CS370

Symbolic Programming
Declarative Programming

LECTURE 8: Input and Output

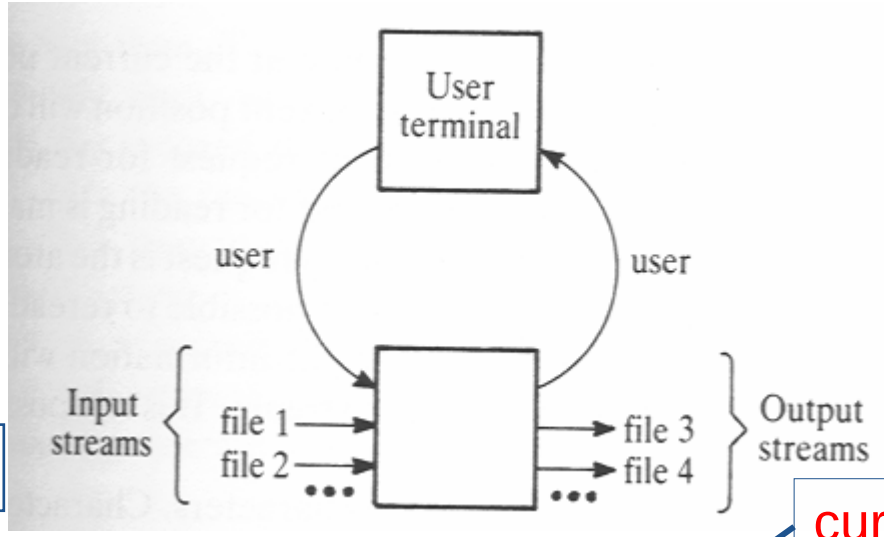
Jong C. Park
park@cs.kaist.ac.kr
Computer Science Department
Korea Advanced Institute of Science and Technology

http://nlp.kaist.ac.kr/~cs370

Input and Output

- Communication with files
- Processing files of terms
- Manipulating characters
- Constructing and decomposing atoms
- Reading programs

Communication with files



current input stream

```
...
see(file1),
read_from_file(Information),
see(user),
...
seen,
```

current output stream

```
...
tell(file3),
write_on_file(Information),
tell(user),
...
told,
```

Communication with files

- File view
  - character-based
    - get
    - get0
    - put
  - term-based
    - read
    - write

Processing files of terms: read and write

- Predicates
  - read(X)
    - X is a term
    - deterministic
    - Each term must be followed by a full stop and a space or carriage-return.
  - write(X)
  - tab(N)
    - Output N spaces

Processing files of terms: read and write

- Examples

```
cube(N,C) :-  C is N * N * N.
?- cube(2,X).
X = 8
?- cube(5,Y).
Y = 125

cube :- read(X),  process(X).
process(stop) :- !.
process(N) :- C is N * N * N, write(C), cube.
?- cube.
2.
8
5.
125
stop.
```

Processing files of terms: read and write

- Example

```
cube :- read(stop), !.
cube :- read(N),
        C is N * N * N,
        write(C), cube.
```

  - Problem?
    - the input stream is chewed up
  - A nicer interface

```
cube :- write('Next: '), read(X), process(X).
process(stop) :- !.
process(N) :- C is N*N*N, write('Cube of'),
        write(N), write('is '), write(C), nl, cube.
```

Processing files of terms: Displaying Lists

- writelist(L)

```
writelist([ ]).
writelist([X|L]) :- write(X), nl, writelist(L).
```

- writelist2(L)

```
?- writelist2([[a,b,c],[d,e,f]]).
a b c
d e f

writelist2([ ]).
writelist2([L|LL]) :- doline(L), nl, writelist2(LL).
doline([ ]).
doline([X|L]) :- write(X), tab(1), doline(L).
```

Processing files of terms: Displaying Lists

- bars(L)

```
?- bars([3,6,5]).
***
******
*****

bars([ ]).
bars([N|L]) :- stars(N), nl, bars(L).
stars(N) :- N > 0, write(*), N1 is N-1, stars(N1).
stars(N) :- N =< 0.
```

Processing files of terms: Processing a file

- Typical goal sequence to process a file

```
..., see(F), processfile, see(user), ...
processfile :-
   read(Term), process(Term).
process(end_of_file) :- !.
process(Term) :-
   treat(Term), processfile.
```

- treat(Term)

```
showfile(N) :- read(Term), show(Term,N).
show(end_of_file,_) :- !.
show(Term,N) :- write(N), tab(2), write(Term), nl,
   N1 is N+1,showfile(N1).
```

Manipulating characters

⊙ put(C), get0(C), get(C)

- get/1 reads non-blank characters, unlike get0/1.

```
?- put(65), put(66), put(67).
ABC

The     robot tried    to pour wine out    of the     bottle.
The robot tried to pour wine out of the bottle.

squeeze :- get0(C), put(C), dorest(C).
dorest(46) :- !.            % 46 is ASCII for full stop.
dorest(32) :- !, get(C), put(C), dorest(C).       % 32 is for
  blank.
dorest(Letter) :- squeeze.
```

Constructing and decomposing atoms

- name(A,L)

```
?- name(baked,L).
L = [98,97,107,101,100]
?- name('미운 오리',L).
L = [185,204,191,238,32,191,192,184,174]
?- name(N,[185,204,184,174]).
N = '미리'
```

- Example

order1, order2, driver1, driver2, taxia1, taxilux

```
taxi(X) :- name(X,Xlist), name(taxi,Tlist), conc(Tlist,_,Xlist).
conc([ ],L,L).    conc([A|L1],L2,[A|L3]) :- conc(L1,L2,L3).
```

Constructing and decomposing atoms

- Example

Mary was pleased to see the robot fail.

Sentence = [‘Mary’,was,pleased,to,see,the,robot,fail]

```
getsentence(Wordlist) :- get0(Char), getrest(Char,Wordlist).
getrest(46,[ ]) :- !.                  % 46 is ASCII for full stop.
getrest(32,Wordlist) :-
        !, getsentence(Wordlist). % 32 is for blank.
getrest(Letter,[Word|Wordlist]) :-
        getletters(Letter,Letters,Nextchar),
        name(Word,Letters),
        getrest(Nextchar,Wordlist).
getletters(46,[ ],46) :- !.
getletters(32,[ ],32) :- !.
getletters(Let,[Let|Letters],Nextchar) :-
  get0(Char), getletters(Char,Letters,Nextchar).
```

Reading programs

- consult(F)
  - ?- consult([program3,program4,queens]).
  - ?- [program3,program4,queens].
- compile(F)
  - ?- compile(program3).
  - ?- compile([program3,queens,program6]).

Summary

- Communication with files
- Processing files of terms
- Manipulating characters
- Constructing and decomposing atoms
- Reading programs