# CSE341 – Programming Languages

## Homework #4

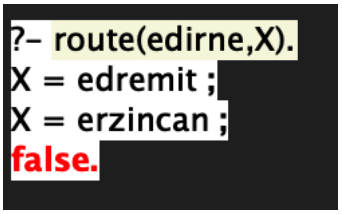
**Mustafa Tokgöz**

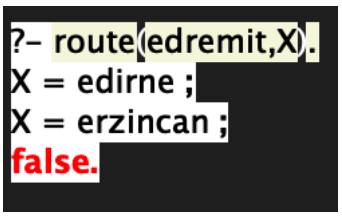
**171044077**

## Part 1)

I wrote the knowledge base part then wrote rules part in part1.pl. Main idea is that If there is a flight a to city and if it is not in the passed list then city is this.If there is city then it adds  city to passed list and calls recursively. It works fine.

OUTPUTS:

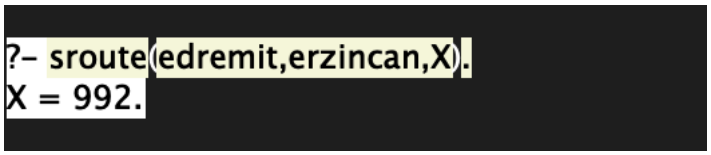


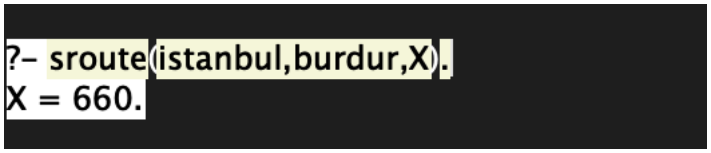




## Part 2)

I wrote the knowledge base part then wrote rules part in part2.pl. Main idea is that If there is a flight a to city and if it is not in the passed list then city is this and distance of them is X .If there is city then it adds  city to passed list and calls recursively and adds distances . For example ,Istanbul and Burdur is our input.After that distance is X = Istanbul-İzmir-Isparta-Burdur =  660 km. Other example , Edremit and Erzincan, X= Edremit-Erzincan = 992 km. It works fine.

OUTPUTS:

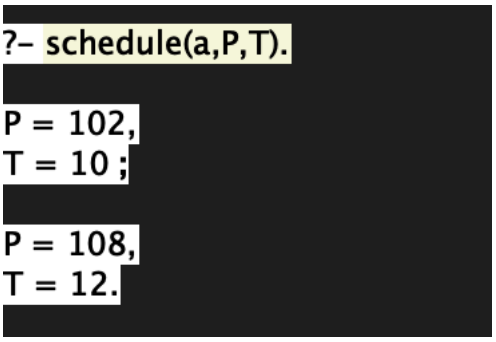


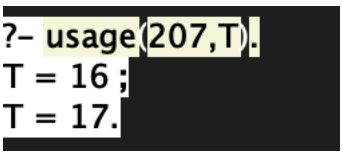



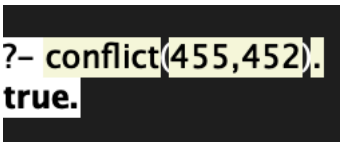
## Part 3)

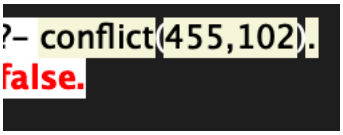
I did this part with respect to table of pdf hw4. I use enroll and when facts for schedule. Because it asks class of student and time of class. I use where and when facts for usage because the given is classroom and I find class by using where and time by using when. For conflict predicate , firstly I find classrooms of classes and if classes are not equal and classrooms are equal then it conflicts or time of classes are equal and classes are not equal and times are equal then it conflicts. For meet predicate, I find classrooms of students by using enroll and where facts then I find times of classes after that if rooms and times are equal, then meet is true else false. All of them works fine.

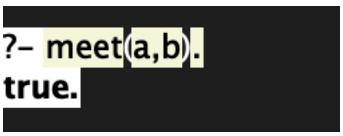
OUTPUTS:

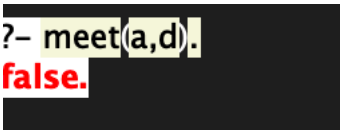


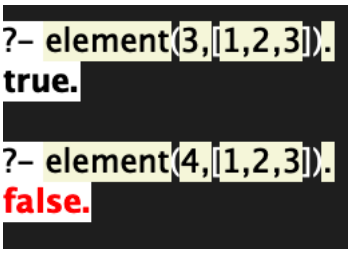



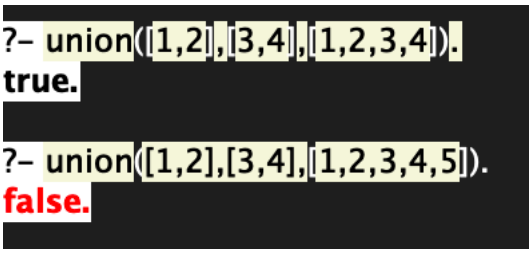



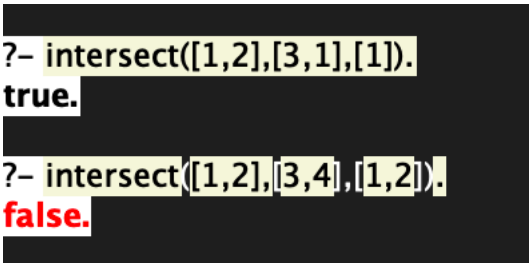
## Part 4)

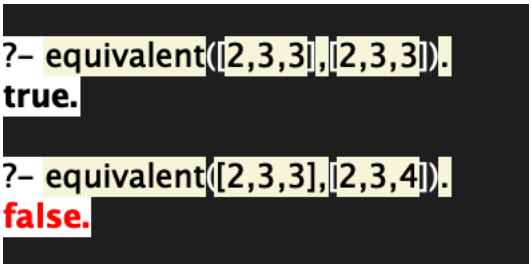
In this part , for element part , I use member to know if E is in Set S. For union part , I create a union list with given inputs S1 and S2 recursively then I control that if the union list of S1 and S2 is equal to S3 with permutation of union list and S3 or not. For intersect part I create intersect list of S1 and S2 recursively then I control that the intersect list of S1 and S2 is equal to S3 with permutation of intersect list and S3. For equivalent part , I use permutation of S1 and S2 sets. All of them works fine.

OUTPUTS:



## Part 5)

I didn’t implement  part 5.

## Part 6)

I didn’t implement part 6.

**Mustafa Tokgöz**

**171044077**