**LAPORAN TUGAS BESAR**

**MATA KULIAH**

**TEORI BAHASA & AUTOMATA (FAR)**

*Simple Parser for Proposition Logic Formula*



Disusun Oleh:

Dani Andhika Permana | 1301180174

Indira Alima Fasyazahra | 1301184051

Edgar Vigo | 1301180149

**UNIVERSITAS TELKOM**

**BANDUNG**

**2019**

1. **DESKRIPSI PROGRAM**

**1.1 Studi Kasus**

Program sederhana untuk melakukan parsing dan validasi terhadap Formula Logika Proposisi.

**1.2 Ruang Lingkup**

* Formula yang diterima dapat bersifat *case-sensitive* atau dapat pula tidak *case-sensitive* (asalkan konsisten)
* Antar satu *lexic* dengan *lexic* sebelum/sesudahnya dapat dipisahkan dengan spasi, tanda kurung (*grouping symbol*), atau tanpa pemisah apapun (asalkan konsisten)
* *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

|  |  |  |  |
| --- | --- | --- | --- |
| String Lexic | Jenis | Token | Keterangan |
| Proposisi:  p,q,r,s | Operand | 1 | * Hanya 1 simbol (di antara p,q,r,s) yang dikenal sebagai 1 proposisi |
| not | Operator | 2 | * Contoh penulisan yang diterima: * not proposisi * not (proposisi) * not (formula) |
| 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 | * Aturan penulisan yang diterima: * if proposisi then proposisi * if (proposisi) then (proposisi) * if (formula) then (formula) |
| then | Operator | 7 |
| iff | Operator | 8 | * Aturan penulisan yang diterima: * proposisi iff proposisi * (proposisi) iff (proposisi) * (formula) iff (formula) |
| ( | Grouping | 9 | * Jumlah kurung buka dan tutup harus seimbang * Ditulis di tempat-tempat yang tepat |
| ) | Grouping | 10 |

1. **TUGAS PROGRAM TAHAP 1**
   1. **Rancangan Finite Automata**
      1. **STD**



* + 1. **Definisi 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}

q0 = {q0}

δ = δ digambar sebagai tabel transisi

F = {q11,q12}

* + 1. **Tabel Transisi**



1. **TUGAS PROGRAM TAHAP 2**
   1. **CFG (Context Free Grammar)**

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

S: *starting symbol*

P: himpunan aturan produksi berikut:

S -> 1 | 2S | 2K | S3S | K3K | S4S | L4L | S5S | K5K | 6S7S | 6K7K | S8S | K8K

K -> 9S10

* 1. **PDA (Pushdown Automata)**

G = (S, , i, F)

S = {i, p, q, f}

Σ = {1,2,3,4,5,6,7,8,9,10}

= {1,2,3,4,5,6,7,8,9,10,#}

i = {i}

F = {f}

T = {(i, λ, λ; p, #), (p, λ, λ; q, S), (q, λ, S; q, 1), (q, λ, S; q, 2S), ), (q, λ, S; q, 2K),

