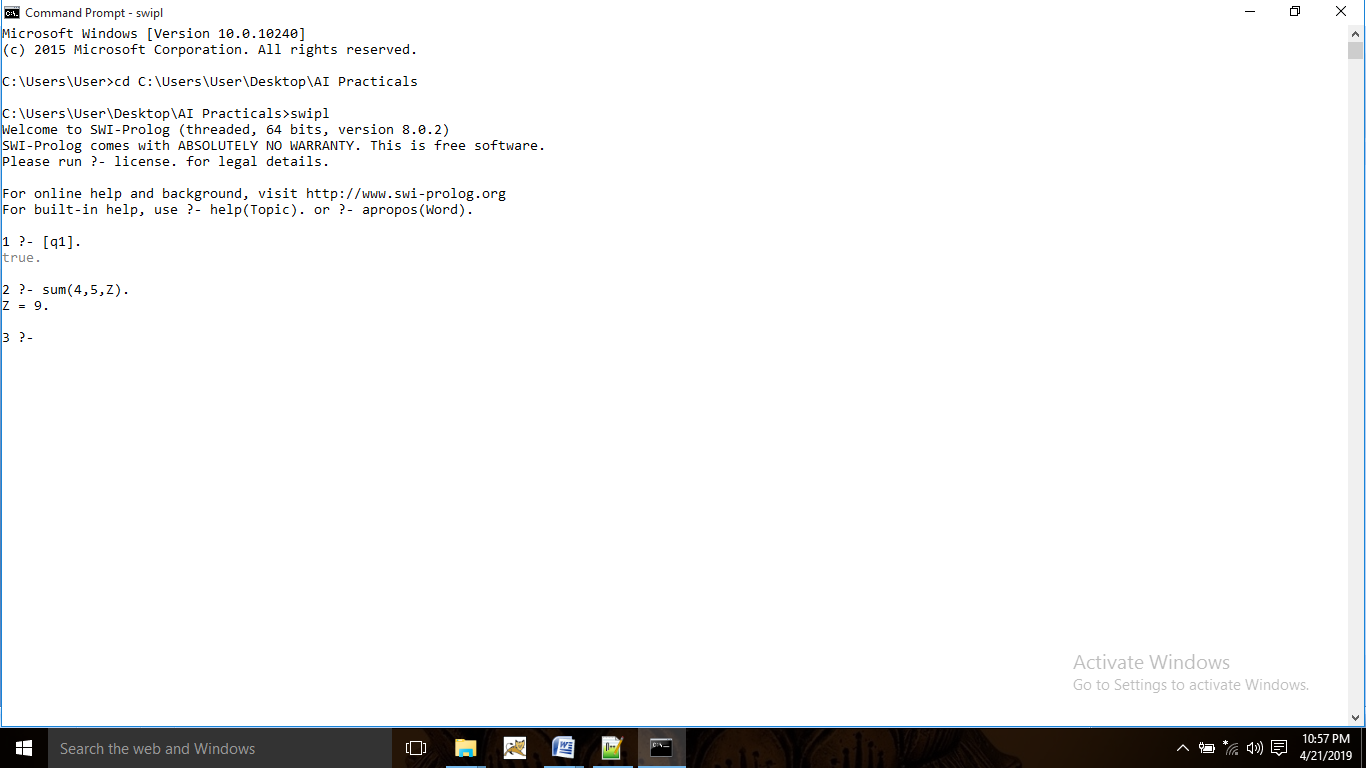
Artificial Intelligence

Practical File

**Q1. Sum:**

sum(X,Y,Z):- Z is X+Y.

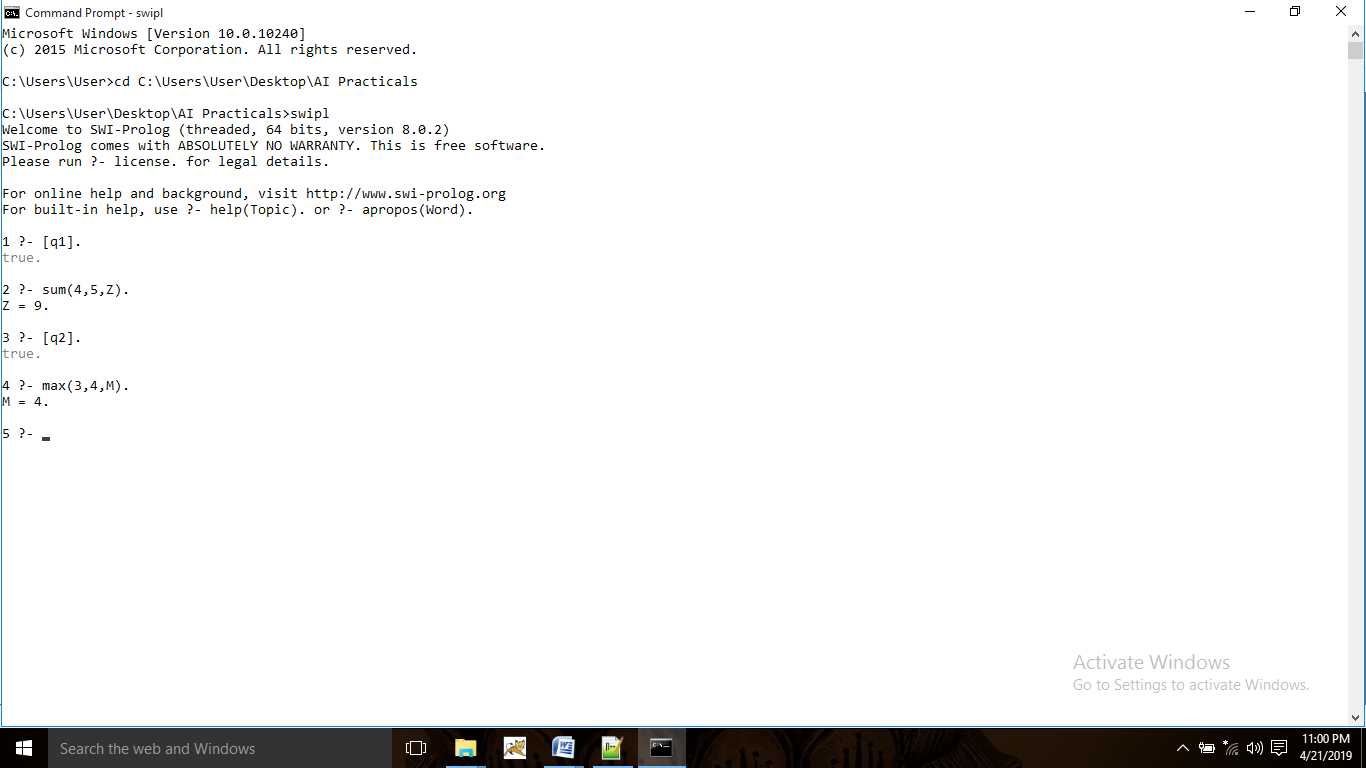
**Output:**



**Q2. Max Number:**

max(X,Y,M):- (X>Y, M is X)|(Y>X, M is Y).

**Output:**

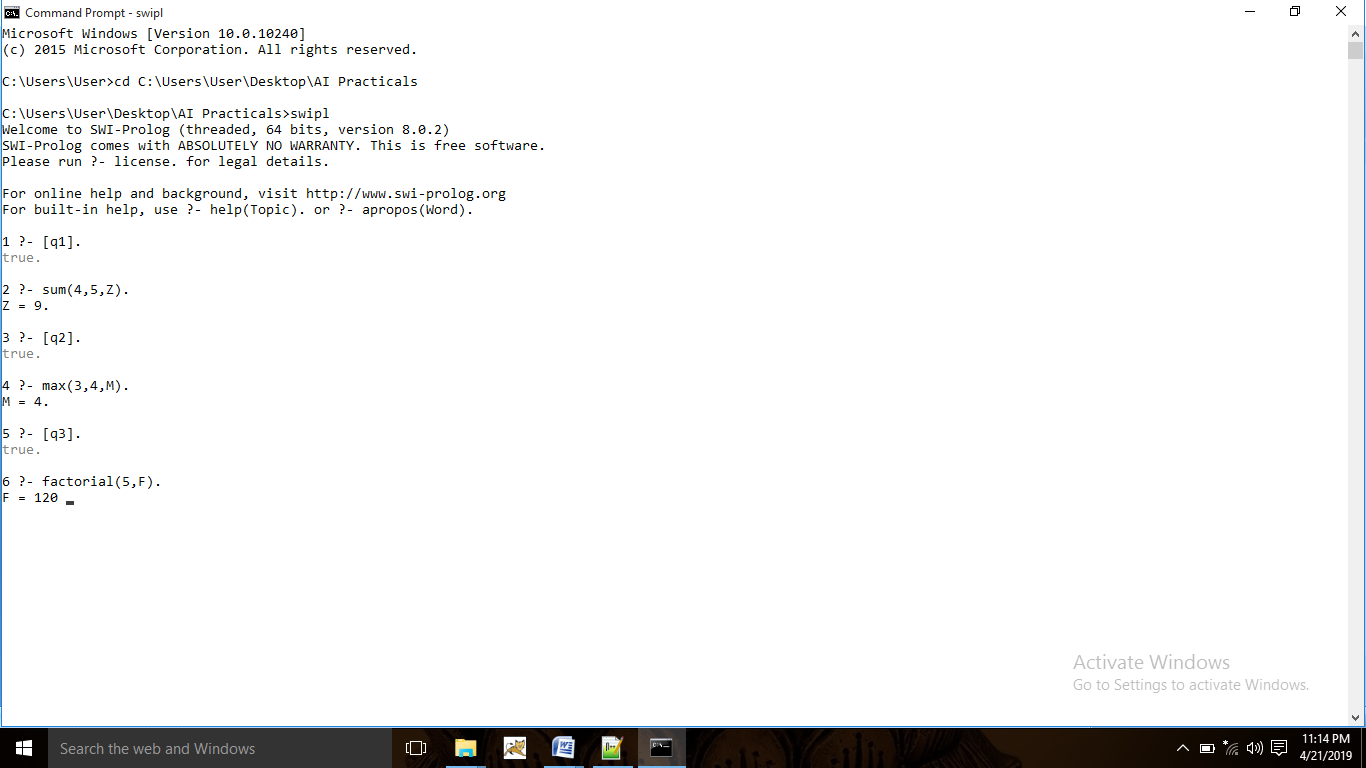


**Q3. Factorial:**

factorial(0,F):- F is 1.

factorial(N,F):- N>0, N1 is N-1, factorial(N1,F1), F is N\*F1.

