|  |
| --- |
| **Simple Parser For Propositional Logic Formula (Tahap 1)** |
| Laporan Tugas Program TBA |
| Dosen Pengampu: Siti Sa’dah  Kelompok: 1  Anggota Kelompok:   1. Lazuardi Zhafran F 1301144275 2. William Adi A K 1301150786 3. M Thariq Sunu R 1301150775 |
|  |

12/6/2018

**2019**

# Spesifikasi Program

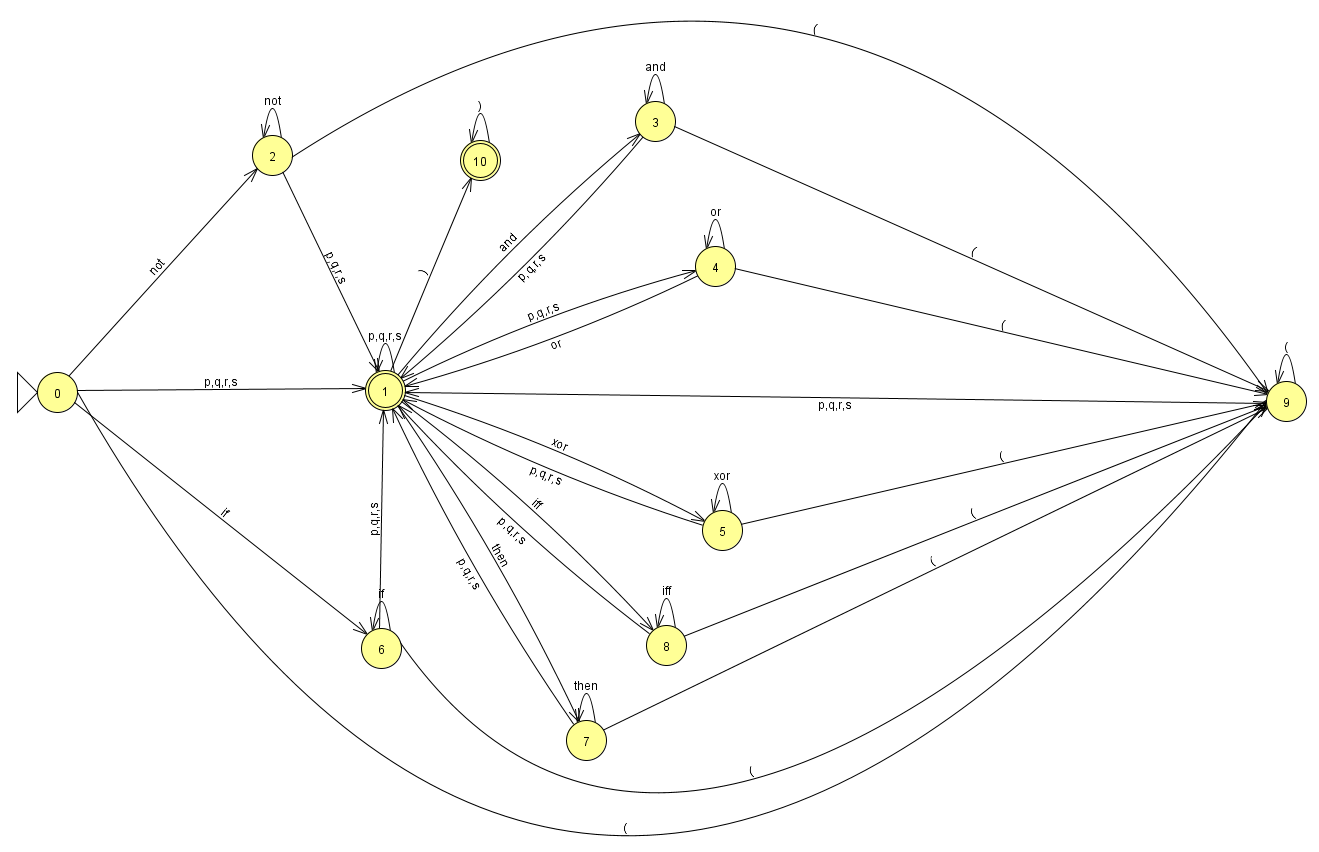
Program ini bertujuan untuk menganalisa formula logika proposisi dengan menerapkan finite automata untuk mengenali setiap lexic yang dituliskan dalam formula. Setelah itu program akan mengganti inputan string lexic menjadi sequence token berdasarkan data yang telah ditentukan. Jika terdapat inputan string yang tidak terdaftar program akan mengoutputkan error

