AI ASSIGNMENT – 4

1. **W.A.P.P to find factorial of a number.**

Source Code:

```
factorial(0, 1).
factorial(N, F) :-
      N > 0,
      Prev is N -1,
      factorial(Prev, R),
      F is R * N.
```

2. **W.A.P.P to print Fibonacci series.**

   The Fibonacci sequence f (1), f (2), f (3)…is: 1, 1, 2, 3, 5, 8, 13, 21, 34, 55....

   Example:

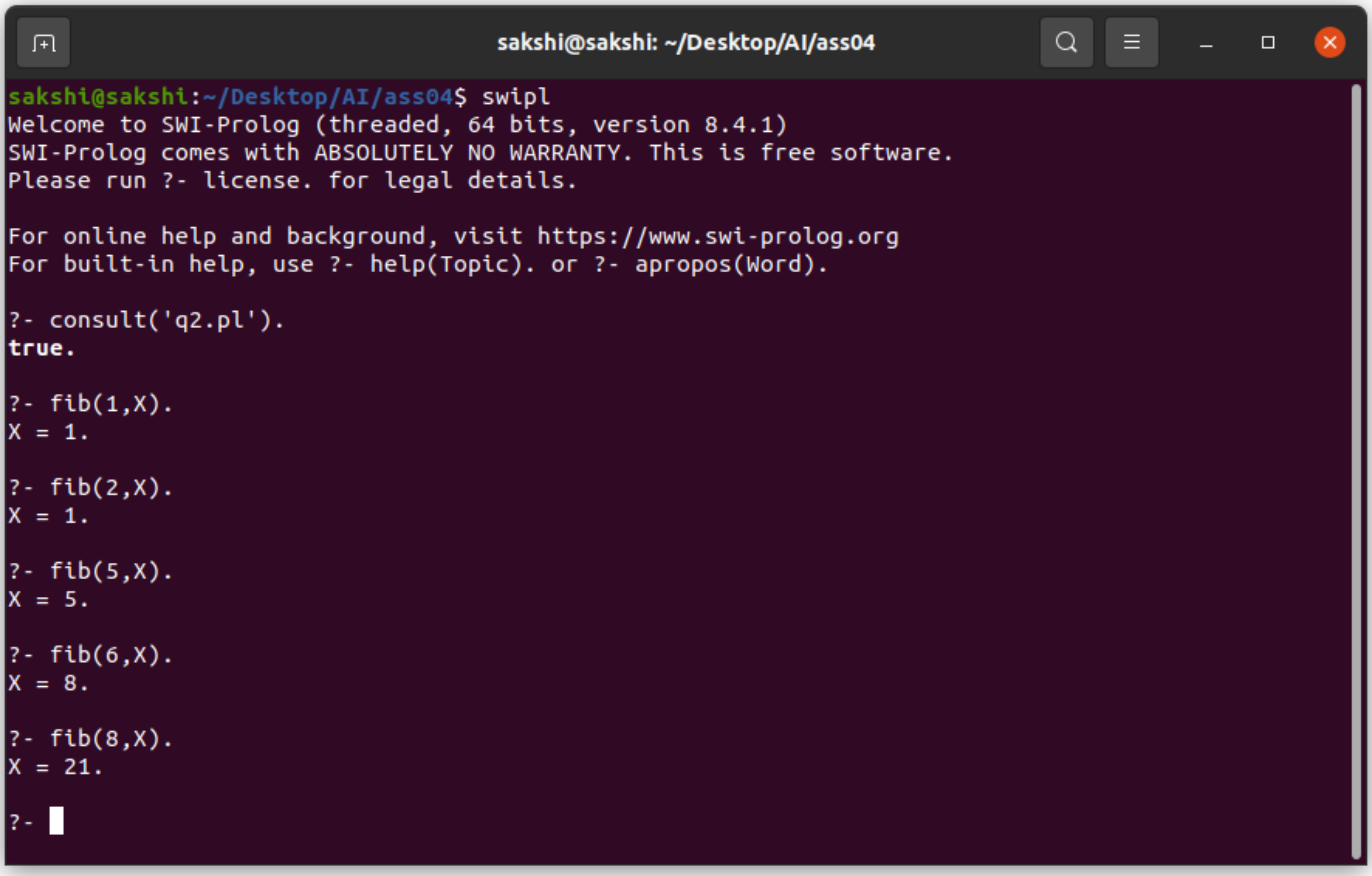
   ?- fib (6, R).

