

# **LAPORAN**

## **TUGAS BESAR IF2124 TEORI BAHASA FORMAL DAN OTOMATA**

**SEMESTER I TAHUN 2021/2022**



**ANGGOTA KELOMPOK :**

Diky Restu Maulana	13520017
Hana Fathiyah	13520047
Yohana Golkaria Nainggolan	13520053

**PROGRAM STUDI TEKNIK INFORMATIKA**

**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA**

**INSTITUT TEKNOLOGI BANDUNG**

**2021**

# DAFTAR ISI

<b>DAFTAR ISI.....</b>	<b>2</b>
<b>BAB I.....</b>	<b>3</b>
<b>DASAR TEORI.....</b>	<b>3</b>
1.1. Python .....	3
1.2. Context-Free Grammar (CFG).....	3
1.3. Chomsky Normal Form (CNF) .....	3
1.4. Algoritma CYK (Cocke-Younger-Kasami) .....	4
<b>BAB II .....</b>	<b>5</b>
<b>ANALISIS PERSOALAN .....</b>	<b>5</b>
2.1. Dasar Permasalahan .....	5
2.2. Grammar CFG.....	5
2.3. Grammar CNF .....	7
<b>BAB III.....</b>	<b>13</b>
<b>SPESIFIKASI TEKNIS PROGRAM .....</b>	<b>13</b>
<b>BAB IV .....</b>	<b>15</b>
<b>HASIL EKSPERIMEN .....</b>	<b>15</b>
<b>BAB V .....</b>	<b>19</b>
<b>KESIMPULAN DAN SARAN .....</b>	<b>19</b>
<b>PEMBAGIAN TUGAS .....</b>	<b>20</b>
<b>REFERENSI.....</b>	<b>21</b>

# BAB I

## DASAR TEORI

### 1.1. Python

Python diciptakan oleh Guido van Rossum di Belanda. Python adalah salah satu bahasa pemrograman yang dapat melakukan eksekusi sejumlah instruksi multi guna secara langsung (interpretatif) dengan metode orientasi objek (Object Oriented Programming) serta menggunakan semantik dinamis untuk memberikan tingkat keterbacaan syntax. Kode-kode yang ada pada Python mudah dibaca dan dapat menjalankan banyak fungsi kompleks dengan mudah karena banyaknya standard library. Namun, Python cukup lambat untuk dijalankan.

### 1.2. Context-Free Grammar (CFG)

Context-Free Grammar adalah sebuah bahasa formal yang digunakan untuk menerima sebuah *language*. Context-Free Grammar didefinisikan sebagai quadruple, yaitu:

$$G = (V, \Sigma, R, S)$$

dengan :

1.  $V$  adalah Non-Terminal Symbol
2.  $\Sigma$  adalah Terminal Symbol
3.  $R$  adalah Production Set
4.  $S$  adalah Start Symbol
5.  $G$  adalah Context-Free Grammar

### 1.3. Chomsky Normal Form (CNF)

CNF adalah salah satu bentuk dari CFG dimana setiap hasil produksi dari CFG berbentuk seperti salah satu dari di bawah ini:

1.  $A \rightarrow BC$  (bagian sebelah kanan keduanya berupa variable)
2.  $A \rightarrow a$  (bagian sebelah kanannya adalah Single Terminal)

#### 1.4. Algoritma CYK (Cocke-Younger-Kasami)

CYK (Cocke-Younger-Kasami) adalah algoritma parsing yang sangat efisien untuk Context-Free Grammar. Oleh karena itu, algoritma ini ideal untuk menentukan word-problem untuk Context-Free Grammar yang diberikan dalam bentuk CNF. Tool ini bisa digunakan untuk mengecek apakah sebuah kata tertentu merupakan bagian dari sebuah language, yang diberikan dalam bentuk CNF. Penemu algoritma ini ada 3 orang, yang kemudian nama ketiganya digunakan untuk menamai algoritma ini.

Algoritma ini bekerja dengan cara sebagai berikut:

- Tulis kata di kolom pertama dan tambahkan non-terminal symbol di bawah baris pertama.
- Kemudian, untuk setiap sel di dalam grid cek secara vertical mulai dari atas ke bawah dan sel kedua dari atas secara diagonal.
- Gabungkan sel dan cek apakah gabungannya muncul di dalam grammar.
- Jika muncul di dalam grammar, tambahkan bagian non-terminal sebelah kiri ke grid-cell.
- Jika setelah semua Langkah-langkah selesai dilakukan dan start symbol ada di baris terakhir, kata tersebut dapat diturunkan dari grammar tersebut.

Implementasinya menggunakan tabel:

	a	a	a	b	b	b	c	c
C	C	C	C	D	D	D	B, E	B, E
			A				B	
			F					
		A						
		F						
	A							
	S							
	S							

Grammar:

```
S -> AB
A -> CD | CF
B -> c | EB
C -> a
D -> b
E -> c
F -> AD
```

## BAB II

### ANALISIS PERSOALAN

#### 2.1. Dasar Permasalahan

Pada tugas besar Teori Bahasa Formal dan Otomata ini, mahasiswa diminta untuk membuat sebuah compiler python dengan memanfaatkan algoritma CYK, CNF, CFG, dan FA.

Compiler ini akan mengecek statement-statement dan juga sintaks-sintaks yang terdapat pada python. Grammar yang disediakan untuk parsing harus memuat 24 dari 33 kata kunci yang merupakan bawaan dari bahasa Python sendiri. Kata kunci yang harus dimuat dalam grammar, dapat dilihat pada tabel di bawah ini:

False	As	def	from	is	raise
None	Break	elif	if	not	return
True	Class	else	import	or	while
And	Continue	for	in	pass	with

*Compiler* akan menerima input file eksternal. Kemudian *compiler* akan membaca file eksternal yang diinput dan mengeluarkan pesan sesuai dengan hasil pembacaan yang diperoleh oleh *compiler* menggunakan *grammar* yang telah diberikan. Bila sintaks dan statement dari file yang diinput dinilai benar, *compiler* akan mengeluarkan statement “Accepted”, jika tidak, *compiler* akan mengeluarkan pesan “Syntax Error”.

#### 2.2. Grammar CFG

S -> IMPORT_METHOD	S -> VAR_METHOD	S -> OBJECT
S -> VAR_ASSIGNMENT	S -> WITH_METHOD	IMPORT_METHOD -> FROM_OBJ
S -> DEF_METHOD	S -> CLASS_METHOD	IMPORT_OBJ AS_OBJ
S -> RETURN_METHOD	S -> LOOP_BREAK	IMPORT_METHOD -> FROM_OBJ
S -> IF_METHOD	S -> LOOP_CONTINUE	IMPORT_OBJ
S -> ELIF_METHOD	S -> PASS	IMPORT_METHOD -> IMPORT_OBJ
S -> ELSE_METHOD	S -> RAISE_METHOD	AS_OBJ
S -> FOR_METHOD	S -> FUNC	IMPORT_METHOD -> IMPORT_OBJ
S -> WHILE_METHOD	S -> EXPRESSION	

