Indian Institute of Engineering Science & Technology, Shibpur,

Department of Computer Science & Technology.

8th Semester Artificial Intelligence Laboratory.

ASSIGNMENT- 4

(Permutation, Combination and Sorting)

Duration- 6 periods. **Full Marks (including Viva Voce)-30**

Write Prolog programs for the following.

1. **Remove the K'th element from a list.**
   Example:
   ?- remove_at(X,[a,b,c,d],2,R).
   {X = b, R = [a,c,d]}
2. **Extract a given number of randomly selected elements from a list.**
   The selected items shall be put into a result list.
   Example:
   ?- rnd_select([a,b,c,d,e,f,g,h],3,L).
   {L = [e,d,a]}

   Hint: Use the built-in random number generator random/2 and the result of problem P1.
3. **Generate a random permutation of the elements of a list.**
   Example:
   ?- rnd_permu([a,b,c,d,e,f],L).
   {L = [b,a,d,c,e,f]}

   Hint: Use the solution of problem P2.
4. **Generate the combinations of K distinct objects chosen from the N elements of a list**
   In how many ways can a committee of 3 be chosen from a group of 12 people? We all know that there are C(12,3) = 220 possibilities (C(N,K) denotes the well-known binomial coefficients). For pure mathematicians, this result may be great. But *we* want to really generate all the possibilities (via backtracking).

Example:
?- combination(3,[a,b,c,d,e,f],L).
{L = [a,b,c]} ;
{L = [a,b,d]} ;
{L = [a,b,e]} ;
...

5. **Sorting a list of lists according to length of sublists**

   a) We suppose that a list (InList) contains elements that are lists themselves. The objective is to sort the elements of InList according to their **length**. E.g. short lists first, longer lists later, or vice versa.

   Example:
   ?- lsort([[a,b,c],[d,e],[f,g,h],[d,e],[i,j,k,l],[m,n],[o]],L).
   {L = [[o], [d, e], [d, e], [m, n], [a, b, c], [f, g, h], [i, j, k, l]]}

   b) Again, we suppose that a list (InList) contains elements that are lists themselves. But this time the objective is to sort the elements of InList according to their **length frequency**; i.e. in the default, where sorting is done ascendingly, lists with rare lengths are placed first, others with a more frequent length come later.

   Example:
   ?- lfsort([[a,b,c],[d,e],[f,g,h],[d,e],[i,j,k,l],[m,n],[o]],L).
   {L = [[i, j, k, l], [o], [a, b, c], [f, g, h], [d, e], [d, e], [m, n]]}

   Note that in the above example, the first two lists in the result L have length 4 and 1, both lengths appear just once. The third and forth list have length 3 which appears, there are two list of this length. And finally, the last three lists have length 2. This is the most frequent length.

6. **Implement Permutation Sort.**
7. **Implement Bubble Sort.**
8. **Implement Selection Sort.**
9. **Implement Insertion Sort.**
10. **Implement Merge Sort.**
11. **Implement Quick Sort using Accumulator.**
12. **Implement Quick Sort without Accumulator.**