   R = 8

Source Code:

```
fib(1,1) :- !.
fib(0,0) :- !.
```

```
fib(N,Ans) :-
      N1 is N-1,
      N2 is N-2,
      fib(N1,Ans1),
      fib(N2,Ans2),
      Ans is Ans1+Ans2.
```

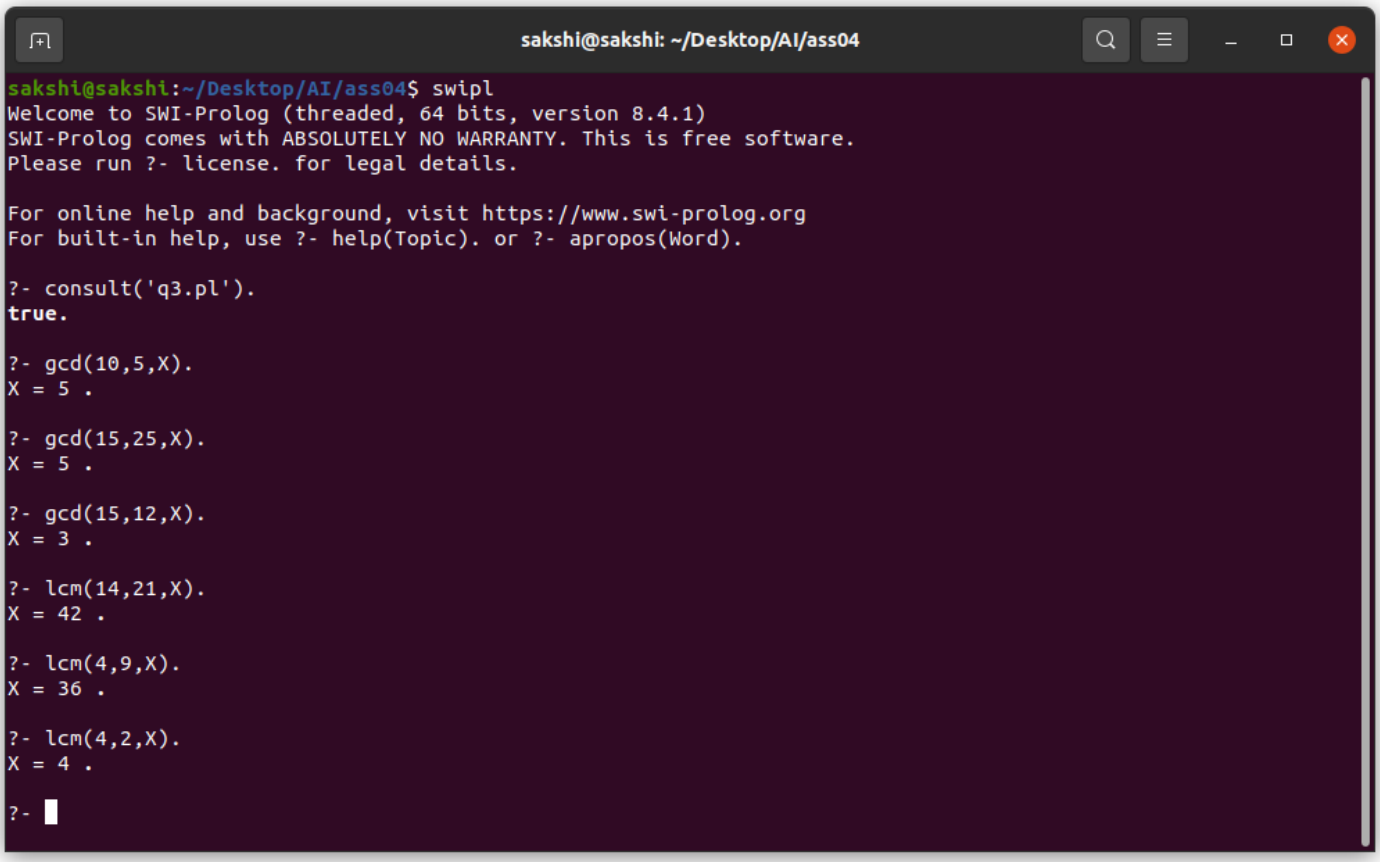


3. **W.A.P.P to finding the greatest common divider (GCD) and the least common multiple (LCM) of two integers.**

Source Code:

```
% Greatest Common Divisor
gcd(X, Y, G) :- X = Y, G = X.

gcd(X, Y, G) :-
  X < Y,
  Y1 is Y - X,
  gcd(X, Y1, G).

gcd(X, Y, G) :- X > Y, gcd(Y, X, G).
```

```
% Least Common Multiple
lcm(X,Y,LCM) :- gcd(X,Y,GCD), LCM is X*Y//GCD.
```



4. **W.A.P.P.**
   - **A. To find length of the list.**
   - **B. To find first and last element of the list.**
   - **C. To find the nth element of the list.**
   - **D. To increment each number in the list.**
   - **E. To reverse the list.**
   - **F. To verify if a list has an even number of elements.**
   - **G. To count vowels in the list**
   - **H. To remove duplicates from the list**

