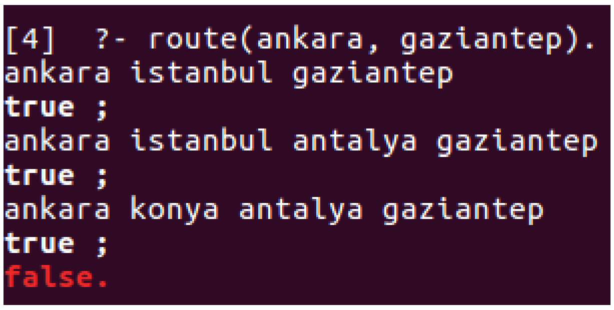
CSE341 PROGRAMMING LANGUAGES HW04 REPORT

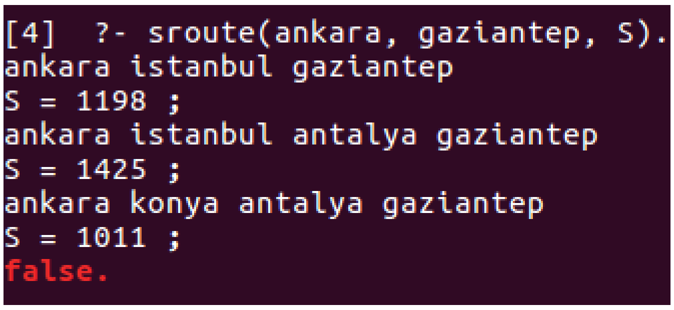
Emire Korkmaz
141044043

<u>Part 1:</u> In this part, every flight possible between cities were written. A function route was written to check if there is any direct or indirect route between two cities. Takes two arguments each of them is a city.



```
[4]  ?- route(ankara, gaziantep).
ankara istanbul gaziantep
true ;
ankara istanbul antalya gaziantep
true ;
ankara konya antalya gaziantep
true ;
false.
```

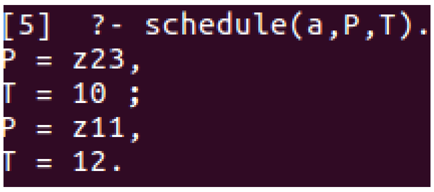
<u>Part 2:</u> In this part, the distance between each cities was added among with the every possible flights between them.



```
[4]  ?- sroute(ankara, gaziantep, S).
ankara istanbul gaziantep
S = 1198 ;
ankara istanbul antalya gaziantep
S = 1425 ;
ankara konya antalya gaziantep
S = 1011 ;
false.
```

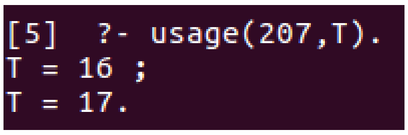
<u>Part 3:</u> In the third part, we were given a program of courses, their times and their classrooms.

→ The first predicate is **schedule** that associates a student to a place and time of class.



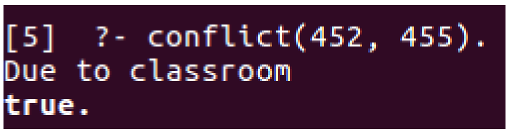
```
[5]  ?- schedule(a,P,T).
P = z23,
T = 10 ;
P = z11,
T = 12.
```

→ The second predicate is **usage** that gives the usage times of a classroom**.**



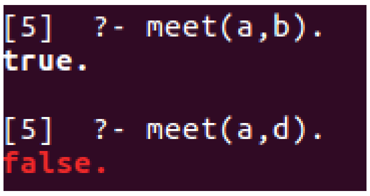
```
[5]  ?- usage(207,T).
T = 16 ;
T = 17.
```

→ The third predicate is **conflict** that gives true if X and Y conflicts due to classroom or time.



```
[5]  ?- conflict(452, 455).
Due to classroom
true.
```

→ The fourth predicate **meet** that gives true if student X and student Y are present in the same classroom at the same time.



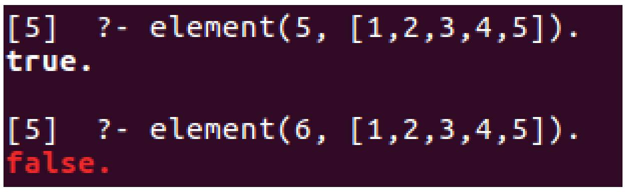
```
[5]  ?- meet(a,b).
true.

[5]  ?- meet(a,d).
false.
```

Part 4:

In this part, we have 4 predicates operating on sets.

