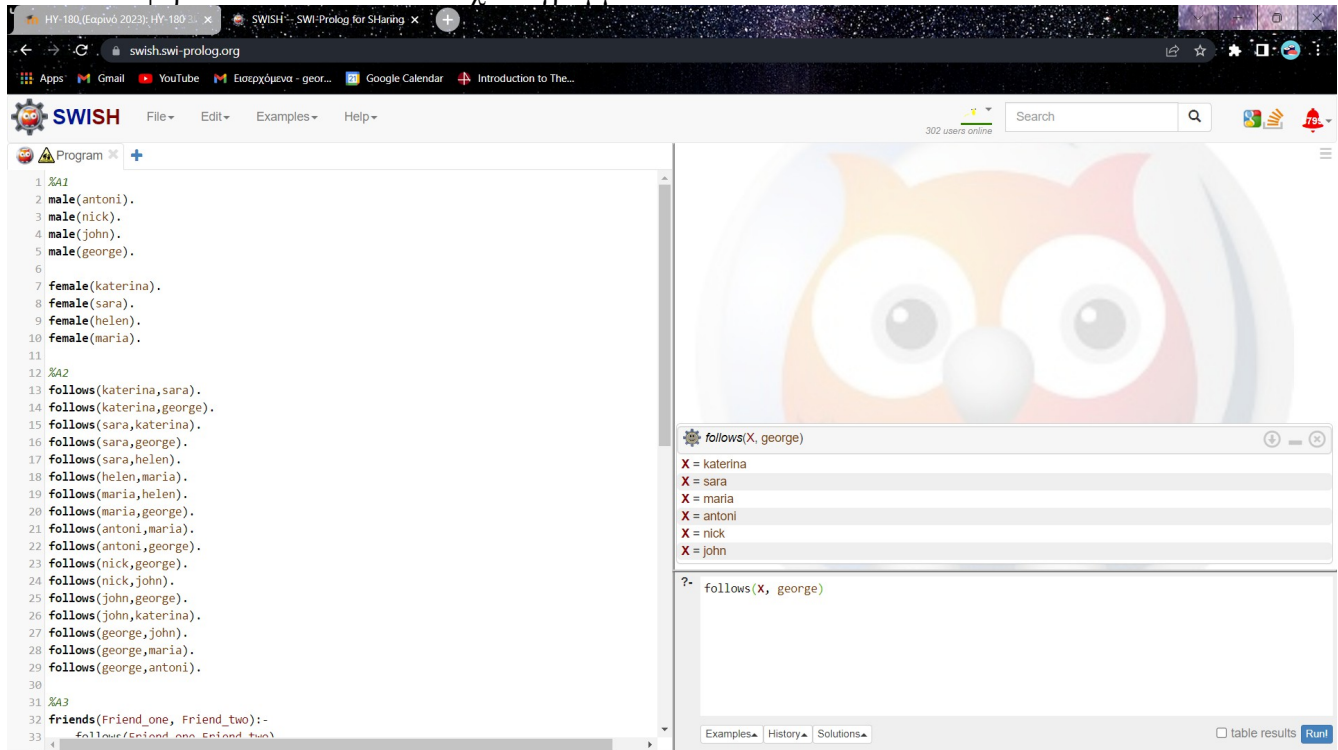


## Ασκηση 1

B.

1. θα ρωτήσουμε `follows(X, george)` και θα πάρουμε όλες τις τιμές που μπορεί να πάρει το `X` αρα ποιο ακολουθούν τον `george`. Αρα περνούμε `X= katerina,sara,maria,antoni,nick,john`. Που επιβεβαιώνεται και απο το σχεδιαγραμμα.



