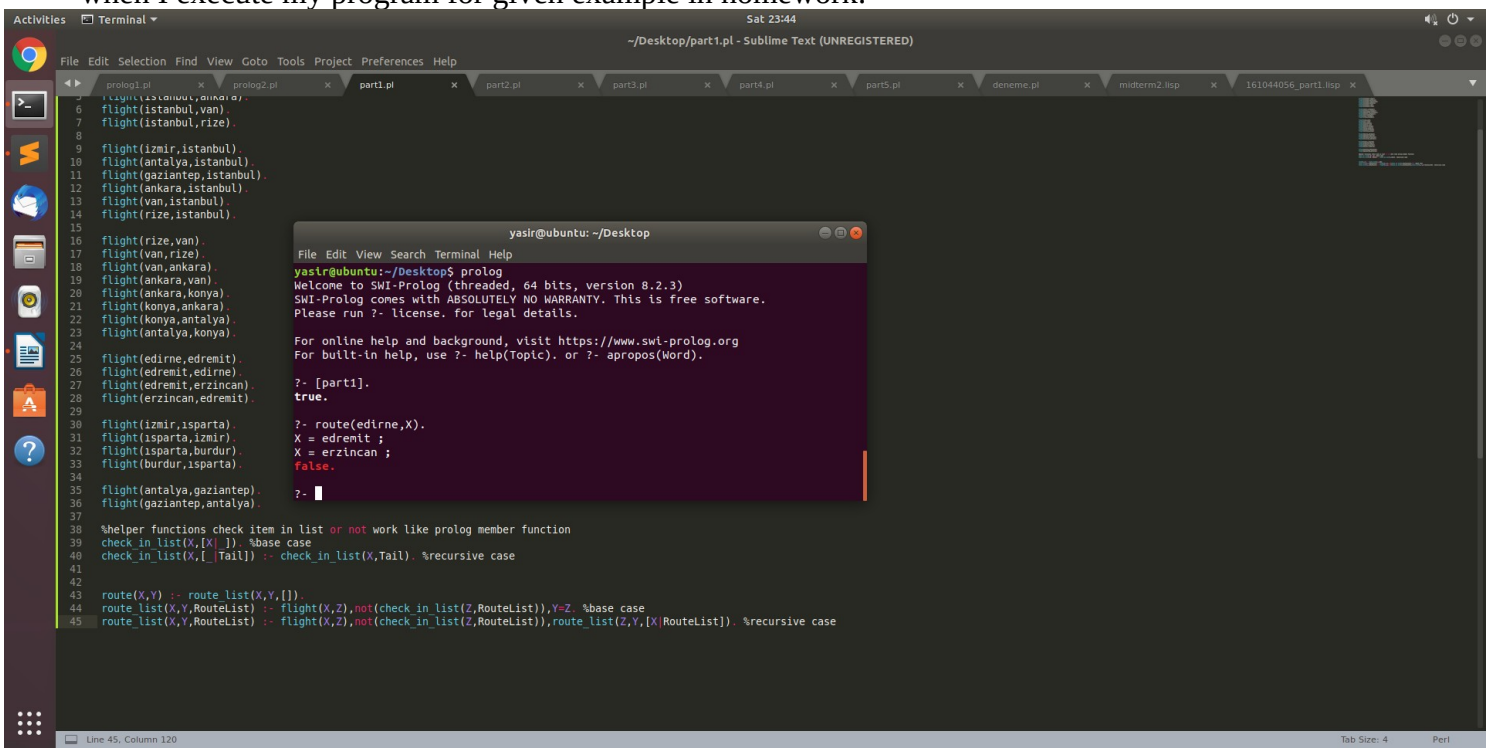
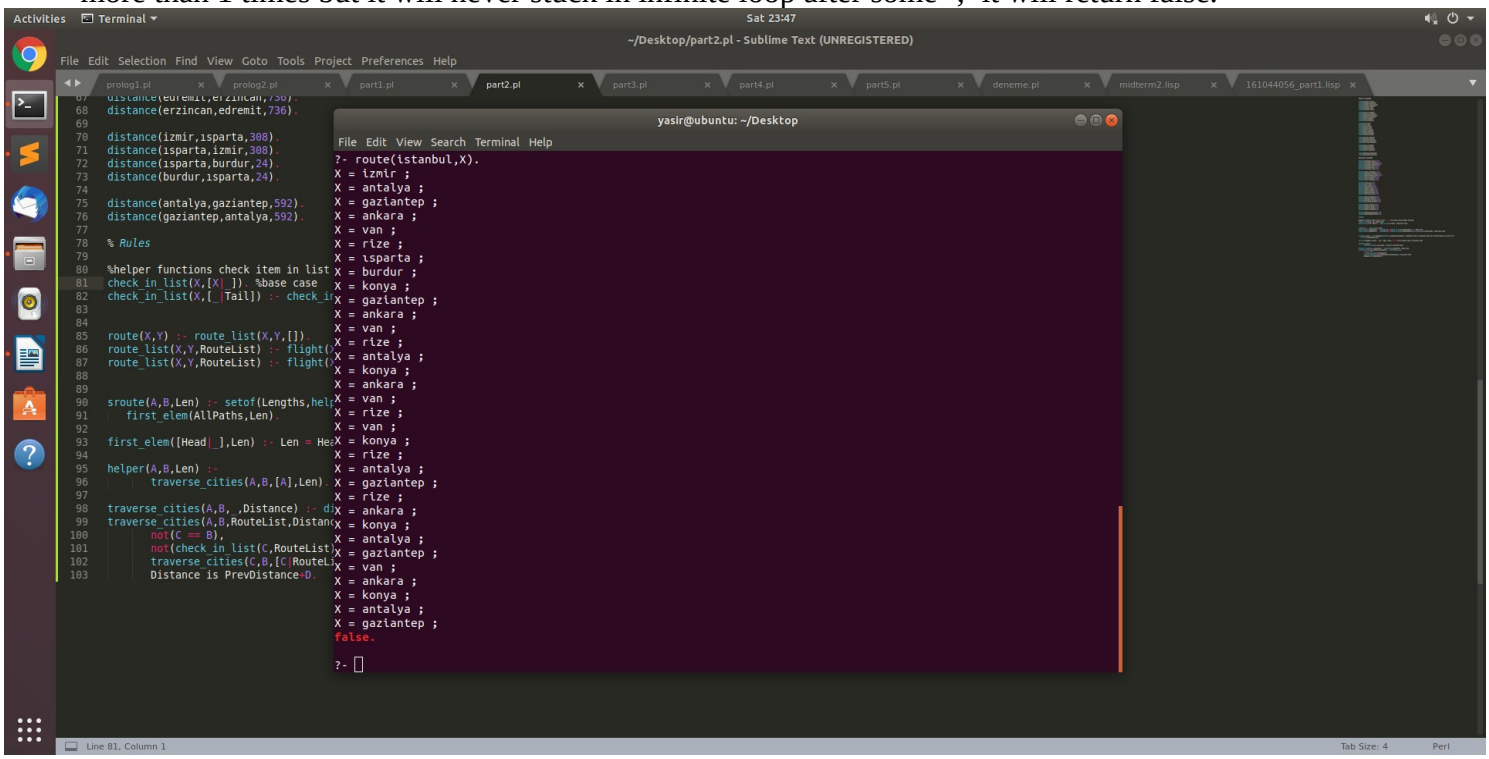
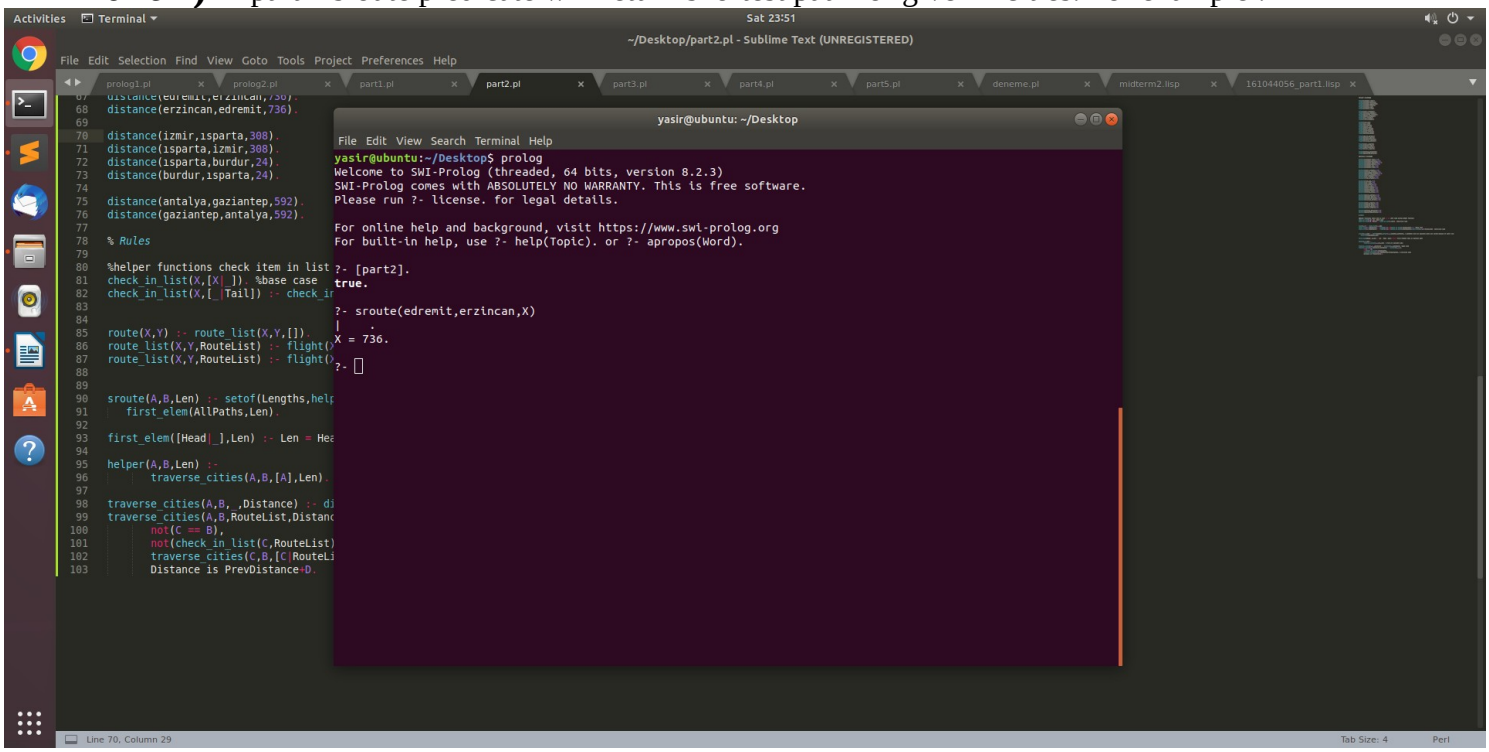
Part 1) I write my flight facts and route predicates. Also there is check_in_list predicate it just check given element is in a list or not. Route call route list predicate and route list is a recursive predicate that check flights
when I execute my program for given example in homework:



Like in example route(edirne,X) return edremit and erzincan.
For route(istanbul,X) it prints possible X but because of recursive calls it will print some possible cities more than 1 times but it will never stuck in infinite loop after some ‘;’ it will return false.

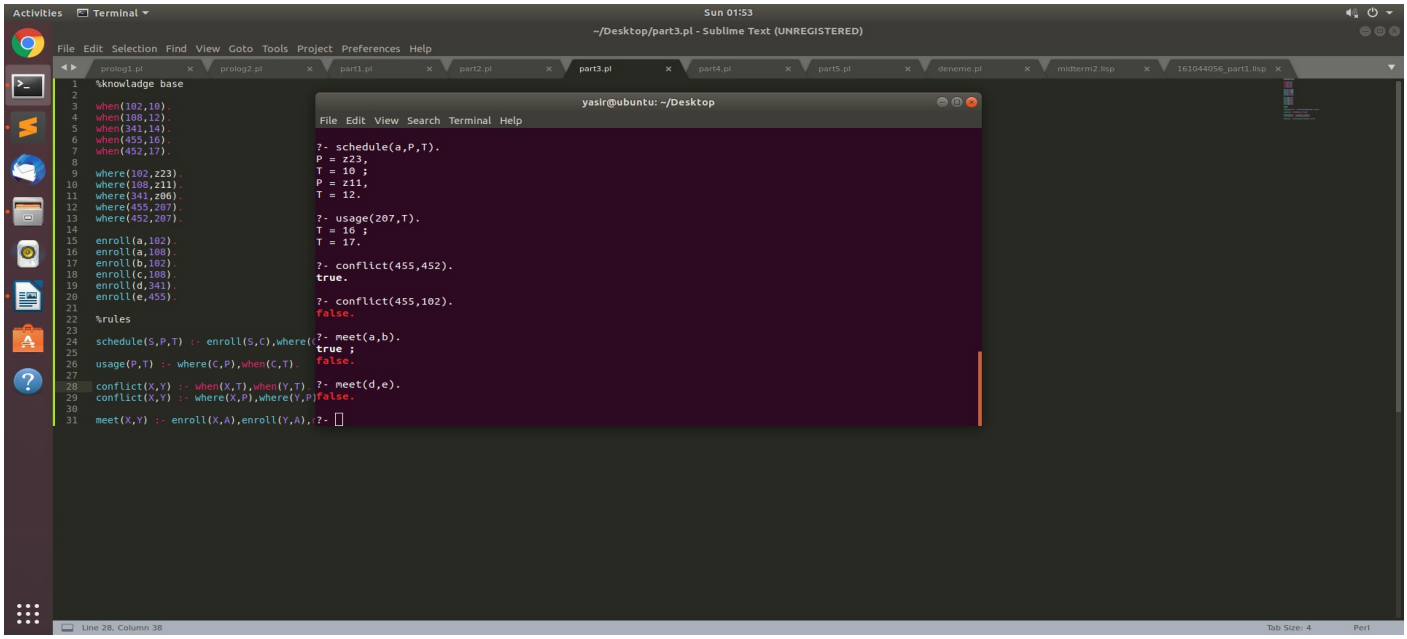


Part 2) in part 2 sroute predicate will return shortest path for given 2 cities. For example :



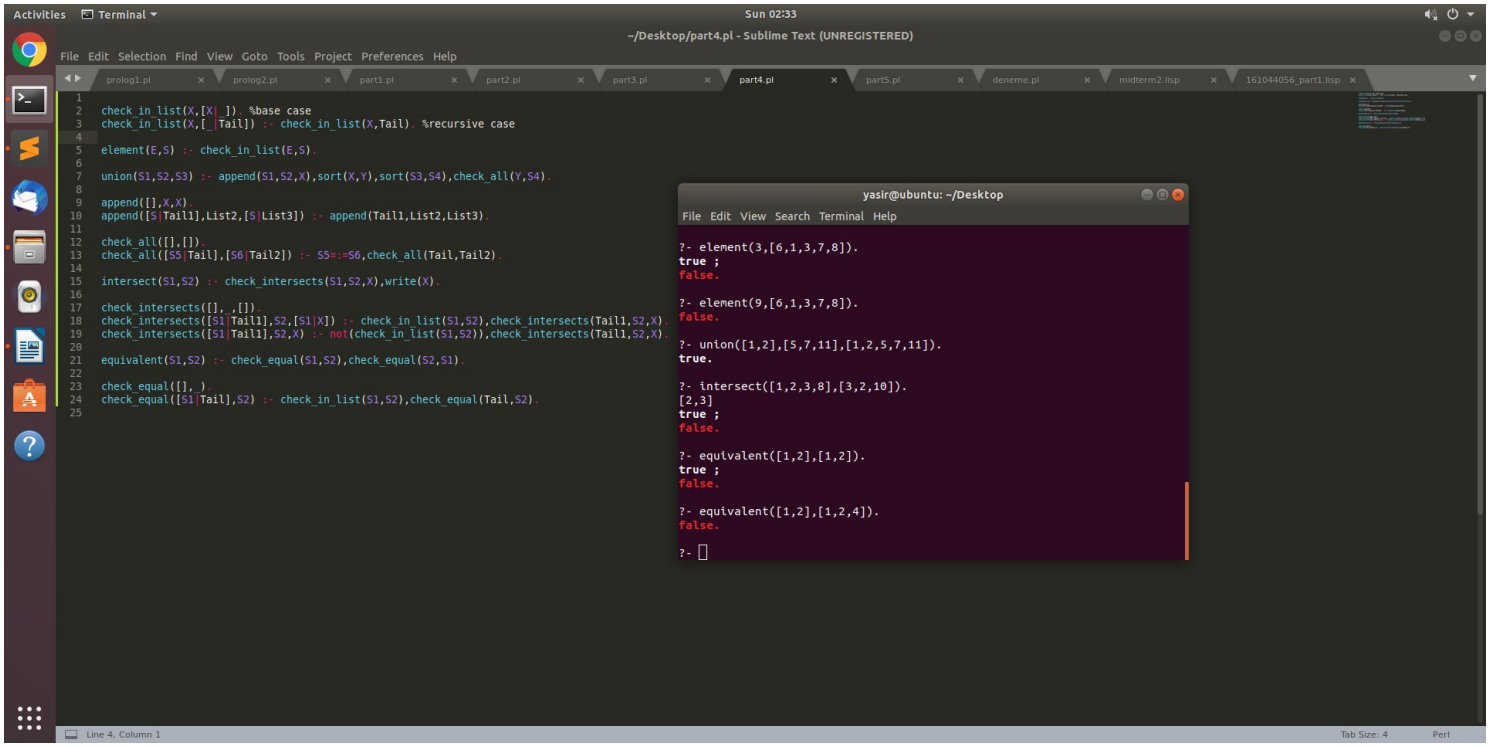
Edremit-Erzincan return 736 km because for this path shortest possible path length is 736 km.

