**Experiment-19**

* **Aim :**

Write a prolog program to find the union of two given sets represented as lists.

* **Description :**

(Y is list1, Z is list2 and W is new list)

• For each element of list1 which is already present in list2 :

union([X|Y],Z,W) :- member(X,Z), union(Y,Z,W).

• For each element of list1 which is not present in list2:

union([X|Y],Z,[X|W]) :- \+ member(X,Z), union(Y,Z,W).

• Copy all the values of list2 into the new list.

union([],Z,Z).

* **Program :**

union([X|Y],Z,W) :-

list\_member(X,Z), union(Y,Z,W).

union([X|Y],Z,[X|W]) :-

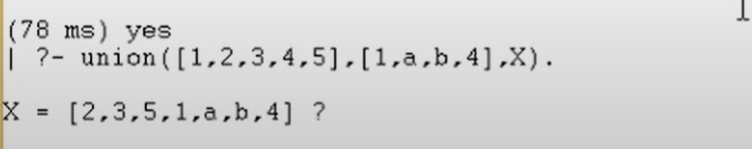
\+ list\_member(X,Z), union(Y,Z,W).

union([],Z,Z). list\_member(X,[X|\_]).

list\_member(X,[\_|TAIL])

list\_member(X,TAIL).

* **Output :**

****

**Experiment-25**

* **Aim :**

Define a LISP function to compute the difference of squares. (if x > y return x2– y2, otherwise return y2 – x2).

* **Description :**

**If:** The if construct has various forms. In the simplest form, it is followed by a test clause, a test action, and some other consequent action(s). If the test clause evaluates to true, then the test action is executed otherwise, the consequent clause is evaluated.

* **Program :**

(defun diff(x y)

(if (> x y)

(setq result (- (\* x x) (\* y y)))

(setq result (- (\* y y) (\* x x)))

)

(format t "difference is : ~d" result)

)

(diff 8 6)

* **Output :**