VAR_ASSIGNMENT -> OBJECT	CLASS_METHOD -> CLASS OBJECT	EXPRESSION -> NUM
ASSOP EXPRESSION	COLON	EXPRESSION -> BOOLEAN
VAR_ASSIGNMENT -> OBJECT	CLASS_METHOD -> CLASS FUNC	EXPRESSION -> OBJECT
ASSIGNMENT EXPRESSION	TYPE_HINTING_COLON	EXPRESSION -> EXPRESSION COMP
DEF_METHOD -> DEF FUNC COLON	CLASS_METHOD -> CLASS OBJECT	EXPRESSION
DEF_METHOD -> DEF FUNC	TYPE_HINTING_COLON	EXPRESSION -> EXPRESSION OP
TYPE_HINTING_COLON	RAISE_METHOD -> RAISE OBJECT	EXPRESSION
RETURN_METHOD -> RETURN	FUNC -> OBJECT IN_PAREN	EXPRESSION -> FUNC
EXPRESSION	TYPE_HINTING_COLON ->	EXPRESSION -> OBJ_DOT_OBJ
IF_METHOD -> IF EXP_COMP_EXP	TYPE_HINTING_TO COLON	EXPRESSION -> OBJ_DOT_FUNC
COLON	OBJ_DOT_OBJ -> OBJ_DOT	EXPRESSION -> IN_BRACKET
IF_METHOD -> IF BOOL_FALSE	OBJ_DOT	EXPRESSION -> IN_PAREN
COLON	OBJ_IN_FUNC -> OBJECT IN FUNC	EXPRESSION -> IN_CBRACKET
IF_METHOD -> IF BOOL_TRUE	OBJ_IN_OBJ -> OBJECT IN	EXPRESSION -> NUM OP NUM
COLON	OBJECT	EXPRESSION -> STRING
IF_METHOD -> IF BOOLEAN COLON	FUNC_AS_OBJ -> FUNC AS OBJECT	OP_MULTIPLY INTEGER
IF_METHOD -> IF IN_PAREN	FROM_OBJ -> FROM OBJECT	EXPRESSION -> STRING OP_PLUS
COLON	IMPORT_OBJ -> IMPORT OBJECT	STRING
IF_METHOD -> IF OBJECT COLON	AS_OBJ -> AS OBJECT	EXPRESSION -> OBJECT OP
IF_METHOD -> IF OBJ_IN_OBJ	IN_PAREN -> OPEN_PAREN	OBJECT
COLON	EXPRESSION_IN_PAREN	EXPRESSION -> BINOP NUM
IF_METHOD -> IF BOOLBINOP	CLOSE_PAREN	EXPRESSION -> BOOLBINOP
FUNC COLON	IN_PAREN -> OPEN_PAREN	BOOLEAN
IF_METHOD -> IF BINOP FUNC	CLOSE_PAREN	EXPRESSION -> BINOP OBJECT
COLON	IN_BRACKET -> OPEN_BRACKET	EXPRESSION -> NONE
ELIF_METHOD -> ELIF	EXPRESSION_IN_BRACKET	EXPRESSION -> EXPRESSION
EXP_COMP_EXP COLON	CLOSE_BRACKET	ASSIGNMENT NONE
ELIF_METHOD -> ELIF IN_PAREN	IN_BRACKET -> OPEN_BRACKET	NUM -> INTEGER
COLON	CLOSE_BRACKET	NUM -> SIGN INTEGER
ELIF_METHOD -> ELIF OBJECT	IN_CBRACKET -> OPEN_CBRACKET	NUM -> FLOAT
COLON	CLOSE_CBRACKET	NUM -> SIGN FLOAT
ELIF_METHOD -> ELIF	EXPRESSION_IN_BRACKET ->	OBJECT -> OBJECT DOT OBJECT
OBJ_IN_OBJ COLON	EXPRESSION	OBJECT -> OBJECT SEPARATOR
ELSE_METHOD -> ELSE COLON	EXPRESSION_IN_BRACKET ->	OBJECT
FOR_METHOD -> FOR OBJ_IN_FUNC	TYPE_HINTING	OBJECT -> OBJECT DOT FUNC
COLON	EXPRESSION_IN_PAREN ->	OBJECT -> OBJECT IS OBJECT
FOR_METHOD -> FOR OBJ_IN_OBJ	EXPRESSION	OBJECT -> OBJECT AS OBJECT
COLON	EXPRESSION_IN_PAREN ->	OBJECT -> OBJECT IN OBJECT
WHILE_METHOD -> WHILE	EXP_COMP_EXP	OBJECT -> OBJECT IN_BRACKET
IN_PAREN COLON	EXPRESSION_IN_PAREN ->	DOT_OBJ
WHILE_METHOD -> WHILE	EXP_ASSIGN_EXP	OBJECT -> OBJECT IN_BRACKET
EXPRESSION COLON	EXPRESSION_IN_PAREN ->	OBJECT -> OBJECT IN_PAREN
VAR_METHOD -> OBJECT DOT FUNC	EXPRESSION TYPE_HINTING	OBJECT -> OBJECT IN_PAREN
VAR_METHOD -> OBJECT ASSOP	EXPRESSION_IN_PAREN ->	DOT_OBJ
OBJECT	EXPRESSION TYPE_HINTING	OBJECT -> 'OBJECT'
VAR_METHOD -> OBJECT	SEPARATOR_EXP	STRING -> STRING DOT STRING
ASSIGNMENT OBJECT	EXPRESSION_IN_PAREN ->	STRING -> STRING DOT OBJECT
WITH_METHOD -> WITH	EXPRESSION SEPARATOR_EXP	STRING -> 'TYPE_STRING'
FUNC_AS_OBJ COLON	SEPARATOR_EXP -> SEPARATOR	INTEGER -> 'TYPE_INT'
CLASS_METHOD -> CLASS FUNC	EXPRESSION_IN_PAREN	FLOAT -> 'TYPE_FLOAT'
COLON	EXPRESSION -> STRING	BOOLEAN -> 'BOOL_TRUE'

BOOLEAN -> 'BOOL_FALSE'	STROP -> 'OP_PLUS'	CLOSE_BRACKET ->
NONE -> 'TYPE_NONE'	STROP -> 'OP_MULTIPLY'	'CLOSE_BRACKET'
TYPE_HINTING -> COLON TYPEH	BOOLBINOP -> 'BINOP_NEGATE'	OPEN_CBRACKET ->
TYPE_HINTING_TO -> TYPEH TO	BOOLBINOP -> 'LOP_NOT'	'OPEN_CBRACKET'
TYPEH	ASSOP -> 'ASSOP_PLUS'	CLOSE_CBRACKET ->
TYPEH -> 'TYPEH_DICT'	ASSOP -> 'ASSOP_MINUS'	'CLOSE_CBRACKET'
TYPEH -> 'TYPEH_LIST'	ASSOP -> 'ASSOP_MULTIPLY'	DOT -> 'DOT'
TYPEH -> 'TYPEH_INT'	ASSOP -> 'ASSOP_DIVIDE'	DOT_OBJ -> DOT OBJECT
TYPEH -> 'TYPEH_STR'	ASSOP -> 'ASSOP_MODULO'	SEPARATOR -> 'SEPARATOR'
TYPEH -> 'TYPEH_FLOAT'	ASSOP -> 'ASSOP_FLOOR_DIVIDE'	FROM -> 'FROM'
TYPEH -> 'TYPEH_BOOL'	ASSOP -> 'ASSOP_EXPONENTIAL'	IMPORT -> 'IMPORT'
TYPEH -> 'TYPEH_BYTES'	ASSIGNMENT -> 'ASSIGNMENT'	AS -> 'AS'
TYPEH_TO -> 'TYPEH_TO'	COMP -> 'COMP_EQUALS'	IN -> 'IN'
OP -> 'OP_PLUS'	COMP -> 'COMP_NOT_EQUALS'	IS -> 'IS'
OP -> 'OP_MINUS'	COMP -> 'COMP_GREATER_EQU'	LOOP_BREAK -> 'LOOP_BREAK'
OP -> 'OP_MULTIPLY'	COMP -> 'COMP_LESS_EQU'	LOOP_CONTINUE ->
OP -> 'OP_DIVIDE'	COMP -> 'COMP_GREATER_THAN'	'LOOP_CONTINUE'
OP -> 'OP_MODULO'	COMP -> 'COMP_LESS_THAN'	CLASS -> 'CLASS'
OP -> 'OP_FLOOR_DIVIDE'	COMP -> IS	DEF -> 'DEF'
OP -> 'OP_EXPONENTIAL'	EXP_COMP_EXP -> EXPRESSION	PASS -> 'PASS'
SIGN -> 'OP_PLUS'	COMP EXPRESSION	RETURN -> 'RETURN'
SIGN -> 'OP_MINUS'	EXP_ASSIGN_EXP -> EXPRESSION	IF -> 'IF'
BINOP -> 'BINOP_NEGATE'	ASSIGNMENT EXPRESSION	ELIF -> 'ELIF'
BINOP -> 'BINOP_XOR'	EXP_ASSIGN_EXP -> EXPRESSION	ELSE -> 'ELSE'
BINOP -> 'BINOP_LEFTSHIFT'	ASSOP EXPRESSION	FOR -> 'FOR'
BINOP -> 'BINOP_RIGHTSHIFT'	OPEN_PAREN -> 'OPEN_PAREN'	WHILE -> 'WHILE'
BINOP -> 'LOP_NOT'	CLOSE_PAREN -> 'CLOSE_PAREN'	RAISE -> 'RAISE'
COMP -> 'LOP_AND'	OPEN_BRACKET ->	WITH -> 'WITH'
COMP -> 'LOP_OR'	'OPEN_BRACKET'	COLON -> 'COLON'