Part 3) There are 4 predicates for this part. Schedule predicate associate a student to a place and time of class
usage predicate gives the usage time of a class.
Conflict predicate gives true if X and Y conflicts because of classroom or time.
Meet predicate gives true if student X and Y in same class at the same time.
Some examples of those:



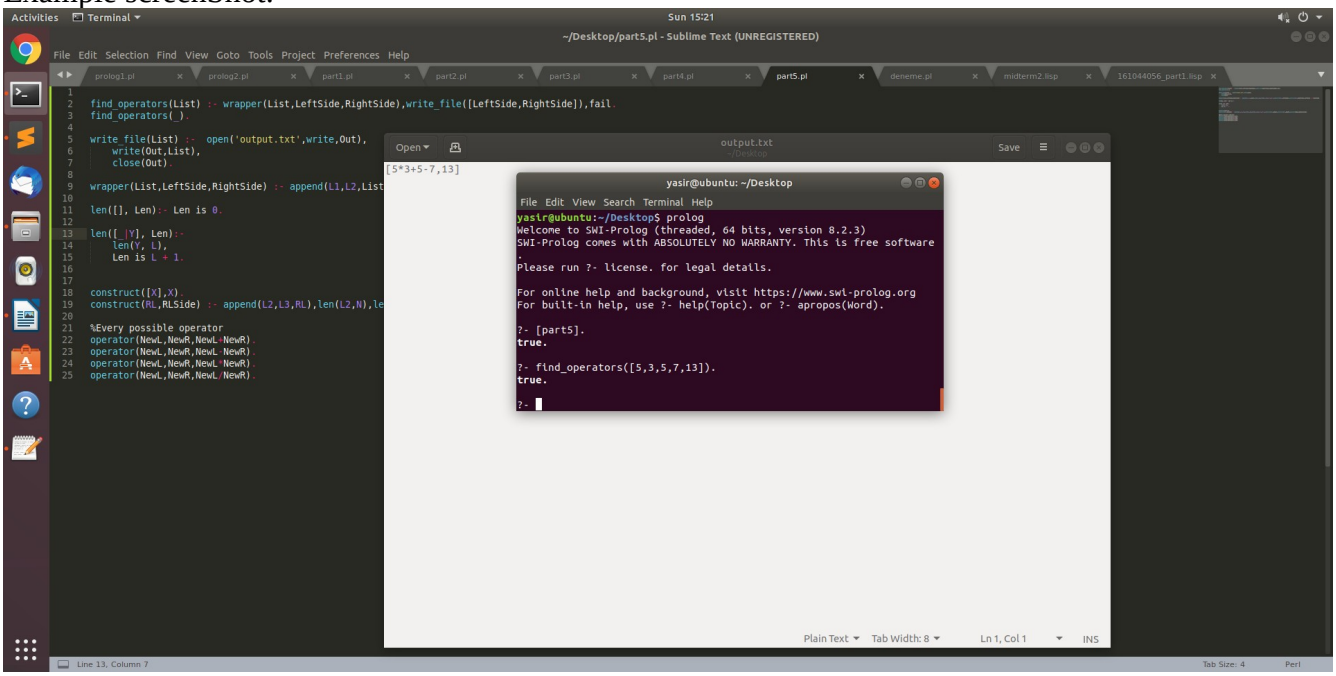
Part 4) In part 4 there is element predicate that check a given input is an element of a set or not, union predicate take 3 input, if first and second sets union is third one return true otherwise false, Intersect predicate take 2 input as set and return intersection of this 2 set, equivalent predicate take 2 list and check this 2 list same equal or not.

Example program output:



Part 5) In part 5 I write my solution to output.txt file. You can execute program with find_operators(List) quary and for this list solution will be in output.txt file.

Example screenShot:



Here I execute my program with [5,3,5,7,13] list and for this list solution will be [5*3+5-7,13] and I write this output.txt