St. Francis Institute of Technology
Borivli (W), Mumbai 400103
Department of Information Technology

Name:Vedant Kadam , Roll no-22 , SEITA ,Batch 1

Experiment-10

1. Aim: To implement list comprehensions using Prolog

2. Objectives: After performing the experiment, the students will be able to
- Understand Lists and construct List
- Perform List operations

3. Lab Objective Mapped: To understand, formulate and implement declarative programming paradigm through logic programming

4. Prerequisite: Knowledge of facts, rules, constants and variables

5. Requirements: The following are the requirements –
- **Internet connection**
- **Laptop/desktop with Windows/Linux/MAC operating system**
- **SWI Prolog**

6. Pre-Experiment Theory:

Lists in Prolog refers to an ordered sequence of elements. It is also a collection of terms, which is useful for grouping items together, or for dealing with large volumes of related data.

Example
[red, white, black, yellow]
Lists are enclosed by square brackets, and items are separated by commas. The length of a list is the number of items it contains. The length of this list is 4.

List is made up of two parts: the first element, known as the **Head,** and everything else, called the **Tail.** Prolog uses a built-in operator, the **pipe (|)** in order to split the list as Head and Tail. If unification is applied with the above example then,

[Head|Tail] = [red, white, black, yellow]. Will result in following output
Head = red
Tail = [white, black, yellow]

The other important points related to lists are-
- A single element in a list can be represented as [a]
- An empty list can be represented as []
- The elements of lists are separated by commas. Compound lists are also possible ▪ [first, second, third] = [A|B] where A = first and B= [second, third]. If the unification succeeds, then A is bound to the first item in the list, and B to the remaining list.
- [ ] is a special list, it is called the empty list because it contains nothing. Its length is 0

The following list comprehensions/ Operations can be performed in Prolog at prolog prompt

Sr. No	List Operation	Definition	Example
1	Membership Checking	To verify whether a given element is member of specified list or not	member(x, [x,y,z]).
2	Length Calculation	To find the length of a list.	length([a,b],M).
3	Append Items	To append one list into another (as an item).	append([a,b],[1,2,3],L).
4	Reverse	To reverse the elements of list	reverse([1,2,3],A).

7. Lab Laboratory Exercise:

A. Procedure

Steps to be implemented

1. Open SWI-Prolog
2. Go to ….File-> new
3. New prolog editor will open up.
4. Write your program (collection of facts, rules, clauses)
5. Save the file at the desired location as ‘abc.pl’ file
6. Follow the steps -> Save buffer, Make and Compile buffer
7. After successful compilation, at the prompt, first change to working directory using cd command Eg. cd('D:/PCPF/AY-2021-22/Course Lab/Prolog Codes').
8. Load the required file Eg. [abc]
9. To check the outputs, fire appropriate queries at the prompt

B. Program Code

1. Execute the following List commands at Prolog Prompt. Make proper observations of the outputs. Make a note of the commands that give error. Analyse the reason for the error

```
member (x, [x,y,z]).
member(p, [x,y,z]).
member(my(x,y,z),[q,r,s,my(x,y,z),w]).
member(v,[]).
length([a,b],M).
length([1,2,3],M).
length([1,2,3],a).
length([1,2,3],X1).
length([1,2,3],X-1).
length([[a,c],[e,f],[h,i]],N).
length([],P).
length([a,b,c],3).
reverse([1,2,3],A).
reverse(B, [1,2,3]).
reverse([[dog,cat],[1,2],[bird,mouse]],L).
reverse([1,2,3,4],[4,3,6,8]).
reverse([1,2,3,4],[4,3,2,1]).
append([],[1,2,3],L).
append([a,b],[1,2,3],L).
```