## 2.3. Grammar CNF

S -> IMPORT OBJECT	S -> IF_METHOD8 COLON	S -> CLASS_METHOD31
S -> IMPORT OBJECT	S -> IF_METHOD7 COLON	TYPE_HINTING_COLON
S -> IMPORT_OBJ AS_OBJ	S -> IF_METHOD6 COLON	S -> CLASS_METHOD30
S -> FROM_OBJ IMPORT_OBJ	S -> IF_METHOD5 COLON	TYPE_HINTING_COLON
S -> IMPORT_METHOD0 AS_OBJ	S -> ELIF_METHOD19 COLON	S -> CLASS_METHOD29 COLON
S -> VAR_ASSIGNMENT2	S -> ELIF_METHOD18 COLON	S -> CLASS_METHOD28 COLON
EXPRESSION	S -> ELIF_METHOD17 COLON	S -> 'LOOP_BREAK'
S -> VAR_ASSIGNMENT1	S -> ELIF_METHOD16 COLON	S -> 'LOOP_CONTINUE'
EXPRESSION	S -> ELSE COLON	S -> 'PASS'
S -> DEF_METHOD4	S -> FOR_METHOD21 COLON	S -> RAISE OBJECT
TYPE_HINTING_COLON	S -> FOR_METHOD20 COLON	S -> OBJECT IN_PAREN
S -> DEF_METHOD3 COLON	S -> WHILE_METHOD23 COLON	S -> 'TYPE_STRING'
S -> RETURN EXPRESSION	S -> WHILE_METHOD22 COLON	S -> STRING54 OBJECT
S -> IF_METHOD15 COLON	S -> VAR_METHOD26 OBJECT	S -> STRING53 STRING
S -> IF_METHOD13 COLON	S -> VAR_METHOD25 OBJECT	S -> 'TYPE_INT'
S -> IF_METHOD11 COLON	S -> VAR_METHOD24 FUNC	S -> 'TYPE_FLOAT'
S -> IF_METHOD10 COLON	S -> WITH_METHOD27 COLON	S -> 'TYPE_INT'
S -> IF_METHOD9 COLON		S -> 'TYPE_FLOAT'

S -> SIGN FLOAT	S -> OBJ_DOT OBJ_DOT	EXPRESSION_IN_BRACKET ->
S -> SIGN INTEGER	S -> OPEN_BRACKET	'BOOL_FALSE'
S -> 'BOOL_FALSE'	CLOSE_BRACKET	EXPRESSION_IN_BRACKET ->
S -> 'BOOL_TRUE'	S -> IN_BRACKET36	'BOOL_TRUE'
S -> 'OBJECT'	CLOSE_BRACKET	EXPRESSION_IN_BRACKET ->
S -> OBJECT52 DOT_OBJ	S -> OPEN_PAREN CLOSE_PAREN	'OBJECT'
S -> OBJECT IN_PAREN	S -> IN_PAREN35 CLOSE_PAREN	EXPRESSION_IN_BRACKET ->
S -> OBJECT IN_BRACKET	S -> OPEN_CBRACKET	OBJECT52 DOT_OBJ
S -> OBJECT51 DOT_OBJ	CLOSE_CBRACKET	EXPRESSION_IN_BRACKET ->
S -> OBJECT50 OBJECT	S -> 'TYPE_NONE'	OBJECT IN_PAREN
S -> OBJECT49 OBJECT	S -> EXPRESSION44 NONE	EXPRESSION_IN_BRACKET ->
S -> OBJECT48 OBJECT	S -> BINOP OBJECT	OBJECT IN_BRACKET
S -> OBJECT47 FUNC	S -> BOOLBINOP BOOLEAN	EXPRESSION_IN_BRACKET ->
S -> OBJECT46 OBJECT	S -> BINOP NUM	OBJECT51 DOT_OBJ
S -> OBJECT45 OBJECT	S -> EXPRESSION43 OBJECT	EXPRESSION_IN_BRACKET ->
S -> OBJECT IN_PAREN	S -> EXPRESSION42 STRING	OBJECT50 OBJECT
S -> OBJ_DOT OBJ_DOT	S -> EXPRESSION41 INTEGER	EXPRESSION_IN_BRACKET ->
S -> OPEN_BRACKET	S -> EXPRESSION40 NUM	OBJECT49 OBJECT
CLOSE_BRACKET	S -> EXPRESSION39 EXPRESSION	EXPRESSION_IN_BRACKET ->
S -> IN_BRACKET36	S -> EXPRESSION38 EXPRESSION	OBJECT48 OBJECT
CLOSE_BRACKET	S -> 'OBJECT'	EXPRESSION_IN_BRACKET ->
S -> OPEN_PAREN CLOSE_PAREN	S -> OBJECT52 DOT_OBJ	OBJECT47 FUNC
S -> IN_PAREN35 CLOSE_PAREN	S -> OBJECT IN_PAREN	EXPRESSION_IN_BRACKET ->
S -> OPEN_CBRACKET	S -> OBJECT IN_BRACKET	OBJECT46 OBJECT
CLOSE_CBRACKET	S -> OBJECT51 DOT_OBJ	EXPRESSION_IN_BRACKET ->
S -> 'TYPE_NONE'	S -> OBJECT50 OBJECT	OBJECT45 OBJECT
S -> 'TYPE_INT'	S -> OBJECT49 OBJECT	EXPRESSION_IN_BRACKET ->
S -> 'TYPE_FLOAT'	S -> OBJECT48 OBJECT	OBJECT IN_PAREN
S -> 'TYPE_STRING'	S -> OBJECT47 FUNC	EXPRESSION_IN_BRACKET ->
S -> STRING54 OBJECT	S -> OBJECT46 OBJECT	OBJ_DOT OBJ_DOT
S -> STRING53 STRING	S -> OBJECT45 OBJECT	EXPRESSION_IN_BRACKET ->
S -> 'TYPE_INT'	IMPORT_METHOD -> IMPORT	OPEN_BRACKET CLOSE_BRACKET
S -> 'TYPE_FLOAT'	OBJECT	EXPRESSION_IN_BRACKET ->
S -> 'TYPE_INT'	EXPRESSION_IN_BRACKET ->	IN_BRACKET36 CLOSE_BRACKET
S -> 'TYPE_FLOAT'	'TYPE_STRING'	EXPRESSION_IN_BRACKET ->
S -> SIGN FLOAT	EXPRESSION_IN_BRACKET ->	OPEN_PAREN CLOSE_PAREN
S -> SIGN INTEGER	STRING54 OBJECT	EXPRESSION_IN_BRACKET ->
S -> 'BOOL_FALSE'	EXPRESSION_IN_BRACKET ->	IN_PAREN35 CLOSE_PAREN
S -> 'BOOL_TRUE'	STRING53 STRING	EXPRESSION_IN_BRACKET ->
S -> 'OBJECT'	EXPRESSION_IN_BRACKET ->	OPEN_CBRACKET CLOSE_CBRACKET
S -> OBJECT52 DOT_OBJ	'TYPE_INT'	EXPRESSION_IN_BRACKET ->
S -> OBJECT IN_PAREN	EXPRESSION_IN_BRACKET ->	'TYPE_NONE'
S -> OBJECT IN_BRACKET	'TYPE_FLOAT'	EXPRESSION_IN_BRACKET ->
S -> OBJECT51 DOT_OBJ	EXPRESSION_IN_BRACKET ->	'TYPE_INT'
S -> OBJECT50 OBJECT	'TYPE_INT'	EXPRESSION_IN_BRACKET ->
S -> OBJECT49 OBJECT	EXPRESSION_IN_BRACKET ->	'TYPE_FLOAT'
S -> OBJECT48 OBJECT	'TYPE_FLOAT'	EXPRESSION_IN_BRACKET ->
S -> OBJECT47 FUNC	EXPRESSION_IN_BRACKET -> SIGN	'TYPE_STRING'
S -> OBJECT46 OBJECT	FLOAT	EXPRESSION_IN_BRACKET ->
S -> OBJECT45 OBJECT	EXPRESSION_IN_BRACKET -> SIGN	STRING54 OBJECT
S -> OBJECT IN_PAREN	INTEGER	



EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_PAREN ->
STRING53 STRING	IN_PAREN35 CLOSE_PAREN	'OBJECT'
EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_PAREN ->
'TYPE_INT'	OPEN_CBRACKET CLOSE_CBRACKET	OBJECT52 DOT_OBJ
EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_PAREN -> OBJECT
'TYPE_FLOAT'	'TYPE_NONE'	IN_PAREN
EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_PAREN -> OBJECT
'TYPE_INT'	EXPRESSION44 NONE	IN_BRACKET
EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_PAREN ->
'TYPE_FLOAT'	BINOP OBJECT	OBJECT51 DOT_OBJ
EXPRESSION_IN_BRACKET -> SIGN	EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_PAREN ->
FLOAT	BOOLBINOP BOOLEAN	OBJECT50 OBJECT
EXPRESSION_IN_BRACKET -> SIGN	EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_PAREN ->
INTEGER	BINOP NUM	OBJECT49 OBJECT
EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_PAREN ->
'BOOL_FALSE'	EXPRESSION43 OBJECT	OBJECT48 OBJECT
EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_PAREN ->
'BOOL_TRUE'	EXPRESSION42 STRING	OBJECT47 FUNC
EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_PAREN ->
'OBJECT'	EXPRESSION41 INTEGER	OBJECT46 OBJECT
EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_PAREN ->
OBJECT52 DOT_OBJ	EXPRESSION40 NUM	OBJECT45 OBJECT
EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_PAREN -> OBJECT
OBJECT IN_PAREN	EXPRESSION39 EXPRESSION	IN_PAREN
EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_PAREN ->
OBJECT IN_BRACKET	EXPRESSION38 EXPRESSION	OBJ_DOT OBJ_DOT
EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_PAREN ->
OBJECT51 DOT_OBJ	COLON TYPEH	OPEN_BRACKET CLOSE_BRACKET
EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_PAREN ->	EXPRESSION_IN_PAREN ->
OBJECT50 OBJECT	'TYPE_STRING'	IN_BRACKET36 CLOSE_BRACKET
EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_PAREN ->	EXPRESSION_IN_PAREN ->
OBJECT49 OBJECT	STRING54 OBJECT	OPEN_PAREN CLOSE_PAREN
EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_PAREN ->	EXPRESSION_IN_PAREN ->
OBJECT48 OBJECT	STRING53 STRING	IN_PAREN35 CLOSE_PAREN
EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_PAREN ->	EXPRESSION_IN_PAREN ->
OBJECT47 FUNC	'TYPE_INT'	OPEN_CBRACKET CLOSE_CBRACKET
EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_PAREN ->	EXPRESSION_IN_PAREN ->
OBJECT46 OBJECT	'TYPE_FLOAT'	'TYPE_NONE'
EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_PAREN ->	EXPRESSION_IN_PAREN ->
OBJECT45 OBJECT	'TYPE_INT'	'TYPE_INT'
EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_PAREN ->	EXPRESSION_IN_PAREN ->
OBJECT IN_PAREN	'TYPE_FLOAT'	'TYPE_FLOAT'
EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_PAREN -> SIGN	EXPRESSION_IN_PAREN ->
OBJ_DOT OBJ_DOT	FLOAT	'TYPE_STRING'
EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_PAREN -> SIGN	EXPRESSION_IN_PAREN ->
OPEN_BRACKET CLOSE_BRACKET	INTEGER	STRING54 OBJECT
EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_PAREN ->	EXPRESSION_IN_PAREN ->
IN_BRACKET36 CLOSE_BRACKET	'BOOL_FALSE'	STRING53 STRING
EXPRESSION_IN_BRACKET ->	EXPRESSION_IN_PAREN ->	EXPRESSION_IN_PAREN ->
OPEN_PAREN CLOSE_PAREN	'BOOL_TRUE'	'TYPE_INT'