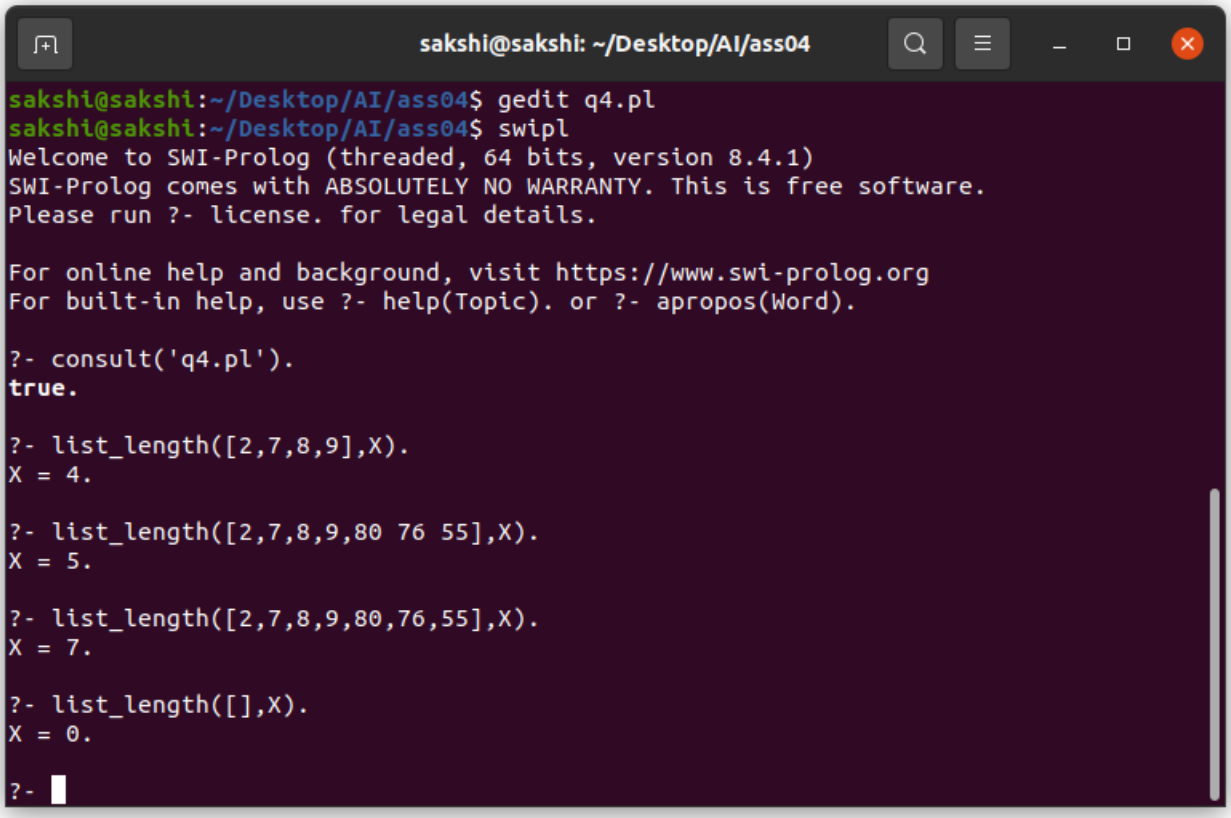
Source Code:

```
% Length of the list
list_length([],0).
list_length([_|Tail],N) :- list_length(Tail,N1),N is N1+1.
```

