**PROPOSAL**

**TUGAS BESAR TEOREMA DAN BAHASA AUTOMATA**

****

Disusun Oleh :

Muhammad Aziz Al- assad 1301180044

Fazainsyah Azka Wicaksono 1301184335

Dimitri Irfan Dzidny 1301180256

Chamadani Faisal Amri 1301184075

IF-42-11

**S1 INFORMATIKA**

**FAKULTAS INFORMATIKA**

**UNIVERSITAS TELKOM**

**BANDUNG**

**2019**

1. **Kelompok**

|  |  |  |
| --- | --- | --- |
| **NO** | **NIM** | **Nama** |
| 1. | 1301180044 | Muhammad Aziz Al-assad |
| 2. | 1301184335 | Fazainsyah Azka Wicaksono |
| 3. | 1301180256 | Dimitri Irfan Dzidny |
| 4. | 1301184075 | Chamadani Faisal Amri |

1. **Spesifikasi Program**

Pada tubes kali ini, kasus yang dihadapi adalah melakukan parsing dan validasi terhadap Formula Logika Proposisi dengan batasan serta spesifikasi kebutuhan program sebagai berikut.

**Spesifikasi**:

* Formula program yang dapat diterima bersifat *case-sensitive* (*lower case*)
* Antar satu lexic dengan lexic sebelum/sesudahnya dapat dipisah dengan spasi, tanda kurung
* String Lexic program yang dapat dikenali pada input Formula program meliputi:

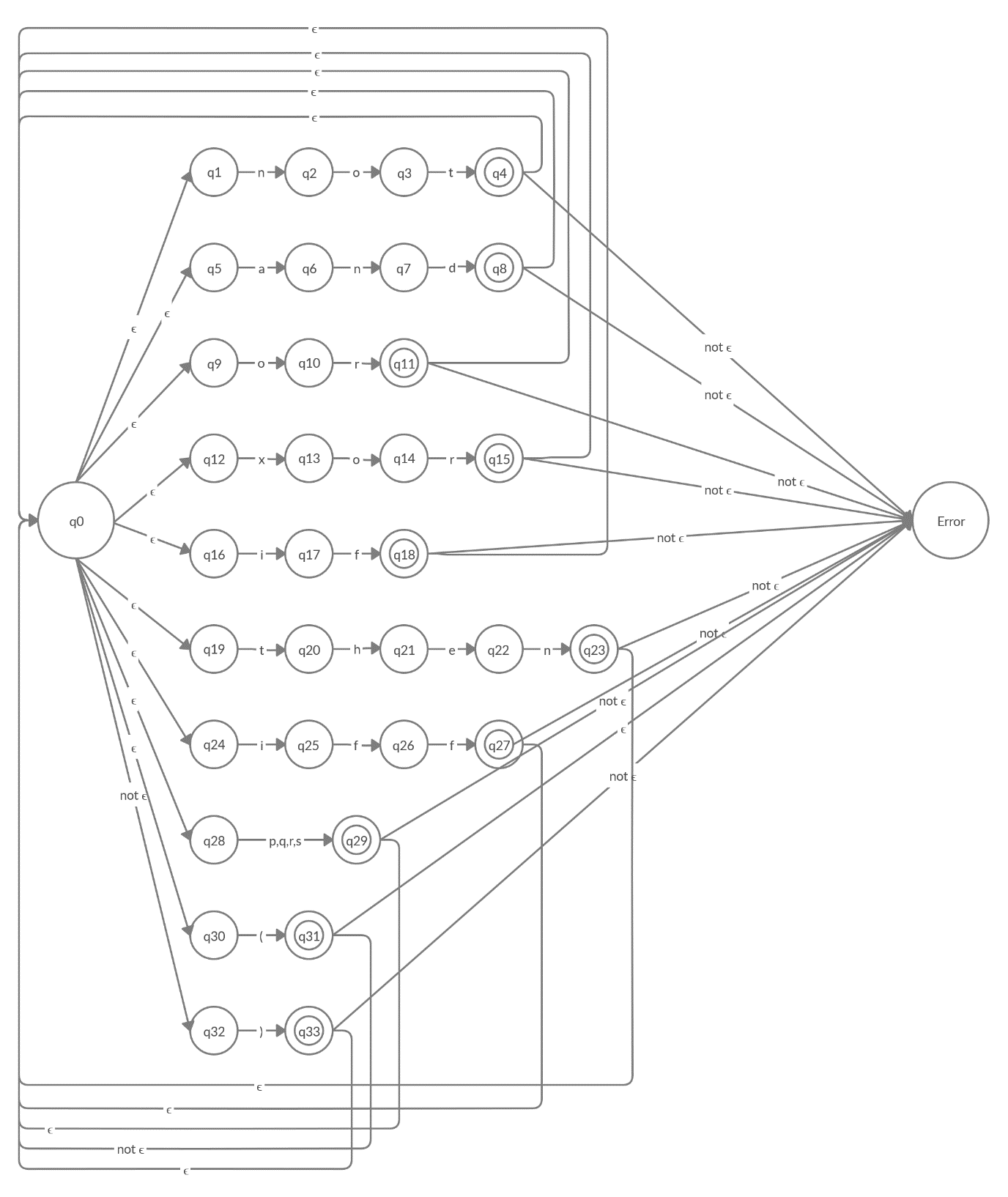
1. Simbol proposisi: p,q,r,s
2. Simbol operator logika: not, and, or, xor, if-then, iff
3. Simbol grouping: (, )