(q, λ, S; q, S3S), (q, λ, S; q, K3K), (q, λ, S; q, S4S), (q, λ, S; q, K4K),

(q, λ, S; q, S5S), (q, λ, S; q, K5K), (q, λ, S; q, 6S7S), (q, λ, S; q, 6K7K),

(q, λ, S; q, S8S), (q, λ, S; q, K8K), (q, λ, K; q, 9S10), (q, λ, #; f, λ)}

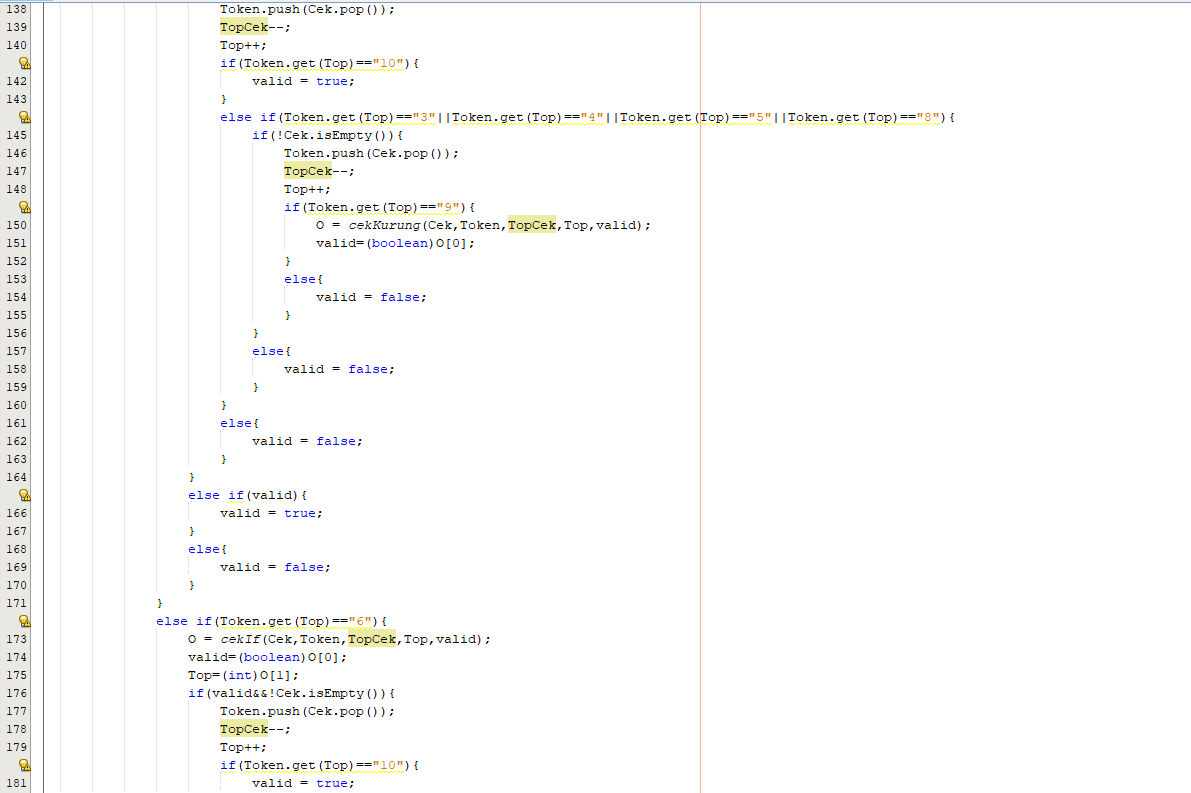


1. **PSEUDO CODE**

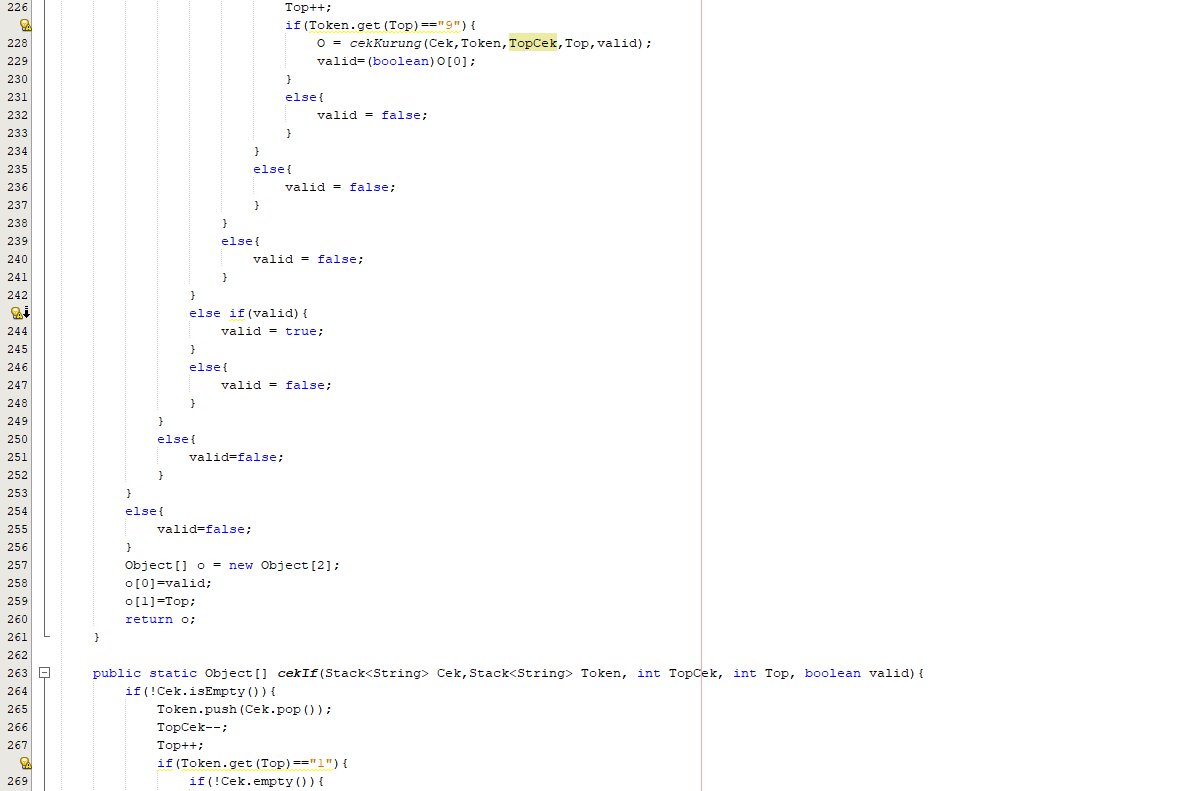


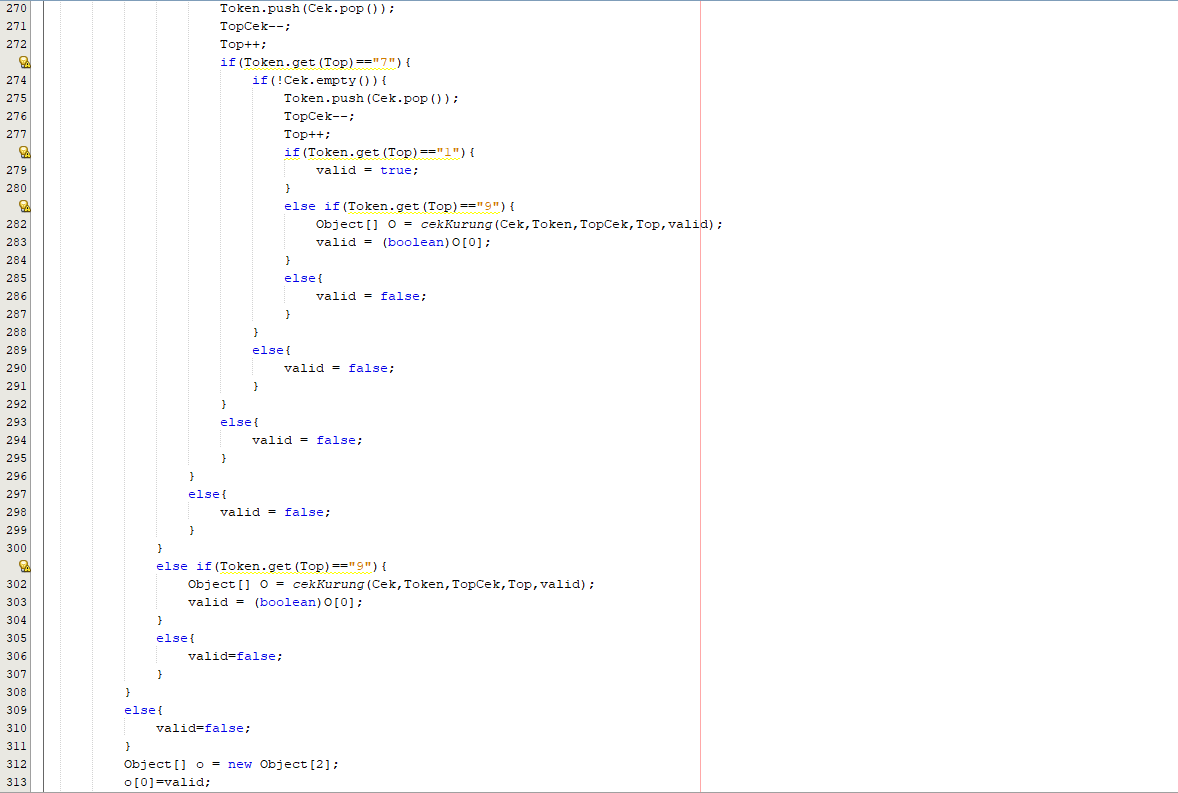


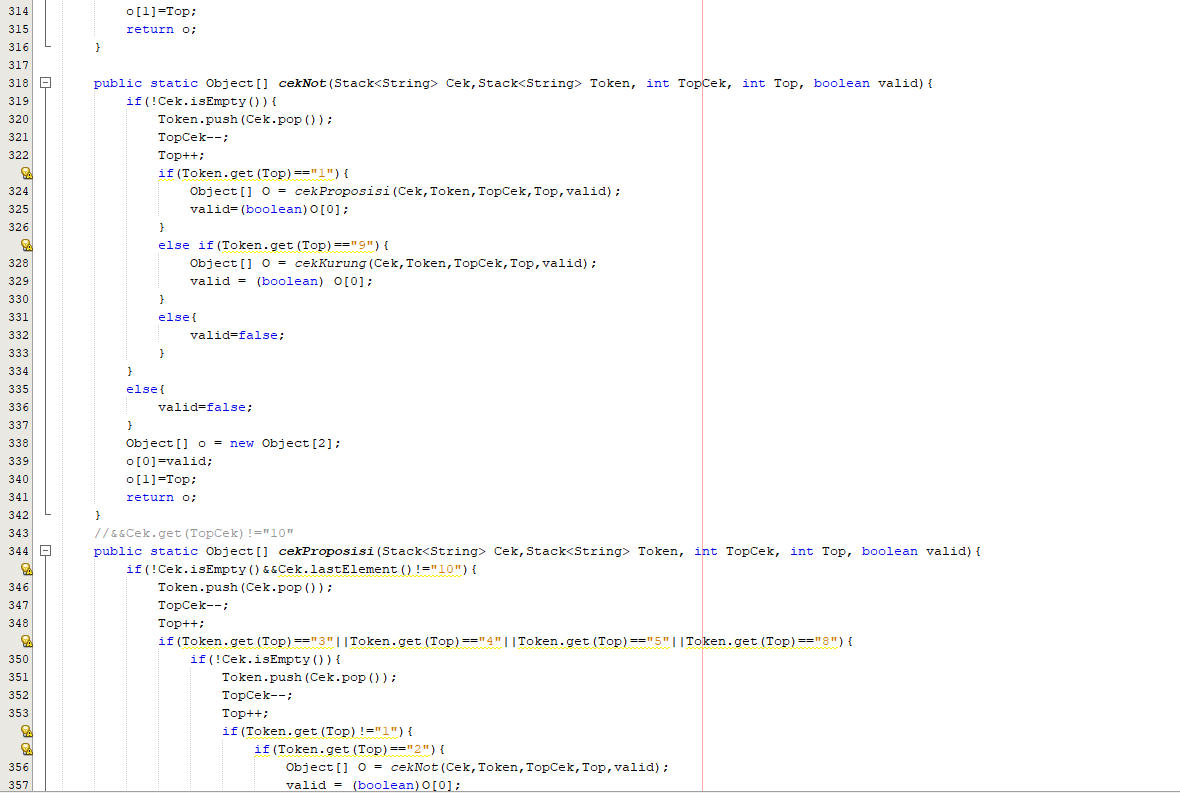


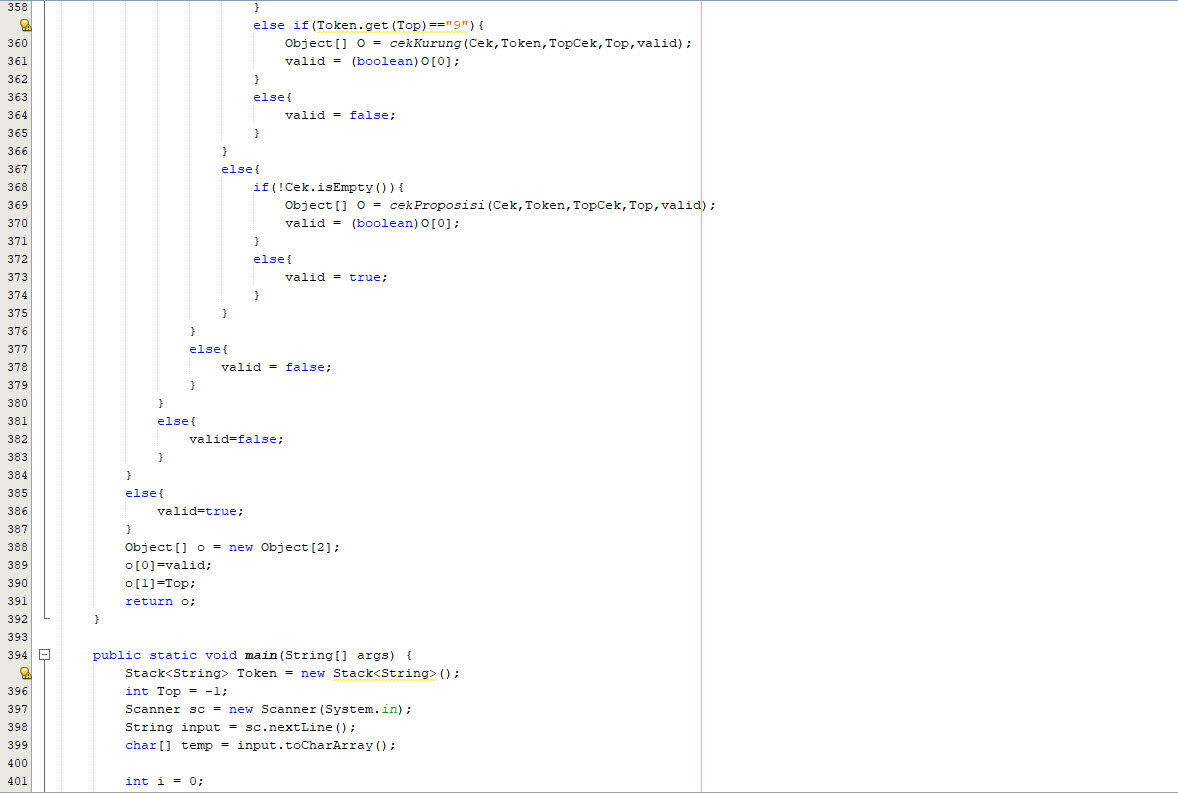




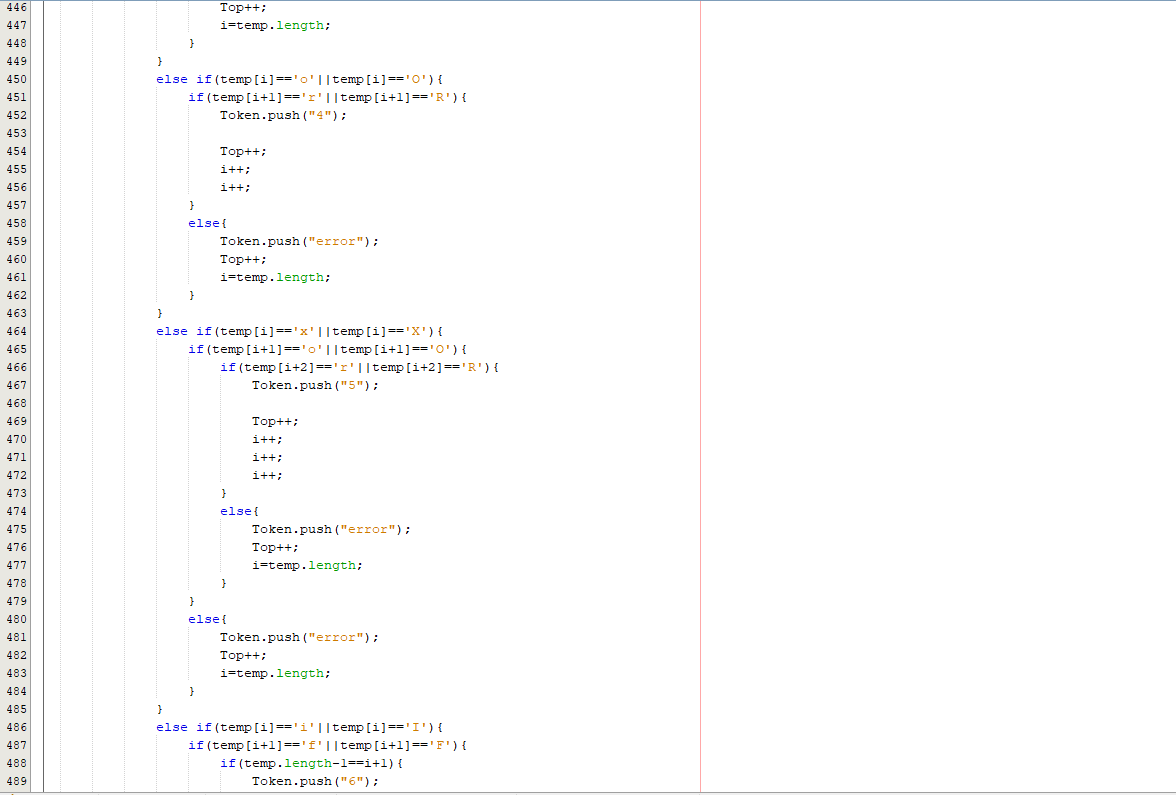








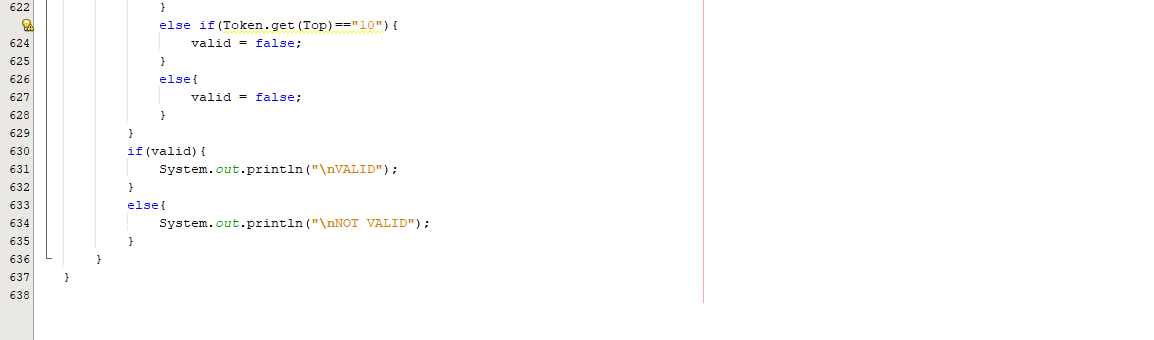