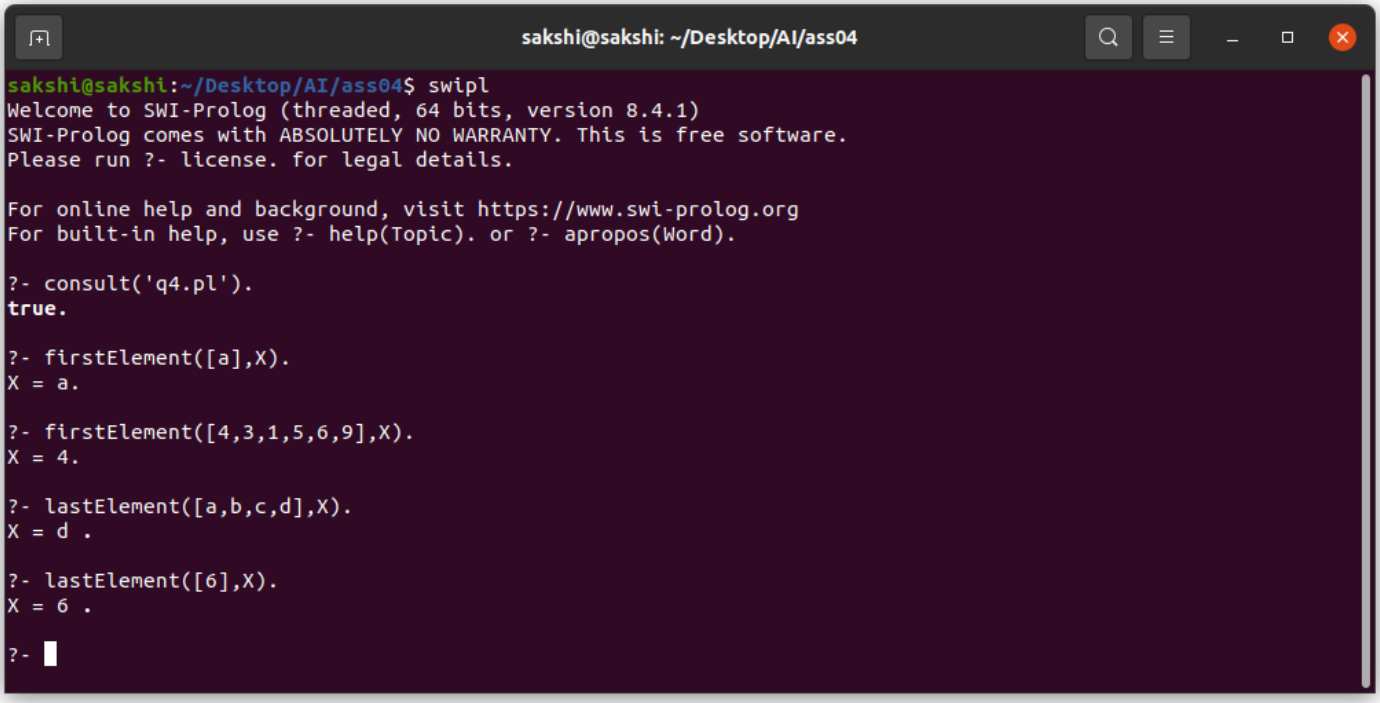
```
% First and Last Element
firstElement([A|_],A).

lastElement([A],A).
lastElement([_|B],Last) :- lastElement(B,Last).


% Nth element of the list
nth_element([A|_],1,A).
nth_element([_|B],N,X) :-
      N1 is N-1,
      nth_element(B,N1,X).


% Increment each number
increment([],[]).
increment([A|B],[A1|B1]) :- A1 is A+1,increment(B,B1).


% Reverse List
list_concat([],L,L).
list_concat([X1|L1],L2,[X1|L3]) :- list_concat(L1,L2,L3).

list_rev([],[]).
list_rev([Head|Tail],Reversed) :-
   list_rev(Tail, RevTail),
   list_concat(RevTail, [Head],Reversed).


% Even number of elements
even_elements([]).
even_elements([_|B]) :- odd_elements(B).

odd_elements([_]).
odd_elements([_|B]) :- even_elements(B).


% Count vowels
vowel(X):- member(X,[a,e,i,o,u]).

count_vowel([], 0).
count_vowel([X|T], N):-
        vowel(X),
        count_vowel(T,N1),
        N is N1+1.
count_vowel([_|T], N):-
        count_vowel(T,N).
```

```
% Remove duplicates
remove_duplicates([],[]).

remove_duplicates([H | T], List) :-
     member(H, T),
     remove_duplicates( T, List).

remove_duplicates([H | T], [H|T1]) :-
      \+member(H, T),
      remove_duplicates( T, T1).
```

