National Taipei University of Technology
Artificial Intelligence (spring, 2014)

Homework #3

(Due: Wednesday, April 9th)

1. (30%) Define the predicate

   `odd_even( List, OddList, EvenList).`

   `OddList` is the odd numbers in the **`List`**, and **`EvenList`** is the even numbers in the **`List`**. For example: **`odd_even([2,4,5,6,8,1,7], [5,1,7], [2,4,6,8]).`**

2. (50%) Modify the two programs of eight queens problem for solving the N-queens problem. (Hint: The program 3 in the book solve the N-queens by using the procedure **`gen`**.)

   Program 1

```
% Program 1
solution([]).
solution([X/Y|Others]) :-
    solution(Others),
    member(Y, [1, 2, 3, 4, 5, 6, 7, 8]),
    noattack(X/Y, Others).

noattack(_,[]).
noattack(X/Y, [X1/Y1|Others]) :-
    Y=\=Y1, Y1-Y=\=X1-X, Y1-Y=\=X-X1,
    noattack(X/Y, Others).

template([1/Y1, 2/Y2, 3/Y3, 4/Y4, 5/Y5, 6/Y6, 7/Y7, 8/Y8]).
```

```
?- template(S), solution(S).
S = [1/4, 2/2, 3/7, 4/3, 5/6, 6/8, 7/5, 8/1];
S = [1/5, 2/2, 3/4, 4/7, 5/3, 6/8, 7/6, 8/1];
...
```

   (1) (20%) Write the procedure **`gen_template(N1, N2, List)`** to produce the list:
   `List = [N1/Y1, N1+1/Y2, N1+2/Y3, ..., N2-1/Y’, N2/Y”]`. For example:

```
?- gen_template(1, 8, L).
L = [1/_G1, 2/_G2, 3/_G3, 4/_G4, 5/_G5, 6/_G6, 7/_G7, 8/_G8].
```

   (2) (10%) Modify the **`solution(Solution)`** to **`solution(N, Solution)`**, so that you can use with the **`gen_template`** to get the solution to the N-queens problem. For example:

```
?- gen_template(1, 8, S), solution(8, S).
S = [1/4, 2/2, 3/7, 4/3, 5/6, 6/8, 7/5, 8/1] ;
S = [1/5, 2/2, 3/4, 4/7, 5/3, 6/8, 7/6, 8/1] .
```

Program 2

```
% Program 2
solution(Queens) :-
    permutation([1, 2, 3, 4, 5, 6, 7, 8], Queens),
    safe(Queens).

safe([]).
safe([Queen|Others]) :-
    safe(Others),
    noattack(Queen, Others, 1).

noattack(_,[],_).
noattack(Y, [Y1|Ylist], Xdist) :-
    Y-Y1=\=Xdist, Y1-Y=\=Xdist,
    Dist1 is Xdist+1,
    noattack(Y, Ylist, Dist1).
```

```
?- solution(S).
S = [1, 5, 8, 6, 3, 7, 2, 4];
S = [1, 6, 8, 3, 7, 4, 2, 5];
...
```

(20%) Modify the **`solution(Solution)`** to **`solution(N, Solution)`**, so you can get the solution to the N-queens problem. For example:

```
?- solution(5, S).
S = [1, 3, 5, 2, 4] ;
S = [1, 4, 2, 5, 3] ;
S = [2, 4, 1, 3, 5] ;
S = [2, 5, 3, 1, 4] ;
S = [3, 1, 4, 2, 5] ;
S = [3, 5, 2, 4, 1] .
```

3. (20%) The following program can get the Fibonacci number:
   fib(1, 1), fib(2, 1), fib(3, 2), fib(4, 3), fib(5, 5), fib(6, 8), fib(7, 13), fib(8, 21), …

```
fib(1, 1).
fib(2, 1).
fib(N, Value) :-
     N1 is N - 1,
     N2 is N - 2,
     fib(N1, Value1),
     fib(N2, Value2),
     Value is Value1 + Value2.
```

```
?- fib(18, Value).
Value = 2584 ;
ERROR: Out of local stack
```

For not showing the error message, you should modify the program to produce exactly one result by using cuts.

Note:

- The programs must be runnable in SWI-Prolog.
- You must package your programs in a single zip file named XXX_hw3.zip, where XXX is your student identity number.
  Submit the whole file to open cyber classroom. The first-time Login ID and password are

student number.

- If the answer is not a program, you must answer the question in a single text file (txt, doc, etc.).
- A penalty will be applied if predicate name, program name or zip file name is not defined as above.