

Nama : Shafa Auliya

NPM : 2117051042

Kelas : CD

## **Tugas 1 Sistem Pakar Praktikum (Latihan Chaining)**

### **1. Daftar Aturan**

Aturan bersumber dari tabel 3 sampel data penyakit THT dengan gejala-gejala.

- a. Aturan 1 :  $A + B + C \Rightarrow W$   
IF A (Sakit kepala) True  
AND B (Keluar cairan) True  
AND C (Terdapat radang di liang telinga) True  
THEN W (Penyakit Otitis Eksterna)
- b. Aturan 2 :  $D + A + F + E + G + H + I \Rightarrow X$   
IF D (Demam) True  
AND A (Sakit kepala) True  
AND E (PUS dan di meatus media) True  
AND F (Hidung tersumbat) True  
AND G (Hidung meler) True  
AND H (Nyeri pipi di bawah mata) True  
AND I (Selaput lender merah dan bengkak) True  
THEN X (Sinusitis)
- c. Aturan 3 :  $J + G + F + K \Rightarrow Y$   
IF J (Bersin-bersin) True  
AND G (Hidung meler) True  
AND F (Hidung tersumbat) True  
AND K (Lendir di tenggorokan) True  
THEN Y (Rinitis Non-Alergika)
- d. Aturan 4 :  $D + A + L + M + N + O + P + Q \Rightarrow Z$   
IF D (Demam) True  
AND A (Sakit kepala) True  
AND L (Nyeri saat berbicara atau menelan) True  
AND M (Sakit pada telinga) True  
AND N (Pembengkakan kelenjar getah bening) True  
AND O (Tenggorokan gatal) True  
AND P (Adanya tonsil yang membengkak) True  
AND Q (Suara serak) True  
THEN Z (Farangitis - Radang Tenggorokan)

Dengan aturan diatas yang sesuai dengan panduan dapat dibuatkan aturan sebagai berikut :

$$A + B + C \Rightarrow W$$

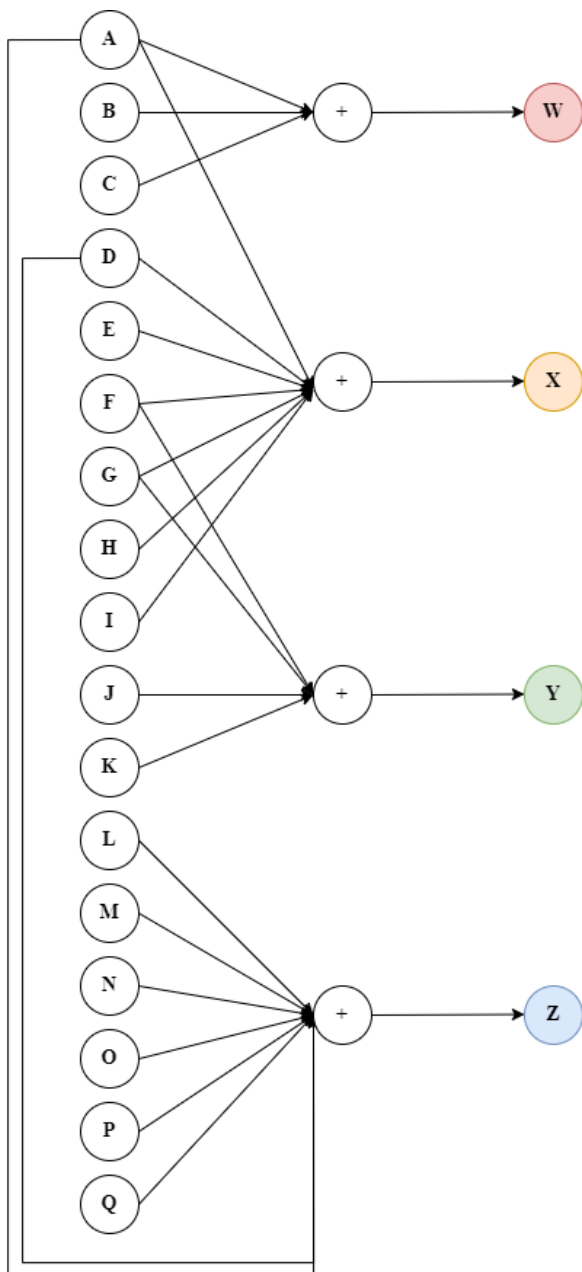
$$D + A + E + F + G + H + I \Rightarrow X$$

$$J + G + F + K \Rightarrow Y$$

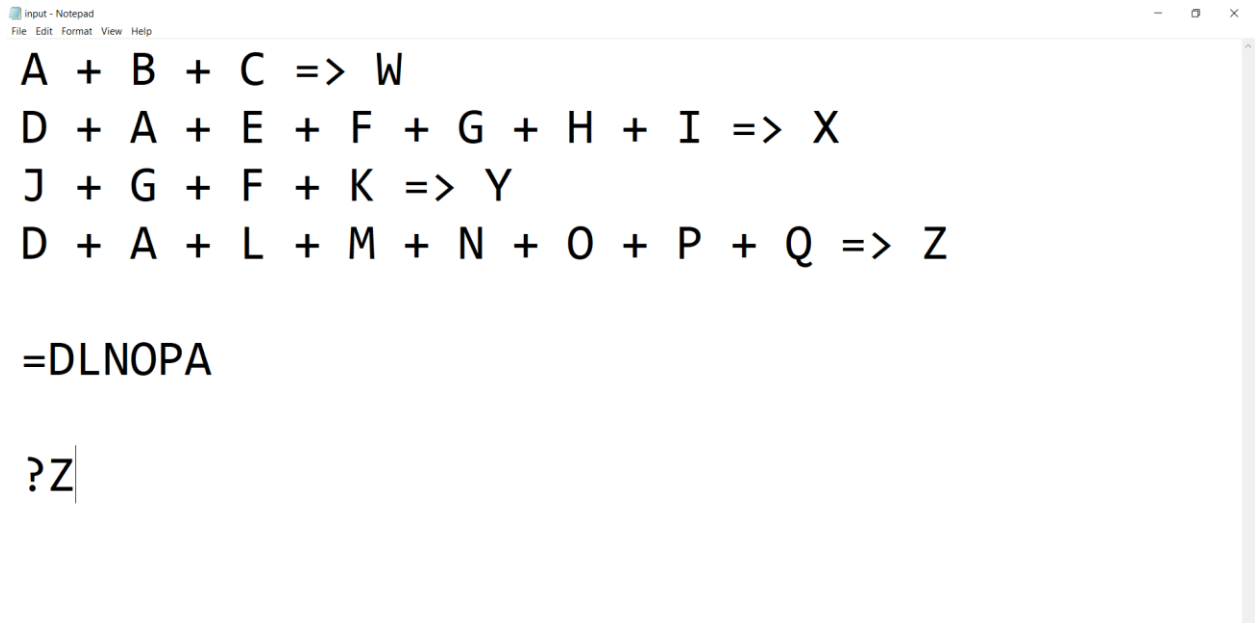
$$D + A + L + M + N + O + P + Q \Rightarrow Z$$

## 2. Graf

Graf sesuai dengan aturan yang dibuat diatas



### 3. Tampilan Input Program



```
Input - Notepad
File Edit Format View Help

A + B + C => W
D + A + E + F + G + H + I => X
J + G + F + K => Y
D + A + L + M + N + O + P + Q => Z

=DLNOPA

?Z
```

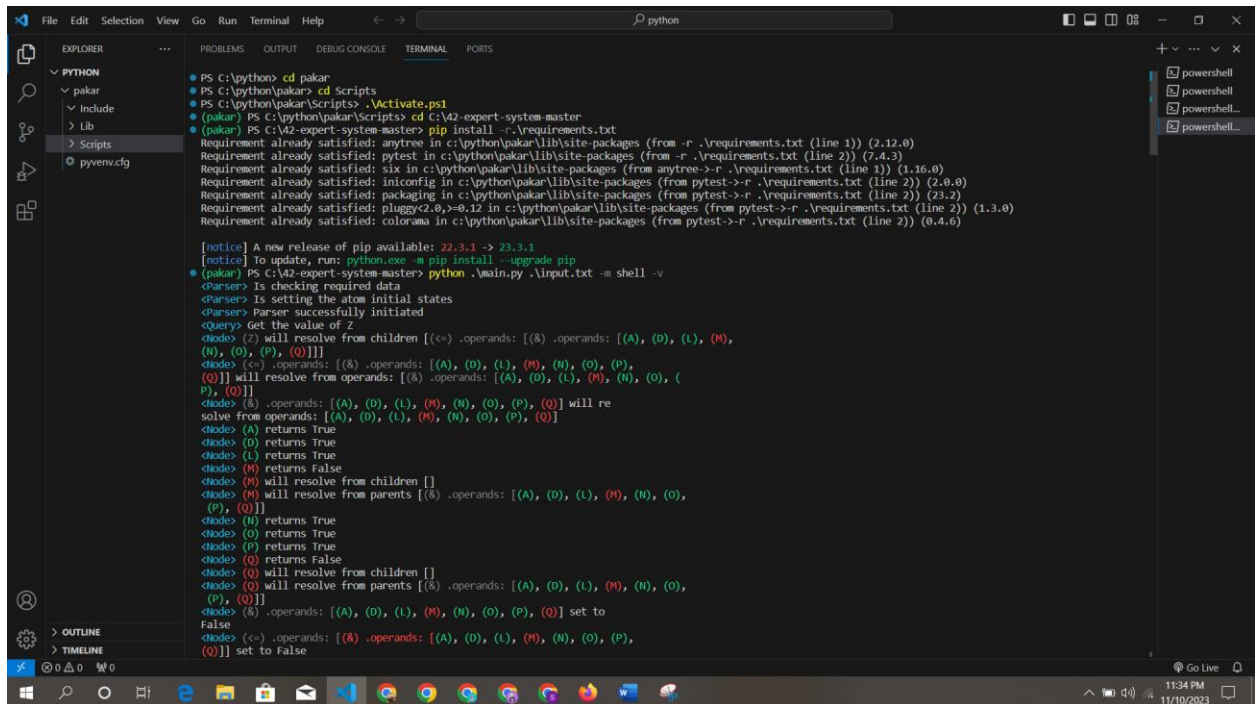
Pada inputan program yang pertama dilakukan adalah membuat file dengan format .txt, pada langkah ini saya membuat file dengan nama 'input.txt'. Pada file tersebut berisi inputan seperti aturan yang sudah dibuat yang bersumber dari tabel 3 sampel data penyakit THT dengan gejala-gejala dan inputan salah satu penyakit untuk membuktikan apakah dari gejala-gejala tersebut akan menghasilkan output True atau False.

Pada aturan tersebut ada tanda yang harus diperhatikan seperti :

- tanda + ditandai dengan operasi AND
- tanda => ditandai dengan THEN
- tanda ? ditandai dengan apakah hasil dari gejala-gejala yang telah diinputkan dan penyakit yang telah diinputkan akan menghasilkan nilai True atau False

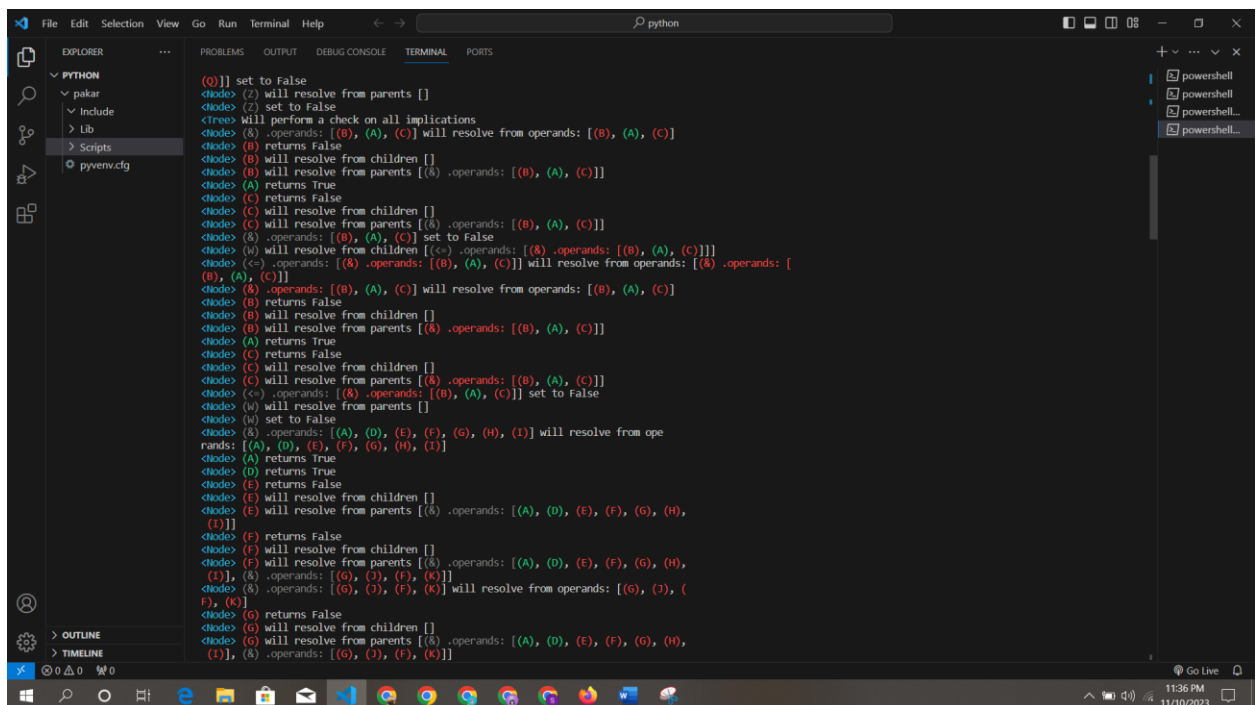
Pada kasus ini saya menginputkan DLNOPA untuk mengecek apakah dari gejala-gejala tersebut akan menimbulkan penyakit Z (Farangitis - Radang Tenggorokan), jika dari gejala tersebut dapat menimbulkan penyakit Farangitis - Radang Tenggorokan maka program nantinya akan menghasilkan True dan jika sebaliknya maka program akan menghasilkan nilai False.