The screenshot shows the SWISH Prolog IDE interface. The main editor displays a Prolog program with the following code:

```
1 %A1
2 male(antoni).
3 male(nick).
4 male(john).
5 male(george).
6
7 female(katerina).
8 female(sara).
9 female(helen).
10 female(maria).
11
12 %A2
13 follows(katerina,sara).
14 follows(katerina,george).
15 follows(sara,katerina).
16 follows(sara,george).
17 follows(sara,helen).
18 follows(helen,maria).
19 follows(maria,helen).
20 follows(maria,george).
21 follows(antoni,maria).
22 follows(antoni,george).
23 follows(nick,george).
24 follows(nick,john).
25 follows(john,george).
26 follows(john,katerina).
27 follows(george,john).
28 follows(george,maria).
29 follows(george,antoni).
30
31 %A3
32 friends(Friend_one, Friend_two):-
33     follow(Friend_one, Friend_two)
```

The right-hand pane shows the execution results for the query `follows(X, george)`. The results are listed as follows:

- X = katerina
- X = sara
- X = maria
- X = antoni
- X = nick
- X = john

The bottom of the right-hand pane shows the query `follows(X, george)` and a `Run!` button.

2. Θα ρωτήσουμε όπως στο προηγούμεν  $\text{follows}(X, \text{maria})$  για να βρούμε ποιο την ακολουθούν και, (και)  $\text{male}(X)$  ωτε να παρουμε μονο τα  $X$  που ειναι αντρες.αρα σηνολοκα  $\text{follows}(X, \text{maria}), \text{male}(X)$ . Και περνουμε  $\text{antoni}, \text{george}$  που επιβεβαιωνεται και απο το σχεδιαγραμμα.

The screenshot shows the SWISH Prolog IDE interface. The left pane contains a Prolog program with the following code:

```
1 %A1
2 male(antoni).
3 male(nick).
4 male(john).
5 male(george).
6
7 female(katerina).
8 female(sara).
9 female(helen).
10 female(maria).
11
12 %A2
13 follows(katerina,sara).
14 follows(katerina,george).
15 follows(sara,katerina).
16 follows(sara,george).
17 follows(sara,helen).
18 follows(helen,maria).
19 follows(maria,helen).
20 follows(maria,george).
21 follows(antoni,maria).
22 follows(antoni,george).
23 follows(nick,george).
24 follows(nick,john).
25 follows(john,george).
26 follows(john,katerina).
27 follows(george,john).
28 follows(george,maria).
29 follows(george,antoni).
30
31 %A3
32 friends(Friend_one, Friend_two):-
33     follow(Friend_one, Friend_two)
```

The right pane shows the execution results for the query  $\text{follows}(X, \text{maria}), \text{male}(X)$ . The results are:

```
X = antoni
X = george
```

Below the results, there is a prompt  $?- \text{follows}(X, \text{maria}), \text{male}(X)$  and a 'Run!' button.

3. Για να βρούμε τα ζευγεί φίλων θα παρουμε το friends/2 και θα του περασουμε δυο μεταβλητες εστω X, Y. Ετρι θα μας επιστρεψει ολους τους συνδιασμων X,Y οπου ισχυει η σχεση friends. (θα παρουμε δυο φορες το καθε ζευγαρι αφου και τα δυο ονοματα θα ανατεθουν και στην δυο μεταβλητες πχ θα παρουμε το x=friend1, y=friend2 και το x= friend2, y=friend1). Και περνουμε (katerina,sara), (helen,maria),(maria,george),(antoni,george),(josh,george),και τα αντιστριφα. Που επιβεβαιωνεται και απο το σχεδιαγραμμα.

