COMP348
PRINCIPLES OF PROGRAMMING LANGUAGES

TUTORIAL – 2

PROLOG – PART II Lists

Acknowlegement

- The following slides are reproduced from the tutorial slides of the course COMP 348 by Dr. Mohamed Taleb.

LIST IN PROLOG :- RECALL

- **A List of terms can be represented between brackets:**
  - E.g. [a, b, c, d]. Its head is "a" and its tail is "[b, c, d]".
  - E.g. The tail of [a] is [ ], the empty list.

- **Another representing form of List.**
- The term [X | Y] matches any list with at least one element:
  - X matches the head of the list, and
  - Y matches the tail.

EXAMPLES ON LIST

- Exercise #1

➢ Define first(F, L) to mean "F is a first element of the list L". Example:

?-first(F, [a, b, c]). Answer is : F = a.

Solution:

```
first_element(F, [F|_]).
```

EXAMPLES ON LIST

- Exercise #2
- Define last(X, L) to mean "X is a last element of the list L". Example:

  ?- last(X, [a, b, c]).

  Answer is X = c.

Solution:

```
last_element(L, [L]).
last_element(L, [ _ | R]) :- last_element(L, R).
```

EXAMPLES ON LIST(CONT.)

- Exercise #3
  - Define list_length(L, R) to mean "R is a length of the list L".

    Example:

    ?- list_length([a, b, c, d, e], N).

    Answer is N = 5.

Solution:

```
list_length([ ], 0).
list_length([ _ | T], R) :- list_length(T, R1), R is R1+1.
```

EXAMPLES ON LIST(CONT.)

- Exercise #4
  - Define append_list(L1, L2, Newlist) to mean "Append a list L1 with list L2 to the new list NewList".

    Example:

    ?- append_list([a, b],[c, d, e,], NewList).

    Answer is : NewList = [a, b, c, d, e].

Solution

```
append_list([ ], L, L).
append_list([H | T], L,[H | R]) :- append_list(T, L, R).
```

EXAMPLES ON LIST(CONT.)

- Exercise #5

- Define prefix(P, L) to mean "P is the prefix of L". Example:

  ?- prefix([1], [1,2, 3]). Answer is : true.

- Define suffix(S, L) to mean "S is the suffix of L".

  Example:

  ?- suffix([3], [1,2, 3]). Answer is : true.

Solution:

```
prefix(P, L) :- append(P, _ , L).
suffix(S, L) :- append( _ , S, L).
```

EXAMPLES ON LIST(CONT.)

- Exercise #6
  - Define sublist(S, L) to mean "S is the sublist of L". (hint: use append(L1, L2, NewList)).

    Example: ?- sublist([2, 3],[1, 2, 3, 4]). Answer is : true.

Solution:

```
sublist(S, L) :- append( _ , S, P) , append(P, _ , L).
```

EXAMPLES ON LIST(CONT.)

- Exercise #7

➢ Define reverse(L, R) to mean "the reverse of list L is R".

Example: ?-reverse([a, b, c], R). Answer is : R = [c, b, a].

Solution:

```
reverse_list([ ], [ ]).
reverse_list([H | T], R) :- reverse_list(T, R1) ,

                append_list(R1, [H], R).
```

EXAMPLES ON LIST(CONT.)

- Exercise #8
- Define nth_element(N, X, L) to mean "X is a nth element in the list L at the position N".

Examples:

?- nth_element(4, X, [a, b, c, d, e]).

Answer is : X = e.

?- nth_element(0, X, [a, b, c, d, e]).

Answer is : X = a.

Solution:

```
nth_element(0, X, [X|_]).
nth_element(N, R, [_|T]) :- M is N-1 , nth_element(M, R, T).
```

EXAMPLES ON LIST(CONT.)

- Exercise #9

➢ Define insert_element(E, L, NL) to mean "insert element E to the head of L to generate a new list NL".

Example:

?- insert_element(1, [2, 3], N).

Answer is : N = [1, 2, 3].

Solution:

```
insert_element(R, L, [R | L]).
```

EXAMPLES ON LIST(CONT.)

- Exercise #10
  - Define insert_nth(E, M, L, NL) to mean "insert an element E into M^{th} position of list L to generate a new list NL". Example:

    ?- insert_nth(b, 1, [a, c], N).

    Answer is : N = [a, b,c].

Solution:

```
insert_nth(R, 0, L, [R | L]).
insert_nth(R, N,[H | L], [H | NL]) :- M is N-1 ,
insert_nth(R, M, L, NL).
```

EXAMPLES ON LIST(CONT.)

- Exercise #11

➢ Define delete_nth(N, L, NL) to mean "delete the nth element from list L to generate a new list NL".

Example: ?- delete_nth(1, [a, b, c, d], N).

Answer is : N = [a, c, d].

Solution:

```
delete_nth(0, [ _ | T], T).
delete_nth(N, [H | T], [H | R]) :- M is N-1 ,
delete_nth(M, T, R).
```

EXAMPLES ON LIST(CONT.)

- Exercise #12
- Define delete_element(E, L, NL) to mean "delete all occurrences of element E in a list L to generate a new list NL".

  Example:

  ?- delete_element(a, [a, b, c, a, c, d, a, a, d], N).

  Answer is : N = [b, c, c, d, d].

Solution:

```
delete_element( _ , [ ], [ ]).
delete_element(X, [X | T], R) :- delete_element(X, T, R).
delete_element(X,[Y | T],[Y | T1]) :- X\=Y , delete_element(X, T, T1).
```