**Output:**



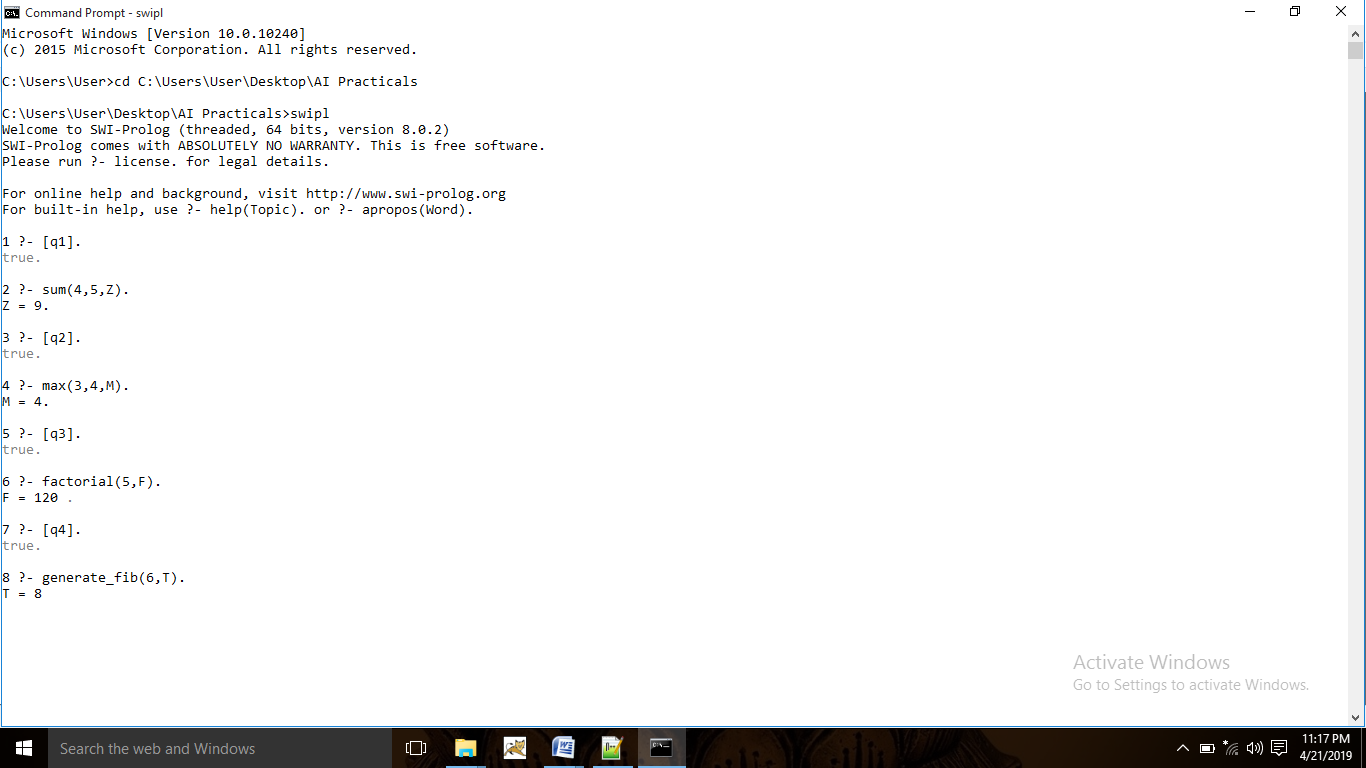
**Q4. Fibonacci Series:**

generate\_fib(0,T):- T is 0.

generate\_fib(1,T):- T is 1.

generate\_fib(N,T) :- N>1, N1 is N-1, N2 is N-2, generate\_fib(N1,T1), generate\_fib(N2,T2), T is T1+T2.

**Output:**



**Q5. GCD:**

gcd(0,B,R):- R is B.

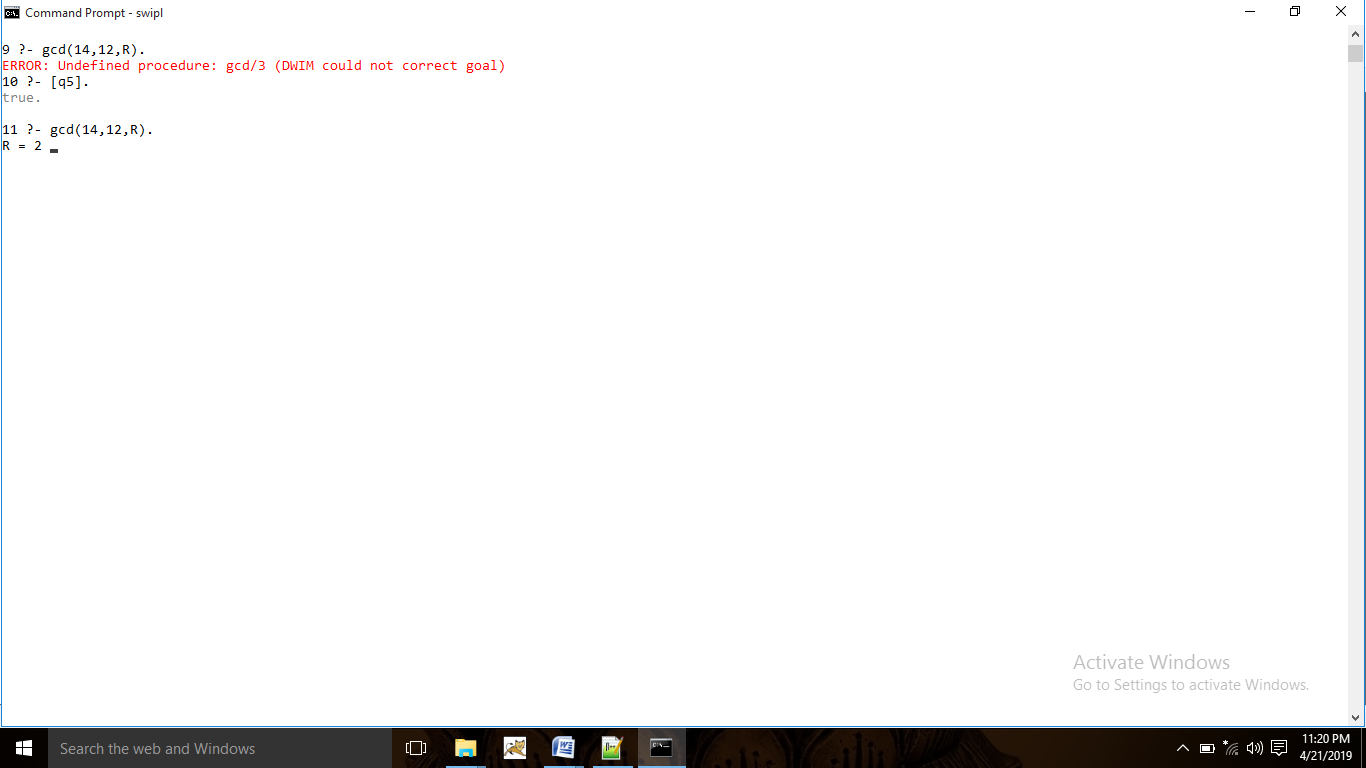
gcd(A,0,R):- R is A.

gcd(A,B,R):- A=B,R is A.

gcd(A,B,R):- A>B,A1 is A-B,gcd(A1,B,R).

gcd(A,B,R):- B>A,B1 is B-A,gcd(A,B1,R).

**Output:**

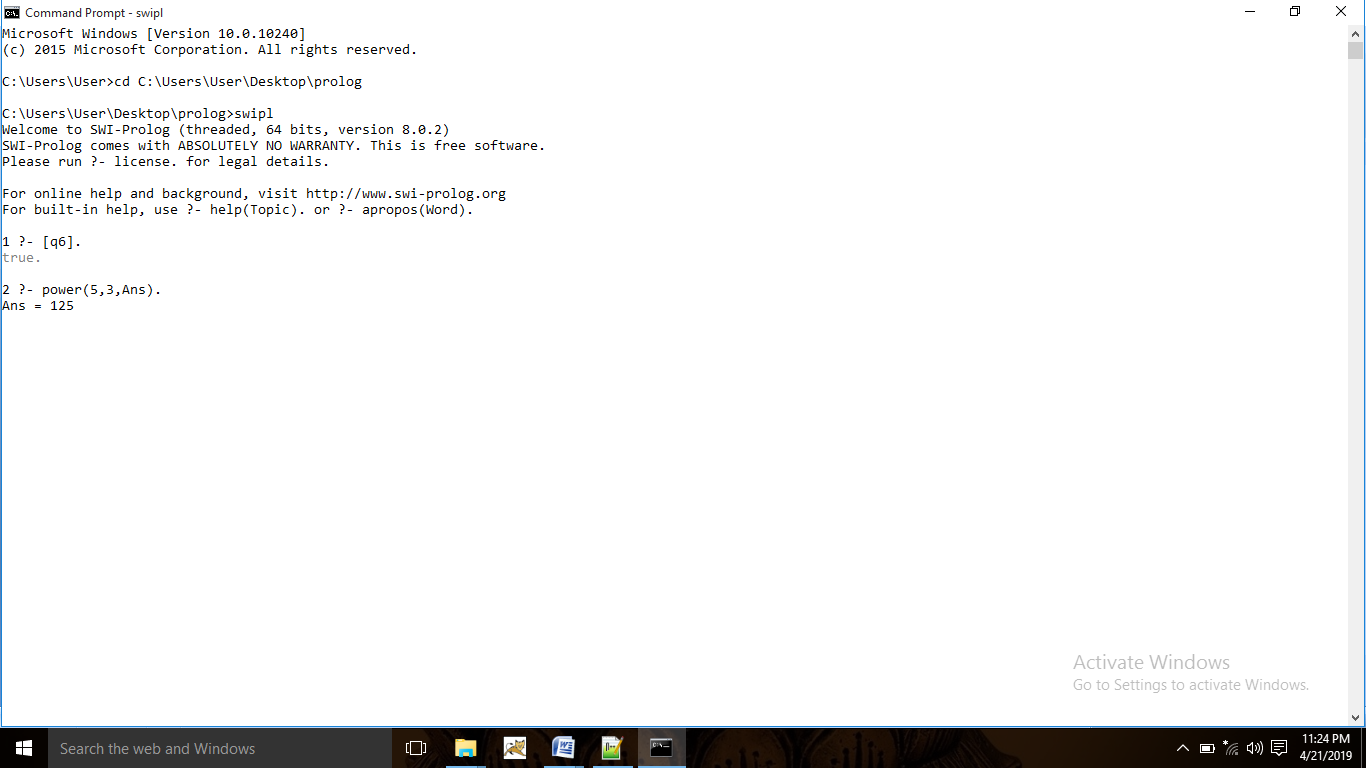


**Q6. Power:**

power(Num,0,Ans):- Ans is 1.

power(Num,Pow,Ans):-Num>0,Pow>0,Pow1 is Pow-1,power(Num,Pow1,Ans1), Ans is Ans1\*Num.