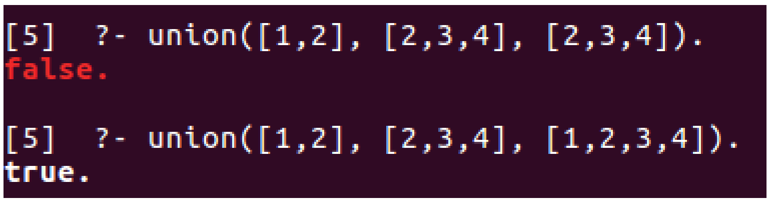
→ The first one is **element(E,S)** that returns true if E is in S.



```
[5]  ?- element(5, [1,2,3,4,5]).
true.

[5]  ?- element(6, [1,2,3,4,5]).
false.
```

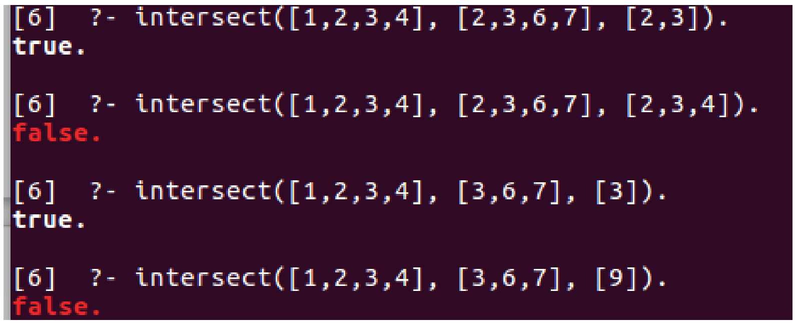
→ The second one is **union(S1,S2,S3)** that returns true if S3 is the union of S1 and S2.



```
[5]  ?- union([1,2], [2,3,4], [2,3,4]).
false.

[5]  ?- union([1,2], [2,3,4], [1,2,3,4]).
true.
```

→ The third predicate is **intersect(S1,S2,S3)** that returns true if S3 is the intersection of of S1 and S2.



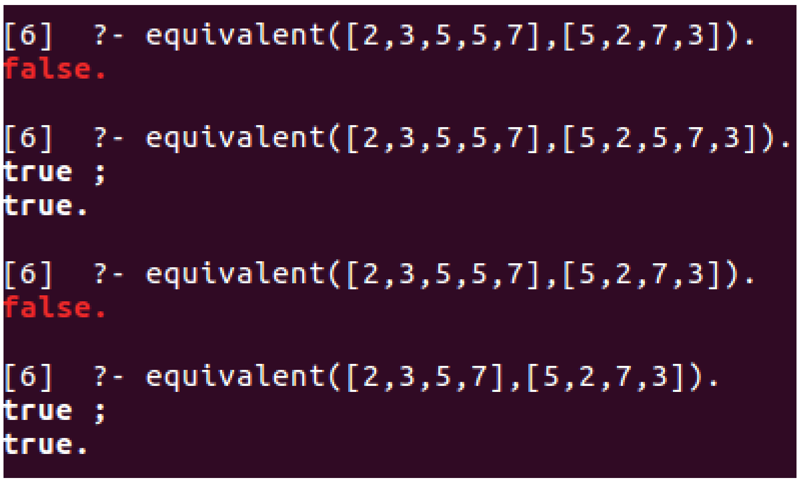
```
[6]  ?- intersect([1,2,3,4], [2,3,6,7], [2,3]).
true.

[6]  ?- intersect([1,2,3,4], [2,3,6,7], [2,3,4]).
false.

[6]  ?- intersect([1,2,3,4], [3,6,7], [3]).
true.

[6]  ?- intersect([1,2,3,4], [3,6,7], [9]).
false.
```

The fourth predicate is **equivalent(S1,S2)** that returns true if S1 and S2 are equivalent sets. I considered the give sets as lists.



```
[6]  ?- equivalent([2,3,5,5,7],[5,2,7,3]).
false.

[6]  ?- equivalent([2,3,5,5,7],[5,2,5,7,3]).
true ;
true.

[6]  ?- equivalent([2,3,5,5,7],[5,2,7,3]).
false.

[6]  ?- equivalent([2,3,5,7],[5,2,7,3]).
true ;
true.
```