|  |
| --- |
| Simple Parser for Propositional Logic Formula (Tahap 2) |
| Laporan Tugas Program TBA |
| Dosen Pengampu : Said Al Faraby  Kelompok : 11 (Sebelas)  Anggota Kelompok :   1. Febry Ghaisani 1301154576 2. Wella Edli Shabrina 1301154562 3. Irfan Kurniawan 1301158587 |

23 Oktober 2016

**2016**

# Kelompok:

|  |  |  |
| --- | --- | --- |
| **No** | **NIM** | **Nama** |
| 1 | 1301154576 | Febry Ghaisani |
| 2 | 1301154562 | Wella Edli Shabrina |
| 3 | 1301158587 | Irfan Kurniawan |

# Spesifikasi Program

1. Studi Kasus

Membangun program Lexical Analyzer (Parser Sederhana) dengan menerapkan Finite Automata untuk mengenali setiap Lexic yang dituliskan dalam Formula Logika Proposisi, Context Free Grammar (CFG) dan Pushdown Automata (PDA) untuk menentukan apakah sesuai dengan penulisan formula logika proposisi atau tidak, lalu mengeluarkan output dalam bentuk pernyataan VALID atau TIDAK VALID berdasarkan string Formula yang dibaca.

1. Batasan Masalah

* Menggunakan 1 buah FA yang dapat mengenali semua string lexic.
* Program dibangun menggunakan bahasa pemrograman C++.
* Tidak menggunakan string matching.
* Dalam menjalankan fungsinya, program membaca satu per satu karakter/simbol dari string yang dibaca.
* Formula yang diterima bersifat case-sensitive, inputan yang diterima program adalah inputan yang lower case.
* Antar satu lexic dengan lexic sebelum/sesudahnya dapat dipisahkan dengan spasi maupun tanpa spasi.
* String Lexic yang dapat dikenali pada input Formula meliputi :
* Simbol proposisi: p, q, r, s.
* Simbol operator logika: not, and, or, xor, if, then, iff.
* Simbol kurung: (, ).
* Berikut ketentuan penulisan Formula yang akan dibaca:



1. Spesifikasi Kebutuhan Program

* PC/Laptop
* OS Minimum Windows 7
* Intel core 2 duos
* CodeBlocks v 16.xx
* RAM maksimal 2Gb

# Rancangan Finite Automata

Rancangan FA



Deskripsi Formal :

M = (Q, Σ, δ, q0, F)

Q = {q0, q1, q2, q3, q4, q5, q6, q7, q8, q9, q10, q11, q12}

Σ = {p, q, r, s, a, n, d, o, t, x, i, f, h, e, (, ), ɛ, space}

δ adalah tabel transisi



q0 adalah starting state

F = {q11,q12}

# Rancangan Context Free Grammar dan Pushdown Automata

1. CFG (Context Free Grammar)

G = ({X,A,B}, {1,2,3,4,5,6,7,8,9,10}, S, P)

S adalah starting symbol

dimana P adalah himpunan aturan produksi berikut :

S 🡪 X

X 🡪 1A | 2B | 6B | 9B | 1

A 🡪 3B | 4B | 5B | 7B | 8B | 10B | 10

B 🡪 1A | 2B | 9B | 1

1. PDA (Pushdown Automata)

GS, i, F)

S = {i, p, q, f}

Σ = {1,2,3,4,5,6,7,8,9,10}

= X,A,B,1,2,3,4,5,6,7,8,9,10,

i = i

F = f

Dan T adalah:

(i, λ, λ; p, #)

(p, λ, λ; q, S)

(q, λ, S; q, X)

(q, λ, X; q, 1A)

(q, λ, X; q, 2B)

(q, λ, X; q, 6B)

(q, λ, X; q, 9B)

(q, λ, X; q, 1)

(q, λ, A; q, 3B)

(q, λ, A; q, 4B)

(q, λ, A; q, 5B)

(q, λ, A; q, 7B)

(q, λ, A; q, 8B)

(q, λ, A; q, 10B)

(q, λ, A; q, 10)

(q, λ, B; q, 1A)

(q, λ, B; q, 2B)

