', 'makan', 'memandu', 'suka', 'epal', 'kereta', 'motosikal', 'oren'];
177     //parse table definition
178     var parse_table = {};
179
180     parse_table[['S', 'awak']] = ['SB', 'VB', 'OB']
181     parse_table[['S', 'dia']] = ['SB', 'VB', 'OB']
182     parse_table[['S', 'kita']] = ['SB', 'VB', 'OB']
183     parse_table[['S', 'makan']] = ['error']
184     parse_table[['S', 'memandu']] = ['error']
185     parse_table[['S', 'suka']] = ['error']
186     parse_table[['S', 'epal']] = ['error']
187     parse_table[['S', 'kereta']] = ['error']
188     parse_table[['S', 'motosikal']] = ['error']
189     parse_table[['S', 'oren']] = ['error']
190     parse_table[['S', 'EOS']] = ['error']
191
192     parse_table[['SB', 'awak']] = ['awak']
193     parse_table[['SB', 'dia']] = ['dia']
194     parse_table[['SB', 'kita']] = ['kita']
195     parse_table[['SB', 'makan']] = ['error']
196     parse_table[['SB', 'memandu']] = ['error']
197     parse_table[['SB', 'suka']] = ['error']
198     parse_table[['SB', 'epal']] = ['error']
199     parse_table[['SB', 'kereta']] = ['error']
200     parse_table[['SB', 'motosikal']] = ['error']
201     parse_table[['SB', 'oren']] = ['error']
202     parse_table[['SB', 'EOS']] = ['error']
203
204     parse_table[['VB', 'awak']] = ['error']
205     parse_table[['VB', 'dia']] = ['error']
206     parse_table[['VB', 'kita']] = ['error']
207     parse_table[['VB', 'makan']] = ['makan']
208     parse_table[['VB', 'memandu']] = ['memandu']
209     parse_table[['VB', 'suka']] = ['suka']
210     parse_table[['VB', 'epal']] = ['error']
211     parse_table[['VB', 'kereta']] = ['error']
212     parse_table[['VB', 'motosikal']] = ['error']
213     parse_table[['VB', 'oren']] = ['error']
214     parse_table[['VB', 'EOS']] = ['error']
215
216     parse_table[['OB', 'awak']] = ['error']
217     parse_table[['OB', 'dia']] = ['error']
218     parse_table[['OB', 'kita']] = ['error']
219     parse_table[['OB', 'makan']] = ['error']
220     parse_table[['OB', 'memandu']] = ['error']
221     parse_table[['OB', 'suka']] = ['error']
222     parse_table[['OB', 'epal']] = ['epal']
223     parse_table[['OB', 'kereta']] = ['kereta']
224     parse_table[['OB', 'motosikal']] = ['motosikal']
225     parse_table[['OB', 'oren']] = ['oren']
226     parse_table[['OB', 'EOS']] = ['error']
227 }
```