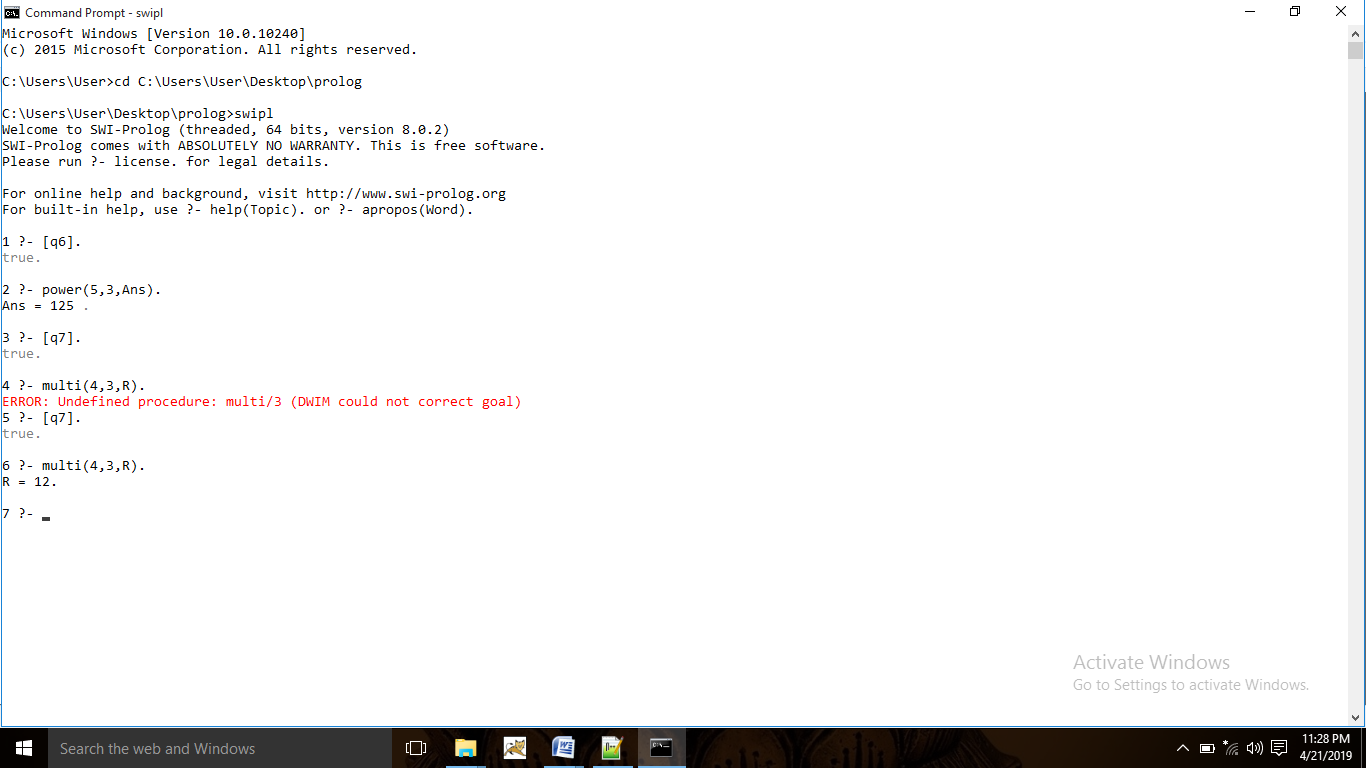
**Output:**



**Q7. Multiply:**

multi(N1,N2,R):- R is N1\*N2.

**Output:**



**Q8. Tower of Hanoi:**

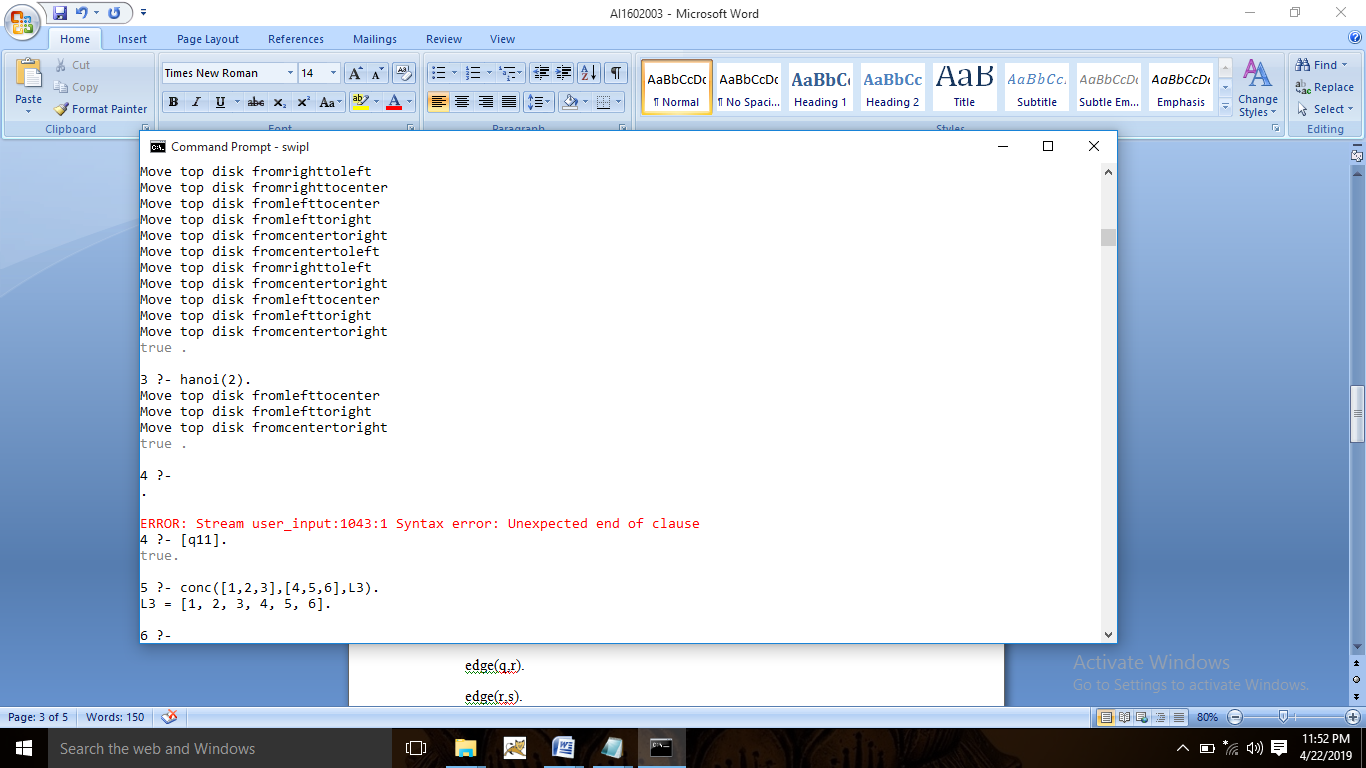
move(1,Left,Right,\_):- write('Move top disk from'), write(Left), write('to'), write(Right),nl.

move(N,Left,Right,Center):- N>1, N1 is N-1,

move(N1,Left,Center,Right), move(1,Left,Right,Center), move(N1,Center,Right,Left).

hanoi(N):- move(N,left,right,center).

**Output:**



**9. Cyclic Directed Graph:**

edge(p,q).

edge(q,r).

edge(r,s).

edge(s,t).

path(X,Y) :- edge(X,Y).