Batasan masalah dari program ini adalah sebagai berikut :

|  |  |  |  |
| --- | --- | --- | --- |
| **String Lexic** | **Jenis** | **Token** | **Keterangan** |
| Proposisi:p,q,r,s | Operand | 1 | Hanya 1 simbol (diantara p,q,r,s) yang dikenal sebagai 1 proposisi |
| not | Operator | 2 | Contoh penulisan yang diterima:   * not proposisi * not (proposisi) |
| and | Operator | 3 | Contoh penulisan yang diterima:   * proposisi and proposisi * (proposisi) and (proposisi) * (formula) and (Formula) |
| or | Operator | 4 | Contoh penulisan yang diterima:   * proposisi or proposisi * (proposisi) or (proposisi) * (formula) or (formula) |
| xor | Operator | 5 | Contoh penulisan yang diterima:   * proposisi xor proposisi * (proposisi) xor (proposisi) * (formula) xor (formula) |
| if | Operator | 6 | Contoh penulisan yang diterima:   * if proposisi then proposisi * if (proposisi) then (proposisi) * if (formula) then (formula) |
| then | Operator | 7 |
| iff | Operator | 8 | Contoh penulisan yang diterima:   * proposisi iff proposisi * (proposisi) iff (proposisi) * (formula) iff (formula) |
| ( | Grouping | 9 | * Jumlah kurung buka dan tutup harus seimbang * dituliskan di tempat yang tepat |
| ) | Grouping | 10 |

# Rancangan Finite Automata

STD untuk mesin Finite Automata yang diimplementasikan.



\*0 : initial state

Definisi formal untuk STD FA :

STD FA terdiri dari 5-tuple (Q, ∑, δ, 0, F), dimana :

1. Q adalah himpunan terbatas dari states,
2. ∑ himpunan terbatas alphabet,
3. δ: Q × ∑ 🡪 Q fungsi transisi, dinotasikan dengan δ(q,a) 🡪 p
4. 0∈ Q adalah start state, dan
5. F ⊆ Q adalah himpunan accept states (atau final states).

Sehingga jika mengikuti FA yang telah dibangun, maka :

1. Q : 0,1,2,3,4, 5, 6, 7, 8, 9, 10
2. ∑ : p, q, r, s, not, and, or, xor, if, then, iff, (, )
3. δ : δ( 0,not) 🡪 2δ : δ( 1,() 🡪 9δ : δ( 4,() 🡪 9δ : δ( 7,() 🡪 9

δ : δ( 0,p,q,r,s) 🡪 1δ : δ( 1,)) 🡪 10δ : δ( 4,or) 🡪 4δ : δ( 7,then) 🡪 7

δ : δ( 0,if) 🡪 6δ : δ( 2,p,q,r,s) 🡪 1δ : δ( 5,p,q,r,s) 🡪 1δ : δ( 8,p,q,r,s) 🡪 1

δ : δ( 0,() 🡪 9δ : δ( 2,() 🡪 9 δ : δ( 5,() 🡪 9 δ : δ( 8,() 🡪 9

δ : δ( 0,p,q,r,s) 🡪 1δ : δ( 2,not) 🡪 2δ : δ( 5,xor) 🡪 5δ : δ( 8,iff) 🡪 8

δ : δ( 1,and) 🡪 3δ : δ( 3,,p,q,r,s) 🡪 1δ : δ( 6,p,q,r,s) 🡪 1δ : δ( 9,p,q,r,s) 🡪 1