* Program dapat menterjemahkan proposisi, operator, dan grouping
* String Lexic setelah kurung buka dan sebelum kurung tutup tidak memakai spasi
* Bahasa pemograman yang digunakan adalah bahasa pemograman Phyton, dan menggunakan aplikasi Jupyter Notebook / Visual Studio Code
* Program dapat menuliskan “error” jika terjadi kesalahan dalam input, “valid” jika input & aturan benar, “tidak valid” jika input benar tetapi aturannya salah.

**Batasan:**

* Program ini dapat dijalan pada sistem operasi Windows dari Windows 7 hingga 10
* Mengenali operand logika, operator dan grouping
* Menerjemahkan kata menjadi sebuah token

1. **Rancangan Finite Automata**

****

* Definisi dari STD tersebut yaitu:

|  |  |  |
| --- | --- | --- |
| String Lexic | Jenis | Token |
| Proposisi: p, q, r, s | Operand | 1 |
| not | Operand | 2 |
| and | Operator | 3 |
| or | Operator | 4 |
| xor | Operator | 5 |
| if | Operator | 6 |
| then | Operator | 7 |
| iff | Operator | 8 |
| ( | Grouping | 9 |
| ) | Grouping | 10 |

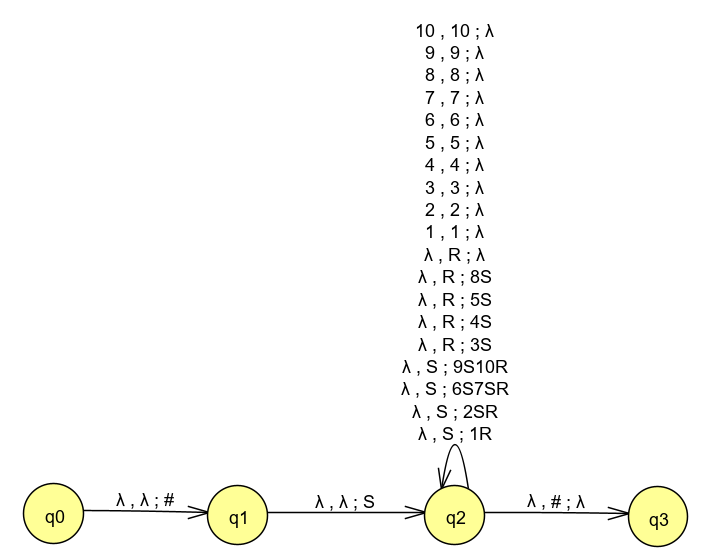
* Contoh logika proposisi: p and q

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | ƹ | p | a | n | d | q | EOS |
| q0 | q28 | {} | {} | {} | {} | {} | {} |
| q28 | {} | q29 | {} | {} | {} | {} | {} |
| q29 | q0 | {} | {} | {} | {} | {} | {} |
| q0 | q5 | {} | {} | {} | {} | {} | {} |
| q5 | {} | {} | q6 | {} | {} | {} | {} |
| q6 | {} | {} | {} | q7 | {} | {} | {} |
| q7 | {} | {} | {} | {} | q8 | {} | {} |
| q8 | q0 | {} | {} | {} | {} | {} | {} |
| q0 | q28 | {} | {} | {} | {} | {} | {} |
| q28 | {} | {} | {} | {} | {} | q29 | {} |
| q29 | {} | {} | {} | {} | {} | {} | Accept |