**Output:**

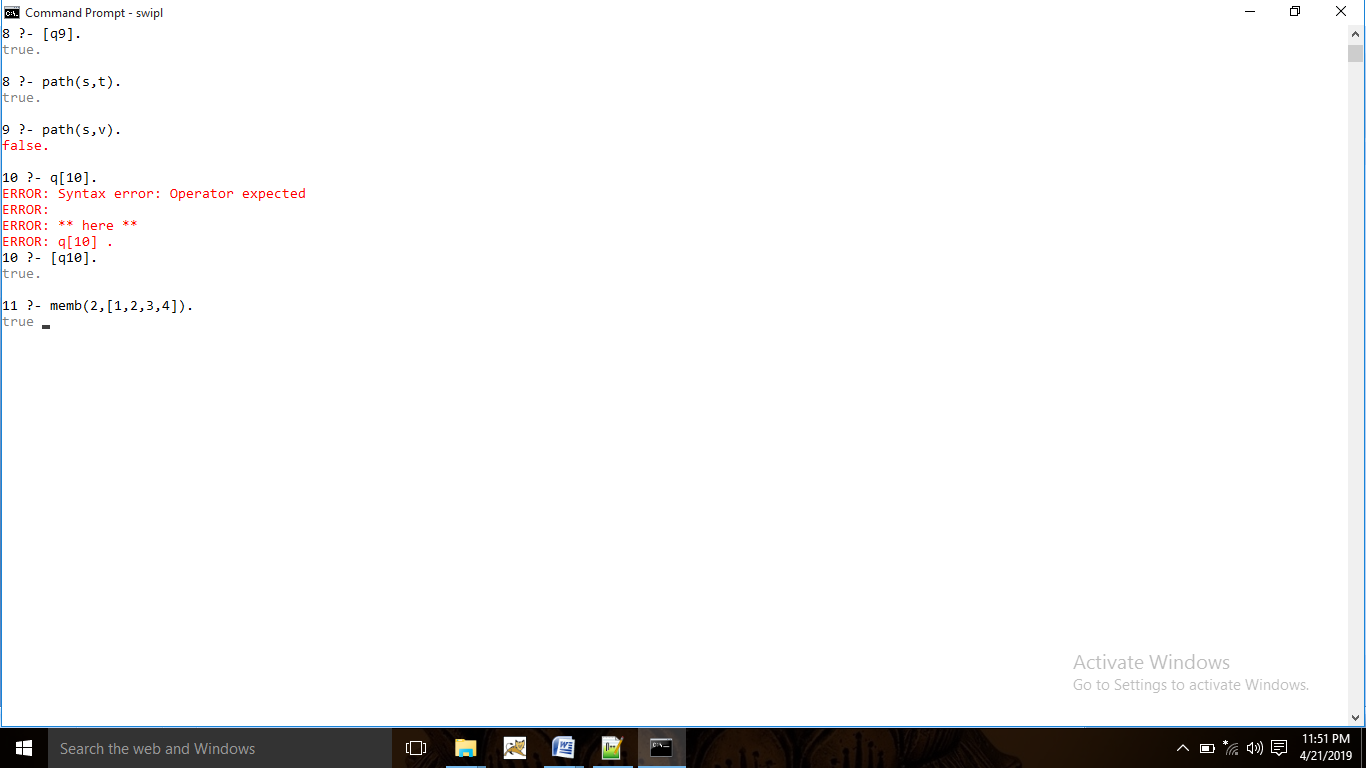


**Q10. Member:**

memb(X,[X|\_]).

memb(X,[\_|Y]):- memb(X,Y).

**Output:**

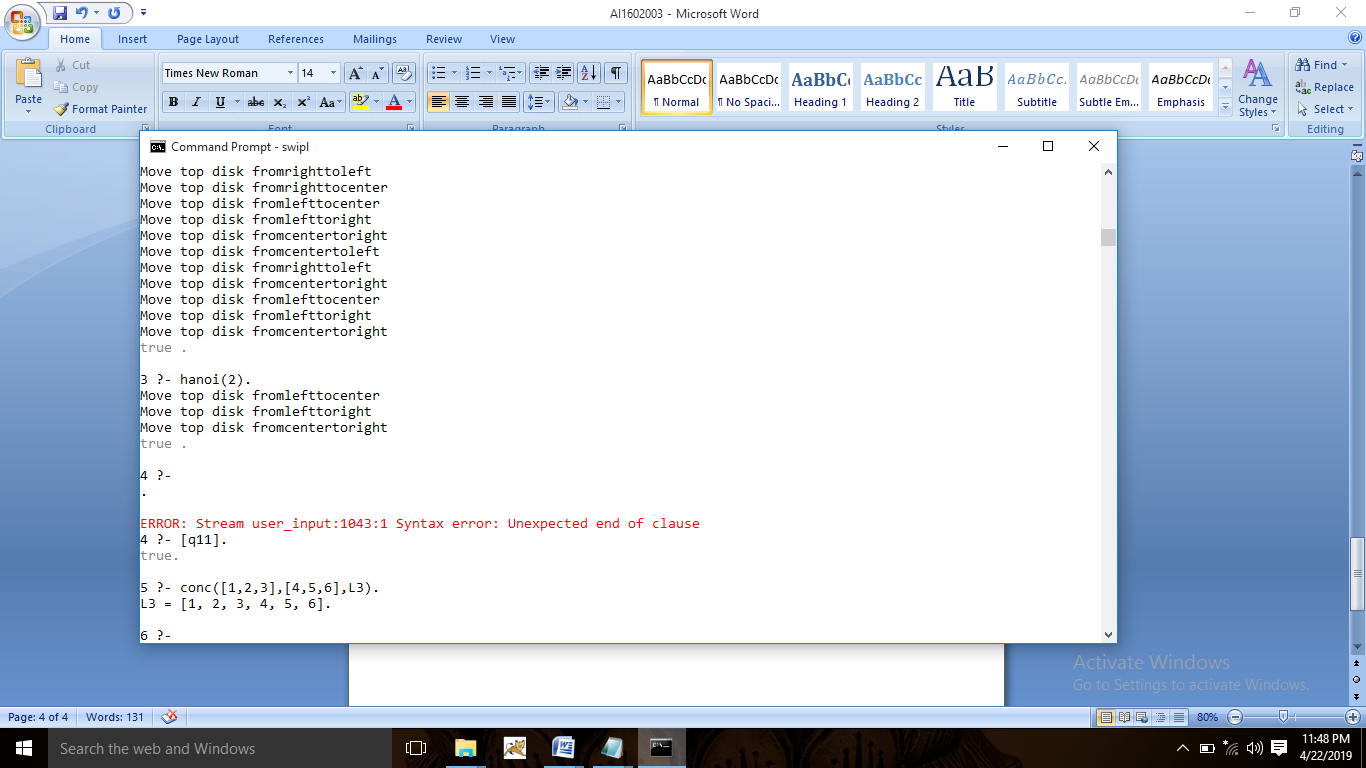


**Q11. Concatenate:**

conc([],L,L).

conc([X|L1],L2,[X|L3]):-conc(L1,L2,L3).

**Output:**



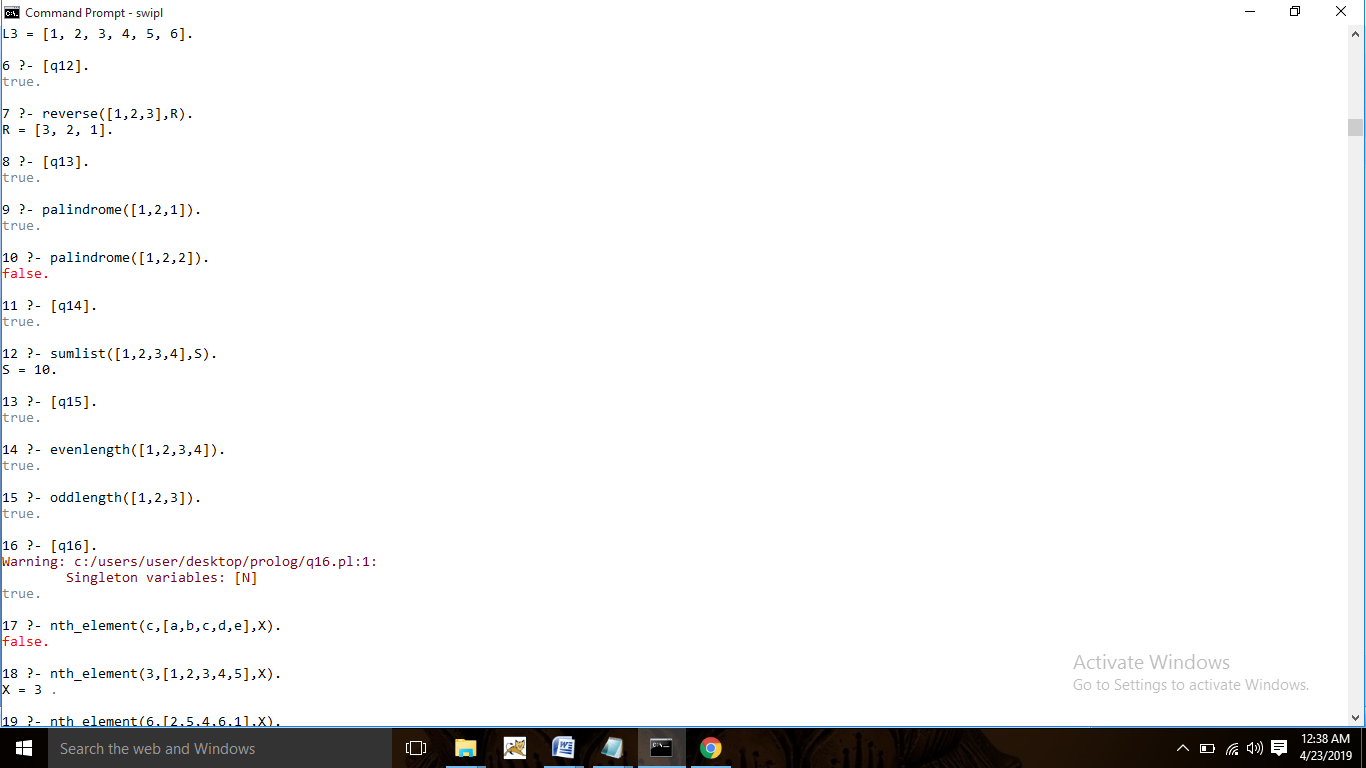
**Q12. Reverse:**

reverse(L,R):- reverseacc(L,[],R).

reverseacc([],R,R).

reverseacc([X|Y],A,R):- A1=[X|A], reverseacc(Y,A1,R).

**Output:**



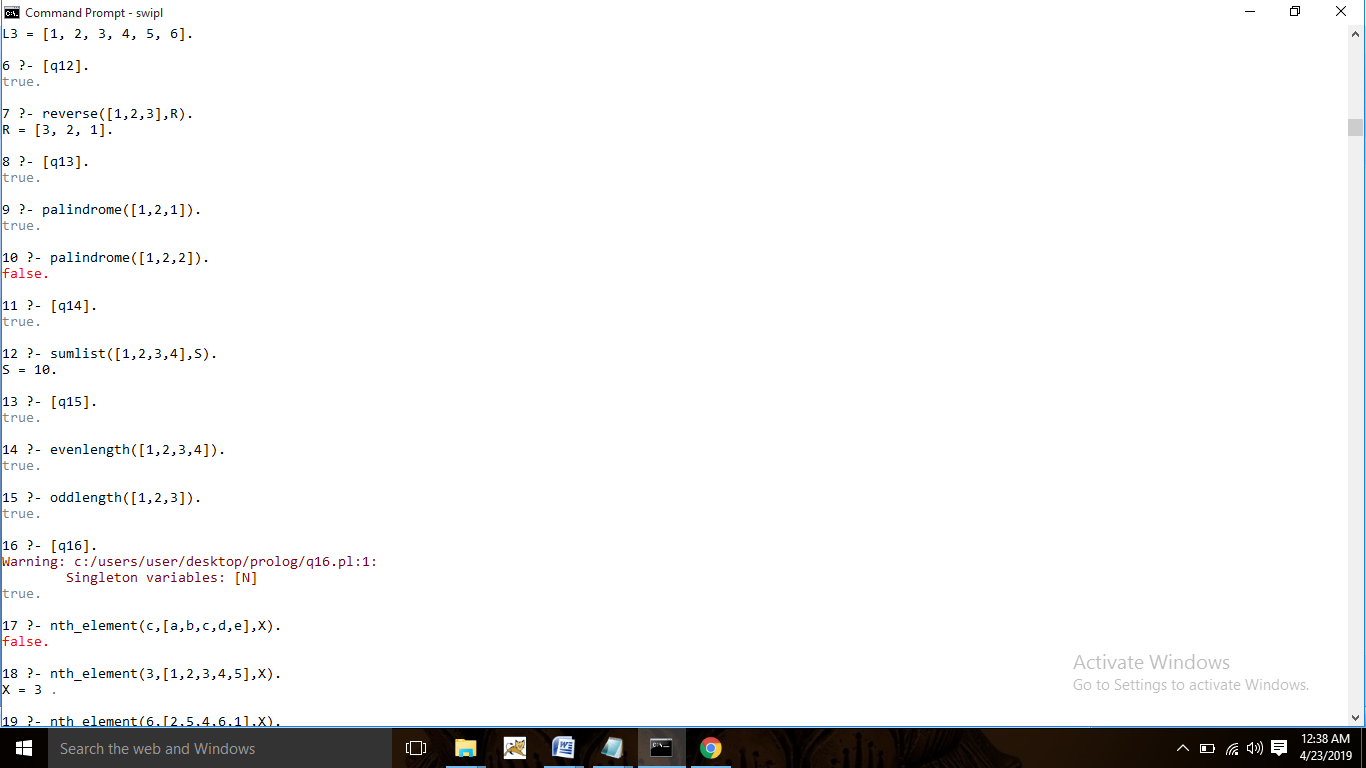
**Q13. Palindrome:**

palindrome(L):- reverse(L,[],R), L=R.

reverse([],R,R).

reverse([H|T],A,R):- A1=[H|A], reverse(T,A1,R).

**Output:**



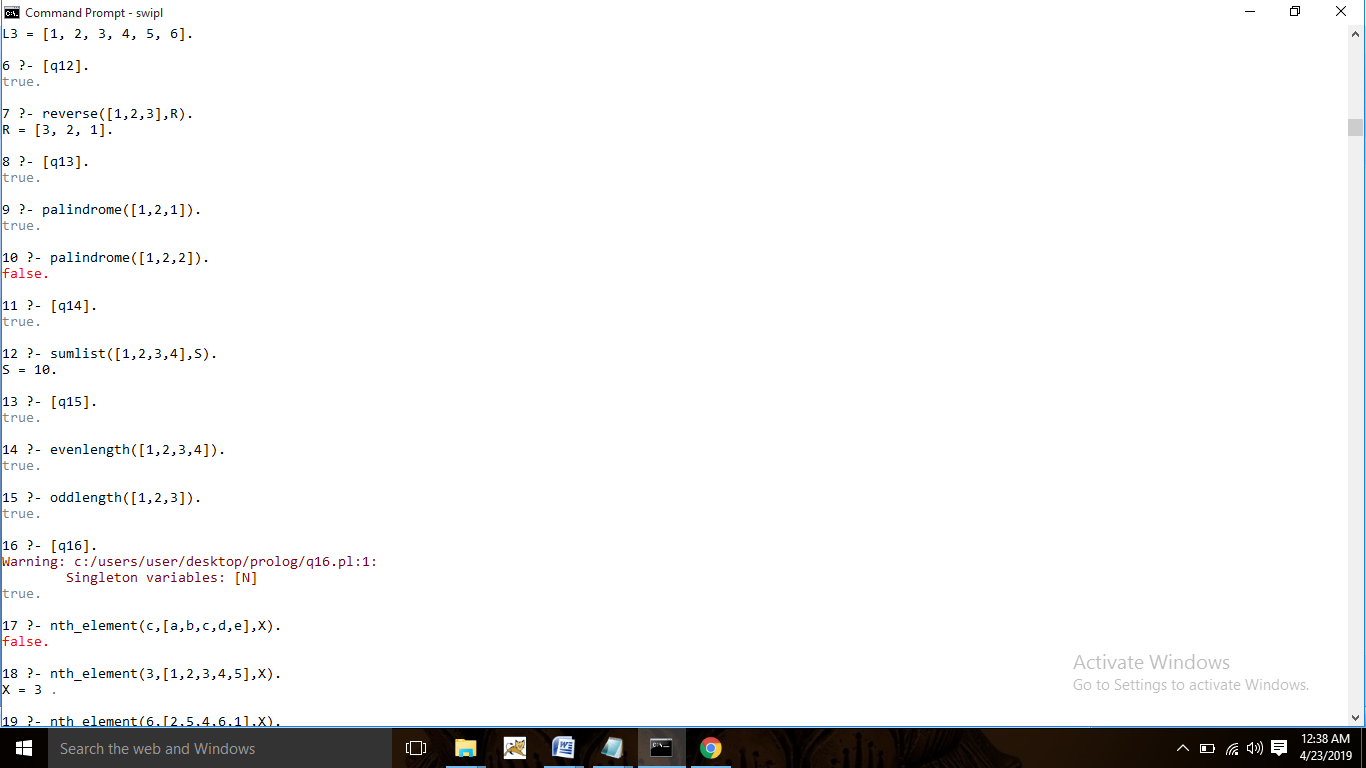
**Q14. Sumlist:**

sumlist(L,S):- sumlistacc(L,0,S).

sumlistacc([],A,A).

sumlistacc([H|T],A,S):- A1 is H+A, sumlistacc(T,A1,S).

**Output:**

****

**Q15. Evenlength & Oddlength:**

evenlength(List):-evenlengthacc(List,0).

evenlengthacc([],A):- A1 is A mod 2, A1=0.

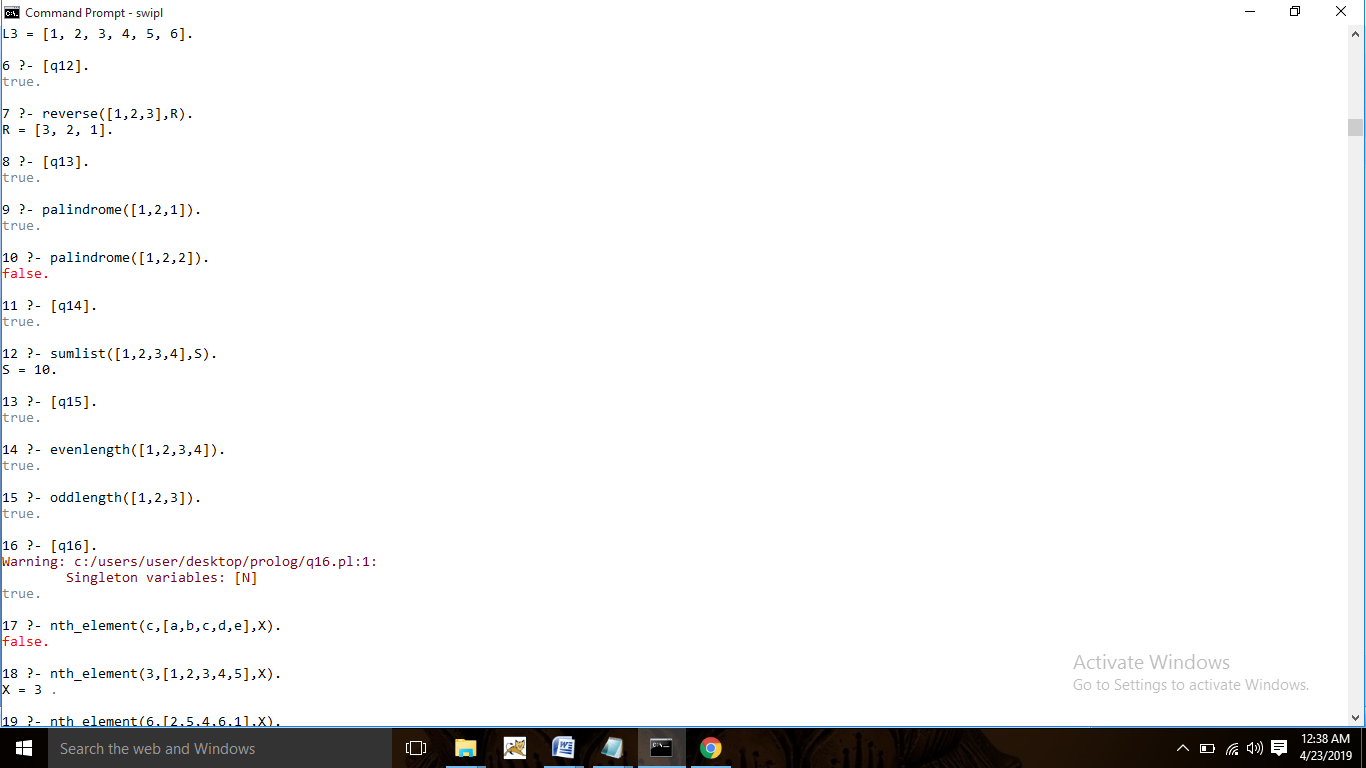
evenlengthacc([\_|T],A):- A1 is A+1, evenlengthacc(T,A1).

oddlength(List):-oddlengthacc(List,0).

oddlengthacc([],A):- A1 is A mod 2, A1=1.

oddlengthacc([\_|T],A):- A1 is A+1, oddlengthacc(T,A1).