## 4. Tampilan Output Program



```
PS C:\python> cd pakar
PS C:\python\pakar> cd Scripts
PS C:\python\pakar\Scripts> .\Activate.ps1
(pakar) PS C:\python\pakar\Scripts> cd C:\42-expert-system-master
(pakar) PS C:\42-expert-system-master> pip install -r requirements.txt
Requirement already satisfied: anytree in c:\python\pakar\lib\site-packages (from -r requirements.txt (line 1)) (2.12.0)
Requirement already satisfied: pytest in c:\python\pakar\lib\site-packages (from -r requirements.txt (line 2)) (7.4.3)
Requirement already satisfied: six in c:\python\pakar\lib\site-packages (from anytree->r requirements.txt (line 1)) (1.16.0)
Requirement already satisfied: iniconfig in c:\python\pakar\lib\site-packages (from pytest->r requirements.txt (line 2)) (2.0.0)
Requirement already satisfied: packaging in c:\python\pakar\lib\site-packages (from pytest->r requirements.txt (line 2)) (23.2)
Requirement already satisfied: pluggy(2.0,>=0.12 in c:\python\pakar\lib\site-packages (from pytest->r requirements.txt (line 2)) (1.3.0)
Requirement already satisfied: colorama in c:\python\pakar\lib\site-packages (from pytest->r requirements.txt (line 2)) (0.4.6)

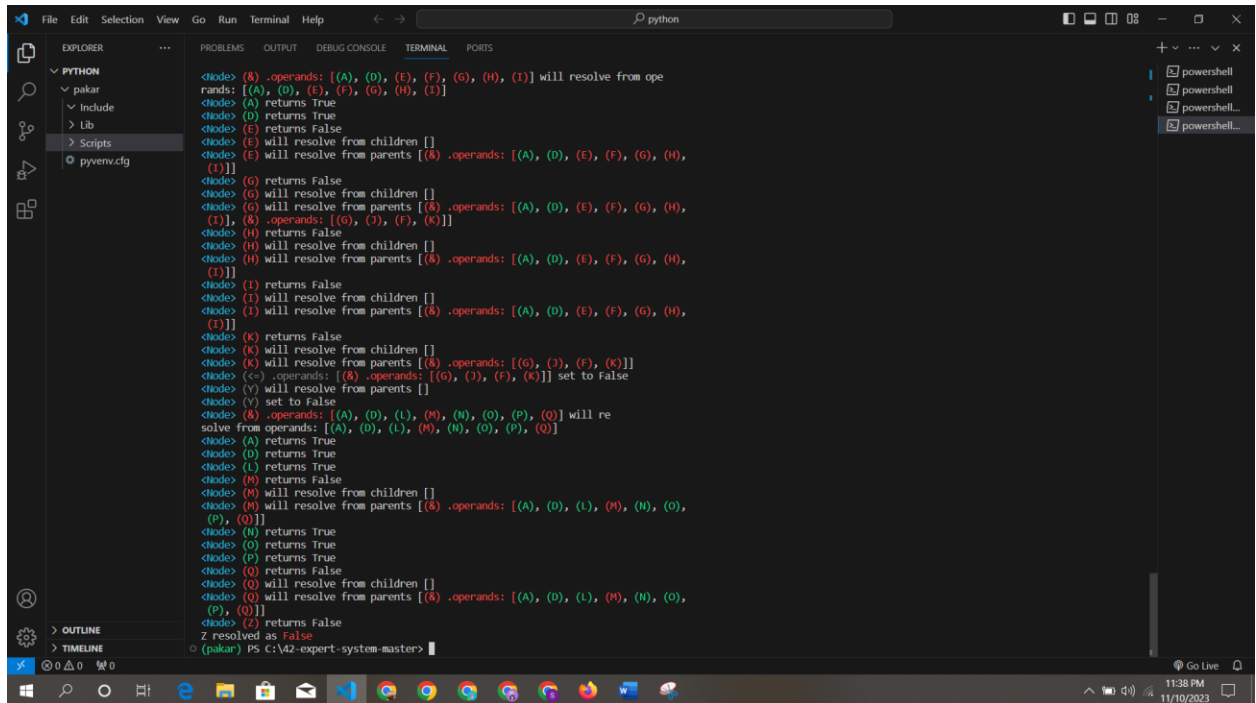
[notice] A new release of pip available: 22.3.1 -> 23.3.1
[notice] To update, run: python.exe -m pip install --upgrade pip
(pakar) PS C:\42-expert-system-master> python .\main.py .\input.txt -m shell -v
<Parser> Is checking required data
<Parser> Is setting the atom initial states
<Parser> Parser successfully initiated
<Query> Get the value of Z
<node> (Z) will resolve from children [(A), (D), (L), (M),
(W), (O), (P), (Q)]
<node> (Z) .operands: [(A), (D), (L), (M), (O), (P),
(W), (O), (P), (Q)]
<node> (Z) will resolve from operands: [(A), (D), (L), (M), (O), (P),
(W), (O)]
<node> (Z) .operands: [(A), (D), (L), (M), (O), (P), (Q)] will re
solve from operands: [(A), (D), (L), (M), (O), (P), (Q)]
<node> (A) returns True
<node> (D) returns True
<node> (L) returns True
<node> (M) returns False
<node> (O) will resolve from children []
<node> (O) will resolve from parents [(A), (D), (L), (M), (O),
(P), (Q)]
<node> (O) returns True
<node> (P) returns True
<node> (P) returns False
<node> (Q) will resolve from children []
<node> (Q) will resolve from parents [(A), (D), (L), (M), (O),
(P), (Q)]
<node> (Q) .operands: [(A), (D), (L), (M), (O), (P), (Q)] set to
False
<node> (Q) .operands: [(A), (D), (L), (M), (O), (P), (Q)] set to
False
```



```
(Q)] set to False
<node> (Z) will resolve from parents []
<node> (Z) set to False
<tree> Will perform a check on all implications
<node> (Z) .operands: [(B), (A), (C)] will resolve from operands: [(B), (A), (C)]
<node> (B) returns False
<node> (B) will resolve from children []
<node> (B) will resolve from parents [(B), (A), (C)]
<node> (A) returns True
<node> (C) returns False
<node> (C) will resolve from children []
<node> (C) will resolve from parents [(B), (A), (C)]
<node> (Z) .operands: [(B), (A), (C)] set to False
<node> (W) will resolve from children [(Z) .operands: [(B), (A), (C)]]
