Logic Programming – Laboratory 4

I/O Operations with files

Isabela Drămnesc

Notions

- I/O in Prolog

1 Examples

1.1 Examples of writing

```
a) ?- write('today'),write(' is sunshine').

b) ?- writeq("abc").
[97, 98, 99]
true.

c) ?- write('abc').
abc
true.

d) ?- writeq('abc').
abc
true.

e) ?- write('la la la').
la la la
true.

f) ?- writeq('la la la').
'la la la'
true.

g) ?- writeq('today'),writeq(' is sunshine').

/* for today does not put quotes */

h) ?- put(98).
b
true.
```

```
/* put writes one character (the representation in the ASCII code) */

i) ?- put("99").
ERROR: put/1: Type error: 'character' expected, found '[57,57]'

j) ?- put('99').
ERROR: put/1: Type error: 'character' expected, found '99'

k) ?- get(L).
|: force

L = 102

l) ?- get(L).
|: f

L = 102

/* get reads one single character */

m) ?- write(today is monday),put(8),write(bye).
today is mondabye
true.

n) ?- put(8).
true.

o) ?- write('hello '), put(8), write(lala),put(98), put(99).
hellolalabc
true.

p) ?- write(hello),put(32),write(man).
hello man
true.
```

1.2 Examples of reading

```
a) ?- read(X).
|    hello.   /* this is what introduces the user */

X = hello.
     /* read unifies X with what the user introduces */

b) ?-read(lala).
|    'today is raining'.
false.

c) ?- read(X).
|    today is sunshine.
```

```
X = (today is sunshine).
```

```
d) ?- read(X).
|: 'today is sunshine'.

X = 'today is sunshine'.
```

1.3 Examples of reading from files

Create a file pb.txt, then ask in Prolog:
a) the case when the file pb.txt is empty

```
?- see('C:\\Documents and Settings\\Desktop\\pb.txt'),
read(X),seen.

X = end_of_file.

?- see('C:\\Documents and Settings\\Desktop\\pb.txt'),
read(X), read(I), seen.

X = end_of_file.
I = end_of_file.
```

b) the case when the file pb.txt contains

```
4.
5.
6.
7.
10.

?- see('C:\\Desktop\\pb.txt'), read(X),read(Y),
read(Z),read(W),read(O),seen.

X = 4,
Y = 5,
Z = 6,
W = 7,
O = 10.
```

1.4 Examples of writing in files

```
a) ?- tell('C:\\Desktop\\pb.txt'),
write('the first exercise'),nl,write('is solved'),
write('the second problem is to write into another file'),
told.

true.

b) ?- tell('C:\\Desktop\\pb.txt'),
write('hello'),tab(5),write('again'),told.
```

```
true.

c) ?- tell('C:\\Desktop\\pb2.txt'),
write('the second problem '),nl,
write('is to write a predicate which calculates
the sum of all the numbers from a file '),told.

true.

d) ?- tell('C:\\Desktop\\An2.pdf'),
write('writes in pdf but open with notepad'),told.

true.

/* so we can write in any kind of files */
```

2 Exercises

2.1 The sum

Read all the elements from the file pb.txt and print the sum of the elements.

a) printing the sum on the SWI-Prolog window

```
example:-see('C:\\Desktop\\pb.txt'),
read(X),read(Y),read(Z),read(V),sum([X,Y,Z,V],W),
write('the sum of the elements from the file pb.txt is '),
write(W),seen.

sum([],0).
sum([X|T],W):-sum(T,S),W is X+S.
```

b) print the sum of all the elements from the file pb.txt into another file called sum.txt

2.2 Generate random lists

Write an efficient program (with accumulators) in Prolog which generates a list of a certain length and its elements are random numbers. Examples:
$?-generate\_elem\_list(10000000, 2, L)$.
will return the list $L = [0, 0, 1, 0, 0, 1, 1, 0, 1, 0|.....]$ which has the length 10000000 and which contains binary elements.
$?-generate\_elem\_list(10, 8, L)$. will return $L = [5, 3, 0, 6, 6, 6, 5, 3, 3|...]$.

2.3 Merge–sort

The file pb.txt contains one single number on each line followed by dot. Create a predicate in Prolog which sorts the numbers from the file pb.txt and print the result into a file called sorted.txt.

Implement the merge-sort algorithm for integers in Prolog – informally it can be formulated as follows: Given a list, divide the list into two halves. Sort

the halves and merge the two sorted lists. Sort the list (of the length 10000000) generated at the previous exercise.

2.4 Left–shift

The file pb.txt contains one single number on each line followed by dot. Create a predicate in Prolog which does shift-left (if we have in the file 4.2.3.1. after we apply shift-left we will obtain 2.3.1.4.) and print the result into the file changed1.txt.

2.5 Even–odd

The file pb.txt contains one single number on each line followed by dot. Create a predicate in Prolog which separates the numbers from the file pb.txt into even numbers and odd numbers. Even numbers are printed into the file even.txt, and odd numbers are printed into the file odd.txt.

2.6 Prefixes

Write a program which returns all the prefixes of a list in 3 ways.
1. The recursive version;
2. Using accumulators;
3. Using open lists/difference lists.
Examples:
1. ?- prefix(L,[1,2,3,f,r,4]).
L = [] ;
L = [1] ;
L = [1, 2] ;
L = [1, 2, 3] ;
L = [1, 2, 3, f] ;
L = [1, 2, 3, f, r] ;
L = [1, 2, 3, f, r, 4] ;

Similar for prefix2 and prefix3.

3 Homework:

Consider as input data a file containing 100 lines. On each line there is e a number followed by dot. Print into another file the sorted numbers (increasing way)! Implement at least two sorting algorithms.