BAB III
LOGIC PROGRAMMING

PROLOG adalah kependekan dari ***PROgramming in LOGic***, yang berarti pemrograman logika. Pemrograman Prolog menggunakan bahasa deklaratif, dimana pemrogram memberi **fakta** dan **aturan** untuk selanjutnya diselesaikan oleh Prolog secara deduktif sehingga menghasilkan suatu kesimpulan. Hal ini berbeda dengan bahasa prosedural seperti Pascal, Fortran, C, atau yang sejenis, dimana pemrogram memberi perintah atau penugasan untuk memecahkan persoalan langkah demi langkah, sehingga sering disebut sebagai *programming with assignment*. Disamping itu, berbeda dengan pemrograman fungsional, pemrograman logika ini menggunakan relasi, bukan fungsi sehingga sangat sesuai untuk implementasi sistem pakar.

LOGIKA PREDIKAT

Logika predikat (kalkulus predikat) merupakan bagian dari komputasi logika yang juga mencakup aljabar Boole (logika proposisional), dimana fakta dan aturan dinyatakan melalui predikat seperti:

lelaki(Joko)	// fakta
menikah(Joko, Tuti)	// fakta
$\forall x\ \forall y$ [**menikah**(x,y) $\wedge$ **lelaki**(x)] $\rightarrow$ ~ **lelaki**(y)	// aturan
$\forall y\ \exists x$ [**orang**(y) $\rightarrow$ **ibu**(x,y)	// aturan

Kalimat pertama menunjukkan adanya fakta bahwa Joko adalah seorang lelaki, dan kalimat kedua menyatakan bahwa Joko menikah dengan Tuti. Kalimat ketiga dan keempat menunjukkan suatu aturan atau kaidah yang umum berlaku, bahwa untuk setiap pasang orang x dan y, jika x menikah dengan y dan x adalah lelaki, maka dapat dipastikan bahwa y adalah bukan seorang lelaki. Sedangkan kalimat terakhir manyatakan bahwa untuk setiap y, ada x sehingga jika y adalah orang maka y mempunyai seorang ibu x (x ibu dari y).

Simbol predikat yang digunakan dalam kalimat-kalimat tersebut adalah **lelaki**, **menikah**, **orang**, dan **ibu** yang sering disebut sebagai **relasi**, sedangkan Joko dan Tuti disebut sebagai simbol konstanta.

BAHASA DEKLARATIF

Seperti yang dijelaskan sebelumnya bahwa pokok perbedaan Prolog dari bahasa lain adalah karena bersifat deskriptif atau deklaratif, sedang bahasa lain umumnya bersifat prosedural atau imperatif. Sebagai bukti bahwa Prolog merupakan bahasa deklaratif adalah dalam menyatakan fakta dan aturan seperti berikut:

1. Jika ingin menyatakan bahwa Prawiro adalah bapak dari Joko, maka dalam Prolog dituliskan sebagai:

```
bapak(prawiro, joko).
```

2. Jika ingin menerangkan suatu kaidah bahwa A adalah kakek dari Z maka harus dibuat dahulu logika dalam bahasa Indonesia sehingga menjadi suatu aturan seperti berikut:

```
A adalah kakek dari Z jika A adalah bapak dari X dan X adalah bapak Z
atau
A adalah kakek dari Z jika A adalah bapak dari X dan X adalah ibu Z
```

Aturan tersebut ditulis dalam Prolog sebagai:

```
kakek(A,Z) :- bapak(A,X), bapak(X,Z).
kakek(A,Z) :- bapak(A,X), ibu(X,Z).
```

DASAR PEMROGRAMAN PROLOG

Pada bagian ini akan diuraikan dasar-dasar pemrograman Prolog, aturan umum penulisan program, bagaimana melakukan dialog dengan Prolog, dan beberapa pengertian dasar yang berkaitan dengan program Prolog.

