CSE341 - Programming Languages Homework #4 REPORT Canberk Arici - 171044062

*** All parts work correctly. ***

PART 1: This part works correctly. I check if there is a direct route between two given cities according to given graph.

Tests of Part1:

```
(base) canberk@canberk-Aspire:~/Desktop/PL 4.odev$ swipl part1.pl
Welcome to SWI-Prolog (threaded, 64 bits, version 8.2.3)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.
For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).
?- route(istanbul,X).
X = izmir;
X = isparta ;
X = burdur;
X = antalya ;
X = konya;
X = ankara ;
X = van ;
X = rize;
X = gaziantep ;
X = gaziantep ;
X = antalya;
X = konya;
X = ankara ;
X = van ;
X = rize;
X = ankara ;
X = konya;
X = antalya ;
X = gaziantep ;
X = van;
X = rize;
X = van;
X = ankara ;
X = konya;
X = antalya ;
X = gaziantep ;
X = rize;
X = rize;
X = van;
X = ankara ;
X = konya;
X = antalya;
X = gaziantep ;
```

```
(base) canberk@canberk-Aspire:~/Desktop/PL 4.odev$ swipl -s part1.pl
Welcome to SWI-Prolog (threaded, 64 bits, version 8.2.3)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.
For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).
?- route(edirne,edremit).
true .
```

PART 2:

I have written facts and then calculated directed or connected shortest distance between two cities in this part and this part works correctly.

Tests of Part2:

PART 3:

I have written predicates according to given database about classes.

```
when(X,Y) - time of the course X is Y
where(X,Y) - place of the course X is Y
enroll(X,Y) - student X is enrolled in course Y
```

Then I have written these predicates:

schedule(S,P,T) that associates a student to a place and time of class,

usage(P,T) that gives the usage times of a classroom,
conflict(X,Y) that gives true if X and Y conflicts due to classroom
or time.

meet(X,Y) that gives true if student X and student Y are present in the same classroom at the same time.

There is no conflict or meet according to given tables in the homework pdf.

```
(base) canberk@canberk-Aspire:~/Desktop/PL ODEVLER/PL 4.odev$ swipl part3.pl
Welcome to SWI-Prolog (threaded, 64 bits, version 8.2.3)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.
For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).
?- schedule(a,P,T).
P = z23,
 = 10 ;
P = z11,
T = 12.
?- usage(z23,T).
T = 10.
?- meet(a,b).
true .
?- meet(b,c).
?- conflict(452,455).
true.
?- conflict(452,108).
```

PART 4:

In this part, I have written these predicates:

element(E,S) that returns true if E is in S.

union(\$1,\$2,\$3) that returns true if \$3 is the union of \$1 and \$2. intersect(\$1,\$2,\$3) that returns true if \$3 is the intersection of \$1 and \$2.

equivalent(S1,S2) that returns true if S1 and S2 are equivalent sets.

```
(base) canberk@canberk-Aspire:~/Desktop/PL 4.odev$ swipl -s part4.pl
Welcome to SWI-Prolog (threaded, 64 bits, version 8.2.3)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license, for legal details.
For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).
?- element(325, [4,21421,325]).
true.
?- element(21412, [1,2]).
?- union([45,2],[3,6],X).
X = [3, 6, 45, 2].
?- union([54,6],[23,2],[3,5,6,8]).
?- intersect([5,3], [4,6,5], X).
X = [5] .
?- intersect([5,3], [4,6], [3,4,5,6]).
?- intersect([5,3], [4,5], [5]).
?- equivalent([76,34,1],[43,6]).
?- equivalent([6,43,1],[1,43,6]).
true .
```

PART 5:

In this part, I am able to take just one line from input.txt that is in a format. Example for input format: [2,3,5,7,11].

Then I write possible correct equations to output.txt. Please run main predicate in terminal to run program.

Algorithm:

- 1 Read input.txt and get contents
- 2 Clean output.txt by opening and closing it
- 3 Get left and right term from the given list then find appropriate equivalent terms
- 4 Write possible answers to output.txt

Please don't put any spaces after line of numbers in input.txt

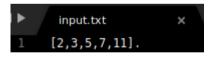
```
(base) canberk@canberk-Aspire:~/Desktop/PL 4.odev$ swipl -s part5.pl
Welcome to SWI-Prolog (threaded, 64 bits, version 8.2.3)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

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

?- main.
true .
```

TESTS FOR PART5:

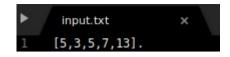
input.txt:



```
output.txt x

1 2 = 3-(5+(7-11))
2 2 = 3-(5+7-11)
3 2 = 3-5-(7-11)
4 2 = 3-(5+7)+11
5 2 = 3-5-7+11
6 2 = (3*5+7)/11
7 2*(3-5) = 7-11
8 2-(3-(5+7)) = 11
9 2-(3-5-7) = 11
10 2-3+(5+7) = 11
11 2-(3-5)+7 = 11
12 2-3+5+7 = 11
```

input.txt:



output.txt:

```
output.txt ×

1 5+3*5 = 7+13
2 5*3+5 = 7+13
3 5+(3*5-7) = 13
4 5*3+(5-7) = 13
5 5+3*5-7 = 13
6 5*3+5-7 = 13
```

PART 6:

In this part, I embedded tests that are given in homework pdf. There are 3 tests, you can use tests by writing to console "test1.", "test2.", "test3.", output is written to output.txt.

* test2 and test3 take too much time to give output.

Algorithm:

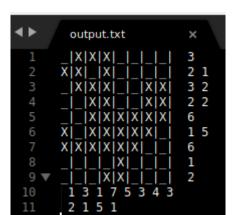
- 1- Generate all possible combinations of rows and columns.
- 2- Search for solution by comparing data of rows and data of appropriate columns then get the solution which is an intersection of these rows and columns.
- 3 Write output bitmap to output.txt

For example, running code for test1.

```
(base) canberk@canberk-Aspire:~/Desktop/PL 4.odev/part 6$ swipl part6.pl Welcome to SWI-Prolog (threaded, 64 bits, version 8.2.3)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software. Please run ?- license. for legal details.

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

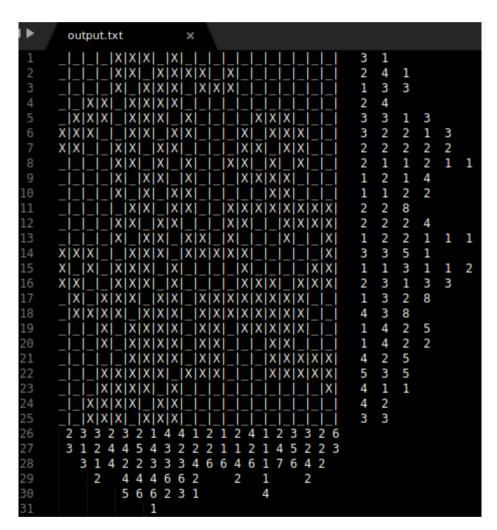
?- test1.
```



Running code for test2.

```
(base) canberk@canberk-Aspire:~/Desktop/PL 4.odev/part 6$ swipl part6.pl
Welcome to SWI-Prolog (threaded, 64 bits, version 8.2.3)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).
?- test2.
```



Running code for test3.

```
(base) canberk@canberk-Aspire:~/Desktop/PL 4.odev/part 6$ swipl part6.pl
Welcome to SWI-Prolog (threaded, 64 bits, version 8.2.3)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.
For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).
?- test3.
```

