Principles of Programming Language

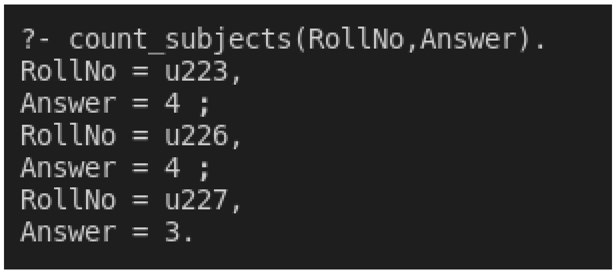
Assignment 5

Student Details

Name: Krunal Rank
Adm No: U18CO081

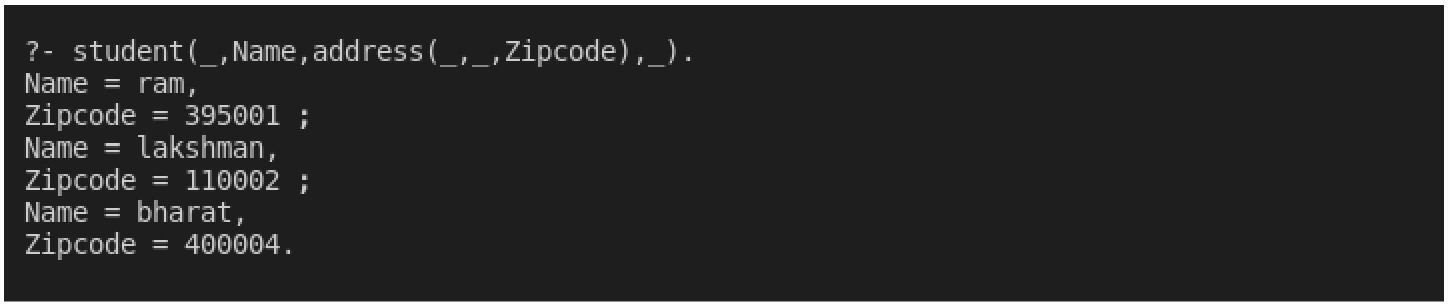
1

```
count([],X) :- X is 0.
count([_ | T],X) :- count(T,X1), X is X1 + 1.

count_subjects(RollNo,N):-
   student(RollNo,_,_,X),
   count(X,N).
```



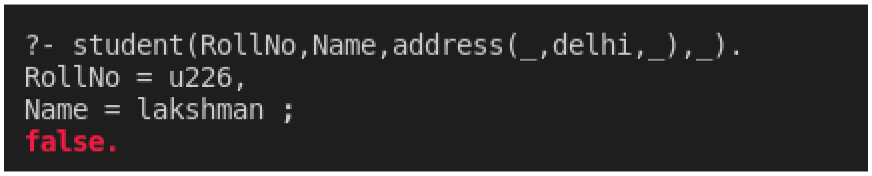
```
?- count_subjects(RollNo,Answer).
RollNo = u223,
Answer = 4 ;
RollNo = u226,
Answer = 4 ;
RollNo = u227,
Answer = 3.
```

2



```
?- student(_,Name,address(_,_,Zipcode),_).
Name = ram,
Zipcode = 395001 ;
Name = lakshman,
Zipcode = 110002 ;
Name = bharat,
Zipcode = 400004.
```

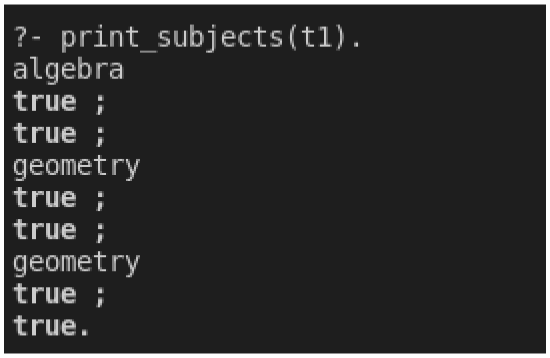
3



```
?- student(RollNo,Name,address(_,delhi,_),_).
RollNo = u226,
Name = lakshman ;
false.
```

4.

```
count_subjects([subject(Teacher,N)|T],Teacher):-
   write(N),
   nl,
   count_subjects(T,Teacher).

count_subjects([_|T],Teacher):-
   count_subjects(T,Teacher).

count_subjects([],_).

print_subjects(Teacher):-
   student(_,_,_,X),
   count_subjects(X,Teacher).
```

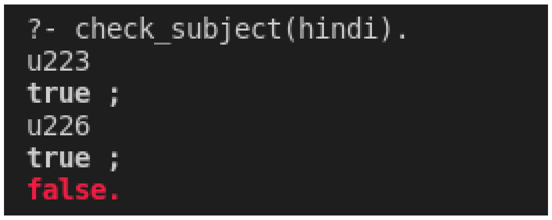


```
?- print_subjects(t1).
algebra
true ;
true ;
geometry
true ;
true ;
geometry
true ;
true.
```

