

# **Parser Bahasa JavaScript (Node.js)**

## **Laporan Tugas Besar**

Diajukan Untuk Memenuhi Tugas Besar Teori Bahasa Formal dan Otomata



Kelompok :

JJR

AFNAN EDSA RAMADHAN	13521011
LAILA BILBINA KHOIRU NISA	13521016
SYARIFA DWI PURNAMASARI	13521018

**TEKNIK INFORMATIKA**

**INSTITUT TEKNOLOGI BANDUNG 2022/2023**

# BAB I

## DASAR TEORI

### 1. LEXER PREPROCESSING

Lexical analyzer adalah fase pertama pada *compiler*. Lexer memodifikasi sebuah program melalui pemrosesan awal. Lexer memecah syntax menjadi deretan token dengan menghapus spasi atau *comment* pada *source code*. Lexer memanfaatkan regex untuk menyeleksi deretan *source* agar sesuai dengan bentuk non terminal sesuai yang diekspektasikan. Pembentukan non terminal pada lexer mengikuti pasangan set pada array rules yang diberikan. Selanjutnya lexer akan diproses mengikuti *grammar* yang ditentukan. Pada tugas kali ini, lexer dibuat dengan finite automata. Program dibuat dengan memanfaatkan *class* atau *object* pada JavaScript. *String* atau file dibaca layaknya mesin kata, di mana pembacaan per karakter kemudian dicocokkan dan dengan enumerasi token yang sudah disiapkan sejak awal.

### 2. CONTEXT-FREE GRAMMAR

Dalam teori bahasa formal, CFG adalah *grammar* formal yang memproduksi aturan dengan bentuk

$$A \rightarrow a$$

Language yang diproduksi CFG disebut dengan CFL, *Content Free Language*. CFG pada linguistic dapat digunakan untuk mendeskripsikan struktur kalimat dan kata dalam bahasa natural. Setiap dari *grammar* regular ada CFG. Namun, tidak semua CFG regular.

Penggunaan CFG pada *compiler* bermaksud untuk mempermudah serta mengurangi ambiguitas dalam parsing. Parsing sendiri merupakan proses pemrosesan string berdasarkan aturan CFG.

Sebagian besar syntax pada bahasa pemrograman didefinisikan dalam CFG. Pada keberjalanannya CFG menggunakan pohon penurunan untuk menggambarkan simbol variabel menjadi simbol terminal.

### 3. CHOMSKY NORMAL-FORM

Chomsky normal form (CNF) adalah bentuk (form) tersimplifikasi dari CFG. Sebuah grammar dikatakan berbentuk CNF jika semua aturan produksinya berbentuk salah satu di antara:

$$A \rightarrow BC$$

$$A \rightarrow a$$

$$S \rightarrow \epsilon$$

dengan  $A, B, C$  adalah simbol nonterminal ( $B$  dan  $C$  bukan simbol start),  $a$  adalah simbol terminal,  $S$  adalah simbol start, dan  $\epsilon$  adalah empty string. Bentuk ketiga hanya valid apabila empty string juga valid dalam grammar tersebut.

Langkah-langkah pengubahan CFG umum ke CNF adalah sebagai berikut:

1. Apabila simbol start muncul di right-hand side pada salah satu rule, buat produksi baru sebagai simbol start
2. Simplifikasi CFG dengan urutan
  - a. Hapus dan ganti null productions
  - b. Hapus dan ganti unit productions
  - c. Hapus useless productions
3. Dekomposisi aturan yang mengandung campuran terminal dan nonterminal, atau lebih dari satu terminal
4. Dekomposisi aturan yang mengandung lebih dari dua non terminal

### 4. COCKE-YOUNGER KASAMI

Cocke-Younger-Kasami (CYK) Algorithm adalah sebuah algoritma yang digunakan untuk membuktikan apakah sebuah word  $w$  di-generate oleh grammar context free atau tidak. Algoritma CYK dikembangkan oleh John Cocke, Daniel Younger, dan Tadao Kasami. Untuk dapat menggunakan algoritma ini dibutuhkan grammar context free  $G$  dalam bentuk Chomsky normal form, dimana word  $w$  adalah sebagai input, dan outputnya adalah sebuah pembuktian apakah word  $w$  merupakan bahasa dari grammar  $G$  atau bukan.

CYK-Algorithm dapat diilustrasikan dalam bentuk tabel sebagai berikut

X <sub>15</sub>				
X <sub>14</sub>	X <sub>25</sub>			
X <sub>13</sub>	X <sub>24</sub>	X <sub>35</sub>		
X <sub>12</sub>	X <sub>23</sub>	X <sub>34</sub>	X <sub>45</sub>	
X <sub>11</sub>	X <sub>22</sub>	X <sub>33</sub>	X <sub>44</sub>	X <sub>55</sub>
	b	a	a	b
		a		a

Tabel diatas merupakan tabel dengan panjang string masukan 5. Dari tabel tersebut, X<sub>15</sub> adalah akar, sehingga bila akar tersebut mengandung start symbol, maka string tersebut merupakan bagian dari grammar G.

Misalkan suatu CFG didefinisikan sebagai

$$G = (\{S, A, B, C\}, \{a, b\}, P, S)$$

G mempunyai aturan produksi:

$S \rightarrow AB \mid BC$

$A \rightarrow BA \mid a$

$B \rightarrow CC \mid b$

$C \rightarrow AB \mid a$

String masukan 'baaba; diterima oleh grammar tersebut, karena akar mengandung start symbol S.

## 5. BAHASA PEMROGRAMAN JAVASCRIPT

JavaScript adalah suatu bahasa pemrograman tingkat tinggi dan dinamis. JavaScript populer di internet dan dapat bekerja di sebagian besar penjelajah web populer seperti Google Chrome, Internet Explorer(IE), Mozilla Firefox, Netscape dan Opera. Kode JavaScript dapat disisipkan dalam halaman menggunakan tag *script*. JavaScript merupakan salah satu teknologi inti World Wide Web selain HTML dan CSS. JavaScript membantu membuat halaman web interaktif dan merupakan bagian aplikasi web yang esensial.

Awalnya hanya diimplementasi sebagai *client-side* dalam penjelajah web, kini *engine* JavaScript disisipkan ke dalam perangkat lunak lain seperti dalam *server-side* dalam server web dan basis data, dalam program non web seperti perangkat lunak

pengolah kata dan pembaca PDF, dan sebagai *runtime environment* yang memungkinkan penggunaan JavaScript untuk membuat aplikasi desktop maupun mobile. JavaScript adalah merek dagang yang dikeluarkan dari Oracle Corporation di Amerika Serikat.

## 6. PYTHON

Python adalah bahasa pemrograman tingkat tinggi yang ditafsirkan, berorientasi objek, dengan semantik dinamis. Pemrograman tingkat tinggi yang dibangun dalam struktur data, dikombinasikan dengan pengetikan dinamis dan pengikatan dinamis, membuatnya sangat menarik untuk pengembangan aplikasi secara cepat, serta digunakan sebagai bahasa scripting untuk menghubungkan komponen yang ada bersama-sama. Sintaksis Python yang sederhana dan mudah dipelajari menekankan keterbacaan dan karenanya mengurangi biaya pemeliharaan program. Python mendukung modul dan paket, yang mendorong modularitas program dan penggunaan kembali kode. Interpreter Python dan pustaka standar yang luas tersedia dalam bentuk sumber atau biner tanpa biaya untuk semua platform utama, dan dapat didistribusikan secara bebas (<https://www.python.org/about/apps/>). Python juga dapat dikolaborasikan dengan beberapa bahasa pemrograman seperti Java, C++, Javascript.

## BAB II

### ANALISIS PERSOALAN DAN DEKOMPOSISI

#### 2.1 CFG Production

Berikut adalah context free grammar yang telah kami buat terkait membuat parser bahasa Java

$$G = (V, T, P, S)$$

NONTERMINAL SYMBOL (V)

BREAK_STATE	COMMENT	FUNC_STATE
S	KALIMAT	FUNC_SENTENCE
SENTENCE	ENTER	RETURN_SENTENCE
SENTENCE_LOOP	BOOLEAN	PARAM
EXPRESI	ID	EXPRESI_FOR
ASSIGNMENT	VAR_STATE	FOR_STATE
OPP_LGC	DICTIONARY	CONTINUE_STATE
OPP_ART	DICT_SENTENCE	VAR_STATE
OPP_LOOP	DELETE_STATE	WHILE_STATE
IF_LOOP	LET_STATE	FINALLY_STATE
ELSE_IF_STATE_LOOP	CONST_STATE	FOR_STATE
ELSE_STATE_LOOP	THROW_STATE	SWITCH_STATE
IF_STATE	TRY_STATE	CASE_STATE
ELSE_IF_STATE	CATCH_STATE	CASE_SENTENCE
STATIC	ELSE_STATE	ELSE_IF_BANYAK

#### TERMINAL SYMBOL (T)

CONST	STRICT_NOT_EQUAL	AND
LET	MULTIPLY_EQUAL	OR
VAR	DIVIDE_EQUAL	TRUE
EQUAL	MODULO_EQUAL	FALSE
IS_EQUAL	PLUS_EQUAL	IF
STRICT_EQUAL	MINUS_EQUAL	ELSE
GREATER_EQUAL	PLUS	FOR
GREATER	MINUS	WHILE
LESS_EQUAL	DIVIDE	FUNCTION
LESS	MODULO	RETURN
NOT_EQUAL	MULTIPLY	BREAK
NOT	POWER	CONTINUE
TRY	THROW	DEFAULT
FINALLY	SWITCH	DELETE
CATCH	CASE	NULL
OPEN_ROUND_BRACKET	OPEN_CURLY_BRACKET	SEMICOLON
CLOSE_ROUND_BRACKET	CLOSE_CURLY_BRACKET	COLON
COMMA	DOT	VARIABLE
INTEGER	STRING	NONE
NEWLINE		

#### PRODUCTION (P)

Production yang kami buat sejumlah 158 baris yang telah mengimplementasikan CFG, Terminal serta non terminal symbol nya yang nantinya akan diolah menjadi CNF.

#### START SYMBOLS (S)

Start symbol yang kami pakai ialah S sehingga akan dilakukan pengecekan apabila sebuah grammar bisa menuju S maka grammar itu akan diterima.

## **2.2 CNF (Chomsky Normal Form)**

Chomsky Normal Form yang dibuat merupakan hasil dari Context Free Grammar yang di-convert dengan fungsi yang telah dibuat untuk kelompok kami. Dikarenakan jumlah dari CNF yang dihasilkan terlalu banyak untuk dimuat di laporan, maka hasil dari CNF yang dihasilkan tidak dimuat di laporan ini.

## **2.3 FA Lexer**

Parsing tiap karakter pada file input menjadi token dilakukan dengan menggunakan prinsip finite automata namun dengan modifikasi sesuai kebutuhan. Token-token yang dihasilkan inilah nantinya yang akan digunakan oleh parser CYK untuk dibandingkan. Oleh karena itu token juga menjadi terminal dalam grammar. Token-token terbagi menjadi sebagai berikut



## **BAB III**

### **IMPLEMENTASI DAN PENGUJIAN**

#### **IMPLEMENTASI**

##### **3.1 File grammar\_conv.py**

File grammar\_conv.py berisi mengenai prosedur dan fungsi dalam mengonversi bentuk CFG menjadi bentuk CNF. File ini akan membaca sebuah *grammar* dalam file grammar.txt.

##### **3.2 File file\_pros.py**

File file\_pros.py berisi mengenai define dari symbol dan keyword dari terminal. Juga membaca teks dan mengubahnya dalam bentuk token serta membuat token agar dapat dibaca di main program.

##### **3.3 File grammar\_pros**

File grammar\_pros.py berisi 3 fungsi. Yang pertama yaitu membaca file dan mengubahnya ke dalam bentuk CFG, lalu yang kedua adalah, mendefinisikan string dalam terminal, dan yang terakhir mendefinisikan variabel string.

