Prolog
List, File Eksternal, Loop

IF1221 Computational Logic
Semester II - 2024/2025

Informatics Engineering Study Program
School of Electrical Engineering and Informatics ITB

Pengenalan List

- Definisi List → sekumpulan elemen dengan keterurutan tertentu yang diketahui elemen pertamanya dan setiap elemen boleh muncul lebih dari satu kali
- Definisi rekursif
  - Basis → list kosong adalah sebuah list
  - Elemen list tidak kosong terdiri atas **head** dan **tail**
  - Head adalah elemen list; tail adalah list
- Elemen list → TERM yang bisa berupa konstanta, variabel,struktur yang mungkin juga berupa list
- Pemanfaatan List → representasi tree, *grammar*, mathematical entity, dsb
- Penulisan list dalam pemrograman Deklaratif
  - Dituliskan di antara tanda kurung siku
  - Pemisahan antara head dan tail menggunakan pipe (|)
- Contoh list sederhana
  - List dengan elemen integer

listint([1,2,3]).

listint([X|Y]). → apa yang dihasilkan?

Exercise

- Unifikasi adalah proses pengikatan suatu variabel dengan konstanta tertentu. Untuk setiap kasus di bawah ini, tentukan apakah unifikasi bisa dilakukan terhadap pasangan list. Jika unifikasi bisa dilakukan, tuliskanlah hasil substitusinya.

Catatan: variabel dinyatakan dengan simbol yang diawali huruf kapital.

a) [a,d,z,c] dan [H|T]
b) [a,b,X] dan [a,b,c,d]
c) [apple,pear,grape] dan [A,pear|Rest]
d) [a,[]] dan [A,B|Rest]
e) [a,b,c] dan [b|T]

Answer Exercise

a) H = a, T = [d,z,c]
b) Tidak bisa
c) A = apple, Rest=[grape]
d) A = a, B = [], Rest = []
e) Tidak bisa.

Contoh

- Terdapat program Prolog sebagai berikut.

```
append([ ], X, X) :- !.
append([A|B], C, [A|D]) :- append(B, C, D).
```

- Tentukan hasil dari query berikut, yang diterapkan pada program di atas

a) `append([a, b, c], [d, e], X).`

b) `append(X, Y, [a, b, c]).`

c) `append(X, [ ], Y).`

Contoh

a) X = [a,b,c,d,e]

yes

b) X = []

Y = [a,b,c] ?

yes

c) X = []

Y = [] ?

yes

Writing to Files

- In order to write to a file we have to open a stream
- To write the string 'Hogwarts' to a file with the name hogwarts.txt we do:

```
…
open(‘hogwarts.txt‘, write, Stream),
write(Stream, 'Harry'),
close(Stream),
…
```

- Predicates:
  - open(filename, mode, stream) → write (overwrite), append (add the existing one), read
  - write(Stream, argument) → notes: different from writing to screen
  - tab(Stream, argument)
  - nl(Stream)
  - format(Stream, control, argument)
  - close (Stream)

Reading from Files

- Example file ‘houses.txt’

gryffindor.
hufflepuff.
ravenclaw.
slytherin.

Example Prolog program:

```
main:-
    open('houses.txt',read,S),
    read(S,H1),
    read(S,H2),
    read(S,H3),
    read(S,H4),
    close(S),
    write([H1,H2,H3,H4]), nl.
```

Reading from Files (2)

- Predicates:
  - Open/3
  - Read/2
  - Close/1
- Only works for Prolog term
- Problem 1: the end of file
- The built-in predicate **at_end_of_stream/1** checks whether the end of a stream has been reached
- It will succeed when the end of the stream (given to it as argument) is reached, otherwise if will fail
- We can modify our code for reading in a file using this predicate

Reading from Files (3)

- Modified program:

```
main:-
    open('houses.txt',read,S),
    readHouses(S,Houses),
    close(S),
    write(Houses), nl.

readHouses(S,[]):-
    at_end_of_stream(S), !.

readHouses(S,[X|L]):-
    \+ at_end_of_stream(S), !,
    read(S,X),
    readHouses(S, L).
```

Reading from Files (4)

- Problem 2: reading arbitrary input
- The predicate **get_code/2** reads the next available character from the stream
  - First argument: a stream
  - Second argument: the character code

Reading from Files (5)

- Example: a predicate **readWord/2** that reads atoms from a file

```
main2:-
    open('coba.txt',read,InStream),
    readWord(InStream,W),
    close(InStream),
    write(W).

readWord(InStream,W):- get_code(InStream,Char),
       checkCharAndReadRest(Char,Chars,InStream),
atom_codes(W,Chars).

checkCharAndReadRest(10,[],_):- !./*return*/
checkCharAndReadRest(-1,[],_):- !./*end_of_stream*/
checkCharAndReadRest(end_of_file,[],_):- !.
checkCharAndReadRest(Char,[Char|Chars],InStream):-
get_code(InStream,NextChar),
checkCharAndReadRest(NextChar,Chars,InStream).
```

Loop in Prolog

- We can use ‘repeat’ in Prolog
- Example:

```
command_loop:-
   repeat,
   write('Enter command (end to exit): '),
   read(X),
   write(X), nl,
   X == end.
```

- Example taken from: http://www.amzi.com/AdventureInProlog/a14cntrl.php

Loop in Prolog (2)

Other example:

```
command_loop:-
    write('Welcome to Nani Search'), nl,
    repeat,
    write('>nani> '),
    read(X),
    do(X), nl,
    end_condition(X).

end_condition(end).
end_condition(X) :- have(X),!,
             write('Congratulations').

do(X):- have(X),!.
do(end).
do(_):- write('Invalid Command').

have(X):- X==nani,!.
```

Program stop when user input 'end' or having nani.

`Example taken from: http://www.amzi.com/AdventureInProlog/a14cntrl.php`

THANK YOU