Fakta

Fakta adalah suatu kenyataan atau kebenaran yang diketahui, dan menyatakan hubungan (relasi) antara dua atau lebih obyek. Fakta dapat pula menunjukkan sifat suatu obyek. Contoh sederhana adalah:

```
bapak(prawiro, joko).
merah(darah).
asin(garam).
```

Aturan

Aturan merupakan logika yang dirumuskan dalam bentuk relasi sebab-akibat dan hubungan implikasi. Misalnya dapat dibuat aturan bahwa jika A adalah bapak dari X dan X adalah bapak atau ibu dari Z maka dapat dipastikan bahwa A adalah kakek dari Z. Contoh untuk ini adalah:

```
kakek(A,Z) :- bapak(A,X), bapak(X,Z).
kakek(A,Z) :- bapak(A,X), ibu(X,Z).
```

Klausa (*clause*)

Aturan yang ditulis ini berupa klausa (*clause*) dan terdiri dari kepala (*kakek*) dan tubuh yang dipisahkan oleh tanda :- (*bapak* dan *ibu*). Klausa adalah suatu frase (ungkapan)

atau susunan kata yang di dalam Prolog dapat berupa fakta atau aturan, yang selalu diakhiri dengan tanda titik (.). Suatu tubuh klausa dapa terdiri dari beberapa sub-klausa yang dihubungkan satu sama lain menggunakan tanda koma (,) yang berarti hubungan **and** dan tanda titik koma (;) yang menunjukkan hubungan **or**. Penggabungan dalam tubuh klausa yang dirangkai dengan **and** disebut sebagai **konjungsi**, sedangkan jika dirangkai dengan **or** disebut **disjungsi**. Berikut disajikan contoh penggabungan disjungsi untuk menuliskan aturan kakek sebelumnya:

```
orangtua(P,Q) :- bapak(P,Q); ibu(P,Q).
kakek(A,Z) :- bapak(A,X), orangtua(X,Z).
```

Relasi

Istilah *merah*, *asin*, *kakek*, *bapak*, *ibu*, dan *orangtua* pada contoh fakta dan aturan sebelumnya disebut sebagai relasi. Relasi adalah tabel dengan n buah kolom dan terdiri dari beberapa baris fakta maupun aturan. Misalkan relasi **append** adalah sekumpulan tuple (X,Y,Z) dimana Z terdiri dari elemen X diikuti dengan Y atau Z=X+Y. Anggota relasi append terdiri dari :

```
([], [], [])
([a], [], [a])
([a],[b],[a,b])
([a,b],[c,d],[a,b,c,d])
```

sedangkan ([a],[b],[]) bukan anggota relasi append.

Secara umum, suatu relasi dinyatakan dalam bentuk aturan atau fakta sebagai berikut:

P if Q_1 and Q_2 and ... and Q_k untuk k>= 0

Sedangkan dalam notasi EBNF dapat dituliskan sebagai:

```
Rule::=Term:-Term {Term}
Term::=Number|Atom|Var|Atom(Term)
Term::=Term{Term}
```

atau dalam Prolog ditulis sebagai:

$P:-Q_1,Q_2,...,Q_k$

Fakta adalah aturan untuk k=0, artinya fakta selalu berlaku tanpa harus memenuhi kondisi tertentu, atau

```
Fact::=Term.
```

Variabel

Argumen suatu predikat dapat berupa konstanta (atom), variabel, atau obyek lain. Atom disebut juga sebagai obyek nyata, sedangkan variabel disebut obyek umum. Suatu atom, variabel, atau obyek lain dalam Prolog disebut **term**, sehingga argumen selalu berupa term.