**Output:**

****

**Q16. Show nth element:**

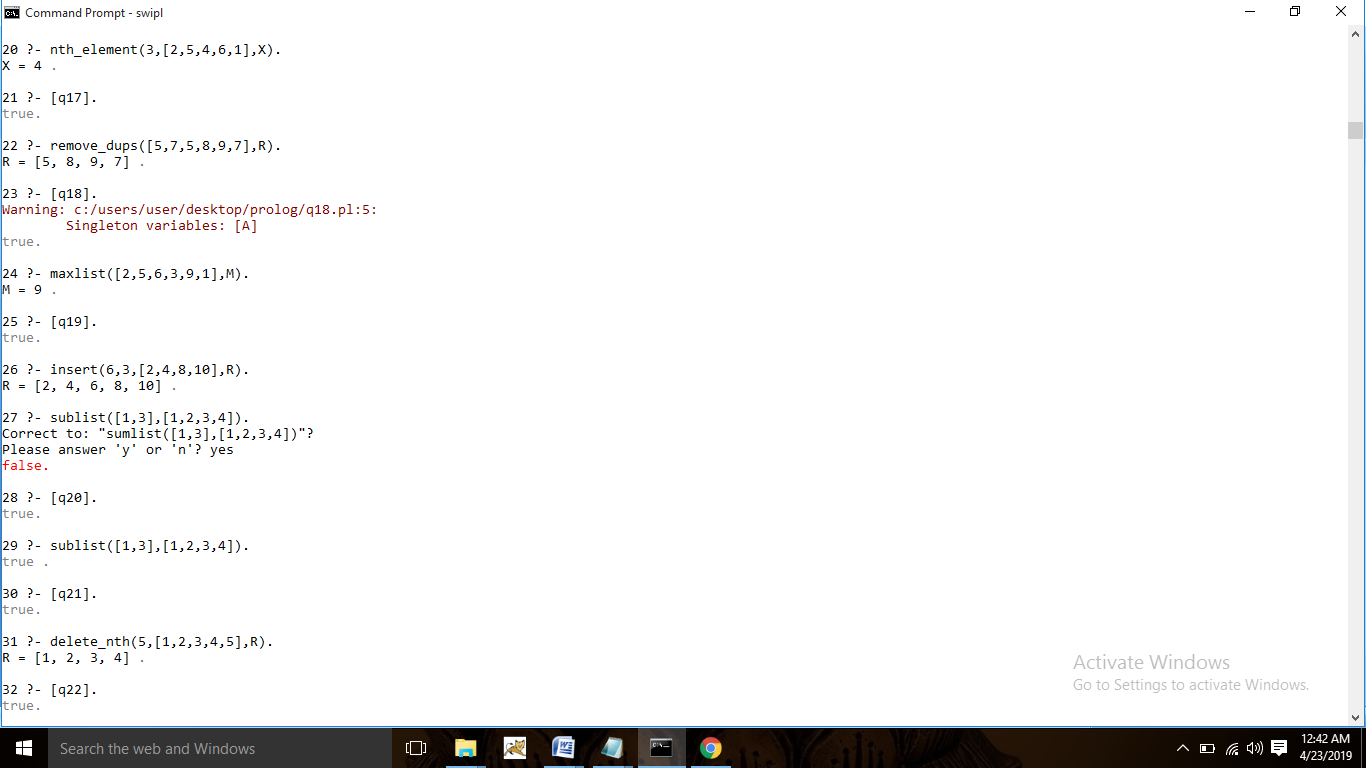
nth\_element(N,[],emptyList).

nth\_element(N,L,X):- nth\_elementacc(N,0,L,X).

nth\_elementacc(N,A,[H|\_],X):- A1 is A+1, A1=N, X=H.

nth\_elementacc(N,A,[\_|T],X):- A1 is A+1, nth\_elementacc(N,A1,T,X).

**Output:**

****

**Q17. Remove Duplicates:**

is\_memb(X, [X|\_]).

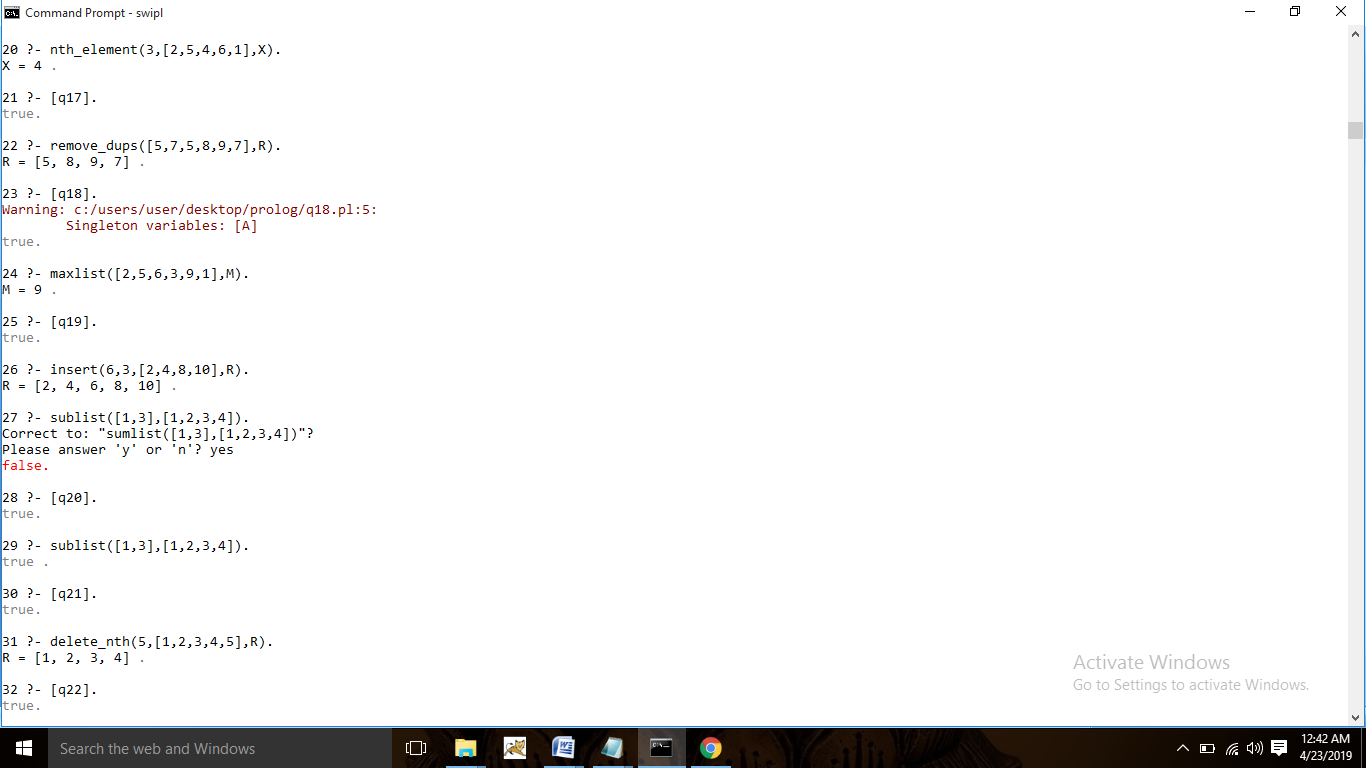
is\_memb(X, [\_|T]):- is\_memb(X,T).

remove\_dups([], []).

remove\_dups([H|T], R):-is\_memb(H, T), remove\_dups(T, R).

remove\_dups([H|T], [H|R]):- not(is\_memb(H, T)), remove\_dups(T, R).

**Output:**

****

**Q18. Maximum in list:**

maxlist([],noElement).

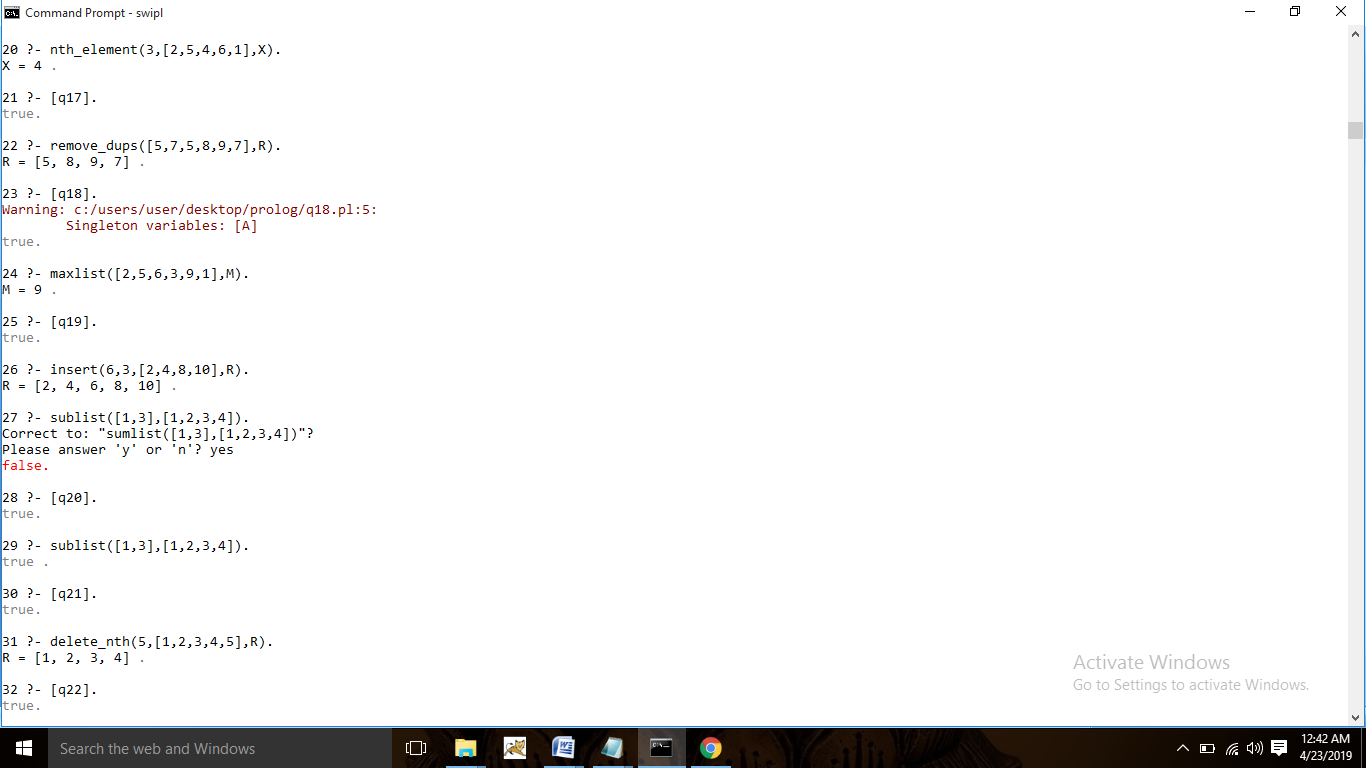
maxlist([H|T],M):- maxlistacc(T,H,M).

maxlistacc([],A,A).

maxlistacc([H1|T1],A,M):- A>=H1, maxlistacc(T1,A,M).

maxlistacc([H1|T1],A,M):-maxlistacc(T1,H1,M).