```

227
228 // stack initialization
229 var stack = [];
230 stack.push('#');
231 stack.push('S');
232
233 //input remothering initialization
234 var idxToken = 0;
235 var symbol = tokens[idxToken];
236
237 while(stack.length > 0) {
238     var top = stack[stack.length - 1];
239     if(terminals.includes(top)) {
240         if(top == symbol) {
241             stack.pop();
242             idxToken += 1;
243             symbol = tokens[idxToken];
244             if(symbol == 'EOS') {
245                 stack.pop();
246             }
247         } else {
248             break;
249         }
250     } else if(non_terminals.includes(top)) {
251         if(parse_table[[top, symbol]][0] != 'error') {
252             stack.pop();
253             var symbolToBePushed = parse_table[[top, symbol]];
254             for(let i = symbolToBePushed.length - 1; i > -1; i--) {
255                 stack.push(symbolToBePushed[i]);
256             }
257         } else {
258             break;
259         }
260     } else {
261         break;
262     }
263 }
264
265 //conclusion
266 console.log('conclusion');
267 if(symbol == 'EOS' && stack.length == 0) {
268     console.log('Input string ', sentence, 'diterima sesuai grammar');
269     hasil.value = `Input string ${sentence} diterima sesuai grammar`;
270     loading.style = 'display: none'
271 } else {
272     console.log('Error, tidak diterima karena tidak sesuai grammar');
273     hasil.value = 'Error, tidak diterima karena tidak sesuai grammar';
274     loading.style = 'display: none'
275 }
276 }
277 lexical(masukkan);

```

Berikut adalah hasil pengujian untuk *parser* yang digunakan untuk menguji kalimat yang dimasukkan oleh user yang diterima oleh *lexical analyzer* dan sesuai *grammar*.

## RV JAYA

**Lexical Analyzer & Parser**

Input kalimat bahasa Melayu (subjek, predikat, dan objek)

awak suka epal

Analyze

Hasil

Input string awak suka epal diterima sesuai grammar

Clear

### Kata Tersedia

awak, dia, kita, makan, memandu, suka, epal, kereta, motosikal, oren

© 2022, RV

## RV JAYA

**Lexical Analyzer & Parser**

Input kalimat bahasa Melayu (subjek, predikat, dan objek)

dia memandu motosikal

Analyze

Hasil

Input string dia memandu motosikal diterima sesuai grammar

Clear

### Kata Tersedia

awak, dia, kita, makan, memandu, suka, epal, kereta, motosikal, oren

© 2022, RV

# RV JAYA

## Lexical Analyzer & Parser

Input kalimat bahasa Melayu (subjek, predikat, dan objek)

kita makan oren

Analyze

Hasil

Input string kita makan oren diterima sesuai grammar

Clear

## Kata Tersedia

awak, dia, kita, makan, memandu, suka, epal, kereta, motosikal, oren

© 2022, RV



Berikut adalah hasil pengujian untuk *parser* yang digunakan untuk menguji kalimat yang dimasukkan oleh user yang diterima oleh *lexical analyzer*, tetapi tidak sesuai *grammar*.

# RV JAYA

## Lexical Analyzer & Parser

Input kalimat bahasa Melayu (subjek, predikat, dan objek)

dia motosikal memandu

Analyze

Hasil

Error, tidak diterima karena tidak sesuai grammar

Clear

### Kata Tersedia

awak, dia, kita, makan, memandu, suka, epal, kereta, motosikal, oren

© 2022, RV

# RV JAYA

## Lexical Analyzer & Parser

Input kalimat bahasa Melayu (subjek, predikat, dan objek)

awak epal memandu dia

Analyze

Hasil

Error, tidak diterima karena tidak sesuai grammar

Clear

### Kata Tersedia

awak, dia, kita, makan, memandu, suka, epal, kereta, motosikal, oren

© 2022, RV

# RV JAYA

## Lexical Analyzer & Parser

Input kalimat bahasa Melayu (subjek, predikat, dan objek)

motosikal makan awak suka makan

Analyze

Hasil

Error, tidak diterima karena tidak sesuai grammar

Clear

## Kata Tersedia

awak, dia, kita, makan, memandu, suka, epal, kereta, motosikal, oren

© 2022, RV

Berikut adalah hasil pengujian untuk *parser* yang digunakan untuk menguji kalimat yang dimasukkan oleh user yaitu “betul betul betul” yang tidak diterima oleh *lexical analyzer*.

# RV JAYA

## Lexical Analyzer & Parser

Input kalimat bahasa Melayu (subjek, predikat, dan objek)

betul betul betul

Analyze

Hasil

error

Clear

### Kata Tersedia

awak, dia, kita, makan, memandu, suka, epal, kereta, motosikal, oren

### 4.3 Program Gabungan (JavaScript)

[illegible]

```
File Edit Selection View Go Run Terminal Help
la parser.js - Tugas Besar TBA - Katompok 9 Bahasa Melayu - IP4408 - Visual Studio Code

la parser.js | @ parser
119 transition_table["q32", "w"] = "q7"
120 transition_table["q32", "o"] = "q32"
121 transition_table["q32", "e"] = "q34"
122 transition_table["q32", "v"] = "q28"
123
124 clear.onclick = (event) => {
125     masukan.value = "";
126     hasil.value = "";
127 }
128
129 submit.onclick = (event) => {
130     var sentence = masukan.value;
131     loading.style = "display: inline-block"
132
133     // lexical analysis
134     var indexChar = 0;
135     var state = "q0";
136     var currentToken = "";
137     var validation = "";
138     var inputChar = masukan.value + " ";
139     console.log(inputChar);
140     while (state != "accept") {
141         var currentChar = inputChar.charAt(indexChar);
142         currentToken = currentChar;
143         state = transition_table[state, currentChar];
144         if (state == "q32") {
145             console.log("valid")
146             validation += "valid "
147             currentToken = ""
148         }
149         if (state == "error") {
150             console.log("error")
151             validation += "error "
152             break;
153         }
154         indexChar++;
155     }
156     if (state == "accept") {
157         // console.log(validation);
158         // hasil.value = validation.trim();
159         parser(sentence);
160         // loading.style = "display: none"
161     } else {
162         console.log(validation);
163         hasil.value = validation.trim();
164         loading.style = "display: none"
165     }
166 }
167
168 }
169
170 function parser(sentence) {
171     sentence = sentence.replace(/\\n/g, " ");
172     var tokens = sentence.toLowerCase().split(' ');
173     tokens.push('EOS');
174     console.log("tokens: ", tokens)
175
176     // Symbols definition
177     nonTerminals = ["S", "SB", "VB", "OB"];
```

```
File Edit Selection View Go Run Terminal Help
la parser.js - Tugas Besar TBA - Katompok 9 Bahasa Melayu - IP4408 - Visual Studio Code

la parser.js | @ parser
178 terminals = ['makan', 'dia', 'kita', 'makan', 'memandu', 'suka', 'epal', 'kereta', 'motosikal', 'oren'];
179
180 //stack initialization
181 var parse_table = {};
182
183 parse_table["S", "makan"] = ["SB", "VB", "OB"]
184 parse_table["S", "dia"] = ["SB", "VB", "OB"]
185 parse_table["S", "kita"] = ["SB", "VB", "OB"]
186 parse_table["S", "makan"] = ["error"]
187 parse_table["S", "memandu"] = ["error"]
188 parse_table["S", "suka"] = ["error"]
189 parse_table["S", "epal"] = ["error"]
190 parse_table["S", "kereta"] = ["error"]
191 parse_table["S", "motosikal"] = ["error"]
192 parse_table["S", "oren"] = ["error"]
193 parse_table["S", "EOS"] = ["error"]
194
195 parse_table["SB", "makan"] = ["makan"]
196 parse_table["SB", "dia"] = ["dia"]
197 parse_table["SB", "kita"] = ["kita"]
198 parse_table["SB", "makan"] = ["error"]
199 parse_table["SB", "memandu"] = ["error"]
200 parse_table["SB", "suka"] = ["error"]
201 parse_table["SB", "epal"] = ["error"]
202 parse_table["SB", "kereta"] = ["error"]
203 parse_table["SB", "motosikal"] = ["error"]
204 parse_table["SB", "oren"] = ["error"]
205 parse_table["SB", "EOS"] = ["error"]
206
207 parse_table["VB", "makan"] = ["error"]
208 parse_table["VB", "dia"] = ["error"]
209 parse_table["VB", "kita"] = ["error"]
210 parse_table["VB", "makan"] = ["makan"]
211 parse_table["VB", "memandu"] = ["memandu"]
212 parse_table["VB", "suka"] = ["suka"]
213 parse_table["VB", "epal"] = ["error"]
214 parse_table["VB", "kereta"] = ["error"]
215 parse_table["VB", "motosikal"] = ["error"]
216 parse_table["VB", "oren"] = ["error"]
217 parse_table["VB", "EOS"] = ["error"]
218
219 parse_table["OB", "makan"] = ["error"]
220 parse_table["OB", "dia"] = ["error"]
221 parse_table["OB", "kita"] = ["error"]
222 parse_table["OB", "makan"] = ["error"]
223 parse_table["OB", "memandu"] = ["error"]
224 parse_table["OB", "suka"] = ["error"]
225 parse_table["OB", "epal"] = ["error"]
226 parse_table["OB", "kereta"] = ["kereta"]
227 parse_table["OB", "motosikal"] = ["motosikal"]
228 parse_table["OB", "oren"] = ["oren"]
229 parse_table["OB", "EOS"] = ["error"]
230
231 // stack initialization
232 var stack = [];
233 stack.push("#");
234 stack.push("S");
235
236 //input remothering initialization
```

