1.Write a prolog program that store countries with their federal language used in a list form using the predicate lang_country. Define the right rule to for the following:

- To display countries that uses some specific language such as English or Amharic as their official or federal language. It should count the number of countries that use the language in addition to the list.
- To display the language used in a country taking the name of the country as input.

```
% Facts: countries and their languages
lang_country(usa, english).
lang_country(uk, english).
lang_country(ethiopia, amharic).
lang_country(kenya, english).
lang_country(sudan, arabic).
lang_country(france, french).
lang_country(italy, italian).
lang_country(japan, japanese).
lang_country(germany, german).
lang_country(spain, spanish).

% Rule 1: Find countries by language
countries_with_language(Language) :-
  findall(Country, lang_country(Country, Language), Countries),
  write('Countries that use '), write(Language), write(' as their official language:'), nl,
  (  Countries \= [] ->
     write_list(Countries),
     length(Countries, Count),
     write('Number of countries: '), write(Count), nl
  ;  write('No countries found using this language.'), nl
  ).

% Rule 2: Find language by country
language_of_country(Country) :-
  lang_country(Country, Language),
  write('The official language of '), write(Country), write(' is '), write(Language), nl.
language_of_country(Country) :-
  \+ lang_country(Country, _),
  write('No data available for the country: '), write(Country), nl.

% Helper: Print list
write_list([]).
write_list([H|T]) :-
  write('- '), write(H), nl,
  write_list(T).
```

2. Create three predicates in a knowledge base that add list element at the beginning, at the middle and at the end of a list.

```
% 1. Predicate to add an element at the beginning of a list
add_at_beginning(Element, List, [Element|List]).
% 2. Predicate to add an element in the middle of a list
add_at_middle(Element, List, Result) :-
   middle_index(List, Index),          % Find the middle index
   insert_at(Element, List, Index, Result).

% 3. Predicate to add an element at the end of a list
add_at_end(Element, List, Result) :-
   reverse(List, Reversed),            % Reverse the list
   add_at_beginning(Element, Reversed, UpdatedReversed), % Add at the beginning
   reverse(UpdatedReversed, Result).    % Reverse again to get the final list

% Helper predicate to compute the middle index of a list
middle_index(List, Index) :-
   length(List, Length),
   Index is Length // 2.

% Helper predicate to insert an element at a specific position in a list
insert_at(Element, List, 0, [Element|List]).  % Base case: insert at position 0
insert_at(Element, [H|T], N, [H|Rest]) :-
   N > 0,
   N1 is N - 1,
   insert_at(Element, T, N1, Rest).
```

3.List any five built in predicates used in prolog related to list. Use example to explain each.

1. **length/2 :-** This predicate determines the length of a list or can generate a list of a specified length.
   Example ?- length([a, b, c, d], L).
   L = 4.
   Example ?- length(List, 3).
   List = [_, _, _].
2. **reverse/2:** This predicate reverses the order of elements in a list.
   Example: ?- reverse([1, 2, 3, 4], Result).
   Result = [4, 3, 2, 1].
3. **nth0/3:** This predicate retrieves an element from a list based on its 0-based index.
   Example: ?- nth0(2, [a, b, c, d], Element).
   Element = c.
4. **nth1/3:** This is similar to nth0/3, but it uses **1-based indexing**
   Example: ?- nth1(3, [a, b, c, d], Element).
   Element = c.
5. **select/3:** Removes an element from a list and returns the rest of the elements.
   Example: ?- select(2, [1, 2, 3, 4], Result).
   Result = [1, 3, 4].

4. A prolog program that define the following predicate

- Cities and countries / 1

  Cities in Africa

  Example Addis Ababa, Ethiopia.
- City Country/2

  A rule that displays list of Cities for a country name given by user.
- Add City with three arguments this should check if that City country pair exist and if the pair doesn't exist it should add it to the list.

```
% 1. Define the initial list of city-country pairs
cities_in_africa([
    city('Casablanca', 'Morocco'),
    city('Abuja', 'Nigeria'),
    city('Dakar', 'Senegal'),
    city('Johannesburg', 'South Africa'),
    city('Luanda', 'Angola'),
    city('Kigali', 'Rwanda'),
    city('Windhoek', 'Namibia'),
    city('Bamako', 'Mali'),
    city('Harare', 'Zimbabwe'),
    city('Tripoli', 'Libya')
]).
% 2. cities_and_countries/0
% Displays all cities and their respective countries directly without passing a list
cities_and_countries :-
    write('Cities in Africa:'), nl,
    cities_in_africa(List),
    forall(member(city(City, Country), List),
        (write(City), write(', '), write(Country), nl)).
```

```
% 3. city_country/2
% Retrieves cities for a specific country and handles empty cases
city_country(Country, Cities) :-
    find_cities_by_country(Country, Cities),
    (   Cities = [] ->
        write('No cities found for '), write(Country), nl
    ;   write('Cities in '), write(Country), write(' are: '), write(Cities), nl
    ).
% Helper predicate to find all cities for a specific country
find_cities_by_country(Country, Cities) :-
    cities_in_africa(List),
    findall(City, member(city(City, Country), List), Cities).
% 4. add_city/3
% Checks if a city-country pair exists, and if not, adds it to the list
add_city(City, Country, UpdatedList) :-
    cities_in_africa(CurrentList),
    (   city_exists(City, Country, CurrentList) ->
        write('The city-country pair already exists.'), nl,
        UpdatedList = CurrentList
    ;   append(CurrentList, [city(City, Country)], UpdatedList),
        write('City-country pair added successfully.'), nl
    ).
% Helper predicate to check if a city-country pair already exists
city_exists(City, Country, List) :-
    member(city(City, Country), List).
```