(q, λ, B; q, 9B)

(q, λ, B; q, 1)

(q, 1, 1; q, λ)

(q, 2, 2; q, λ)

(q, 3, 3; q, λ)

(q, 4, 4; q, λ)

(q, 5, 5; q, λ)

(q, 6, 6; q, λ)

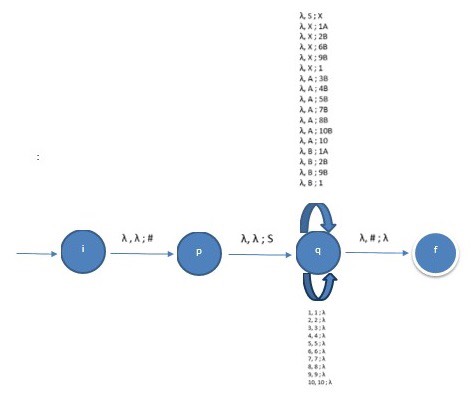
(q, 7, 7; q, λ)

(q, 8, 8; q, λ)

(q, 9, 9; q, λ)

(q, 10, 10; q, λ)

(q, λ, #; f, λ)



# Cara Kerja Program

Pertama user diminta menginputkan formula logika proposisi. Kemudian formula tersebut disimpan dalam variabel bertipe string. Setelah itu akan dilakukan pengecekan satu persatu karakter dengan menggunakan array karena string merupakan array of char. Dalam pengecekan tersebut menggunakan perulangan dari indeks pertama sampai indeks terakhir string tersebut dan didalam perulangan ada pengecekan dengan 2 kondisi dimana kondisi pertama yaitu jika karakter yang dicek adalah karakter spasi maka langsung cek karakter berikutnya, kondisi kedua yaitu jika karakter yang dicek merupakan anggota dari alphabet yang sudah kita tentukan sebelumnya maka akan diset nilai tokennya sesuai dengan jenis string lexicnya kemudian dimasukkan ke dalam array. Jika string lexicnya lebih dari satu karakter maka akan dicek lagi karakter berikutnya dengan menggunakan if..then. Jika sesuai maka akan diset nilai tokennya sesuai dengan jenis string lexicnya dan dimasukkan ke dalam array. Jika hasil pengecekan ternyata karakter tersebut tidak termasuk dalam alphabet yang sudah dibuat maka nilai token akan diset error atau pengecekan di set false. Jika pengecekan diset false maka perulangan akan langsung berhenti (tidak mengecek karakter berikutnya meskipun ada).

Setelah perulangan berhenti dan pengecekan bernilai true maka tahap selanjutnya adalah validasi. Pada tahap ini yang di cek adalah nilai token dalam array. Tahap pertama, isi array pertama akan dicek (yang diperbolehkan untuk mengawali formula logika proposisi adalah p,q,r,s,not,if dan tanda kurung buka) jika memenuhi aturan penulisan formula logika proposisi maka akan dilakukan pengecekan ke array berikutnya. Untuk pengecekan if..then dan “(“ dan “)” menggunakan stack sehingga terdapat dua array, satu untuk array token dan yang lainnya untuk stack. Kemudian pengecekan dilakukan sampai pengecekan berada di posisi terakhir dari array. Tahap terakhir adalah pengecekan isi stack, jika pengecekan bernilai true dan isi dari array index ke top dari stack adalah tanda pagar (“#”) maka formula proposisi yang diinputkan dapat disimpulkan VALID jika tidak memenuhi kedua kondisi tersebut maka disimpulkan TIDAK VALID.

# Pengujian Program

|  |  |  |
| --- | --- | --- |
| **Input** | **Output Program** | **Keterangan** |
| p and q or r | VALID | Sesuai |
| if p then (not q s) | TIDAK VALID | Sesuai karena setelah operand tidak boleh operand. |
| If P then q | TIDAK VALID | Sesuai karena dalam formula tersebut terdapat huruf kapital. |
| (p and q ifg(r or s)) | TIDAK VALID | Sesuai karena tidak ada operator ifg. |

# Screenshot Program

Beberapa screenshot program yang telah dibangun :