<node> (W) .operands: [(Z) .operands: [(B), (A), (C)]] will resolve from operands: [(Z) .operands: [
(B), (A), (C)]
<node> (Z) .operands: [(B), (A), (C)] will resolve from operands: [(B), (A), (C)]
<node> (B) returns False
<node> (B) will resolve from children []
<node> (B) will resolve from parents [(B), (A), (C)]
<node> (A) returns True
<node> (C) returns False
<node> (C) will resolve from children []
<node> (C) will resolve from parents [(B), (A), (C)]
<node> (Z) .operands: [(B), (A), (C)] set to False
<node> (W) will resolve from parents []
<node> (W) set to False
<node> (Z) .operands: [(A), (D), (E), (F), (G), (H), (I)] will resolve from ope
rands: [(A), (D), (E), (F), (G), (H), (I)]
<node> (A) returns True
<node> (D) returns True
<node> (E) returns False
<node> (E) will resolve from children []
<node> (E) will resolve from parents [(A), (D), (E), (F), (G), (H),
(I)]
<node> (F) returns False
<node> (F) will resolve from children []
<node> (F) will resolve from parents [(A), (D), (E), (F), (G), (H),
(I)]
<node> (G) .operands: [(G), (I), (F), (K)]
<node> (K) .operands: [(G), (I), (F), (K)] will resolve from operands: [(G), (I), (
F), (K)]
<node> (G) returns False
<node> (O) will resolve from children []
<node> (O) will resolve from parents [(A), (D), (E), (F), (G), (H),
(I)], (K) .operands: [(G), (I), (F), (K)]
```







```
<node> (&).operands: [(A), (D), (E), (F), (G), (H), (I)] will resolve from ope
rands: [(A), (D), (E), (F), (G), (H), (I)]
<node> (A) returns true
<node> (D) returns true
<node> (E) returns false
<node> (E) will resolve from children []
<node> (E) will resolve from parents [(&).operands: [(A), (D), (E), (F), (G), (H),
(I)]]
<node> (G) returns false
<node> (G) will resolve from children []
<node> (G) will resolve from parents [(&).operands: [(A), (D), (E), (F), (G), (H),
(I)], (&).operands: [(G), (I), (F), (K)]]
<node> (H) returns false
<node> (H) will resolve from children []
<node> (H) will resolve from parents [(&).operands: [(A), (D), (E), (F), (G), (H),
(I)]]
<node> (I) returns false
<node> (I) will resolve from children []
<node> (I) will resolve from parents [(&).operands: [(A), (D), (E), (F), (G), (H),
(I)]]
<node> (K) returns false
<node> (K) will resolve from children []
<node> (K) will resolve from parents [(G), (I), (F), (K)]]
<node> (<=).operands: [(&).operands: [(G), (I), (F), (K)]] set to false
<node> (<=) will resolve from parents []
<node> (<=) set to false
<node> (&).operands: [(A), (D), (L), (M), (N), (O), (P), (Q)] will re
solve from operands: [(A), (D), (L), (M), (N), (O), (P), (Q)]
<node> (A) returns true
<node> (D) returns true
<node> (L) returns true
<node> (M) returns false
<node> (M) will resolve from children []
<node> (M) will resolve from parents [(&).operands: [(A), (D), (L), (M), (N), (O),
(P), (Q)]]
<node> (N) returns true
<node> (O) returns true
<node> (P) returns true
<node> (Q) returns false
<node> (Q) will resolve from children []
<node> (Q) will resolve from parents [(&).operands: [(A), (D), (L), (M), (N), (O),
(P), (Q)]]
<node> (Z) returns false
Z resolved as false
(pakar) PS C:\42-expert-system-master>
```