**Output:**

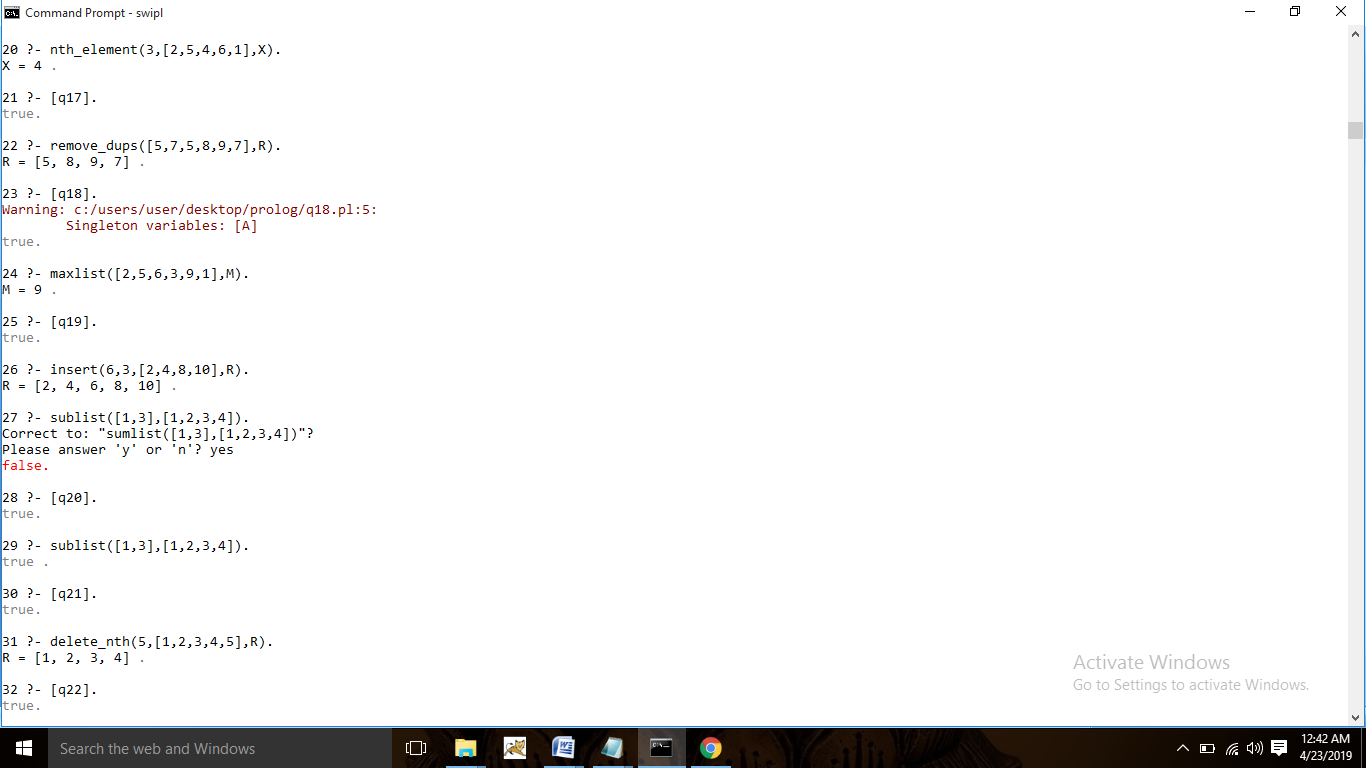
****

**Q19. Insert in list:**

insert\_nth(I,1,L,[I|L]).

insert\_nth(I,N,[H|T],[H|R]):- N1 is N-1,insert\_nth(I,N1,T,R).

**Output:**

****

**Q20. Sublist:**

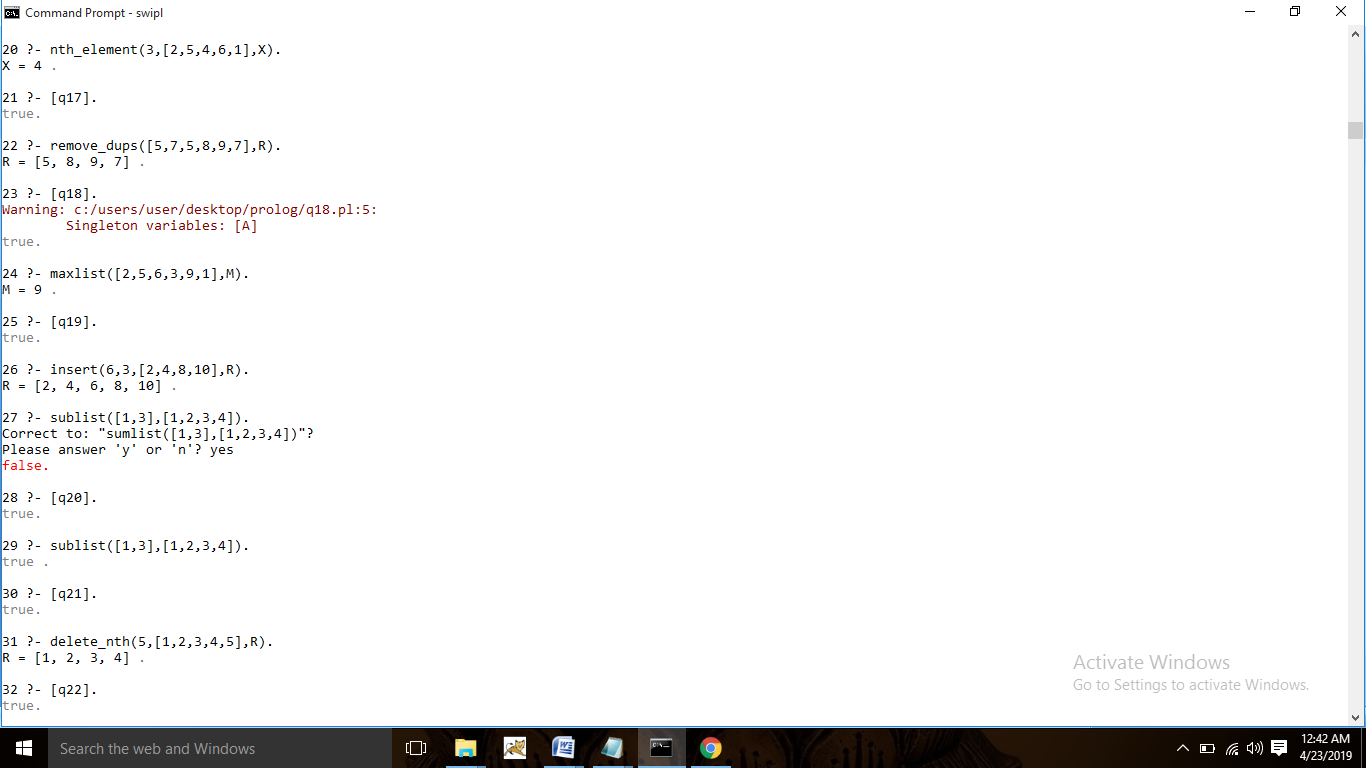
sublist([],[\_|\_]).

sublist([],[]).

sublist([H|T],[H|T1]):- sublist(T,T1).

sublist(L,[\_|T1]):- sublist(L,T1).

**Output:**

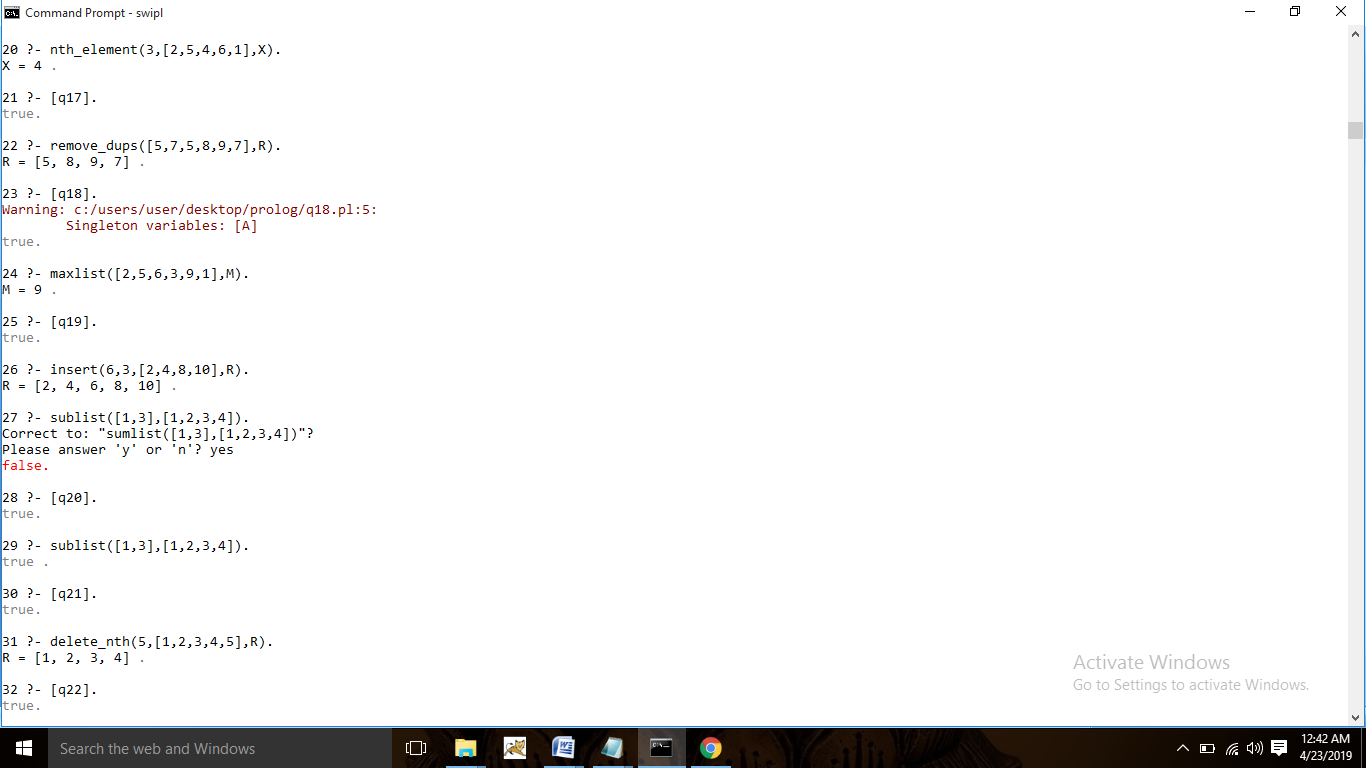
****

**Q21. Delete from list:**

delete\_nth(1,[\_|T],T).

delete\_nth(N,[H|T],[H|R]):- N1 is N-1,delete\_nth(N1,T,R).

**Output:**

****

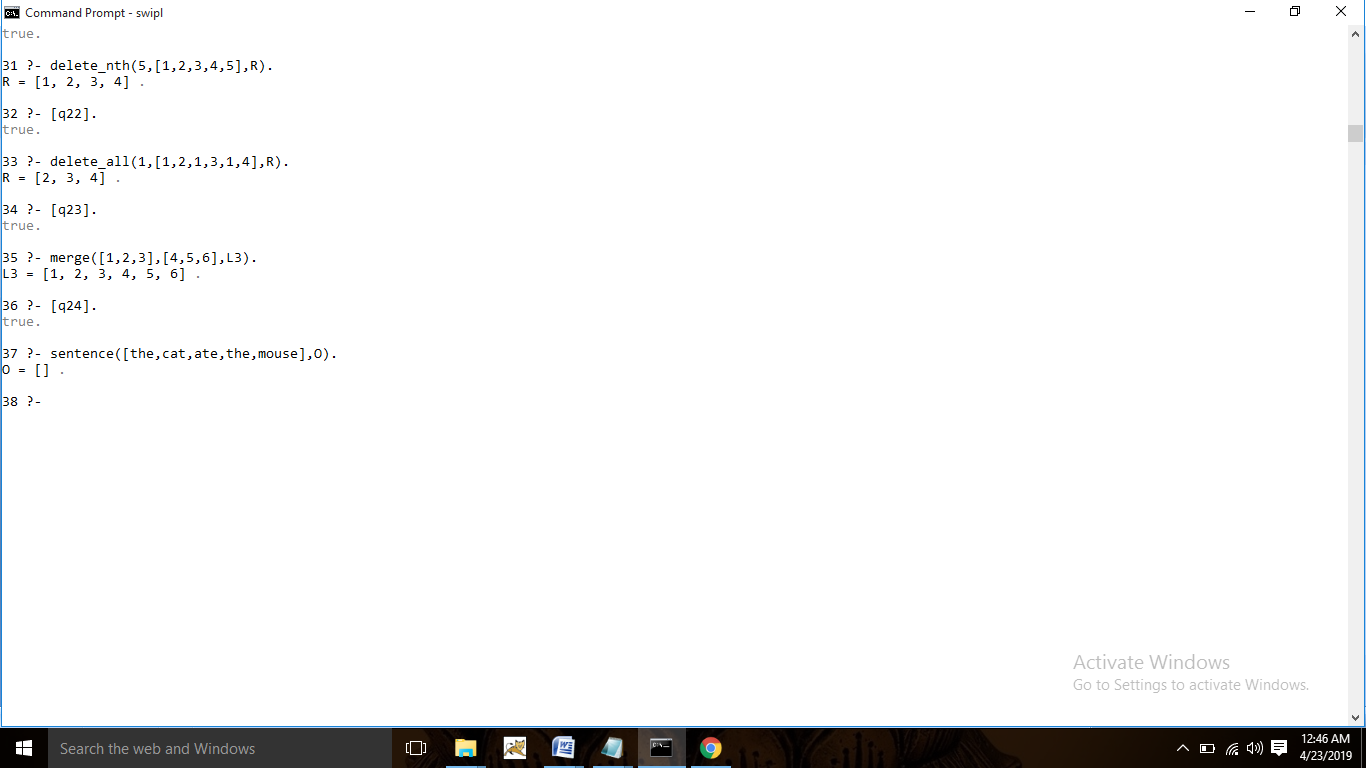
**Q22. Delete multiple occurrences:**

delete\_all(\_,[],[]).

delete\_all(X,[X|T],R):-delete\_all(X,T,R).

delete\_all(X,[H|T],[H|R]):-delete\_all(X,T,R).

**Output:**

****

**Q23. Merge:**

merge([],[],[]).

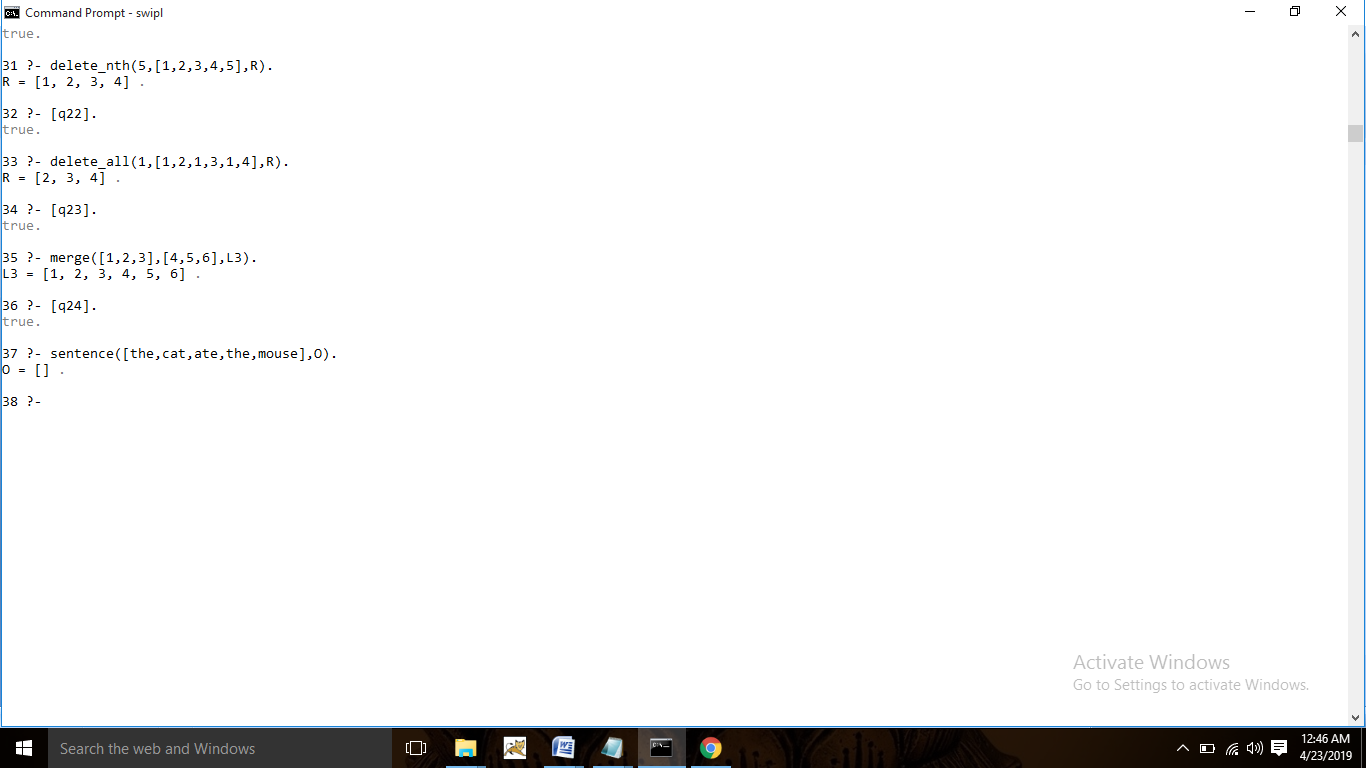
merge([],L,L).

merge(L,[],L).

merge([H1|T1],[H2|T2],[H3|T3]):- H1=<H2, H3=H1, merge(T1,[H2|T2],T3).

merge([H1|T1],[H2|T2],[H3|T3]):- H3=H2, merge([H1|T1],T2,T3).

**Output:**

****

**Q24. Grammar Rules:**

noun([dog|X],X).

noun([cat|X],X).

noun([mouse|X],X).

verb([ate|X],X).

verb([chases|X],X).

adjective([big|X],X).

adjective([brown|X],X).

adjective([lazy|X],X).

determiner([the|X],X):-!.

nounphrase(NP,X):- determiner(NP,R), noun(R,X).

nounphrase(NP,X):- determiner(NP,R), adjective(R,S2), noun(S2,X).

verbphrase(VP,X):- verb(VP,R), nounphrase(R,X).

sentence(I,O):- nounphrase(I,R), verbphrase(R,O).

**Output:**