##### **3.4 File parser\_grammar.py**

File parser\_grammar.py berisi fungsi parser\_CYK yang digunakan untuk parsing string dengan CNF grammar yang telah dibuat dalam grammar\_conv.py.

#### **HASIL PENGUJIAN**

##### **1. Hasil Finite Automata**

Pada tahap awal, finite automata digunakan untuk membaca file Java dan mengubahnya ke dalam terminal. Tiap karakter dari file dibaca, tiap pembacaan dimasukkan pada stack-stack tertentu. Pembacaan yang tepat akan memasukkan komponen program ke sebuah stack yang pasti, jika stack pembacaan tak tersedia, maka hasil pembacaan akan dimasukkan dalam stack illegal.

Pembacaan dibagi menjadi dua sistem, yakni keywords tertentu, serta char tertentu. Char yang sesuai dengan simbol tertentu akan dimasukkan ke dalam bentuk simbol tersebut (token), sedangkan string yang match dengan keywords tertentu (syntax) akan dimasukkan atau dikelompokkan pada stack tersebut. Susunan character

yang tak illegal, tetapi tak match dengan keywords akan dibaca sebagai identifier (variabel).

## 2. Hasil CFG

Pada program ini, grammar atau CFG disimpan dalam bentuk .txt. CFG diparsing menjadi right hand side and left hand side yang kemudian dipecah menjadi bagian non terminal serta terminal. Pemecahan dari CFG akan disimpan dalam .txt berjudul CNF. Seperti yang dijelaskan pada bagian teori dasar, CNF ini lah yang digunakan komputer sebagai panduan dalam menjalankan algoritma CYK.

Sebelum menjalankan CYK, CNF disimpan dengan pasangan key value dalam bentuk dictionary. Dimana left hand side menjadi value dan right hand side menjadi key. CYK akan mencocokkan pasangan key value tadi dengan bentukan terminal pada tabel CYK.

## 3. Pengerjaan Tugas Besar

Tugas besar ini dikerjakan pada repository: <https://github.com/syrifaa/Tubes-TBFO.git>

Pembagian tugas anggota kelompok:

### 1. Afnan Edsa Ramadhan (13521011)

- grammar\_pros.py
- parser\_grammar.py
- grammar.txt
- laporan

### 2. Laila Bilbina Khoiru Nisa (13521016)

- grammar\_conv.py
- grammar.txt
- laporan

### 3. Syarifa Dwi Purnamasari (13521018)

- file\_pros.py
- grammar.txt
- parser\_grammar.py
- Laporan

## BAB IV

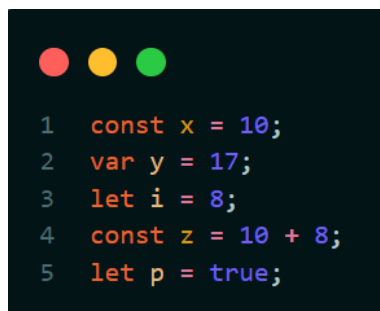
### HASIL PENGUJIAN

#### 4.1 Hasil Pengujian

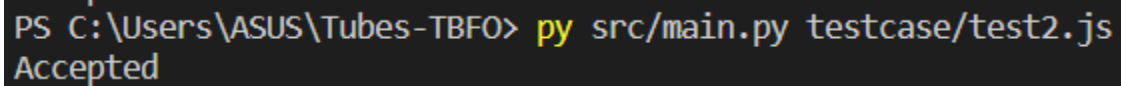
Berikut hasil pengujian terhadap program kami,

a. Input dan Output Sederhana Pada pengujian pertama,

program diuji dengan operasi sederhana. Program diuji dengan source code berisi input output serta operasi matematika sederhana, sebagai berikut.



