Experiment-3:

Implementing the prolog programs for example BFS, DFS, GCD, Area of Circle, Area of Triangle, Factorial and Quadratic Equation.

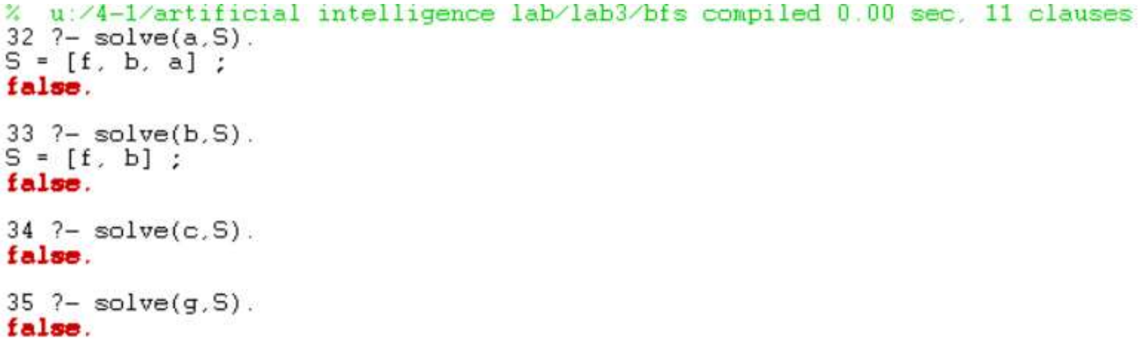
Objective:

We have to create some prolog codes of BFS, DFS, GCD, Area of Circle, Area of Triangle, Factorial and Quadratic Equation and then checking the validation of these codes.

BFS(Breadth First Search):

```
s(a, b).
s(a, c).
s(b, g).
s(b, f).
s(c, r).
s(c, e).
goal(f).

solve( Start, Solution) :-
    breadthfirst( [ [Start] ], Solution).

breadthfirst( [ [Node | Path] |_], [Node | Path] ) :-
    goal( Node).

breadthfirst( [ [N | Path] | Paths], Solution) :-
    bagof([M,N|Path],
    ( s( N, M), \+ member( M, [N | Path] ) ), NewPaths),
    append(Paths, NewPaths, Pathsl), !,
    breadthfirst( Pathsl, Solution);
    breadthfirst( Paths, Solution).
```

Output:

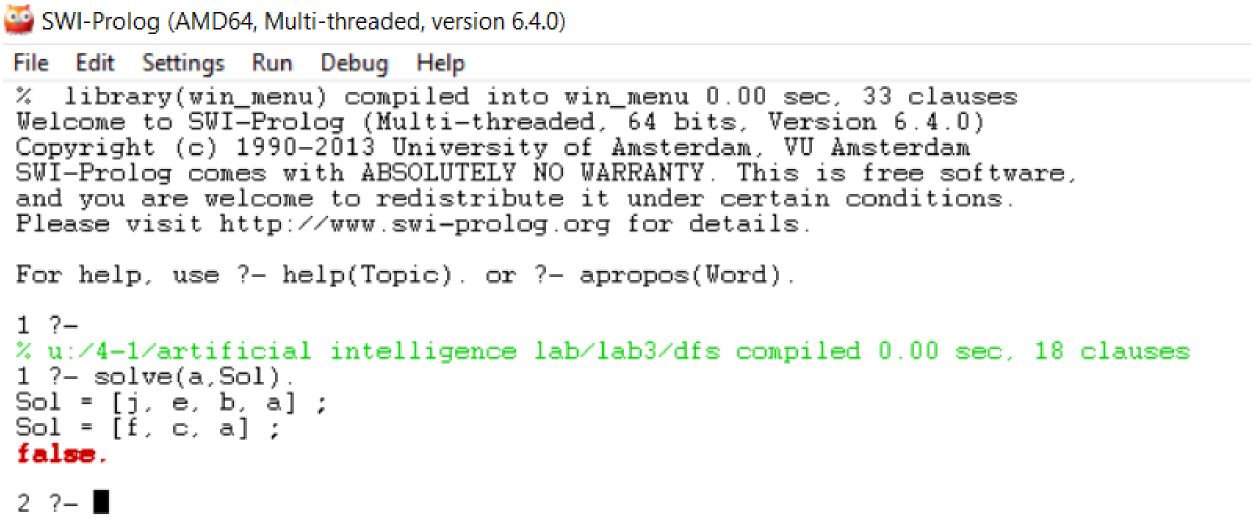


```
%  u:/4-1/artificial intelligence lab/lab3/bfs compiled 0.00 sec, 11 clauses
32 ?- solve(a,S).
S = [f, b, a] ;
false.

33 ?- solve(b,S).
S = [f, b] ;
false.

34 ?- solve(c,S).
false.

35 ?- solve(g,S).
false.
```