EXPRESSION_IN_PAREN ->	EXPRESSION_IN_PAREN ->	EXPRESSION -> OBJECT46 OBJECT
'TYPE_FLOAT'	'TYPE_NONE'	EXPRESSION -> OBJECT45 OBJECT
EXPRESSION_IN_PAREN ->	EXPRESSION_IN_PAREN ->	EXPRESSION -> OBJECT IN_PAREN
'TYPE_INT'	EXPRESSION44 NONE	EXPRESSION -> OBJ_DOT OBJ_DOT
EXPRESSION_IN_PAREN ->	EXPRESSION_IN_PAREN -> BINOP	EXPRESSION -> OPEN_BRACKET
'TYPE_FLOAT'	OBJECT	CLOSE_BRACKET
EXPRESSION_IN_PAREN -> SIGN	EXPRESSION_IN_PAREN ->	EXPRESSION -> IN_BRACKET36
FLOAT	BOOLBINOP BOOLEAN	CLOSE_BRACKET
EXPRESSION_IN_PAREN -> SIGN	EXPRESSION_IN_PAREN -> BINOP	EXPRESSION -> OPEN_PAREN
INTEGER	NUM	CLOSE_PAREN
EXPRESSION_IN_PAREN ->	EXPRESSION_IN_PAREN ->	EXPRESSION -> IN_PAREN35
'BOOL_FALSE'	EXPRESSION43 OBJECT	CLOSE_PAREN
EXPRESSION_IN_PAREN ->	EXPRESSION_IN_PAREN ->	EXPRESSION -> OPEN_CBRACKET
'BOOL_TRUE'	EXPRESSION42 STRING	CLOSE_CBRACKET
EXPRESSION_IN_PAREN ->	EXPRESSION_IN_PAREN ->	EXPRESSION -> 'TYPE_NONE'
'OBJECT'	EXPRESSION41 INTEGER	NUM -> 'TYPE_INT'
EXPRESSION_IN_PAREN ->	EXPRESSION_IN_PAREN ->	NUM -> 'TYPE_FLOAT'
OBJECT52 DOT_OBJ	EXPRESSION40 NUM	COMP -> 'IS'
EXPRESSION_IN_PAREN -> OBJECT	EXPRESSION_IN_PAREN ->	IMPORT_METHOD ->
IN_PAREN	EXPRESSION39 EXPRESSION	IMPORT_METHOD0 AS_OBJ
EXPRESSION_IN_PAREN -> OBJECT	EXPRESSION_IN_PAREN ->	IMPORT_METHOD0 -> FROM_OBJ
IN_BRACKET	EXPRESSION38 EXPRESSION	IMPORT_OBJ
EXPRESSION_IN_PAREN ->	EXPRESSION_IN_PAREN ->	IMPORT_METHOD -> FROM_OBJ
OBJECT51 DOT_OBJ	EXP_COMP_EXP55 EXPRESSION	IMPORT_OBJ
EXPRESSION_IN_PAREN ->	EXPRESSION_IN_PAREN ->	IMPORT_METHOD -> IMPORT_OBJ
OBJECT50 OBJECT	EXP_ASSIGN_EXP57 EXPRESSION	AS_OBJ
EXPRESSION_IN_PAREN ->	EXPRESSION_IN_PAREN ->	VAR_ASSIGNMENT ->
OBJECT49 OBJECT	EXP_ASSIGN_EXP56 EXPRESSION	VAR_ASSIGNMENT1 EXPRESSION
EXPRESSION_IN_PAREN ->	EXPRESSION -> 'TYPE_STRING'	VAR_ASSIGNMENT1 -> OBJECT
OBJECT48 OBJECT	EXPRESSION -> STRING54 OBJECT	ASSOP
EXPRESSION_IN_PAREN ->	EXPRESSION -> STRING53 STRING	VAR_ASSIGNMENT ->
OBJECT47 FUNC	EXPRESSION -> 'TYPE_INT'	VAR_ASSIGNMENT2 EXPRESSION
EXPRESSION_IN_PAREN ->	EXPRESSION -> 'TYPE_FLOAT'	VAR_ASSIGNMENT2 -> OBJECT
OBJECT46 OBJECT	EXPRESSION -> 'TYPE_INT'	ASSIGNMENT
EXPRESSION_IN_PAREN ->	EXPRESSION -> 'TYPE_FLOAT'	DEF_METHOD -> DEF_METHOD3
OBJECT45 OBJECT	EXPRESSION -> SIGN FLOAT	COLON
EXPRESSION_IN_PAREN -> OBJECT	EXPRESSION -> SIGN INTEGER	DEF_METHOD3 -> DEF_FUNC
IN_PAREN	EXPRESSION -> 'BOOL_FALSE'	DEF_METHOD -> DEF_METHOD4
EXPRESSION_IN_PAREN ->	EXPRESSION -> 'BOOL_TRUE'	TYPE_HINTING_COLON
OBJ_DOT OBJ_DOT	EXPRESSION -> 'OBJECT'	DEF_METHOD4 -> DEF_FUNC
EXPRESSION_IN_PAREN ->	EXPRESSION -> OBJECT52	RETURN_METHOD -> RETURN
OPEN_BRACKET CLOSE_BRACKET	DOT_OBJ	EXPRESSION
EXPRESSION_IN_PAREN ->	EXPRESSION -> OBJECT IN_PAREN	IF_METHOD -> IF_METHOD5 COLON
IN_BRACKET36 CLOSE_BRACKET	EXPRESSION -> OBJECT	IF_METHOD5 -> IF_EXP_COMP_EXP
EXPRESSION_IN_PAREN ->	IN_BRACKET	IF_METHOD -> IF_METHOD6 COLON
OPEN_PAREN CLOSE_PAREN	EXPRESSION -> OBJECT51	IF_METHOD6 -> IF_BOOL_FALSE
EXPRESSION_IN_PAREN ->	DOT_OBJ	IF_METHOD -> IF_METHOD7 COLON
IN_PAREN35 CLOSE_PAREN	EXPRESSION -> OBJECT50 OBJECT	IF_METHOD7 -> IF_BOOL_TRUE
EXPRESSION_IN_PAREN ->	EXPRESSION -> OBJECT49 OBJECT	IF_METHOD -> IF_METHOD8 COLON
OPEN_CBRACKET CLOSE_CBRACKET	EXPRESSION -> OBJECT48 OBJECT	IF_METHOD8 -> IF_BOOLEAN
	EXPRESSION -> OBJECT47 FUNC	IF_METHOD -> IF_METHOD9 COLON

IF_METHOD9 -> IF IN_PAREN	VAR_METHOD24 -> OBJECT DOT	IN_PAREN -> OPEN_PAREN
IF_METHOD -> IF_METHOD10	VAR_METHOD -> VAR_METHOD25	CLOSE_PAREN
COLON	OBJECT	IN_BRACKET -> IN_BRACKET36
IF_METHOD10 -> IF OBJECT	VAR_METHOD25 -> OBJECT ASSOP	CLOSE_BRACKET
IF_METHOD -> IF_METHOD11	VAR_METHOD -> VAR_METHOD26	IN_BRACKET36 -> OPEN_BRACKET
COLON	OBJECT	EXPRESSION_IN_BRACKET
IF_METHOD11 -> IF OBJ_IN_OBJ	VAR_METHOD26 -> OBJECT	IN_BRACKET -> OPEN_BRACKET
IF_METHOD -> IF_METHOD13	ASSIGNMENT	CLOSE_BRACKET
COLON	WITH_METHOD -> WITH_METHOD27	IN_CBRACKET -> OPEN_CBRACKET
IF_METHOD12 -> IF BOOLBINOP	COLON	CLOSE_CBRACKET
IF_METHOD13 -> IF_METHOD12	WITH_METHOD27 -> WITH	EXPRESSION_IN_PAREN ->
FUNC	FUNC_AS_OBJ	EXPRESSION TYPE_HINTING
IF_METHOD -> IF_METHOD15	CLASS_METHOD ->	EXPRESSION_IN_PAREN ->
COLON	CLASS_METHOD28 COLON	EXPRESSION_IN_PAREN37
IF_METHOD14 -> IF BINOP	CLASS_METHOD28 -> CLASS FUNC	SEPARATOR_EXP
IF_METHOD15 -> IF_METHOD14	CLASS_METHOD ->	EXPRESSION_IN_PAREN37 ->
FUNC	CLASS_METHOD29 COLON	EXPRESSION TYPE_HINTING
ELIF_METHOD -> ELIF_METHOD16	CLASS_METHOD29 -> CLASS	EXPRESSION_IN_PAREN ->
COLON	OBJECT	EXPRESSION SEPARATOR_EXP
ELIF_METHOD16 -> ELIF	CLASS_METHOD ->	SEPARATOR_EXP -> SEPARATOR
EXP_COMP_EXP	CLASS_METHOD30	EXPRESSION_IN_PAREN
ELIF_METHOD -> ELIF_METHOD17	TYPE_HINTING COLON	EXPRESSION -> EXPRESSION38
COLON	CLASS_METHOD30 -> CLASS FUNC	EXPRESSION
ELIF_METHOD17 -> ELIF	CLASS_METHOD ->	EXPRESSION38 -> EXPRESSION
IN_PAREN	CLASS_METHOD31	COMP
ELIF_METHOD -> ELIF_METHOD18	TYPE_HINTING COLON	EXPRESSION -> EXPRESSION39
COLON	CLASS_METHOD31 -> CLASS	EXPRESSION
ELIF_METHOD18 -> ELIF OBJECT	OBJECT	EXPRESSION39 -> EXPRESSION OP
ELIF_METHOD -> ELIF_METHOD19	RAISE_METHOD -> RAISE OBJECT	EXPRESSION -> EXPRESSION40
COLON	FUNC -> OBJECT IN_PAREN	NUM
ELIF_METHOD19 -> ELIF	TYPE_HINTING COLON ->	EXPRESSION40 -> NUM OP
OBJ_IN_OBJ	TYPE_HINTING_TO COLON	EXPRESSION -> EXPRESSION41
ELSE_METHOD -> ELSE COLON	OBJ_DOT_OBJ -> OBJ_DOT	INTEGER
FOR_METHOD -> FOR_METHOD20	OBJ_DOT	EXPRESSION41 -> STRING
COLON	OBJ_IN_FUNC -> OBJ_IN_FUNC32	OP_MULTIPLY
FOR_METHOD20 -> FOR	FUNC	EXPRESSION -> EXPRESSION42
OBJ_IN_FUNC	OBJ_IN_FUNC32 -> OBJECT IN	STRING
FOR_METHOD -> FOR_METHOD21	OBJ_IN_OBJ -> OBJ_IN_OBJ33	EXPRESSION42 -> STRING
COLON	OBJECT	OP_PLUS
FOR_METHOD21 -> FOR	OBJ_IN_OBJ33 -> OBJECT IN	EXPRESSION -> EXPRESSION43
OBJ_IN_OBJ	FUNC_AS_OBJ -> FUNC_AS_OBJ34	OBJECT
WHILE_METHOD ->	OBJECT	EXPRESSION43 -> OBJECT OP
WHILE_METHOD22 COLON	FUNC_AS_OBJ34 -> FUNC AS	EXPRESSION -> BINOP NUM
WHILE_METHOD22 -> WHILE	FROM_OBJ -> FROM OBJECT	EXPRESSION -> BOOLBINOP
IN_PAREN	IMPORT_OBJ -> IMPORT OBJECT	BOOLEAN
WHILE_METHOD ->	AS_OBJ -> AS OBJECT	EXPRESSION -> BINOP OBJECT
WHILE_METHOD23 COLON	IN_PAREN -> IN_PAREN35	EXPRESSION -> EXPRESSION44
WHILE_METHOD23 -> WHILE	CLOSE_PAREN	NONE
EXPRESSION	IN_PAREN35 -> OPEN_PAREN	EXPRESSION44 -> EXPRESSION
VAR_METHOD -> VAR_METHOD24	EXPRESSION_IN_PAREN	ASSIGNMENT
FUNC		NUM -> SIGN INTEGER