Pada inputan yang yang ada di file ‘input.txt’ yang sudah dibuat sebelumnya menghasilkan output nilai Z yaitu **False**.

## 5. Alur Kerja Program

Dalam file ‘input.txt’ yang sudah dimasukkan sebelumnya terdapat inputan bahwa DLNOPA dan menghasilkan nilai Z. Pada proses ini menerapkan metode backward chaining, dimana prosesnya dimulai dengan pemeriksaan inputan terlebih dahulu. Pada inputan juga memasukan aturan yang sudah dibuat sebelumnya agar program dapat membaca aturan tersebut dan dapat menghasilkan output. Berikut merupakan alur kerja program :

- 1) Program akan menelusuri rules yang menghasilkan nilai dari query yang diinputkan, dalam kasus yang saya buat ini adalah Z . Salah satu rule yang memproduksi Z adalah  $D + A + L + M + N + O + P + Q \Rightarrow Z$
- 2) Fakta yang diinputkan DLNOPA adalah yang memiliki nilai true. Program kemudian membaca bahwa dalam fakta tersebut terdapat DLNOPA yang bernilai true tetapi ada bagian rules yang tidak dimasukkan didalam inputan yang kita inginkan seperti MQ.
- 3) Dengan mengetahui bahwa nilai DLNOPA true, namun ada bagian rules yang tidak dimasukkan maka program ini akan menghasilkan nilai False.

Namun, dengan demikian jika kita menginputkan dengan kondisi yang ada di penyakit dan menginputkannya semua bagian rules maka program tersebut akan menghasilkan nilai True, contohnya jika kita menginputkan nilai BCA dengan penyakit W maka hasil nya akan True.