```
File Edit Selection View Go Run Terminal Help
le.parser.js - Tugan Besar TBA - Kelompok 9 Bahasa Melayu - IF402 - Visual Studio Code

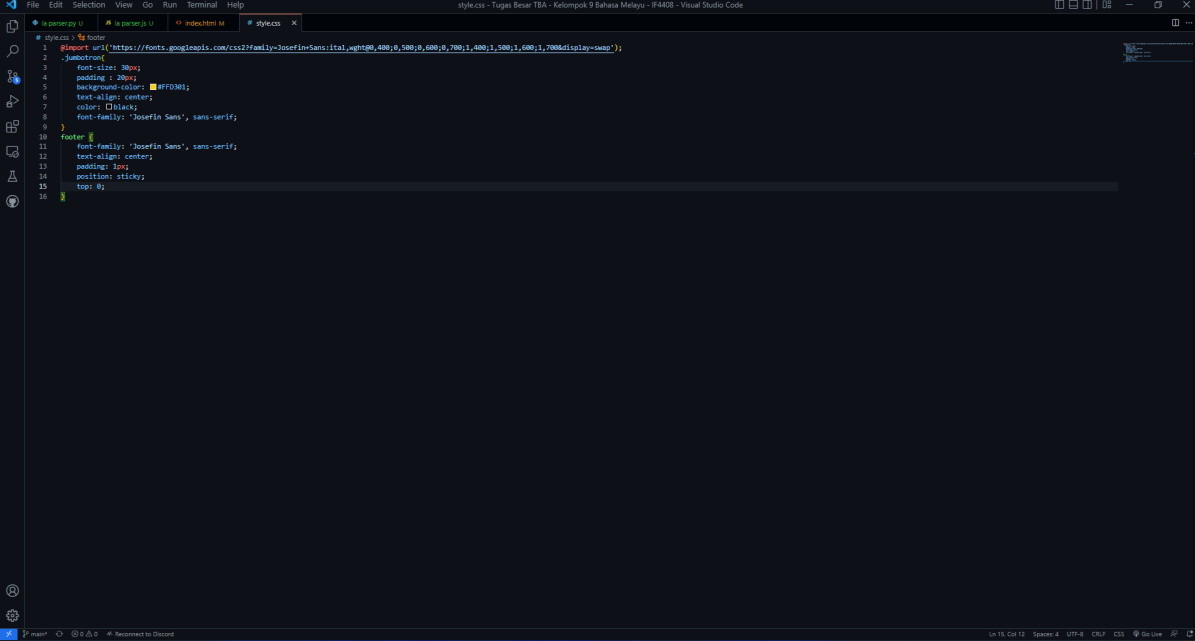
le.parser.js > @ parser
236 //input remothering initialization
237 var idfoken = 0;
238 var symbol = tokens[idfoken];
239
240 while(stack.length > 0) {
241     var top = stack[stack.length - 1];
242     if(terminal.includes(top)) {
243         if(top == symbol) {
244             stack.pop();
245             idfoken++;
246             symbol = tokens[idfoken];
247             if(symbol == 'EoS') {
248                 stack.pop();
249             }
250         } else {
251             break;
252         }
253     } else if(non_terminals.includes(top)) {
254         if(parse_table[top, symbol][0] != "error") {
255             stack.pop();
256             var symbolIndexPushed = parse_table[top, symbol];
257             for(let i = symbolIndexPushed.length - 1; i > -1; i--) {
258                 stack.push(symbolIndexPushed[i]);
259             }
260         } else {
261             break;
262         }
263     } else {
264         break;
265     }
266 }
267
268 //conclusion
269 console.log('conclusion');
270 if(symbol == 'EoS' && stack.length == 0) {
271     console.log('Input string ', sentence, 'diterima sesuai grammar');
272     hasil.value = "Input string ("sentence, diterima sesuai grammar";
273     loading.style = "display: none";
274 } else {
275     console.log('error, tidak diterima karena tidak sesuai grammar');
276     hasil.value = "error, tidak diterima karena tidak sesuai grammar";
277     loading.style = "display: none";
278 }
279
280 lexical(masukkan);
```

## 4.4 HTML

