**ASSIGNMENT-1 PROLOG**

**Note:** Do not copy any part of code from any online or offline sources. Your code will be checked using MOSS and plagiarism will be dealt seriously. Instruction for submission will be intimated soon.

**Last date of submission:** 16th Aug.

**Demo date:** 17th/18th Aug

**LISTS - (10 Marks)**

Q1. Write rules in prolog to implement set operations-

1. *create\_set(X,Y)* : Create set from a list entered by user i.e., remove duplicates.
2. *union(X,Y,Z)* : takes X and Y two sets (represented by lists) as parameters,   
   Z = X ∪ Y
3. *intersection(X,Y,Z)* : takes X and Y two sets (represented by lists) as parameters,   
   Z = X ⋂ Y.
4. *difference(X,Y,Z)* : takes X and Y two sets (represented by lists) as parameters,   
   Z = X - Y.
5. *product(A,B,Z)* : takes A and B two sets (represented by lists) as parameters,   
   Z = A ⨯ B i.e., *cartesian product* of set A and B.

**TREES - (10 Marks)**

Q2. Write rules in prolog to implement tree operations-

Node Structure

*node(A,B,C)* : where A is the left subtree, B will be an integer value and C is the right subtree.

1. *create\_BST(X,Y)* : Create binary search tree from a list Y entered by user.
2. *pre\_order(X,Y) , in\_order(X,Y) &*  *post\_order(X,Y)*: where X is the binary tree on which traversal needs to be done and Y is a list which will contain the final traversal node values.
3. *count(X,Y)* : counts the no. of nodes in binary tree X and returns the count as Y.
4. *height(X,Y)* : finds the height of binary tree X and returns the height as Y. Assuming height of the root node is 0.
5. *find(X,Y)* : find value Y in binary search tree X and return true if value exists otherwise false.