1. **Rancangan Context Free Grammar dan Pushdown Automata**

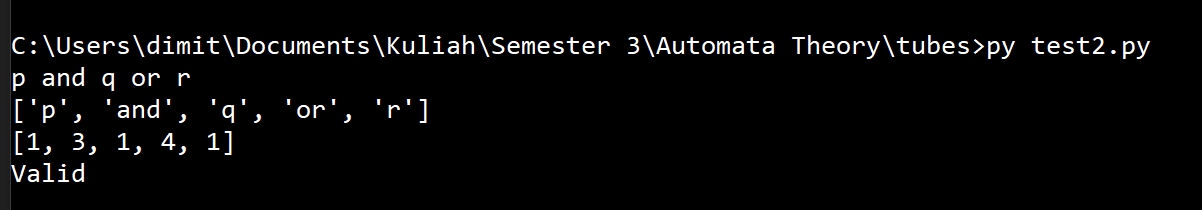
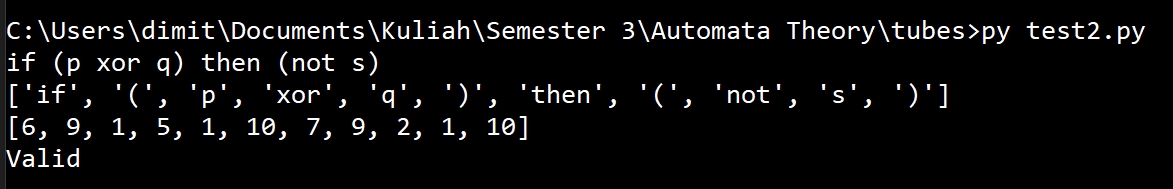
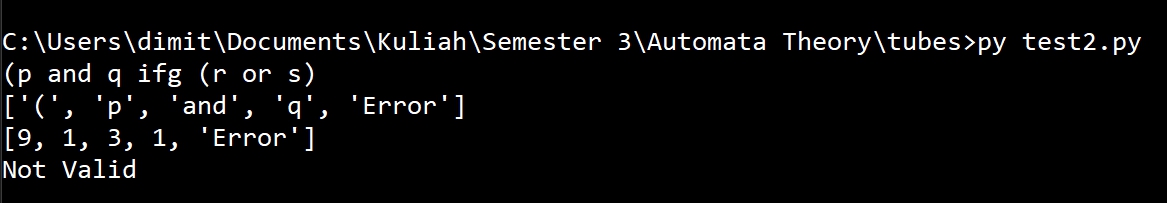
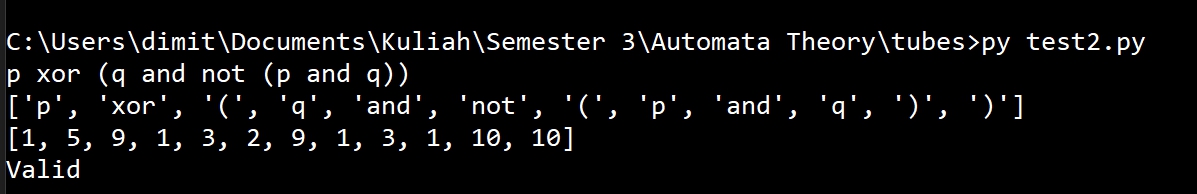
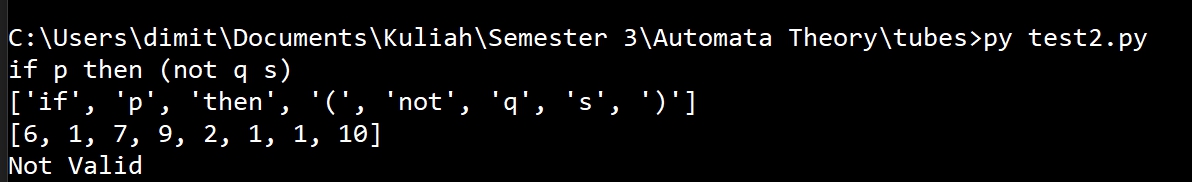
S -> 1R | 2SR | 6S7SR | 9S10R

R -> 3S | 4S | 5S | 8S | lambda



1. **Cara Kerja Program**
2. Pertama user akan memberikan inputan yang akan dipisahkan oleh spasi kecuali pada saat setelah kurung buka ‘(‘ dan sebelum ‘)’ tanpa menggunakan spasi. Input bisa dimulai dengan operand, operator (kecuali ‘then’), dan grouping (kecuali ‘)’ ).
3. Program akan membaca inputan karakter satu persatu mulai dari kiri hingga ke kanan.
4. Program akan memberikan output token selama saat pembacaan string lexic tidak ditemukan error
5. Program akan memberikan output ‘Error’ jika inputan yang diberikan tidak sesuai dengan String Lexic.
6. Program akan memberikan output berupa token dan output valid atau tidak validnya dari inputan. Program ini menggunakan PDA yang akan mengecek kevalidan inputan tersebut berdasarkan dengan CFG yang telah ditentukan.
7. **Pengujian Program**

|  |  |  |
| --- | --- | --- |
| **Input** | **Output Token** | **Keterangan** |
| p and q or r | ‘1’ 3’ ’1’ ’4’ ’1’ | Valid |
| if p then (not q s ) | ‘6’ ‘1’ ‘7’ ‘9’ ‘2’ ‘1’ ‘1’ ‘10’ | Not Valid |
| p xor (q and not (p and q)) | ‘1’ ‘5’ ‘9’ ‘1’ ‘3’ ‘2’ ‘9’ ‘1’ ’3’ ‘1’ ‘10’ ‘10’ | Valid |
| (p and q ifg(r or s) | ‘9’ ‘1’ ‘3’ ‘1’ ‘Error’ | Not Valid |
| if (p xor q) then (not s) | ‘6’ ‘9’ ‘1’ ‘5’ ‘1’ ‘10’ ‘7’ ‘9’ ‘2’ ‘1’ ‘10’ | Valid |

****

1. **Screenshot Program**