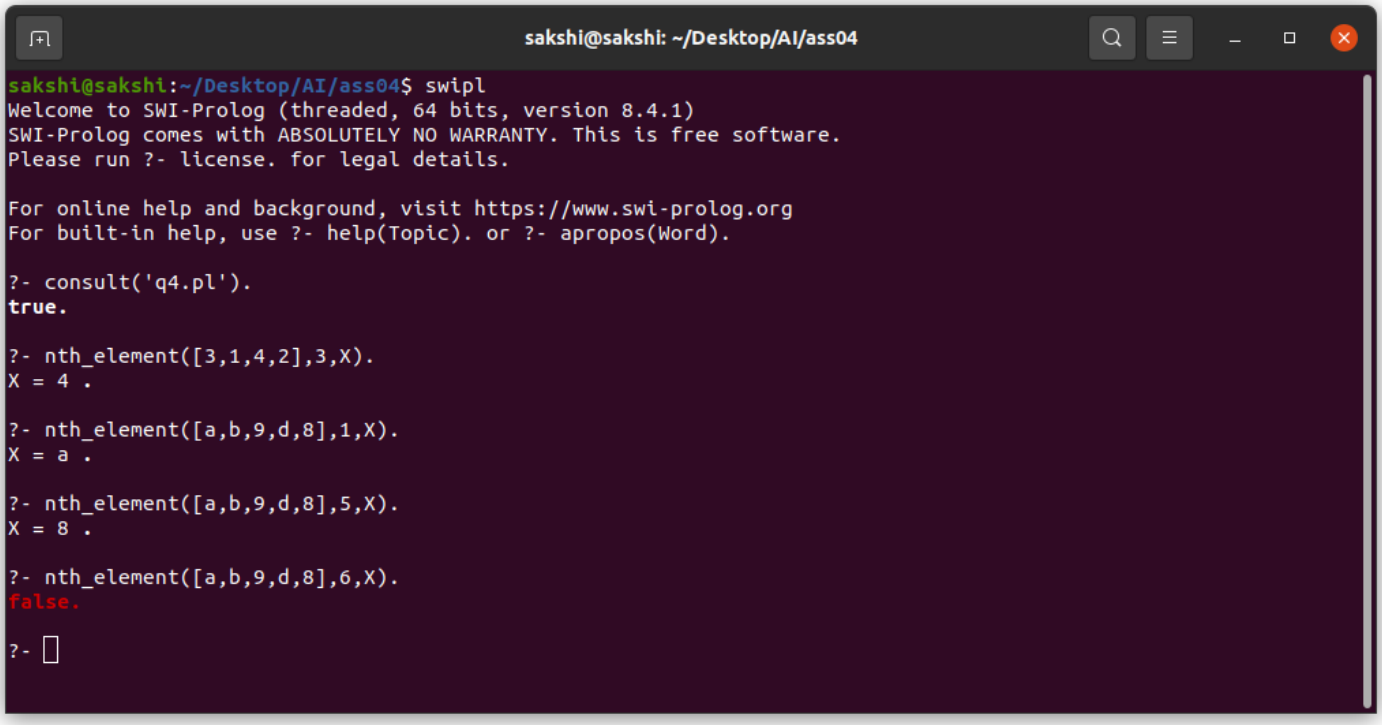
Length of the list:



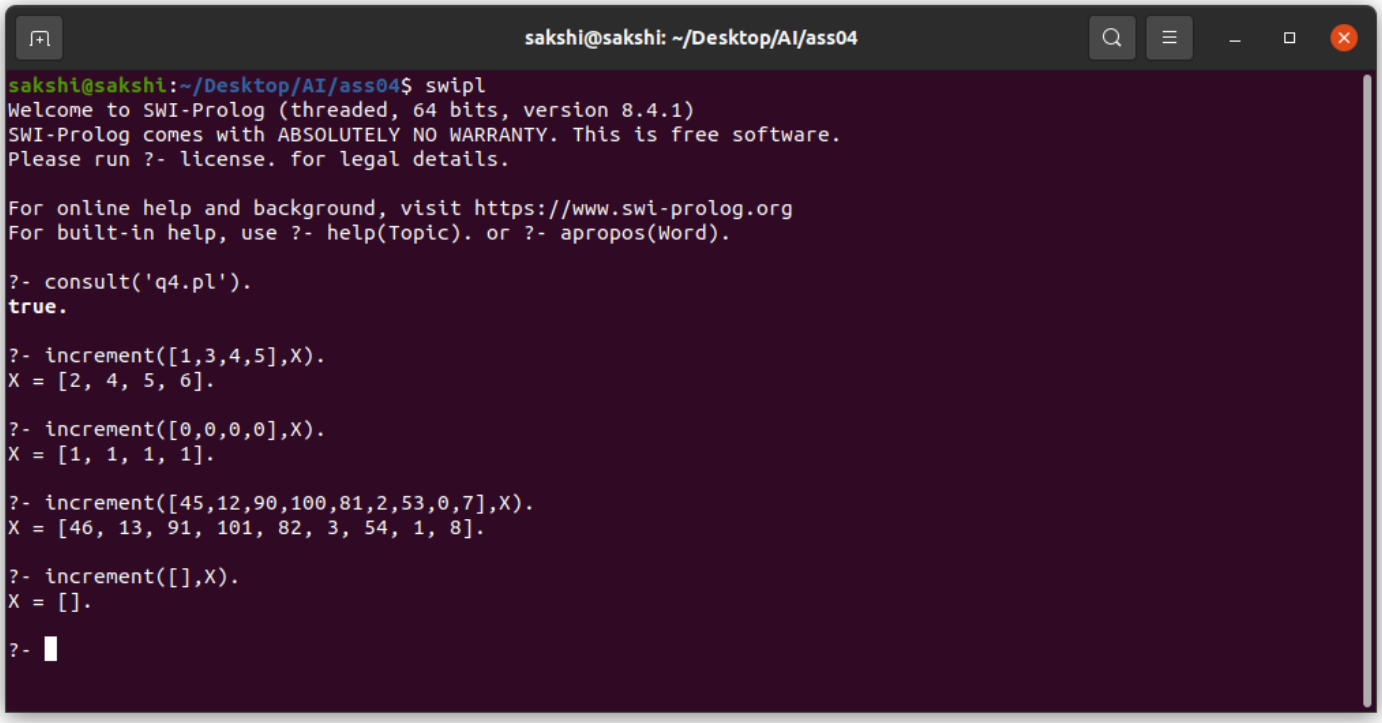
First and last element of the list:



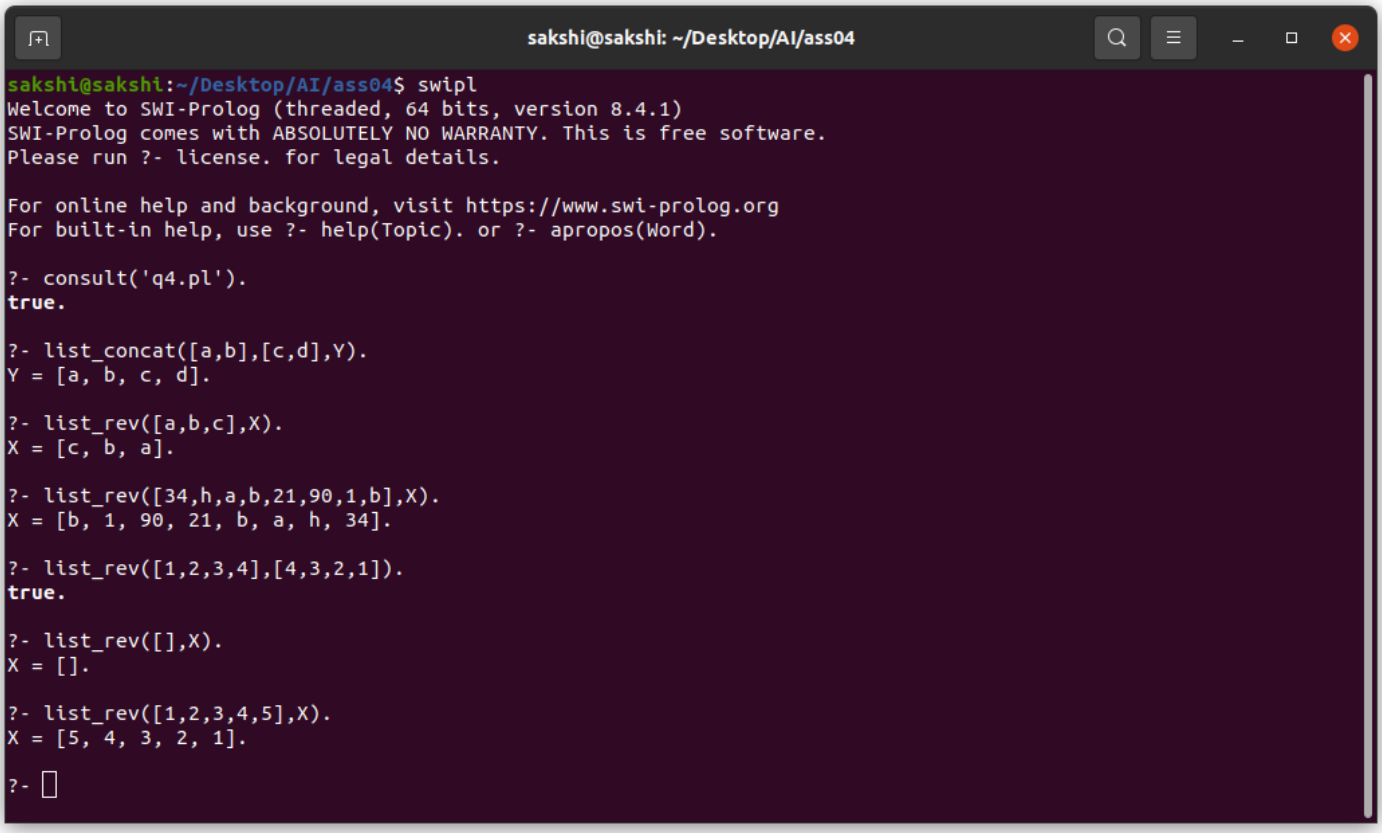
Nth element of the list:



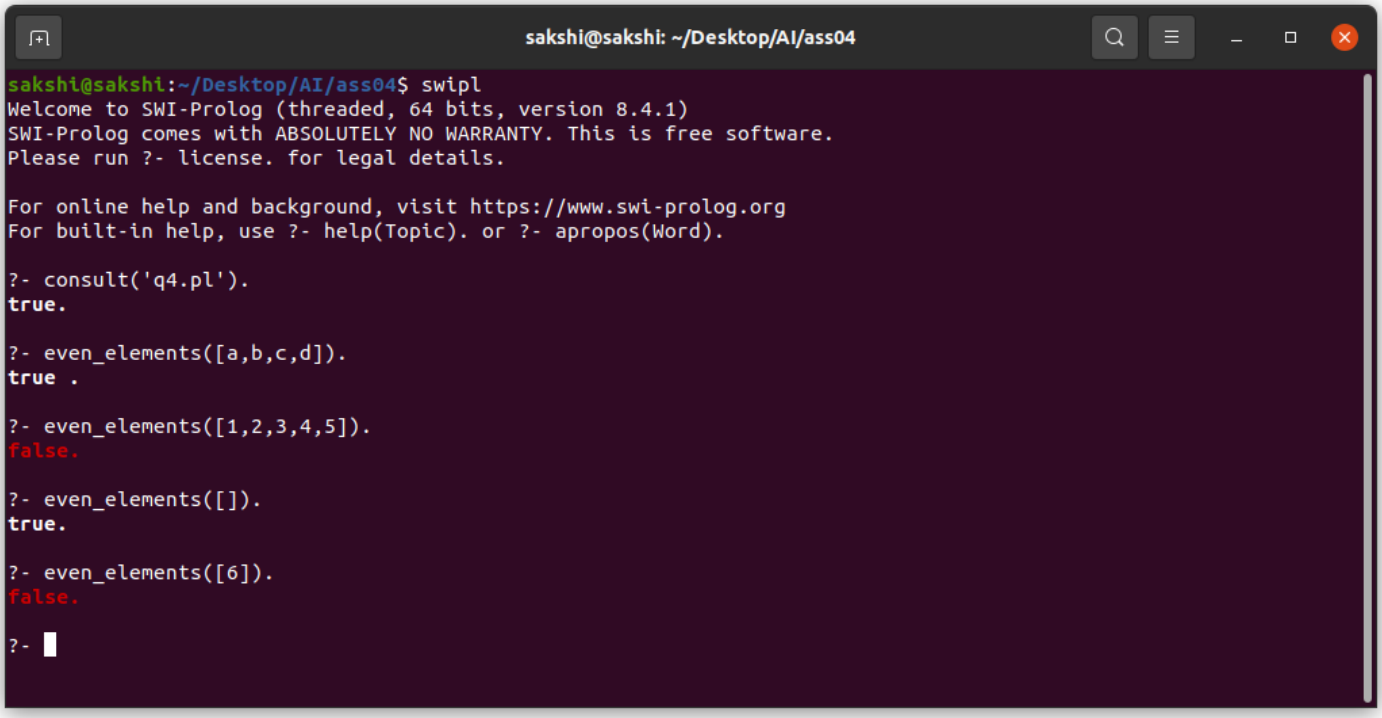
Increment each number in the list:



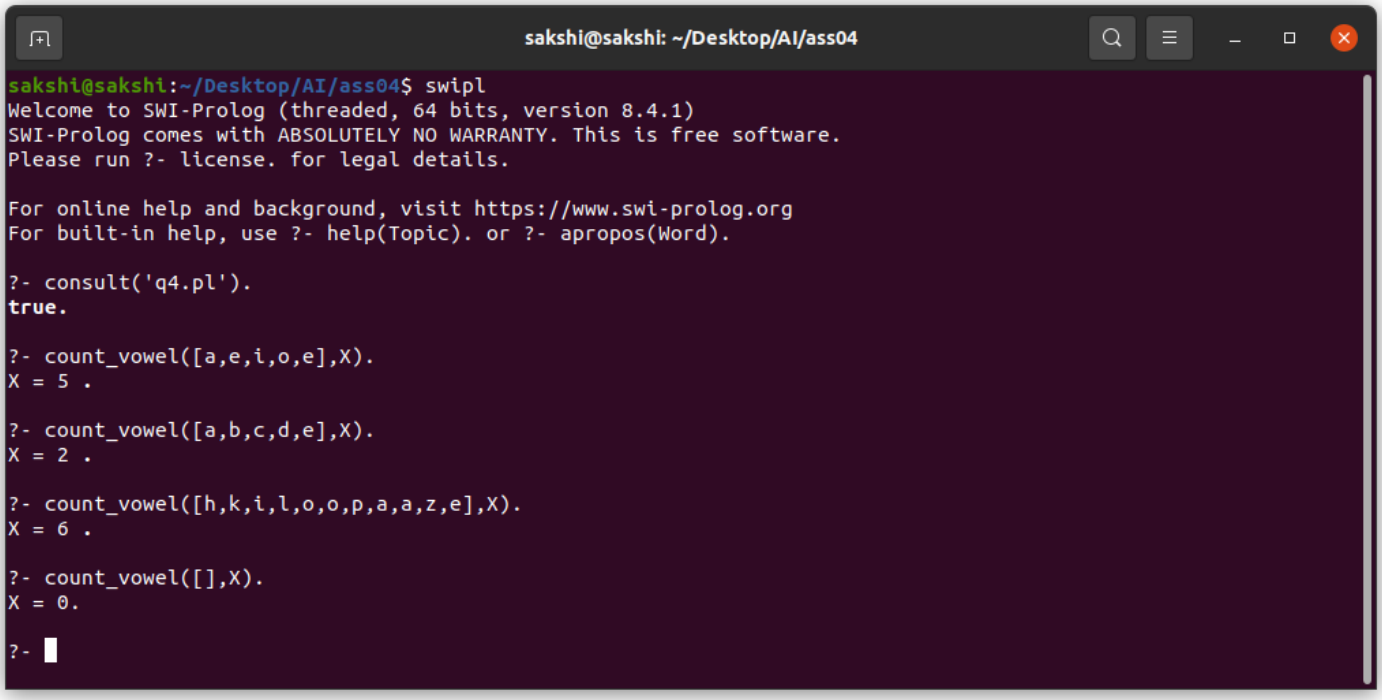
Reverse the list:



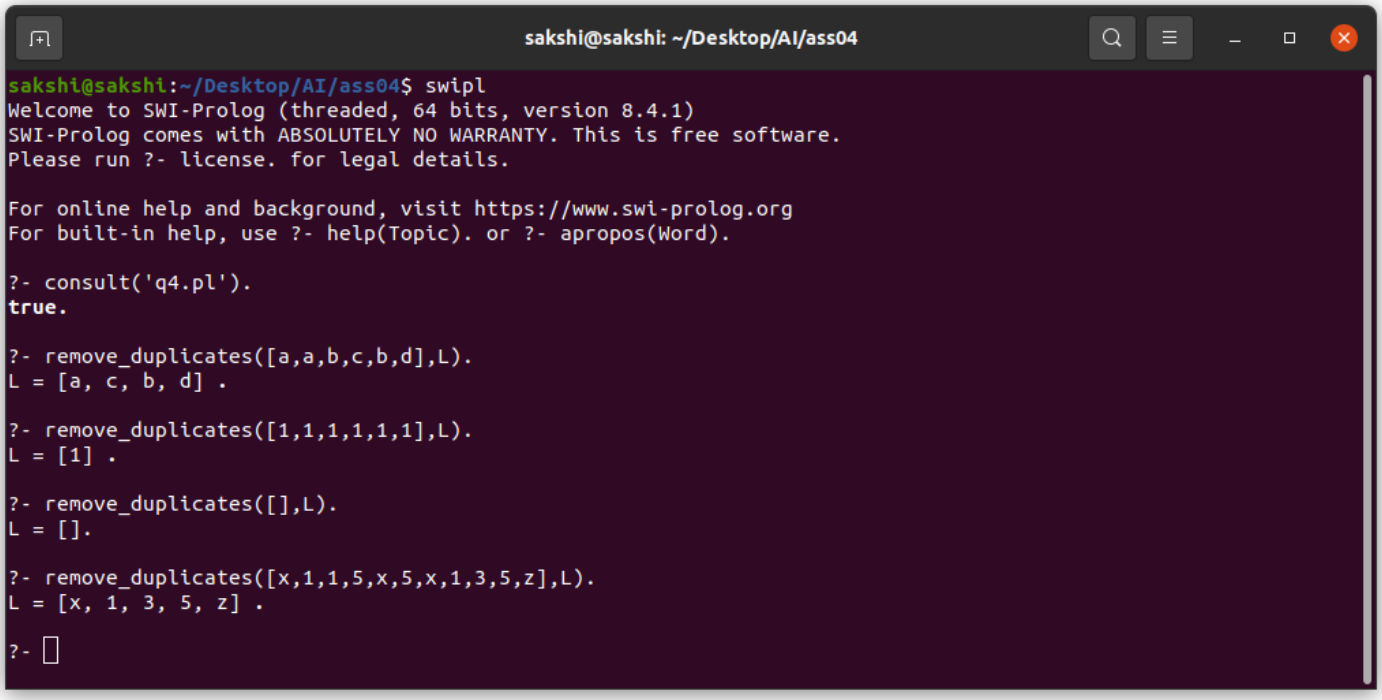
Verify if a list has an even number of elements:



Count vowels in the list:



Remove duplicates from the list:


sakshi@sakshi: ~/Desktop/AI/ass04
sakshi@sakshi:~/Desktop/AI/ass04$ swipl
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.1)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.
For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).
?- consult('q4.pl').
true.
?- remove_duplicates([a,a,b,c,b,d],L).
L = [a, c, b, d] .
?- remove_duplicates([1,1,1,1,1,1],L).
L = [1] .
?- remove_duplicates([],L).
L = [].
?- remove_duplicates([x,1,1,5,x,5,x,1,3,5,z],L).
L = [x, 1, 3, 5, z] .
?-