```
File Edit Selection View Go Run Terminal Help
index.html - Tugan Besar TBA - Kelompok 9 Bahasa Melayu - IF402 - Visual Studio Code

index.html > @ html
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <meta http-equiv="X-UA-Compatible" content="IE=edge">
6     <meta name="viewport" content="width=device-width, initial-scale=1.0">
7     <link rel="stylesheet" href="style.css">
8     <link rel="shortcut icon" href="image/AV LOGO-01.png">
9     <!-- CSS only -->
10    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0-beta1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-4wEaPiDd6OKnIDwJh4v36FLyrj7QX2iAuIiqYf6xwq9F5qni7j1o2c0s9y" crossorigin="anonymous">
11    <title>Lexical Analyzer & Parser</title>
12 </head>
13 <body class="container mt-5">
14     <div class="jumbotron">
15         <div class="text-center">
16             <h1>AV JAWAB</h1>
17         </div>
18     </div>
19     <div class="card text-center text-bg-dark mb-3">
20         <div class="card-header">Lexical Analyzer & Parser</div>
21         <div class="card-body">
22             <div class="mb-3">
23                 <label for="masukkan_kalimat" class="form-label">Input kalimat bahasa Melayu (subjek, predikat, dan objek) </label>
24                 <input type="text" class="form-control" id="masukkan_kalimat" placeholder="ex: makan saka epal">
25             </div>
26             <button type="button" id="btn-analyze" class="btn btn-outline-light">
27                 Analyze
28             </button>
29             <div class="spinner-border text-light spinner-border-sm mt-2" id="loading" style="display: none;" role="status">
30                 <span class="visually-hidden">Loading...</span>
31             </div>
32             <div class="mb-3 mt-3">
33                 <label for="result" class="form-label">Hasil</label>
34                 <input type="text" class="form-control" disabled id="result">
35             </div>
36             <button type="button" id="btn-clear" class="btn btn-outline-danger">
37                 Clear
38             </button>
39         </div>
40     </div>
41     <div class="text-center">
42         <div class="card-header">Kata Tersedia</div>
43         <div class="card-body">
44             <div class="mb-3">
45                 <label for="masukkan_kalimat" class="form-label">mamak, dia, kita, makar, mamandu, saka, epal, kereta, motosikal, oren</label>
46             </div>
47         </div>
48     </div>
49
50     <!-- JavaScript Bundle with Popper -->
51     <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0-beta1/dist/js/bootstrap.bundle.min.js" integrity="sha384-pprn3073Q6p26YajwXtd30+yaoiQn186wvM7QU8YSJysKi3n06pDQM7QU8" crossorigin="anonymous"></script>
52     <script type="text/javascript" src="js/parser.js"></script>
53 </body>
54 </html>
```

## 4.5 CSS



```
1 @import url('https://fonts.googleapis.com/css2?family=Josefin+Sans:ital,wght@0,400;0,500;0,600;0,700;1,400;1,500;1,600;1,700&display=swap');
2
3 .jumbotron
4 {
5   font-size: 28px;
6   padding: 20px;
7   background-color: #FFD381;
8   text-align: center;
9   color: #1a2b3c;
10  font-family: 'Josefin Sans', sans-serif;
11 }
12
13 footer
14 {
15   font-family: 'Josefin Sans', sans-serif;
16   text-align: center;
17   padding: 10px;
18   position: sticky;
19   top: 0;
20 }
```

Link web :

<https://vaniaas.github.io/Lexical-Analyzer-dan-Parser-Sederhana-untuk-Teks-Bahasa-Alami-Melayu/>

## BAB 5

### CARA MENJALANKAN PROGRAM

#### 5.1 JavaScript, HTML, dan CSS

Dengan membuka file secara langsung:

1. Unduh file dari <https://github.com/vaniaas/Lexical-Analyzer-dan-Parser-Sederhana-untuk-Teks-Bahasa-Alami-Melayu>
2. Extract file .zip
3. Buka folder Lexical-Analyzer-dan-Parser-Sederhana-untuk-Teks-Bahasa-Alami-Melayu-main
4. Buka file index.html pada browser
5. Setelah dibuka, akan muncul tampilan seperti ini

The screenshot shows a web application titled "RV JAYA" in a yellow header. Below it is a dark-themed box titled "Lexical Analyzer & Parser". Inside this box, there is a label "Input kalimat bahasa Melayu (subjek, predikat, dan objek)" above a text input field containing the example "ex: awak suka epal". Below the input field is an "Analyze" button. Underneath the button is the label "Hasil" above a large, empty light-gray rectangular box. At the bottom of the dark box is a "Clear" button. Below the dark box, the text "Kata Tersedia" is followed by a list of words: "awak, dia, kita, makan, memandu, suka, epal, kereta, motosikal, oren". At the very bottom, the copyright notice "© 2022, RV" is displayed.

6. Masukkan kalimat dalam bahasa Melayu.
7. Klik tombol "Analyze".
8. Hasil akan ditampilkan apakah kalimat tersebut:
  - a. Diterima oleh *lexical analyzer* dan dengan *grammar* yang sesuai (subjek, predikat, dan objek).
  - b. Diterima oleh *lexical analyzer* dan dengan *grammar* yang tidak sesuai.
  - c. Tidak diterima oleh *lexical analyzer* (error).
9. Klik tombol "Clear" untuk menghapus input dan hasil



Dengan membuka link website:

1. Buka link berikut ini

<https://vaniaas.github.io/Lexical-Analyzer-dan-Parser-Sederhana-untuk-Teks-Bahasa-Alami-Melayu/>

2. Setelah dibuka, akan muncul tampilan seperti ini

The screenshot shows a web application titled "RV JAYA" with a yellow header. Below the header is a dark-themed interface for a "Lexical Analyzer & Parser". It features an input field with the placeholder text "Input kalimat bahasa Melayu (subjek, predikat, dan objek)" and an example "ex: awak suka epal". There is an "Analyze" button, a "Hasil" label, and a "Clear" button. Below the interface, a list of "Kata Tersedia" (Available Words) is shown: "awak, dia, kita, makan, memandu, suka, epal, kereta, motosikal, oren". At the bottom, there is a copyright notice "© 2022, RV".

3. Masukkan kalimat dalam bahasa Melayu.
4. Klik tombol "Analyze".
5. Hasil akan ditampilkan apakah kalimat tersebut:
  - a. Diterima oleh *lexical analyzer* dan dengan *grammar* yang sesuai (subjek, predikat, dan objek).
  - b. Diterima oleh *lexical analyzer* dan dengan *grammar* yang tidak sesuai.
  - c. Tidak diterima oleh *lexical analyzer* (error).
6. Klik tombol "Clear" untuk menghapus input dan hasil

## 5.2 Python (Bukan Web)

Apabila ingin melihat output yang lebih detail setiap tahapnya, dapat membuka file .py dengan cara:

1. Sudah menginstall bahasa pemrograman python
2. Unduh file dari <https://github.com/vaniaas/Lexical-Analyzer-dan-Parser-Sederhana-untuk-Teks-Bahasa-Alami-Melayu>
3. Extract file .zip
4. Buka folder Lexical-Analyzer-dan-Parser-Sederhana-untuk-Teks-Bahasa-Alami-Melayu-main
5. Buka folder "python ver"
6. Buka file "laparser.py"
7. Setelah dibuka, akan muncul tampilan seperti ini

```

python ver > la parser.py > parser
1  #library
2  import string
3
4  #input example
5  # sentence = 'awak memandu motosikal'
6
7  def lexical (sentence):
8      print("\n===== Lexical Analyzer =====\n")
9      input_string = sentence. lower () + '#'
10
11     #initialization
12     alphabet_list = list(string.ascii_lowercase)
13     state_list = [
14         'q0', 'q1', 'q2', 'q3', 'q4', 'q5', 'q6', 'q7', 'q8', 'q9', 'q10',
15         'q11', 'q12', 'q13', 'q14', 'q15', 'q16', 'q17', 'q18', 'q19', 'q20',
16         'q21', 'q22', 'q23', 'q24', 'q25', 'q26', 'q27', 'q28', 'q29', 'q30', 'q31', 'q32'
17     ]
18
19     transition_table = {}
20
21     for state in state_list:
22         for alphabet in alphabet_list:
23             transition_table[(state, alphabet)] = 'error'
24         transition_table[(state, '#')] = 'error'
25         transition_table[(state, ' ')] = 'error'
26
27     #spaces before input string
28     transition_table [('q0', ' ')] = 'q0'
29     #final state
30     transition_table [('q31', ' ')] = 'q32'
31     transition_table [('q31', '#')] = 'accept'
32
33     transition_table [('q32', ' ')] = 'q32'
34     transition_table [('q32', '#')] = 'accept'
35
36     #update the transition table for the following token: awak
37     transition_table [('q0', 'a')] = 'q1'
38     transition_table [('q1', 'w')] = 'q2'
39     transition_table [('q2', 'a')] = 'q3'
40     transition_table [('q3', 'k')] = 'q31'
41     transition_table [('q31', ' ')] = 'q32'
42
43     #update the transition table for the following token: suka
44     transition_table [('q0', 's')] = 'q4'
45     transition_table [('q4', 'u')] = 'q5'
46     transition_table [('q5', 'k')] = 'q11'
47     transition_table [('q11', 'a')] = 'q31'
48     transition_table [('q31', ' ')] = 'q32'
49
50     #update the transition table for the following token: dia
51     transition_table [('q0', 'd')] = 'q6'
52     transition_table [('q6', 'i')] = 'q11'
53     transition_table [('q11', 'a')] = 'q31'
54     transition_table [('q31', ' ')] = 'q32'
55
56     #update the transition table for the following token: kita
57     transition_table [('q0', 'k')] = 'q7'
58     transition_table [('q7', 'i')] = 'q8'
59     transition_table [('q8', 't')] = 'q11'

```

8. Run kode program dengan vscode

```

python ver> python laparser.py
1 library
2 import string
3
4 #input example
5 # sentence = "maksudnya metotikal"
6
7 def lexical(sentence):
8     print("===== Lexical Analyzer =====\n")
9     input_string = sentence.lower() + " "
10
11 #initialization
12 alphabet_list = list(string.ascii_lowercase)
13 state_list = [
14     'q0', 'q1', 'q2', 'q3', 'q4', 'q5', 'q6', 'q7', 'q8', 'q9', 'q10',
15     'q11', 'q12', 'q13', 'q14', 'q15', 'q16', 'q17', 'q18', 'q19', 'q20',
16     'q21', 'q22', 'q23', 'q24', 'q25', 'q26', 'q27', 'q28', 'q29', 'q30', 'q31', 'q32'
17 ]
18
19 transition_table = {}
20
21 for state in state_list:
22     for alphabet in alphabet_list:
23         transition_table[(state, alphabet)] = 'error'
24     transition_table[(state, ' ')] = 'error'
25     transition_table[(state, '\n')] = 'error'
26
27 #spaces before input string
28 transition_table[('q0', ' ')] = 'q0'
29 #initial state
30 transition_table[('q31', '\n')] = 'q32'
31 transition_table[('q31', ' ')] = 'accept'
32
33 transition_table[('q32', '\n')] = 'q32'
34 transition_table[('q32', ' ')] = 'accept'
35
36 #update the transition table for the following token: maks
37 transition_table[('q0', 'a')] = 'q1'
38 transition_table[('q1', 'a')] = 'q2'
39 transition_table[('q2', 'a')] = 'q3'
40 transition_table[('q3', 'k')] = 'q31'
41 transition_table[('q31', '\n')] = 'q32'
42
43 #update the transition table for the following token: maksud
44 transition_table[('q0', 'a')] = 'q4'
45 transition_table[('q4', 'u')] = 'q5'
46 transition_table[('q5', 'k')] = 'q11'
47 transition_table[('q11', 'a')] = 'q31'
48 transition_table[('q31', '\n')] = 'q32'
49
50 #update the transition table for the following token: dia
51 transition_table[('q0', 'd')] = 'q6'
52 transition_table[('q6', 'i')] = 'q31'
53 transition_table[('q31', 'a')] = 'q31'
54 transition_table[('q31', '\n')] = 'q32'
55
56 #update the transition table for the following token: kita
57 transition_table[('q0', 'k')] = 'q7'
58 transition_table[('q7', 'i')] = 'q8'
59 transition_table[('q8', 't')] = 'q11'

```

atau dengan terminal langsung