1. **PETUNJUK MENJALANKAN PROGRAM**

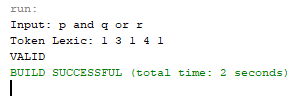
Buka *Project* didalam *folder* “Java Project” yang sudah disediakan didalam *.zip* menggunakan aplikasi NetBeans. Jalankan *Project* tersebut didalam aplikasi NetBeans.

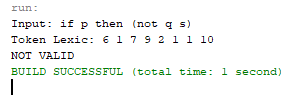
1. **CARA KERJA PROGRAM**

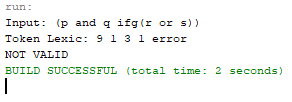
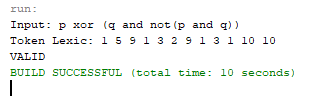
Pertama, user akan diminta meng-*input* suatu formula logika proposisi. Formula tersebut disimpan didalam variabel bertipe String dan membuat variabel Stack baru bertipe String yang masih kosong. String yang menyimpan Formula tadi diubah menjadi Array of Char. Array of Char dicek menggunakan perulangan dari indeks ke-0 atau pertama sampai indeks terakhir. Didalam perulangan, jika Char yang dicek sesuai dengan ketentuan String Lexic, maka Stack akan mem-*push* Token Lexic sesuai dengan Stringnya. Jika Char yang dicek adalah spasi, maka indeks bertambah satu dan perulangan dilanjutkan. Jika Char yang dicek tidak sesuai dengan ketentuan String Lexic, maka Stack akan mem-*push* String “error” dan perulangan berhenti.