File Edit Settings Run Debug Help

```
?- member(x,[x,y,z]).
true .

?- member(p,[x,y,z]).
false.

?- member(my(x,y,z),[q,r,s,my(x,y,z),w]).
true .

?- member(v,[]).
false.

?- length([a,b],M).
M = 2.

?- length([1,2,3],M).
M = 3.

?- length([1,2,3],a).
ERROR: Type error: `integer' expected, found `a' (an atom)
ERROR: In:
ERROR:   [11] throw(error(type_error(integer,a),context(...,_24108)))
ERROR:    [9] <user>
ERROR:
ERROR: Note: some frames are missing due to last-call optimization.
ERROR: Re-run your program in debug mode (:- debug.) to get more detail.
?- length([1,2,3],X1).
X1 = 3.

?- length([1,2,3],X-1).
ERROR: Type error: `integer' expected, found `_26104-1' (a compound)
ERROR: In:
ERROR:   [11] throw(error(type_error(integer,...),context(...,_26168)))
ERROR:    [9] <user>
ERROR:
ERROR: Note: some frames are missing due to last-call optimization.
ERROR: Re-run your program in debug mode (:- debug.) to get more detail.
?- length([[a,c],[e,f],[h,i]],N).
N = 3.

?- length([],P).
P = 0.

?- length([a,b,c],3).
true.

?- reverse([a,b,c],3).
false.

?- reverse([1,2,3],A).
A = [3, 2, 1].

?- reverse(B,[1,2,3]).
B = [3, 2, 1] .

?- everse([[dog,cat],[1,2],[bird,mouse]],L).
Correct to: "reverse([[dog,cat],[1,2],[bird,mouse]],L)"?
Please answer 'y' or 'n'? yes
L = [[bird, mouse], [1, 2], [dog, cat]].

?-
|    reverse([1,2,3,4],[4,3,6,8]).
false.

?- reverse([1,2,3,4],[4,3,2,1]).
true.

?- append([],[1,2,3],L).
L = [1, 2, 3].

?- append([a,b],[1,2,3],L).
L = [a, b, 1, 2, 3].

?- append([],[1,2,3],L).
L = [1, 2, 3].
```

```
?- append([a,b,23],[1,2,3],L).
L = [a, b, 23, 1, 2, 3].
```


1. Write a program in Prolog to concatenate two lists.

concat.pl

File Edit Browse Compile Prolog Pce Help

concat.pl

```
concatenation([],[],[]).
concatenation([],L,L).
concatenation([X1|L1],L2,[X1|L3]):-concatenation(L1,L2,L3).
```

OUTPUT:

```
?- [concat].
true.

?- concatenation([8],[7],L1).
L1 = [8, 7].

?- concatenation
% c:/users/91992/documents/prolog/concat compiled 0.00 sec, 2 clauses
% c:/users/91992/documents/prolog/concat compiled 0.00 sec, 0 clauses
?- concatenation([1,2,3],[6,7,9],L3).
L3 = [1, 2, 3, 6, 7, 9]
% c:/users/91992/documents/prolog/concat compiled 0.00 sec, -1 clauses
```

2. Write a program in Prolog to delete an item from a given list.

del.pl [modified]

File Edit Browse Compile Prolog Pce Help

concat.pl | del.pl [modified]

```
del([],[],[]).
del(Y,[Y],[]).
del(X,[X|LIST1],LIST1).
del(X,[Y|LIST],[Y|LIST1]):-del(X,LIST,LIST1).
```

OUTPUT:

```
?- del(1,[1,2,3],X1).
X1 = [2, 3] ;
false.

?- del([],[],L1).
L1 = [] ;
false.

?-
|    del(2,[4,1,2,3],C1).
C1 = [4, 1, 3]
```

3. Write a program in Prolog to insert an item in a given list.

insert.pl

File Edit Browse Compile Prolog Pce Help

concat.pl | del | insert.pl

```
insert([],[],[]).
insert(X,[],[X]).
insert(X,L,L1):-delete(X,L1,L).
```