NUM -> SIGN FLOAT	OP -> 'OP_PLUS'	EXP_ASSIGN_EXP56 ->
OBJECT -> OBJECT45 OBJECT	OP -> 'OP_MINUS'	EXPRESSION ASSIGNMENT
OBJECT45 -> OBJECT DOT	OP -> 'OP_MULTIPLY'	EXP_ASSIGN_EXP ->
OBJECT -> OBJECT46 OBJECT	OP -> 'OP_DIVIDE'	EXP_ASSIGN_EXP57 EXPRESSION
OBJECT46 -> OBJECT SEPARATOR	OP -> 'OP_MODULO'	EXP_ASSIGN_EXP57 ->
OBJECT -> OBJECT47 FUNC	OP -> 'OP_FLOOR_DIVIDE'	EXPRESSION ASSOP
OBJECT47 -> OBJECT DOT	OP -> 'OP_EXPONENTIAL'	OPEN_PAREN -> 'OPEN_PAREN'
OBJECT -> OBJECT48 OBJECT	SIGN -> 'OP_PLUS'	CLOSE_PAREN -> 'CLOSE_PAREN'
OBJECT48 -> OBJECT IS	SIGN -> 'OP_MINUS'	OPEN_BRACKET ->
OBJECT -> OBJECT49 OBJECT	BINOP -> 'BINOP_NEGATE'	'OPEN_BRACKET'
OBJECT49 -> OBJECT AS	BINOP -> 'BINOP_XOR'	CLOSE_BRACKET ->
OBJECT -> OBJECT50 OBJECT	BINOP -> 'BINOP_LEFTSHIFT'	'CLOSE_BRACKET'
OBJECT50 -> OBJECT IN	BINOP -> 'BINOP_RIGHTSHIFT'	OPEN_CBRACKET ->
OBJECT -> OBJECT51 DOT_OBJ	BINOP -> 'LOP_NOT'	'OPEN_CBRACKET'
OBJECT51 -> OBJECT IN_BRACKET	COMP -> 'LOP_AND'	CLOSE_CBRACKET ->
OBJECT -> OBJECT IN_BRACKET	COMP -> 'LOP_OR'	'CLOSE_CBRACKET'
OBJECT -> OBJECT IN_PAREN	STROP -> 'OP_PLUS'	DOT -> 'DOT'
OBJECT -> OBJECT52 DOT_OBJ	STROP -> 'OP_MULTIPLY'	DOT_OBJ -> DOT OBJECT
OBJECT52 -> OBJECT IN_PAREN	BOOLBINOP -> 'BINOP_NEGATE'	SEPARATOR -> 'SEPARATOR'
OBJECT -> 'OBJECT'	BOOLBINOP -> 'LOP_NOT'	FROM -> 'FROM'
STRING -> STRING53 STRING	ASSOP -> 'ASSOP_PLUS'	IMPORT -> 'IMPORT'
STRING53 -> STRING DOT	ASSOP -> 'ASSOP_MINUS'	AS -> 'AS'
STRING -> STRING54 OBJECT	ASSOP -> 'ASSOP_MULTIPLY'	IN -> 'IN'
STRING54 -> STRING DOT	ASSOP -> 'ASSOP_DIVIDE'	IS -> 'IS'
STRING -> 'TYPE_STRING'	ASSOP -> 'ASSOP_MODULO'	LOOP_BREAK -> 'LOOP_BREAK'
INTEGER -> 'TYPE_INT'	ASSOP -> 'ASSOP_FLOOR_DIVIDE'	LOOP_CONTINUE ->
FLOAT -> 'TYPE_FLOAT'	ASSOP -> 'ASSOP_EXPONENTIAL'	'LOOP_CONTINUE'
BOOLEAN -> 'BOOL_TRUE'	ASSIGNMENT -> 'ASSIGNMENT'	CLASS -> 'CLASS'
BOOLEAN -> 'BOOL_FALSE'	COMP -> 'COMP_EQUALS'	DEF -> 'DEF'
NONE -> 'TYPE_NONE'	COMP -> 'COMP_NOT_EQUALS'	PASS -> 'PASS'
TYPE_HINTING -> COLON TYPEH	COMP -> 'COMP_GREATER_EQU'	RETURN -> 'RETURN'
TYPE_HINTING_TO -> TYPEH_TO	COMP -> 'COMP_LESS_EQU'	IF -> 'IF'
TYPEH	COMP -> 'COMP_GREATER_THAN'	ELIF -> 'ELIF'
TYPEH -> 'TYPEH_DICT'	COMP -> 'COMP_LESS_THAN'	ELSE -> 'ELSE'
TYPEH -> 'TYPEH_LIST'	EXP_COMP_EXP ->	FOR -> 'FOR'
TYPEH -> 'TYPEH_INT'	EXP_COMP_EXP55 EXPRESSION	WHILE -> 'WHILE'
TYPEH -> 'TYPEH_STR'	EXP_COMP_EXP55 -> EXPRESSION	RAISE -> 'RAISE'
TYPEH -> 'TYPEH_FLOAT'	COMP	WITH -> 'WITH'
TYPEH -> 'TYPEH_BOOL'	EXP_ASSIGN_EXP ->	COLON -> 'COLON'
TYPEH -> 'TYPEH_BYTES'	EXP_ASSIGN_EXP56 EXPRESSION	
TYPEH_TO -> 'TYPEH_TO'		

## **BAB III**

### **SPESIFIKASI TEKNIS PROGRAM**

Kami membuat compiler python berbasis Command Line Interface (CLI) dengan menggunakan CFG, CNF, dan CYK. Pada mulanya, kami mendesain grammar sesuai dengan spesifikasi program yang diberikan pada tugas besar mata kuliah Teori Bahasa Formal dan Otomata semester 1 2021/2022. Grammar yang berbentuk CFG tersebut kami ubah menjadi CNF menggunakan program CFG2CNF.py dengan mengambil referensi dari <https://github.com/RobMcH/CYK-Parser>, sebuah repository yang telah mendapat lisensi MIT. Setelah CFG tersebut berubah menjadi CNF, proses parsing pun dilakukan.