Dalam Prolog terdapat dua variabel, yaitu:

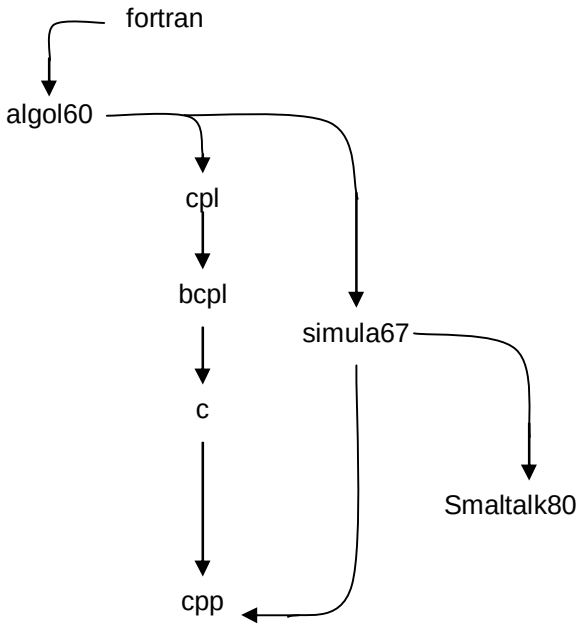
1. Variabel bernama, yaitu variabel yang diberi nama seperti X, Orang, dan sebagainya
2. Variabel tak bernama (*placeholder*), dilambangkan dengan tanda garis bawah (_).

Setiap term yang ditulis dengan awalan huruf kapital selalu dianggap sebagai variabel bernama dalam Prolog, sedangkan awalan dengan huruf kecil dianggap sebagai suatu relasi atau konstanta. Variabel tak bernama digunakan untuk mengabaikan nilai suatu variabel, yang berarti bisa bernilai apa saja. Berikut adalah contoh penggunaan variabel bernama dan tidak bernama.

```
member(X,[X|_]).
member(X,[_|Y]):-member(X,Y).
```

Di bawah ini disajikan beberapa contoh program Prolog sebagai gambaran awal pemahaman pemrograman logika:

Contoh 1 (hubungan atau link antara beberapa bahasa pemrograman).



```
clauses
    link(fortran,algol60).
    link(algol60, simula67).
    link(algol60, cpl).
    link(simula67, smalltalk80).
    link(simula67, cpp).
    link(cpl, bcpl).
    link(bcpl, c).
    link(c, cpp).
    path(L,L).
    path(L,M):-link(L,X),path(X,M).
```

Contoh 2.

```
clauses
    grandfather(X,Z):-father(X,Y),father(Y,Z).
    grandfather(X,Z):-father(X,Y),mother(Y,Z).
    father(john,bill).
    father(bill,mary).
    father(bill,tom).
    father(tom,chris).
    father(tom,bob).
    mother(mary,june).
    mother(mary,katie).
```

Contoh 3.

```
clauses
    append([],Y,Y).
    append([H|X1],Y,[H|Z1]):-append(X1,Y,Z1).
```

QUERIES

Query atau pertanyaan digunakan untuk memperoleh jawaban dari suatu problem (secara deduktif). Dalam notasi EBNF, query didefinisikan sebagai:

```
Query::=Term {Term}
```

sedangkan dalam Prolog, query dinyatakan dalam **goal**. Ada dua jenis goal, yaitu internal yang dituliskan langsung di dalam tubuh program, sedangkan goal eksternal dituliskan di luar program dan diberikan pada saat program dijalankan. Berikut ini beberapa contoh goal:

Contoh 1 (goal internal)

```
father(Bapak,Chris), write(Bapak)
grandfather(Kakek,Chris), write(kakek)
```

Contoh 2 (goal internal 2)

```
append([a,b],[c,d],Z), write(Z), [a,b,c,d]
append(X,[c,d],[a,b,c,d]),write(X)
append([a,b],Y,[a,b,c,d]),write(Y)
```

Contoh 3 (goal eksternal)

goal = link(cpl,bcpl), link(bcpl, c).
goal = link(algol60, L), link(L,M).
L = simula67, M = smalltalk80
L = simula67, M = cpp
L = cpl,
M = bcpl

3 solutions
goal = link(L,N), link(M,N), (L=M).
L = simula67, N = cpp, M = c
L = c,
N = cpp, M = simula67

2 solutions