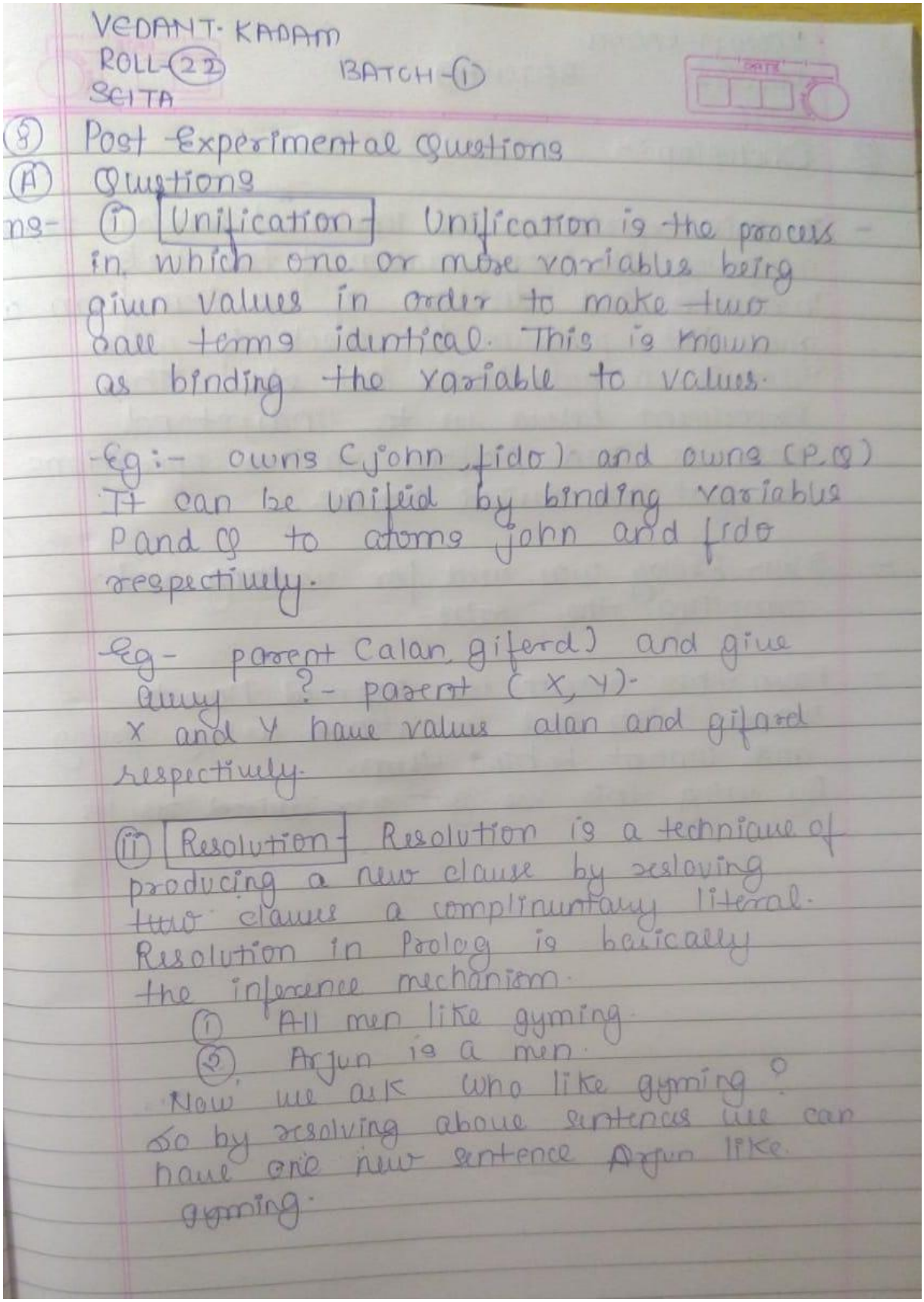
OUTPUT:

```
?- insert([1,2],[],l1).
false.

?- insert([1,2],[],L1).
L1 = [[1, 2]]
```

8. Post Experimental Exercise

A. Questions:

1. Define the terms-> (i) Unification (ii) Resolution. Give suitable examples.



VEDANT- KADAM
ROLL-(22) BATCH-(1)
SEITA

(8) Post Experimental Questions
(A) Questions
ns- (i) [Unification] Unification is the process in which one or more variables being given values in order to make two sall terms identical. This is known as binding the variable to values.