#### **Main Program**

Pada main program, kami menggunakan beberapa boolean, yaitu isBlockComment (untuk mendeteksi blok komentar), isSkipUntilNextBC, isDef (untuk mendeteksi blok fungsi atau prosedur), isAccepted (untuk mendeteksi apakah program tersebut sukses dikompilasi atau malah menghasilkan syntax error), dan isIfLevel (untuk mendeteksi blok percabangan dan levelnya).

Pertama, program kami meminta input berupa nama file yang dimasukkan ke dalam variabel bernama inputfile. Selanjutnya, program kami menggunakan file grammar yang dimasukkan ke dalam grammarfile. File grammar yang dimasukkan tersebut merupakan file CNF yang sudah kami bentuk melalui hasil konversi CFG menjadi CNF menggunakan program CFG2CNF.py dan disimpan dalam file cnf.txt. Pada awal program, kami melakukan import library isfile dari os.path sebagai isExist yang tujuannya adalah untuk mengecek apakah file uji tersebut ada atau tidak ada. Jika file tersebut tidak ada, maka program kami akan langsung berakhir dan mengeluarkan output berupa tulisan "File not exist!". Jika ternyata file tersebut ada dan ditemukan, maka program akan lanjut ke proses berikutnya.

Proses berikutnya adalah setup file konfigurasi menggunakan mesin kata. Mesin kata yang kami gunakan adalah lexer. Kami melakukan setup lexer yang dimasukkan ke dalam variable lx dengan memanggil fungsi Lexer dalam file lexer. Kami juga melakukan setup cyk yang dimasukkan ke dalam variable CYK dengan memanggil fungsi Parser dalam file cyk untuk melakukan parsing file grammar.

Selanjutnya, kami melakukan pembacaan file input dengan membaca inputfile sebagai file. File kami baca secara bertahap baris demi baris menggunakan `readlines()` yang disimpan di dalam variabel `lines` yang kemudian diproses secara per baris.

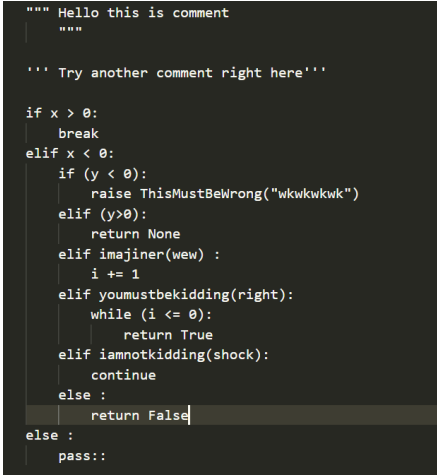
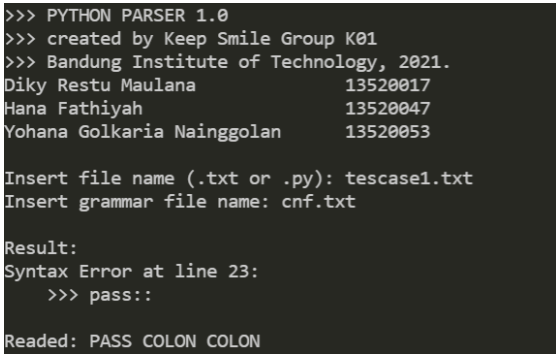
Setelah itu, dilakukan pemrosesan token. Token adalah bagian terkecil dari suatu bahasa. Pemrosesan token ini dilakukan dengan memanfaatkan regular expression. Setiap karakter diperiksa dan dilakukan parsing di setiap baris dengan mengabaikan whitespace. Selanjutnya dilakukan pencocokan dengan lexer rules yang disediakan dan menghasilkan sebuah kalimat.

Terakhir, kalimat tersebut dicocokkan dengan grammar. Jika tidak ditemukan di dalam grammar, program akan dihentikan dan menampilkan pesan “Syntax Error” disertai dengan letak kesalahannya. Jika belum ditemukan kesalahan, program akan terus berlanjut hingga baris terakhir dan menampilkan pesan “Accepted”.

## BAB IV

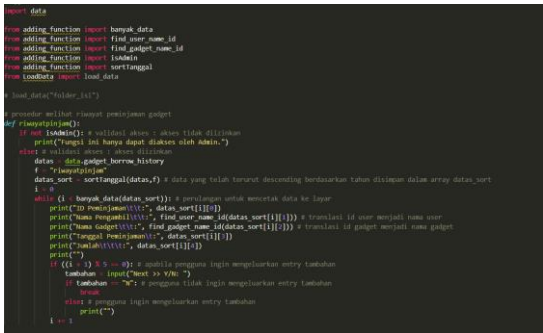
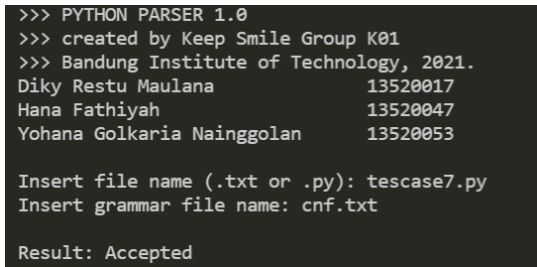
### HASIL EKSPERIMEN

Program	Hasil
Test case pada spesifikasi tugas besar TBFO:	
 <pre> inputAcc.py &gt; do_something 1 def do_something(x): 2     ''' This is a sample multiline comment 3     ... 4     if x == 0: 5         return 0 6     elif x + 4 == 1: 7         if True: 8             return 3 9         else: 10            return 2 11    elif x == 32: 12        return 4 13    else: 14        return "Doodoo" </pre>	 <pre> &gt;&gt;&gt; PYTHON PARSER 1.0 &gt;&gt;&gt; created by Keep Smile Group K01 &gt;&gt;&gt; Bandung Institute of Technology, 2021. Diky Restu Maulana      13520017 Hana Fathiyah           13520047 Yohana Golkaria Nainggolan 13520053  Insert file name (.txt or .py): inputAcc.py Insert grammar file name: cnf.txt  Result: Accepted </pre>
<p>Berdasarkan program di atas, dapat dilihat bahwa program tersebut benar secara syntax.</p>	<p>Hasil eksekusi program inputAcc.py menggunakan compiler yang kami buat adalah Accepted, dengan kata lain sesuai dengan kondisi program yang benar secara syntax.</p>
 <pre> inputReject.py &gt; ... 1 def do_something(x): 2     ''' This is a sample multiline comment 3     ... 4     x + 2 = 3 5     if x == 0 + 1: 6         return 0 7     elif x + 4 == 1: 8         else: 9             return 2 10    elif x == 32: 11        return 4 12    else: 13        return "Doodoo" </pre>	 <pre> &gt;&gt;&gt; PYTHON PARSER 1.0 &gt;&gt;&gt; created by Keep Smile Group K01 &gt;&gt;&gt; Bandung Institute of Technology, 2021. Diky Restu Maulana      13520017 Hana Fathiyah           13520047 Yohana Golkaria Nainggolan 13520053  Insert file name (.txt or .py): inputReject.py Insert grammar file name: cnf.txt  Result: Syntax Error at line 4: &gt;&gt;&gt; x + 2 = 3  Readed: OBJECT OP_PLUS TYPE_INT ASSIGNMENT TYPE_INT </pre>