```

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Van\Desktop\Tugas Besar TBA - Kelompok 9 Bahasa Melayu - IF4408\python ver> python laparser.py
Input Sentence : |

```

9. Masukkan kalimat dalam bahasa Melayu.
10. Hasil akan ditampilkan apakah kalimat tersebut:
  - a. Diterima oleh *lexical analyzer* dan dengan *grammar* yang sesuai (subjek, predikat, dan objek).
  - b. Diterima oleh *lexical analyzer* dan dengan *grammar* yang tidak sesuai.
  - c. Tidak diterima oleh *lexical analyzer* (error).

contoh :

```
Windows PowerShell
Windows Terminal can be set as the default terminal application in your settings.  Open Settings

PS C:\Users\Van\Desktop\Tugas Besar TBA - Kelompok 9 Bahasa Melayu - IF4408\python ver> python laparser.py
Input Sentence : awak suka epal

===== Lexical Analyzer =====

current token:  awak , valid
current token:  suka , valid
current token:  epal , valid
semua token di input:  awak suka epal , valid

===== Parser =====

top = S
symbol = awak
top adalah simbol non-terminal
isi stack: ['#', 'OB', 'VB', 'SB']

top = SB
symbol = awak
top adalah simbol non-terminal
isi stack: ['#', 'OB', 'VB', 'awak']

top = awak
symbol = awak
top adalah simbol terminal
isi stack: ['#', 'OB', 'VB']

top = VB
symbol = suka
top adalah simbol non-terminal
isi stack: ['#', 'OB', 'suka']

top = suka
symbol = suka
top adalah simbol terminal
isi stack: ['#', 'OB']

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

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

Input string awak suka epal diterima, sesuai Grammar

```
Input Sentence : betul betul risma

===== Lexical Analyzer =====

eror
```

## REFERENSI

- (n.d.). TEORI BAHASA DAN AUTOMATA. Retrieved June 8, 2022, from [https://repository.dinus.ac.id/docs/ajar/bab%2010\\_file\\_2013-05-31\\_080824\\_dr.r.\\_heru\\_tjahjana\\_s.si\\_m.si\\_.pdf](https://repository.dinus.ac.id/docs/ajar/bab%2010_file_2013-05-31_080824_dr.r._heru_tjahjana_s.si_m.si_.pdf)
- BAB 2 LANDASAN TEORI 2.1 Finite Automata Finite automata adalah mesin abstrak berupa sistem model matematika dengan masukan dan.* (n.d.). Library Binus. Retrieved June 7, 2022, from <http://library.binus.ac.id/eColls/eThesisdok/Bab2/2011-2-00004-MTIF%20Bab2001.pdf>
- PENYEDERHANAAN CONTEXT FREE GRAMMAR.* (2018, December 20). School of Computer Science | BINUS University. Retrieved June 8, 2022, from <https://socs.binus.ac.id/2018/12/20/penyederhanaan-context-free-grammar/>
- Teknik Kompilasi : TAHAPAN KOMPILASI.* (2019, December 23). School of Computer Science | BINUS University. Retrieved June 8, 2022, from <https://socs.binus.ac.id/2019/12/23/teknik-kompilasi-first-set-pada-top-down-parsing/>
- Yulianto, Alfiah, F., Wijaya, A. N., Ramadhan, M. R., Sakti, L. K., Mubtasir, & Mukti, A. (2015, February 8). *IMPLEMENTASI PENGGUNAAN SISTEM APLIKASI WEB PDF PARSER UNTUK MENAMPILKAN INFORMASI ISI DOKUMEN.* AMIKOM OJS Journal. Retrieved June 8, 2022, from <https://ojs.amikom.ac.id/index.php/semnasteknomedia/article/viewFile/806/772>