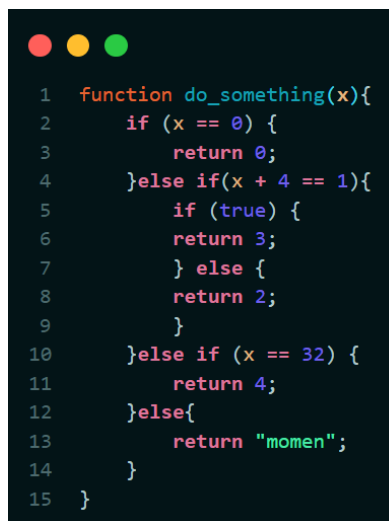
```
1  const x = 10;
2  var y = 17;
3  let i = 8;
4  const z = 10 + 8;
5  let p = true;
```



```
PS C:\Users\ASUS\Tubes-TBFO> py src/main.py testcase/test2.js
Accepted
```

b. Fungsi serta Conditional

Pada pengujian kedua, program diuji dengan syntax berupa kondional serta fungsi sederhana, sebagai berikut



```
1  function do_something(x){
2    if (x == 0) {
3      return 0;
4    }else if(x + 4 == 1){
5      if (true) {
6        return 3;
7      } else {
8        return 2;
9      }
10   }else if (x == 32) {
11     return 4;
12   }else{
13     return "momen";
14   }
15 }
```

```
PS C:\Users\ASUS\Tubes-TBFO> py src/main.py testcase/test.js
Accepted
```

### c. Iterasi

Pada pengujian ketiga, program diuji dengan syntax berupa iterasi, sebagai berikut

```
1 while(true){
2     for(var i = 0; i < 10; i++){
3         if(i == 3){
4             break;
5         }
6     }
7 }
```

```
PS C:\Users\ASUS\Tubes-TBFO> py src/main.py testcase/test1.js
Accepted
```

### d. Contoh program yang menghasilkan syntax error

```
1 function do_something(x){
2     if (x == 0) {
3         return 0;
4     }else if x + 4 == 1{
5         if (true) {
6             return 3;
7         } else {
8             return 2;
9         }
10    }else if (x == 32) {
11        return 4;
12    }else{
13        return "momen";
14    }
15 }
```

```
PS C:\Users\ASUS\Tubes-TBFO> py src/main.py testcase/test3.js
Syntax Error
```

## BAB V

### PENUTUP

#### 5.1 Simpulan

Dari tugas besar berjudul Parser Bahasa JavaScript maka terbukti tugas yang kami buat berjalan, meskipun masih terdapat banyak kekurangan. Dari tugas ini, dapat ditarik kesimpulan, bahwa setiap proses sangat penting dan berpengaruh pada keberjalan program, baik preprocessing hingga implementasi.

Preprocessing yang buruk akan menyebabkan proses parsing terhambat akibat dari banyaknya noise pada text yang dianalisis. Pembacaan parsing akan error jika noise pada terminal terjadi cukup fatal.

Grammar harus di convert menjadi pasangan key value yang lebih dipahami komputer, sehingga syarat butuh dari program ini adalah program yang dapat memecah grammar secara universal. Algoritma CYK merupakan algoritma berjenis dynamic programming. Sehingga perlu diperhatikan kompleksitasnya.

#### 5.2 Saran

Berikut adalah saran-saran yang kami himpun mengenai tubes besar ini.

1. Banyak menggali mengenai implementasi regular expression serta materi lain pada mata kuliah TBFO.
2. Mempunyai jiwa pembelajar yang kuat dan tangguh dalam mempelajari hal-hal baru.
3. Tingkatkan komunikasi antar anggota kelompok.
4. Jangan mudah menyerah, mulailah dari hal-hal kecil terlebih dahulu.

## BAB VI

### DAFTAR PUSTAKA

Eisele, R. (n.d.). *The CYK Algorithm • Computer Science and Machine Learning*.

<https://www.xarg.org/tools/cyk-algorithm/>

GeeksforGeeks. (2022, June 22). *Cocke–Younger–Kasami (CYK) Algorithm*.

<https://www.geeksforgeeks.org/cocke-younger-kasami-cyk-algorithm/>

GeeksforGeeks. (2019, May 21). *Converting Context Free Grammar to Chomsky*

*Normal Form*.

<https://www.geeksforgeeks.org/converting-context-free-grammar-chomsky-normal-form/>

[l-form/](https://www.geeksforgeeks.org/converting-context-free-grammar-chomsky-normal-form/)

*Python RegEx (With Examples)*. (n.d.).

<https://www.programiz.com/python-programming/regex>