δ : δ( 1,or) 🡪 4δ : δ( 3,() 🡪 9δ : δ( 6,() 🡪 9 δ : δ( 9,() 🡪 9

δ : δ( 1,xor) 🡪 5δ : δ( 3,and) 🡪 3δ : δ( 6,if) 🡪 6δ : δ( 10,)) 🡪 1

δ : δ( 1,then) 🡪 7δ : δ( 4,p,q,r,s) 🡪 1δ : δ( 7,p,q,r,s) 🡪 1

1. 0:0
2. F : 1, 10

Tabel transisi :

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| δ | p,q r,s | not | and | or | xor | if | then | iff | ( | ) |
| 0 | {1} | {2} | {} | {} | {} | {6} | {} | {} | {9} | {} |
| 1 | {1} | {} | {3} | {4} | {5} | {} | {6} | {7} | {} | {10} |
| 2 | {1} | {2} | {} | {} | {} | {} | {} | {} | {9} | {} |
| 3 | {1} | {} | {3} | {} | {} | {} | {} | {} | {9} | {} |
| 4 | {1} | {} | {} | {4} | {} | {} | {} | {} | {9} | {} |
| 5 | {1} | {} | {} | {} | {5} | {} | {} | {} | {9} | {} |
| 6 | {1} | {} | {} | {} | {} | {6} | {} | {} | {9} | {} |
| 7 | {1} | {} | {} | {} | {} | {} | {7} | {} | {9} | {} |
| 8 | {1} | {} | {} | {} | {} | {} | {} | {8} | {9} | {} |
| 9 | {1} | {} | {} | {} | {} | {} | {} | {} | {9} | {} |
| 10 | {} | {} | {} | {} | {} | {} | {} | {} | {} | {10} |

104

# Cara Kerja Program

1. Program tahap 1 Lexical Analyzer

Program akan menanalisa inputan yang dimasukkan lalu mengganti masing-masing string ke bentuk token yang sudah di tentukan. Jika terdapat inputan yang tidak sesuai program akan mengoutputkan error

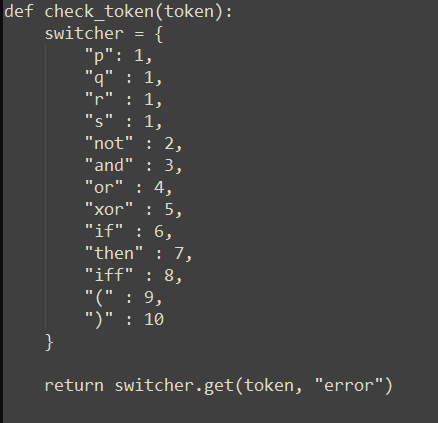
# Pengujian Program

Berikut beberapa pengujian yang telah dilakukan.

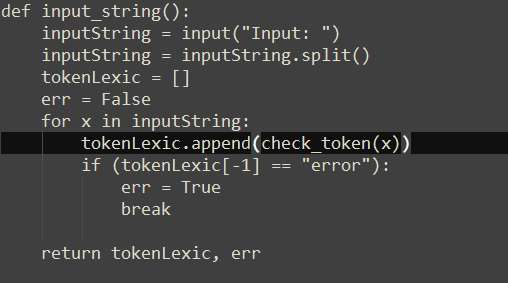
|  |  |  |
| --- | --- | --- |
| Input | Output analisa program | Screenshoot |
| p and q or r | 1, 3, 1, 4, 1 |  |
| if p then ( not q s ) | 6, 1, 7, 9, 2, 1, 1, 10 |  |
| p xor ( q and not ( p and q )) | 1, 5, 9, 1, 3, 2, 9, 1, 3, 1, 10, 10 |  |
| ( p and q ifg (r or s ) | 9, 1, 3, 1, ‘error’ |  |

# Screenshot Program

Analyzer



Input String



Main