Setelah perulangan berhenti dan pengecekan selesai, maka tahap selanjutnya adalah mem-validasi Stack yang sudah terisi tadi. Program akan membuat Stack baru bertipe String yang masih kosong. Stack yang pertama akan di *pop* lalu hasil dari *pop* tersebut di *push* ke Stack yang baru. Proses validasi akan dimulai dari Top Stack yang baru. Top dari Stack yang berupa Token Lexic di *pop* lalu akan dicek apakah Token Lexic tersebut termasuk operand, operator, atau grouping ( p, q, r, s, not, and, or, xor, if, then, iff, (, atau ) ). Token Lexic yang dicek pertama kali harus p, q, r, s, not, if, atau kurung buka. Jika tidak, maka proses validasi berhenti, lalu validasi akan di assign false dan Program akan mengoutputkan TIDAK VALID. Jika Stack yang di *pop* bernilai “error”, maka proses validasi berhenti, lalu validasi akan di assign false dan Program akan mengoutputkan TIDAK VALID. Jika validasi masih bernilai true, maka Stack akan terus di *pop* dan dicek sesuai dengan ketentuan Formula yang ada sampai Stack kosong atau validasi bernilai false. Jika Stack sudah kosong dan validasi bernilai true, maka Program akan mengoutputkan VALID.

1. **PENGUJIAN PROGRAM**







|  |  |  |
| --- | --- | --- |
| **Input** | **Output Program** | **Keterangan** |
| p and q or r | VALID | Sesuai Formula  proposisi and proposisi  proposisi or proposisi |
| if p then (not q s) | TIDAK VALID | Setelah operand tidak boleh operand |
| p xor (q and not(p and q)) | VALID | Sesuai Formula  proposisi xor (formula)  proposisi and proposisi  not (formula) |
| (p and q ifg(r or s)) | TIDAK VALID | Tidak ada operator ifg |