Mini Assignment-2

P16 (**) Drop every N'th element from a list.

Code:

```
drop_element([], _, []).
drop_element(X, N, Y) :-
  X \= [],
  N_ is N-1,
  take(N_, X, S),
  put(N, X, E),
  drop_element(E, N, E_),
  append(S, E_, Y).

take(_, [], []).
take(0, _, []).
take(N, [X|Xs], [X|Y]) :- N > 0, N_ is N-1, take(N_, Xs, Y).

put(_, [], []).
put(0, X, X).
put(N, [_|Xs], Y) :- N > 0, N_ is N-1, put(N_, Xs, Y).
```

Output:

```
drop_element([p,q,r,s,t,u,v,w], 6, X).
X = [p, q, r, s, t, v, w]

drop_element([1,2,3,4,5,6,7], 4, X).
X = [1, 2, 3, 5, 6, 7]
```

P17 (*) Split a list into two parts; the length of the first part is given.

Code:

```
split(X, N, P1, P2) :- take(N, X, P1), put(N, X, P2).
```

Output:

```
split([a, b, c, d, e, f, g], 3, P1, P2)
P1 = [a, b, c],
P2 = [d, e, f, g]

split([1,2,3,4,5,6,7], 3, P1, P2)
P1 = [1, 2, 3],
P2 = [4, 5, 6, 7]
```

P19 (**) Rotate a list N places to the left.

Code:

```
rotate_n(X, N, Y) :-
  length(X, L), N_ is N mod L, split(X, N_, L1, L2), append(L2, L1, Y).
```

Output:

rotate_n([a, b, c, d, e, f, g],4 , X).
X = [e, f, g, a, b, c, d]

rotate_n([1,2,3,4,5,6,7],3 , X).
X = [4, 5, 6, 7, 1, 2, 3]

P21 (*) Insert an element at a given position into a list.

Code:

```
insert(X, L, N, R) :- N_ is N-1, split(L, N_, L1, L2), append(L1, [X|L2], R).
```

Output:

insert(hi,[a,b,c,d],4,L).
L = [a, b, c, hi, d]

insert(Hello,[a,b,c,d],2,L).
L = [a, Hello, b, c, d]

P22 (*) Create a list containing all integers within a given range.

Code:

```
create_list(N, N, [N]).
create_list(A, B, [A|R]) :- A \= B, A_ is A + sign(B-A), create_list(A_, B, R).
```

Output:

create_list(3, 12, L).
L = [3, 4, 5, 6, 7, 8, 9, 10, 11, 12]

P26 (**) Generate the combinations of K distinct objects chosen from the N elements of a list

Code:

```
combination(0, _, []).
combination(N, X, [H|R]) :-
  0 < N, tails(X, [H|T]), N_ is N-1, combination(N_, T, R).
```

```
tails(X, X).
tails([_|Xs], T) :- tails(Xs, T).
```

Output:

```
combination(3,[a,b,c,d],L).
L = [a, b, c]
L = [a, b, d]
L = [a, c, d]
L = [b, c, d]
false

combination(4,[1,2,3,4,5,6],L).
L = [1, 2, 3, 4]
L = [1, 2, 3, 5]
L = [1, 2, 3, 6]
L = [1, 2, 4, 5]
L = [1, 2, 4, 6]
L = [1, 2, 5, 6]
L = [1, 3, 4, 5]
L = [1, 3, 4, 6]
L = [1, 3, 5, 6]
L = [1, 4, 5, 6]
L = [2, 3, 4, 5]
L = [2, 3, 4, 6]
L = [2, 3, 5, 6]
L = [2, 4, 5, 6]
L = [3, 4, 5, 6]
```

P31 (**) Determine whether a given integer number is prime.

Code:

```
prime(N) :- integer(N), N > 1, \+ has_factor(N, 2).

has_factor(N, K) :- K * K =< N, N mod K =:= 0.
has_factor(N, K) :- K * K =< N, K_ is K + 1, has_factor(N, K_).
```

Output:

```
prime(23).
true

prime(4).
false
```

P32 (**) Determine the greatest common divisor of two positive integer numbers.

Code:

```
prime_factors(N, Fs) :- N > 1, prime_factors(N, 2, Fs).
prime_factors(1, _, []) :- !.
prime_factors(N, P, Fs) :-
  P =< N, N mod P =\= 0, !, next_prime(P, P_), prime_factors(N, P_, Fs).
prime_factors(N, P, [P|Fs]) :-
  P =< N, N_ is N / P, prime_factors(N_, P, Fs).

next_prime(P, P_) :- P_ is P + 1, prime(P_), !.
next_prime(P, N) :- P_ is P + 1, next_prime(P_, N).
```

Output:

```
prime_factors(225, L).
L = [3, 3, 5, 5]

prime_factors(1225, L).
L = [5, 5, 7, 7]
```