-Eg:- owns (john, fido) and owns (P, Q)
It can be unified by binding variables P and Q to atoms john and fido respectively.

Eg- parent (alan, gifard) and give query ?- parent (X, Y).
X and Y have values alan and gifard respectively.

(ii) [Resolution] Resolution is a techniaue of producing a new clause by resloving two clause a complimentary literal. Resolution in Prolog is baically the inference mechanism.
(1) All men like gyming.
(2) Arjun is a men.
Now we ask who like gyming?
So by resolving above sentences we can have one new sentence Arjun like gyming.

2. Write a program in Prolog to check if the given list is a palindrome list.

palindrome1.pl

```
palindrome(List):-list_reverse(List,List).
list_reverse([],[]).
list_reverse([First|Rest],Reversed):-list_reverse(Rest,ReversedRest), concatenation(ReversedRest,[First],Reversed).
concatenation([],L,L).
concatenation([X1|L1],L2,[X1|L3]):- concatenation(L1,L2,L3).
```

OUTPUT:

```
?- palindrome([m,a,d,a,m]).
true.

?- palindrome([a,d,a,m]).
false.
```

3. Write a program in Prolog to find all possible subsets of a given list.

palindrome1.pl | subset.pl

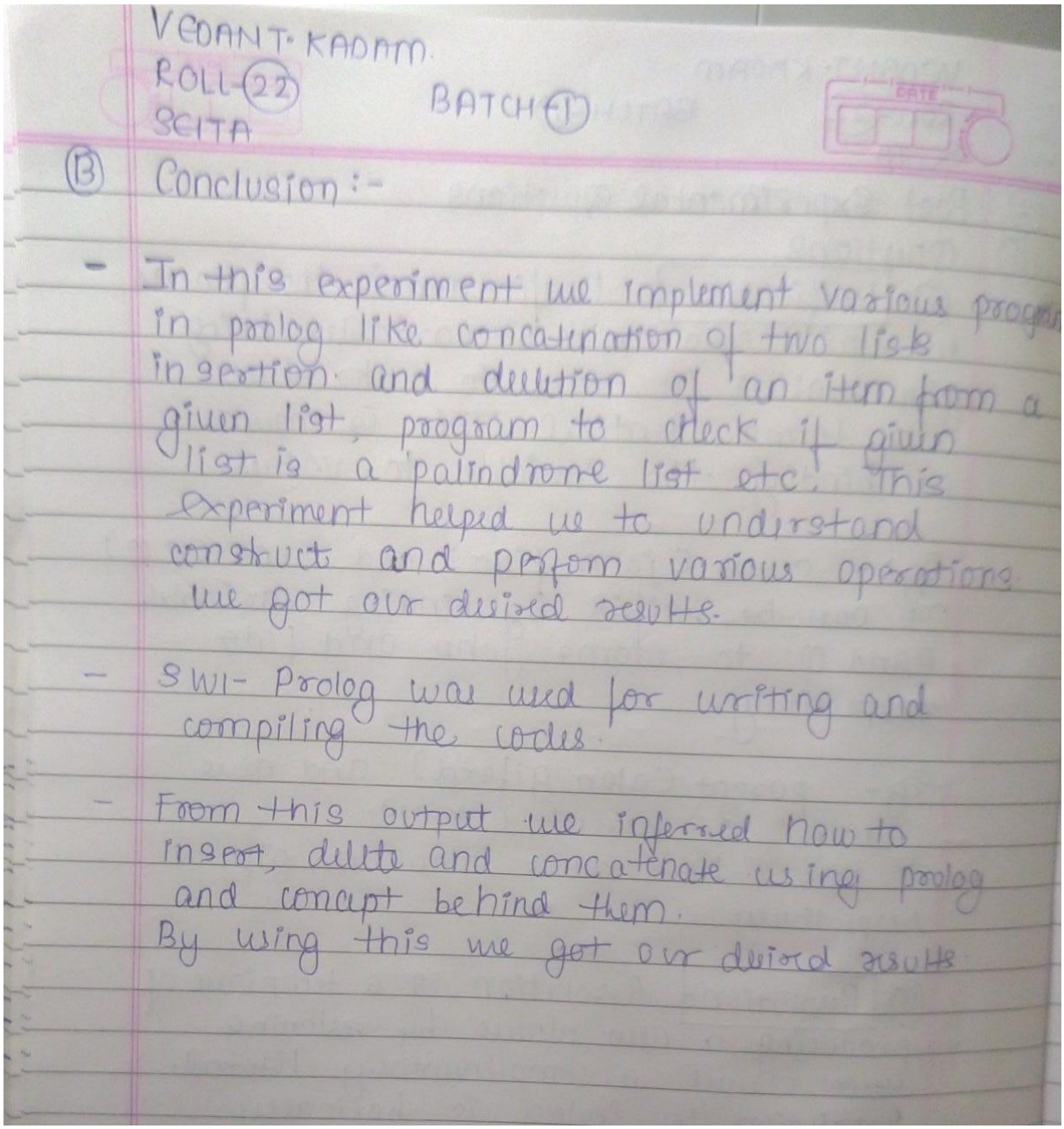
```
subseq(List,List).
subseq(List,Rest):-subseq1(List,Rest).

subseq1([_|Tail],Rest):-subseq(Tail,Rest).
subseq1([Head|Tail],[Head|Rest]):-subseq1(Tail,Rest).
```

