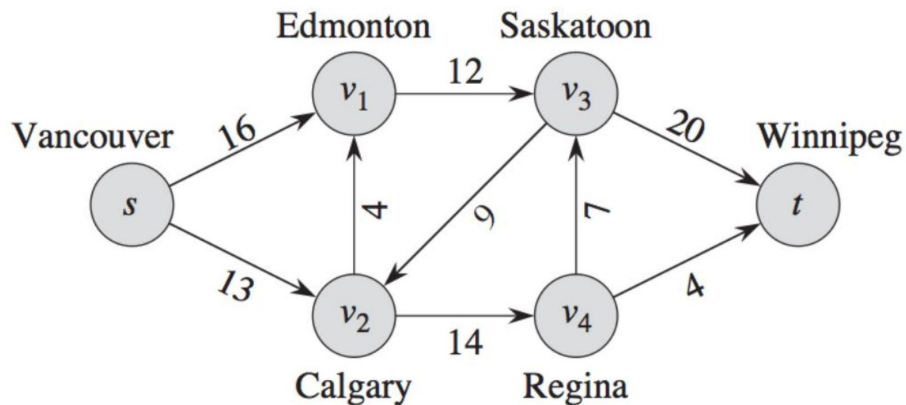


ACTIVIDAD 2



LOGICA (HECHOS Y REGLAS)

```

conexion(vancouver,edmonton,16).
conexion(vancouver,calgary,13).
conexion(edmonton,saskatoon,12).
conexion(calgary,edmonton,4).
conexion(calgary,regina,14).
conexion(saskatoon,calgary,9).
conexion(saskatoon,winnipeg,20).
conexion(regina,saskatoon,7).
conexion(regina,winnipeg,4).

```

% Existe una conexión entre Saskatoon y Vancouver?

```

existeConexion(Origen,Destino):-
    conexion(Origen,Destino,_).

```

% Con qué nodos está conectado Regina y cual es el costo de cada conexión?

```

conectadoDesdeyCosto(Origen,Destino,Costo):-
    conexion(Origen,Destino,Costo).

```

% Regla para determinar si un nodo tiene aristas

```

tieneAristas(Nodo) :-
    conexion(Nodo, _, _);
    conexion(_, Nodo, _).

```

% Regla para determinar cuál es el costo para ir de un nodo X a un Z pasando por Y

```

costoPorCiudades(X, Y, Z, CostoTotal) :-
    conexion(X, Y, Costo1),
    conexion(Y, Z, Costo2),
    CostoTotal is Costo1 + Costo2.

```

% Es posible viajar desde Edmonton a Calgary?

True

% Regla recursiva para conocer si existe un viaje de un Origen y un Destino sin importar las ciudades Intermedias y su costo total.

viaje(Origen, Destino, Costo):-
 conexion(Origen, Destino, Costo).

viaje(Origen, Destino, CostoTotal):-
 conexion(Origen, Intermedio, Costo1),
 viaje(Intermedio, Destino, Costo2),
 CostoTotal is Costo1 + Costo2.

CONSULTAS

existeConexion(saskatoon, vancouver).



existeConexion(calgary, regina).



conectadoDesdeyCosto(regina, Destino, Costo).




A screenshot of a Prolog query interface. The top panel shows a query: `conectadoDesdeyCosto(regina, Destino, Costo).` with a gear icon. Below it, the results are displayed in a list: `Costo = 7,` `Destino = saskatoon`, `Costo = 4,` `Destino = winnipeg`. The bottom panel shows the query prompt: `?- conectadoDesdeyCosto(regina, Destino, Costo).`

tieneAristas(regina).



A screenshot of a Prolog query interface. The top panel shows a query: `tieneAristas(regina).` with a gear icon. Below it, the result `true` is displayed. A row of buttons is shown: `Next`, `10`, `100`, `1,000`, and `Stop`. The bottom panel shows the query prompt: `?- tieneAristas(regina).`



A screenshot of a Prolog query interface. The top panel shows a query: `tieneAristas(regina).` with a gear icon. Below it, the result `true` is displayed three times, each on a new line. The bottom panel shows the query prompt: `?- tieneAristas(regina).`

costoPorCiudades(vancouver, _, saskatoon, CostoTotal).

```
costoPorCiudades(vancouver, _, saskatoon, CostoTotal).  
CostoTotal = 28  
false  
?- costoPorCiudades(vancouver, _, saskatoon, CostoTotal).
```

costoPorCiudadesCiudades(calgary, Y, winnipeg, CostoTotal).

```
costoPorCiudades(calgary, Y, winnipeg, CostoTotal).  
CostoTotal = 18,  
Y = regina  
?- costoPorCiudades(calgary, Y, winnipeg, CostoTotal).
```

costoPorCiudades(edmonton, _, calgary, _).

```
costoPorCiudades(edmonton, _, calgary, _).  
true  
?- costoPorCiudades(edmonton, _, calgary, _).
```

CONSULTA RECURSIVA:

viaje(vancouver, regina, Costo).

 **viaje**(vancouver, regina, Costo).

Costo = 51

Costo = 76

Costo = 101

Costo = 126

Next

10

100

1,000

Stop

?- **viaje**(vancouver, regina, Costo).