<p>Program di atas sengaja dibuat keliru untuk menguji kebenaran compiler yang kami buat. Terdapat dua kesalahan, yaitu operator penjumlahan di sebelah kiri assignment dan <i>else</i> yang tidak didahului oleh <i>if</i>.</p>	<p>Hasil eksekusi program inputReject.py menggunakan compiler yang kami buat menampilkan pesan “Syntax Error at line 4”. Terbukti bahwa kesalahan syntax terdeteksi. Kesalahan yang ditampilkan adalah yang berada di baris teratas karena pengecekan dilakukan dari baris atas ke bawah.</p>
<p>Test case buatan sendiri</p>	
 <pre> """ Hello this is comment """  ''' Try another comment right here'''  if x &gt; 0:     break elif x &lt; 0:     if (y &lt; 0):         raise ThisMustBeWrong("wkwkwkwk")     elif (y&gt;0):         return None     elif imajiner(wew) :         i += 1     elif youmustbekidding(right):         while (i &lt;= 0):             return True     elif iamnotkidding(shock):         continue     else :         return False else :     pass: </pre>	 <pre> &gt;&gt;&gt; PYTHON PARSER 1.0 &gt;&gt;&gt; created by Keep Smile Group K01 &gt;&gt;&gt; Bandung Institute of Technology, 2021. Diky Restu Maulana      13520017 Hana Fathiyah           13520047 Yohana Golkaria Nainggolan 13520053  Insert file name (.txt or .py): tescase1.txt Insert grammar file name: cnf.txt  Result: Syntax Error at line 23:     &gt;&gt;&gt; pass::  Readed: PASS COLON COLON </pre>
<p>Program di atas sebenarnya program yang tidak memiliki makna. Program dibuat untuk menguji beberapa kata kunci python yang wajib ada dalam tugas besar ini. Program sengaja dibuat salah secara sintaks pada bagian <i>pass</i>.</p>	<p>Program menunjukkan sintaks error pada line 23 bagian <i>pass::</i> . Program salah karena seharusnya setelah <i>pass</i> tidak ada <i>colon</i>. Hasil eksekusi program sesuai dengan pengujian sintaks yang sebenarnya.</p>



<pre>import numpy as np import cv2 import vektor          # untuk cari nilai dan vektor eigen  def matrixSVD(M):     # mendekomposisi M menjadi U, s, dan V_T     M_T = np.transpose(M)      # transpose(M)     M_M_T = np.dot(M, M_T)     # M*transpose(M)     M_T_M = np.dot(M_T, M)     # transpose(M)*M</pre>	<pre>&gt;&gt;&gt; PYTHON PARSER 1.0 &gt;&gt;&gt; created by Keep Smile Group K01 &gt;&gt;&gt; Bandung Institute of Technology, 2021. Diky Restu Maulana      13520017 Hana Fathiyah           13520047 Yohana Golkaria Nainggolan 13520053  Insert file name (.txt or .py): testcase2.py Insert grammar file name: cnf.txt  Result: Accepted</pre>
<p>Program di atas mengimport modul eksternal sekaligus penggunaan fungsi di dalamnya. Program dibuat untuk menguji beberapa kata kunci Python yang wajib ada dalam tugas besar ini.</p>	<p>Hasil eksekusi adalah Accepted. Dengan kata lain, program tersebut benar secara syntax.</p>
<pre># input waktu = 1000  # waktu Hari = waktu // (24 * 3600) # 24 jam dikonversi menjadi detik print("Hari =", Hari) waktu %= 24 * 3600 Jam = waktu // 3600 print("Jam =", Jam) waktu %= 3600 Menit = waktu // 60 print("Menit =", Menit) waktu %= 60 Detik = waktu print("Detik =", Detik)</pre>	<pre>&gt;&gt;&gt; PYTHON PARSER 1.0 &gt;&gt;&gt; created by Keep Smile Group K01 &gt;&gt;&gt; Bandung Institute of Technology, 2021. Diky Restu Maulana      13520017 Hana Fathiyah           13520047 Yohana Golkaria Nainggolan 13520053  Insert file name (.txt or .py): testcase4.py Insert grammar file name: cnf.txt  Result: Accepted</pre>
<p>Program di atas berisi berbagai variasi operator, termasuk operator assignment. Tujuannya untuk menguji kebenaran syntax operan dan operator dalam Bahasa Python.</p>	<p>Hasil eksekusi adalah Accepted. Dengan kata lain, program tersebut benar secara syntax.</p>
<pre>testVarName.py &gt; ... 1  import math 2 3  # Fungsi kuadrat 4  # Mengembalikan hasil angka pertama dipangkatkan angka kedua 5  def pangkat(x,y): 6      1num = 99 7      2num = 46 8      result (1num ** 2num)</pre>	<pre>&gt;&gt;&gt; PYTHON PARSER 1.0 &gt;&gt;&gt; created by Keep Smile Group K01 &gt;&gt;&gt; Bandung Institute of Technology, 2021. Diky Restu Maulana      13520017 Hana Fathiyah           13520047 Yohana Golkaria Nainggolan 13520053  Insert file name (.txt or .py): testVarName.py Insert grammar file name: cfg.txt  Result: Syntax Error at line 6: &gt;&gt;&gt; 1num = 99  Readed: TYPE_INT OBJECT ASSIGNMENT TYPE_INT</pre>

<p>Program di atas bertujuan untuk menguji aturan penamaan variabel dalam Bahasa Python. Nama variabel sengaja dibuat salah, yakni diawali dengan angka.</p>	<p>Terdapat syntax error di baris 6, tepat di lokasi assignment variabel lnum. Program tidak mendeteksinya sebagai nama variabel, melainkan sebuah nilai bertipe integer. Kemudian, bagian huruf dideteksi sebagai object. Syntax semacam ini tidak diterima oleh compiler sehingga ditampilkan pesan kesalahan.</p>
	
<p>Program di atas cukup kompleks secara syntax. Kami menyertakan program ini sebagai salah satu testcase untuk menguji kemampuan compiler yang kami buat dalam membaca program yang rumit dan terdiri dari banyak baris.</p>	<p>Hasil eksekusi adalah Accepted. Dengan kata lain, program tersebut benar secara syntax.</p>

## **BAB V**

### **KESIMPULAN DAN SARAN**

Berdasarkan program yang telah kami buat, dapat disimpulkan bahwa compiler pada dasarnya dibentuk dari sekumpulan grammar yang disusun menggunakan Context Free Grammar (CFG) dan kemudian diubah menjadi Chomsky Normal Form (CNF) serta diolah menggunakan Cocke-Younger Kasami (CYK). Penggunaan regular expression digunakan dalam program lexer untuk mengoperasikan token. Mata kuliah Teori Bahasa Formal dan Otomata sangat bermanfaat dalam membentuk compiler suatu bahasa pemrograman yang dalam hal ini adalah bahasa Python. Saran yang dapat kami berikan adalah ke depannya penggunaan program ini dapat dikembangkan, sehingga dapat menghasilkan bahasa pemrograman baru yang lebih dekat dengan manusia.

## PEMBAGIAN TUGAS

NO.	NIM	NAMA	TUGAS
1.	13520017	Diky Restu Maulana	Grammar
2.	13520047	Hana Fathiyah	Convert CFG ke CNF
3.	13520053	Yohana Golkaria Nainggolan	CYK

Link ke repository github: <https://github.com/Yohanagn/TBFO-Python-Compiler-using-CYK>

## REFERENSI

Hopcroft, J., Motwani, R., & Ullman, J. (2006). *Introduction to Automata Theory, Languages, and Computation 3rd Edition*. Boston: Addison-Wesley Longman Publishing Co., Inc.

MChardy, R., & Leung, G. (2021, November 22). Retrieved from  
<https://github.com/RobMcH/CYK-Parser>

(n.d.). Retrieved from <https://www.xarg.org/tools/cyk-algorithm/>

(n.d.). Retrieved from  
[https://www.baktikominfo.id/id/informasi/pengetahuan/bahasa\\_pemrograman\\_python\\_pengertian\\_sejarah\\_kelebihan\\_dan\\_kekurangannya-954](https://www.baktikominfo.id/id/informasi/pengetahuan/bahasa_pemrograman_python_pengertian_sejarah_kelebihan_dan_kekurangannya-954)

(n.d.). Retrieved from <https://www.programiz.com/python-programming/keyword-list>