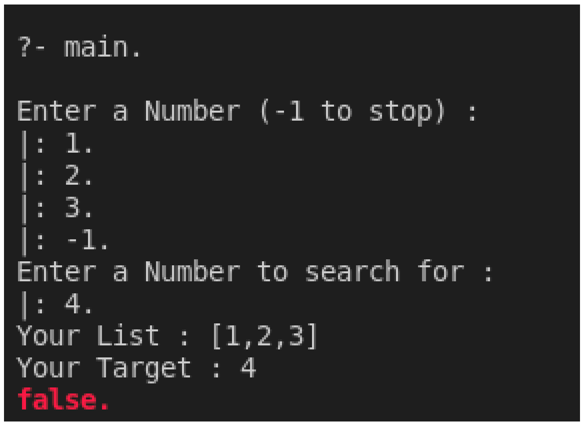
5

```
check([subject(_,Subject)|_],Subject).

check([_|T],Subject):-
   check(T,Subject).

check([],_):-fail.
```

```
check_subject(Subject):-
    student(RollNo,_,_,X),
    check(X,Subject),
    write(RollNo),
    nl.
```



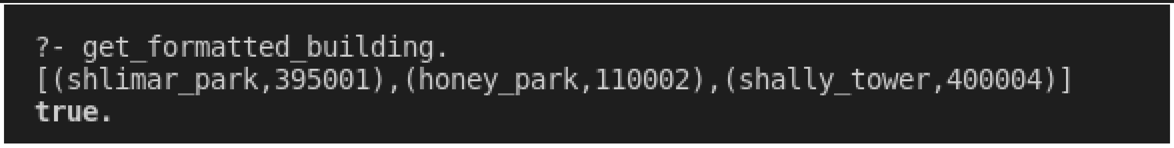
```
?- check_subject(hindi).
u223
true ;
u226
true ;
false.
```



```
?- main.

Enter a Number (-1 to stop) :
|: 1.
|: 2.
|: 3.
|: -1.
Enter a Number to search for :
|: 4.
Your List : [1,2,3]
Your Target : 4
false.
```

6

```

get_formatted_building:-
    findall((B,C),student(_,_,address(B,_,C),_), L),
    write(L).
```

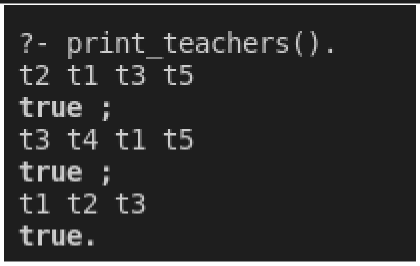


```
?- get_formatted_building.
[(shlimar_park,395001),(honey_park,110002),(shally_tower,400004)]
true.
```

7

```

print_teacher([subject(Teacher,_)|T]):-
    write(Teacher),write(" "),
    print_teacher(T).

print_teacher([]).
```

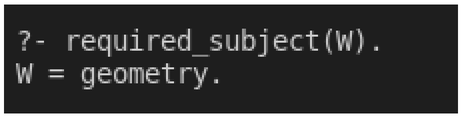
```
print_teachers():-
   student(_,_,_,X),
   print_teacher(X).
```



```
?- print_teachers().
t2 t1 t3 t5
true ;
t3 t4 t1 t5
true ;
t1 t2 t3
true.
```

8

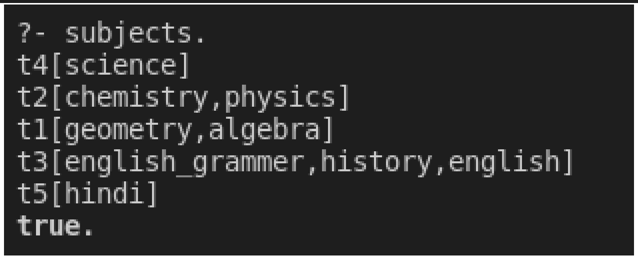
```
append_list([],[]).
append_list([H|T],W):-
   append_list(T,W1),
   append(W1,H,W).

count([],_,0).
count([subject(_,S)|T],S,C):-
   count(T,S,C1),
   C is C1 + 1,
   !.
count([_|T],S,C):-
   count(T,S,C).

req_sub(S,W):-
   count(S,W,C),
   forall(member(subject(_,W1), S),(count(S,W1,C1), C >=C1)).

required_subject(W):-
   findall(X,student(_,_,_,X), LL),
   append_list(LL,L),
   req_sub(L,W).
```



```
?- required_subject(W).
W = geometry.
```

9

```
append_list([],[]).
append_list([H|T],W):-
   append_list(T,W1),
   append(W1,H,W).

unique([],[]).
unique([subject(H,_)|T],[H|T1]):-
   forall(member(subject(K,_),T), K \= H),unique(T,T1),!.
unique([_|T],T1):-unique(T,T1).

teaches(_,[],[]).
teaches(T,[subject(T,Subj)|Tail], Y):-
   member(subject(T,Subj), Tail),
   teaches(T,Tail,Y).
teaches(T,[subject(T,Subj)|Tail], [Subj|Rest]):-
   teaches(T,Tail,Rest).
teaches(T,[_|Tail], Y):-
   teaches(T,Tail,Y).

subjects:-
   findall(X,student(_,_,_,X),LL),
   append_list(LL,L),
   unique(L,Teachers),
   forall(member(T,Teachers),
       (
           teaches(T, L, Ans),
           write(T),
           write(Ans),
           nl
       )
   ).
```



```
?- subjects.
t4[science]
t2[chemistry,physics]
t1[geometry,algebra]
t3[english_grammer,history,english]
t5[hindi]
true.
```