Name – Indranil Bain DEPT. – Computer Science & Technology
Enrolment No. – 2020CSB039, Assignment - 01
Subject – Artificial Intelligence LAB

```
#1. To find last element of a list.
last_element([X], X).
last_element([_|Tail], Last) :- last_element(Tail, Last).
```

```
#2. To append two lists in a third list.
append_list(L1, L2, L3) :- append(L1, L2, L3).
```

```
#2. To append two lists in a third list.
reverse_list(L1, L2) :- reverse(L1, L2).
```

```
# 4. To determine whether a list is a palindrome.
# The structure of predicate:
# Palindrome(L)
same([], []).
same([L1|L1T], [L2|L2T]) :- L1=L2, same(L1T, L2T).
palindrome(L) :- reverse(L, L1), same(L, L1).
```

```
# 5. To find the kth element of a list.
element_at(X, [H|T], 1) :- X is H.
element_at(X, [H|T], P) :- P2 is P-1, element_at(X, T, P2).
```

```
#6. To find the sum and average of all elements of a list using sum and length.
sum([], 0).
sum([H|T], Sum) :- sum(T, X), Sum is H + X.
average(L, Avg) :- sum(L, Sum), length(L, Len), Avg is Sum/Len.
```

```
#7. To find gcd of two integers.
gcd(A, B, 1, _).
gcd(A, B, GCD, X) :- A mod GCD =:= 0, B mod GCD =:= 0, X is GCD ; gcd(A, B, GCD-1, X).
get_gcd(A, B, X) :- max_list([A,B], Y), gcd(A, B, Y, X).
```

```
#8. To determine whether a given integer number is prime or not.
check(A, 1).
check(A, X) :- X2 is X-1, A mod X =\= 0, check(A, X2).
is_prime(A) :- A=\=1, A2 is A-1, check(A, A2).
```

```
#9. To determine the prime factors of a given positive integer.
smallest_factor(A, X, A1) :- A mod X =:= 0, is_prime(X), A1 is X; X<A, smallest_factor(A, X+1, A1).
```

```
prime_factors(1, []).
prime_factors(A, L) :- smallest_factor(A, 2, A1), A2 is A/A1, prime_factors(A2, L1), append([A1], L1, L).

# #10.Goldbach’s conjecture says that every positive even number greater than 2 is the sum
# of two prime numbers. Example: 28 = 5 + 23. It is one of the most famous facts in
# number theory that has not been proved to be correct in the general case. It has been
# numerically confirmed up to very large numbers. Write a predicate to find the two prime
# numbers that sum up to a give even integer.
# solve(A, X, L) :- X<A-1, is_prime(X), is_prime(A-X), X3 is A-X, L = [X, X3]; X2 is X+1, solve(A, X2, L).
# goldbach(A, L) :- solve(A, 2, L).

#11. To generate first N Fibonacci numbers.
fibonacci_term(1, 1).
fibonacci_term(2, 1).
fibonacci_term(A, X) :- A1 is A-1, fibonacci_term(A1, X1), A2 is A-2, fibonacci_term(A2, X2), X is X1+X2.
fibonacci(1, [1]).
fibonacci(A, L) :- A > 1, A_PREV is A-1, fibonacci(A_PREV, L1), fibonacci_term(A, X1), append(L1, [X1], L).


# #12. Consider a database of facts that describe parent rela4onships as well as gender
# rela4onships. The predicate parent(john,ann) is interpreted as: "John is a parent of
# Ann". The predicate male(john) is interpreted as: "John is a man". The predicate
# female(ann) is interpreted as: "Ann is a woman". So an example database of facts is:
parent(john,ann).
parent(jim,john).
parent(jim,keith).
parent(mary,ann).
parent(mary,sylvia).
parent(brian,sylvia).
male(keith). male(jim).
female(sylvia).
female(ann).
male(brian).

# a) Write a Prolog predicate uncle(X,Y) that is true if X is Y's uncle. Note that we are
# not considering uncles "by marriage", meaning that for X to be Y's uncle the two
# must be related by blood. For instance :
uncle(A, B) :- male(A), parent(X, B), parent(Y, X), parent(Y, A).

# b) Write a Prolog predicate halfsister(X,Y) that is true if X is Y's half-sister. For
# instance
halfsister(A, B) :- female(A), parent(X, A), parent(X, B), parent(Y, A), parent(Z, B), not(Y=Z), not(A=B).
```

Answer:

Relations:-

```
parent(john,ann).
parent(jim,john).
parent(jim,keith).
parent(mary,ann).
parent(mary,sylvia).
parent(brian,sylvia).
male(keith).
male(jim).
male(brian).
male(john).
female(sylvia).
female(ann).
female(mary).
female(sylvia).
```

Predicates:

```
uncle(X,Y):-parent(Z,Y),brother(X,Z).
same(X,X).
brother(X,Y):-male(X),siblings(X,Y).
siblings(X,Y):-parent(Z,X),parent(Z,Y).
halfsister(X,Y):-
parent(A,X),parent(B,X),parent(C,Y),parent(D,Y),same(A,C),not(same(B,D)),not(same(X,Y)).
```

Queries:

```
a) ?- uncle(X,Y).
X = keith,
Y = ann .

b) ?- halfsister(X,ann).
X = sylvia .
```