CSE 341 – PROGRAMMING LANGUAGES
HW4

PART 1 : I implement the possible flights between some of the cities in Turkey. For example flight(edirne, edremit). It shows that edirne and edremit has a flight.

I write the predicate route(X,Y) and it returns true if there is a flight between to city.

I use the online prolog terminal. " https://swish.swi - prolog.org/ " .

I try this predicate this way :

route(edirne,erzincan).

It prints true. And I try :

route(edirne,X).

It prints X=edremit and click the next button it prints X=erzincan and next again it terminates.

```
route(edirne,erzincan).
true

route(edirne,X).
X = edremit
X = erzincan
false

?- route(edirne,X).
```

PART 2 :

distance(edremit,erzincan,1044). distance(erzincan,edremit,1044).

I write the distance between two cities. I found the shortest distance between them.

```
sroute(edremit,erzincan,X).
X = 1044

?- sroute(edremit,erzincan,X).
```

PART 3 :

3.1 : This prediction takes a student and find its place and time. It uses enroll, when and where predicates.

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

3.2 : This prediction takes a place and find the usage times of classrooms. I use classes facts.

```
usage(207,T).
T = 16
T = 17
?- usage(207,T).
```

3.3 : I write this prediction two seperate part. One is controlling the conflict of places and the other is time. I used when and where predicates.

```
conflict(455,452).
true
Next | 10 | 100 | 1,000
?- conflict(455,452).
```

3.4 : I check that if student are in same room at the same time. Returns true if they are meet otherwise false.

```
meet(a,b)
true
?- meet(a,b)
```

```
meet(a,d)
false
?- meet(a,d)
```

PART 4 :

4.1 : In this part I wrote element predicate. Its outputs are:

```
element(5,[1,2,3,4,5])
true
element(7,[1,2,3,4,5])
false
?- element(7,[1,2,3,4,5])
```

4.2 : I wrote union predicate in this part. I used 2 helper predicate. Check if all elements of S3 is also S1 or S2. Predicate true if all elements in S1 and S2 into the S3. But if there are different elements in the S3 that are not in S1 and S2. Its outputs are:

```
union([3,5,6],[1,2,4,7],[2,1,3,5,4,7,6]).
true
Next | 10 | 100 | 1,000 | Stop
union([1,2,3],[4,5,6],[6,5,4,3,2,1]).
true
Next | 10 | 100 | 1,000 | Stop
union([1,2,0],[4,5,6],[9,5,8,3,0,1]).
false
```

4.3 : I wrote intersect predicate in this part. I used 2 helper predicate. Predicate true if all elements in the S3 in the S1 and S2. Its outputs are:

```
intersect([1,5,4],[1,3,4],[1,4]).
true
Next | 10 | 100 | 1,000 | Stop
intersect([1,5,4],[1,3,4],[1,5]).
false
?- intersect([1,5,4],[1,3,4],[1,5]).
```

4.4 : I wrote equivalent predicate in this part. It checks two set are equal or not. Its outputs are:

```
equivalent([1,5,4],[1,5,4]).
true
equivalent([1,5,4],[1,5,6]).
false
?- equivalent([1,5,4],[1,5,6]).
```

PART 5:

In my code:
equation(L,LT,RT) :- L is the list of numbers which are the leaves in the arithmetic terms LT and RT.
term(L,T) :- L is the list of numbers which are the leaves in the arithmetic term T
do(L) :- Find all solutions to the problem as given by the list of numbers L, and print them.

```
medo@medo:~/Desktop$ swipl part5.pl
Welcome to SWI-Prolog (threaded, 64 bits, version 7.6.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit http://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- do([2,3,6,7,11]).
2*(3+6) = 7+11
2*(3+6)-7 = 11
2/(3/6)+7 = 11
2/3*6+7 = 11
true.

?- do([5,3,5,7,13]).
5+3*5 = 7+13
5*3+5 = 7+13
5+(3*5-7) = 13
5*3+(5-7) = 13
5+3*5-7 = 13
5*3+5-7 = 13
true.

?-
```