DFS(Depth First Search):

```
s(a,b).
s(a,c).
s(b,d).
s(b,e).
s(c,f).
s(c,g).
s(d,h).
s(e,i).
s(e,j).
s(f,k).

goal(f).
goal(j).

member(X, [X|_]).
member(X, [_|Tail]) :-member(X, Tail).

solve(Node,Solution) :-
        depthfirst([],Node,Solution).

depthfirst(Path,Node,[Node|Path]) :-
        goal(Node).

depthfirst(Path,Node,Sol) :-
        s(Node,Node1),
        not(member(Node1,Path)),
        depthfirst([Node|Path],Node1,Sol).
```

Output:


SWI-Prolog (AMD64, Multi-threaded, version 6.4.0)
File Edit Settings Run Debug Help

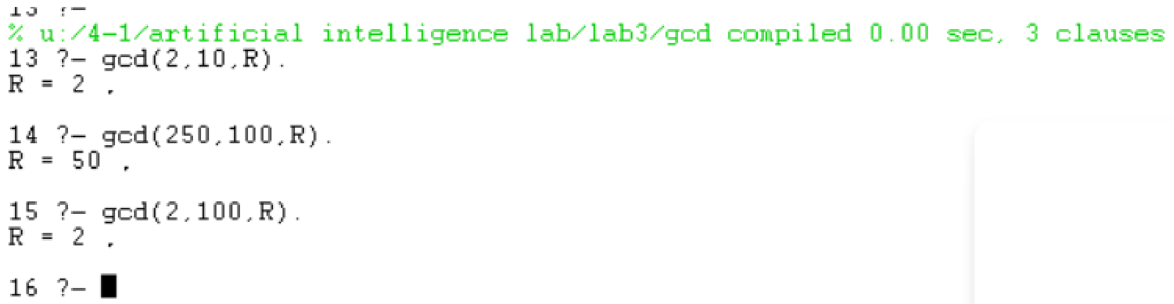
```
%  library(win_menu) compiled into win_menu 0.00 sec, 33 clauses
Welcome to SWI-Prolog (Multi-threaded, 64 bits, Version 6.4.0)
Copyright (c) 1990-2013 University of Amsterdam, VU Amsterdam
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions.
Please visit http://www.swi-prolog.org for details.

For help, use ?- help(Topic). or ?- apropos(Word).

1 ?-
% u:/4-1/artificial intelligence lab/lab3/dfs compiled 0.00 sec, 18 clauses
1 ?- solve(a,Sol).
Sol = [j, e, b, a] ;
Sol = [f, c, a] ;
false.

2 ?-
```

GCD(Greatest Common Divisor):

% Write a Prolog program to implement GCD of two numbers.

```
/* GCD of two numbers. */
gcd(X,0,X).
gcd(X,Y,Z):-
 R is mod(X,Y),
 gcd(Y,R,Z).
```

Output:

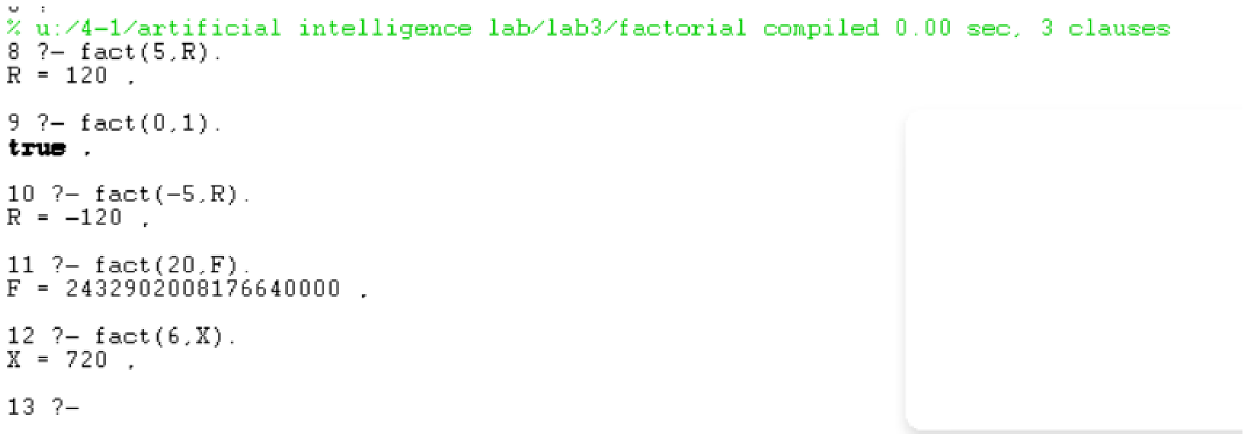


```
13 ?-
% u:/4-1/artificial intelligence lab/lab3/gcd compiled 0.00 sec, 3 clauses
13 ?- gcd(2,10,R).
R = 2 .

14 ?- gcd(250,100,R).
R = 50 .

15 ?- gcd(2,100,R).
R = 2 .

16 ?-
```

Factorial:

```
fact(0,1).
fact(N,F):-
(

 % The below is for +ve factorial.
 N>0 ->
 (
  N1 is N-1,
  fact(N1,F1),
  F is N*F1
 )
 ;

 % The below is for -ve factorial.
 N<0 ->
 (
  N1 is N+1,
  fact(N1,F1),
  F is N*F1
 )
).
```

Output:

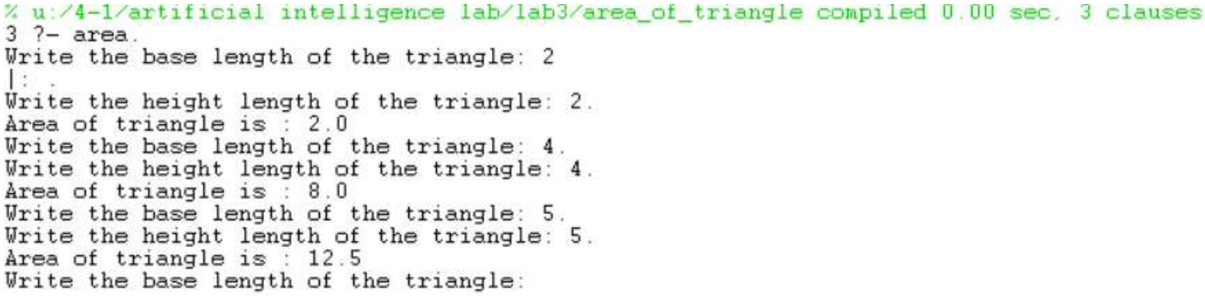


```
% u:/4-1/artificial intelligence lab/lab3/factorial compiled 0.00 sec, 3 clauses
8 ?- fact(5,R).
R = 120 .

9 ?- fact(0,1).
true .

10 ?- fact(-5,R).
R = -120 .

11 ?- fact(20,F).
F = 2432902008176640000 .

12 ?- fact(6,X).
X = 720 .

13 ?-
```

Area of Triangle:

```
area :-
   write('Write the base length of the triangle: '),
   read(B),
   write('Write the height length of the triangle: '),
   read(H),
   process(B,H).

process(B,H) :-
   A is 0.5 * B * H,
   write('Area of triangle is : '),write(A),nl, area.
```

Output:

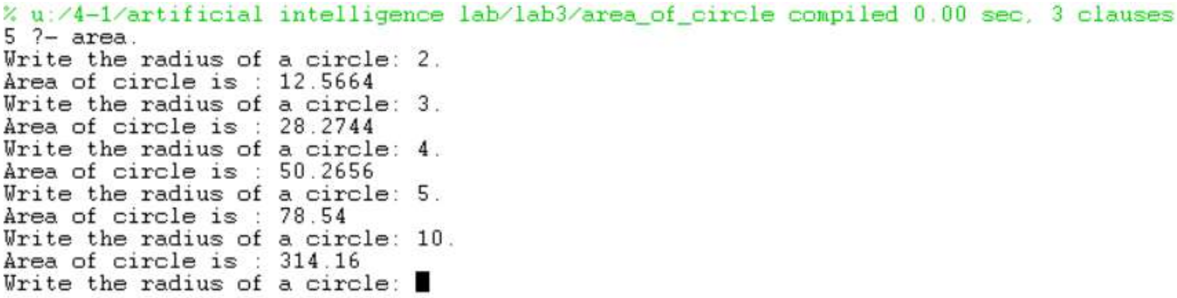


```
% u:/4-1/artificial intelligence lab/lab3/area_of_triangle compiled 0.00 sec, 3 clauses
3 ?- area.
Write the base length of the triangle: 2
|: .
Write the height length of the triangle: 2.
Area of triangle is : 2.0
Write the base length of the triangle: 4.
Write the height length of the triangle: 4.
Area of triangle is : 8.0
Write the base length of the triangle: 5.
Write the height length of the triangle: 5.
Area of triangle is : 12.5
Write the base length of the triangle:
```

Area of Circle:

```
area :-
   write('Write the radius of a circle: '),
   read(R),
   process(R).
process(R) :-
   A is 3.1416 * R * R,
   write('Area of circle is : '),write(A),nl, area.
```

Output:

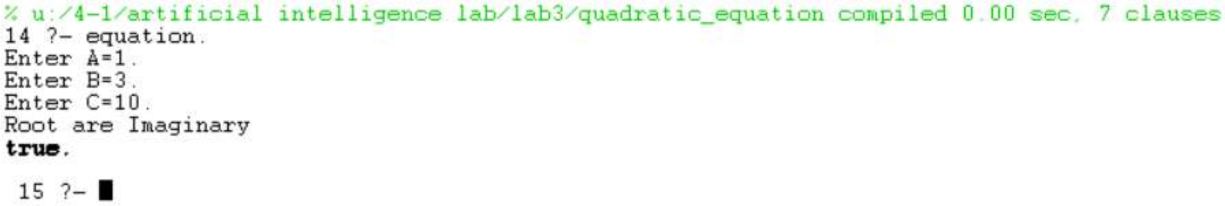


```
% u:/4-1/artificial intelligence lab/lab3/area_of_circle compiled 0.00 sec, 3 clauses
5 ?- area.
Write the radius of a circle: 2.
Area of circle is : 12.5664
Write the radius of a circle: 3.
Area of circle is : 28.2744
Write the radius of a circle: 4.
Area of circle is : 50.2656
Write the radius of a circle: 5.
Area of circle is : 78.54
Write the radius of a circle: 10.
Area of circle is : 314.16
Write the radius of a circle: █
```

Quadratic Equation:

```
equation:-
 write('Enter A='),
 read(A),
 write('Enter B='),
 read(B),
 write('Enter C='),
 read(C),
 quadroot(A,B,C),!.

 quadroot(A,B,C):-
 D=(B*B)-(4*A*C),
 sol(A,B,C,D),!.

 sol(_,_,_,D):-
 D<0,
 write('Root are Imaginary'),!.
 sol(A,B,_,D):-
 X1=(-B+sqrt(D))/2*A,
 X2=(-B-sqrt(D))/2*A,
 write('X1=',X1),
 write('X2=',X2),!.

 equal(X1,X2):-
 X1=X2,
 write('Roots are equal'),!.

 equal(X1,X2):-
 X1==X2,
 write('Roots are real'),!.
```

Output:



```
% u:/4-1/artificial intelligence lab/lab3/quadratic_equation compiled 0.00 sec, 7 clauses
14 ?- equation.
Enter A=1.
Enter B=3.
Enter C=10.
Root are Imaginary
true.

 15 ?- 
```