OUTPUT:

```
?- consult('sub.pl').
true.

?- subseq0([a,b],X).
X = [a, b] .

?- subseq0([a,b,c,d],X),
|    .
ERROR: Syntax error: Unexpected ',' before '.'
ERROR: subseq0([a,b,c,d],X),
ERROR: ** here **
ERROR:   .
?- subseq0([a,b,c,d],X).
X = [a, b, c, d] .

?- subseq0([a,b,c,d],X).
X = [a, b, c, d] ;
X = [b, c, d] ;
X = [c, d] ;
X = [d] ;
X = [] ;
X = [c] ;
X = [b, d] ;
X = [b] ;
X = [b, c] ;
X = [a, c, d] ;
X = [a, d] ;
X = [a] ;
X = [a, c] ;
X = [a, b, d] ;
X = [a, b] ;
X = [a, b, c] ;
false.

?-
```

B. Conclusion:

1. Write what was performed in the experiment
2. Write which tools you used to perform the experiment
3. Write what you inferred from the output obtained


VEDANT KADAM
ROLL 22 BATCH T1
SEITA

(B) Conclusion :-

- In this experiment we implement various progra in prolog like concatenation of two lists insertion and deletion of an item from a given list, program to check if given list is a palindrome list etc. This experiment helped us to understand construct and perform various operations we got our desired results.
- SWI- Prolog was used for writing and compiling the codes.
- From this output we inferred how to insert, delete and concatenate using prolog and concept behind them.
By using this we got our desired results

D. References:

[1] Michael L Scott, "Programming Language Pragmatics", Third edition, Elsevier publication

[2] Max Bramer, " Logic Programming with Prolog", Springer, 2005

[3] https://www.youtube.com/watch?v=iJhtgWAGUAQ [Lecture 14 Prolog Programming]