The screenshot shows the SWISH Prolog IDE interface. The left pane contains a Prolog program with the following code:

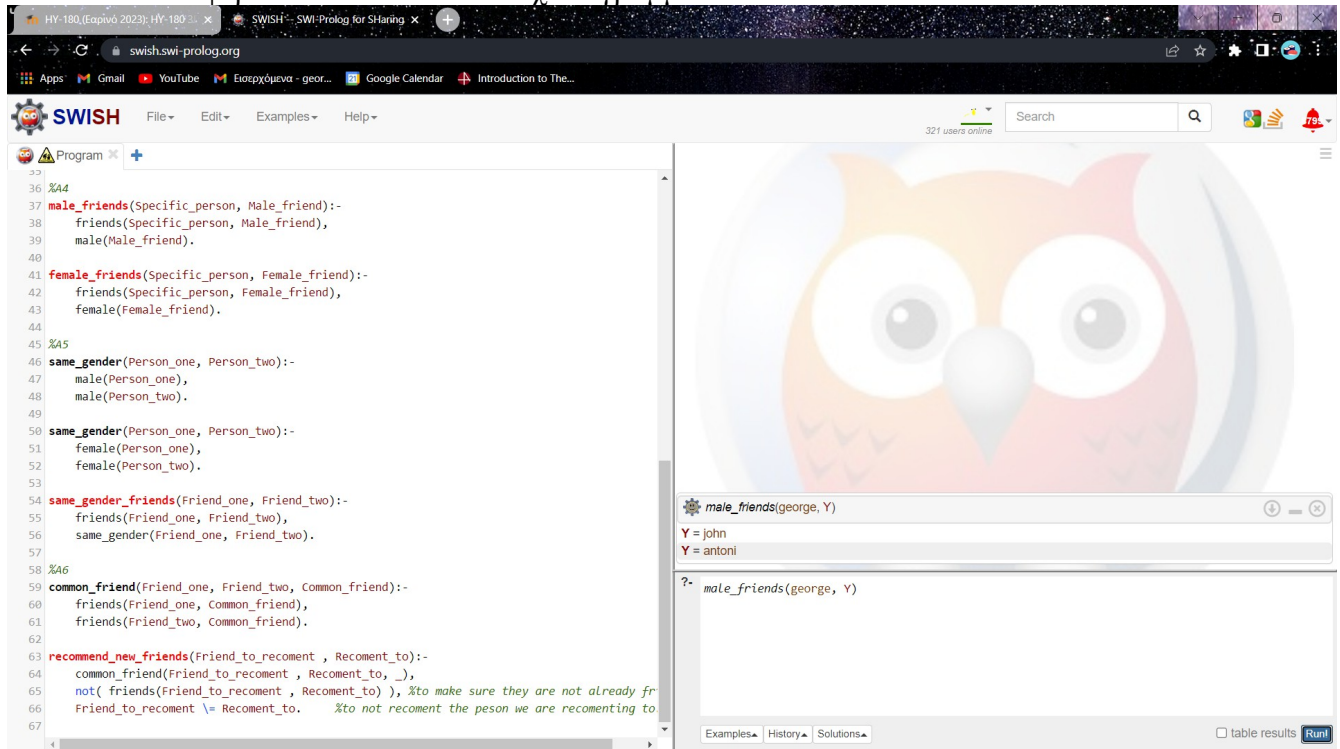
```
36 %A4
37 male_friends(Specific_person, Male_friend):-
38     friends(Specific_person, Male_friend),
39     male(Male_friend).
40
41 female_friends(Specific_person, Female_friend):-
42     friends(Specific_person, Female_friend),
43     female(Female_friend).
44
45 %A5
46 same_gender(Person_one, Person_two):-
47     male(Person_one),
48     male(Person_two).
49
50 same_gender(Person_one, Person_two):-
51     female(Person_one),
52     female(Person_two).
53
54 same_gender_friends(Friend_one, Friend_two):-
55     friends(Friend_one, Friend_two),
56     same_gender(Friend_one, Friend_two).
57
58 %A6
59 common_friend(Friend_one, Friend_two, Common_friend):-
60     friends(Friend_one, Common_friend),
61     friends(Friend_two, Common_friend).
62
63 recommend_new_friends(Friend_to_recoment , Recoment_to):-
64     common_friend(Friend_to_recoment , Recoment_to, _),
65     not( friends(Friend_to_recoment , