Indian Institute of Engineering Science & Technology, Shibpur

Department of Computer Science & Technology.

8th Semester Artificial Intelligence Laboratory.

ASSIGNMENT- 1

(Simple List Processing and Arithmetic in PROLOG)

Duration- 3 periods. **Full Marks (including Viva Voce)-20**

Write PROLOG programs

1. To determine whether the first two elements of a list are same.
2. To determine whether a list is *not* a two-element list.
3. To determine whether two lists are of same length.
4. To determine length of a list using your own number system, that does not contain more than two symbols.
5. To determine whether two lists are of same length using the length predicate developed in 4 (previous problem).
6. To find the last element of a list.
7. To find whether an element is a member of a list.
8. To find whether two elements are next to each other in a list.
9. To append two lists in a third list.
10. To find the last element of a list using append predicate developed in 9.
11. To find whether an element is a member of a list using append predicate developed in 9.
12. To find whether two elements are next to each other in a list using append predicate developed in 9.
13. To reverse a list in another list.
14. To determine whether a list is a palindrome.
    [the structure of predicate:
    palindrome(L)].

15. Write a Prolog program for double (List, ListList), where every element in List appears twice in ListList, e. g., double ([1,2,3], [1,1,2,2,3,3]) is true.

16. To find the **sum** of all elements of a list.
17. To find the **length** of a list.
18. To find the **average** of all elements of a list using **sum** and **length** defined in Problem 16 and 17.
19. To find the **maximum** number from a list.
20. To find **gcd** of two integers.
21. To **generate** all integers **between** two integers N1 and N2, both N1 and N2 included and N2>N1.
22. To **count** numbers **greater** than 100.0 in a list.
23. To **split** a list of numbers in two lists such that one contains negative numbers and other contains positive numbers.
24. To find **N!**
25. To generate first N **Fibonacci** numbers.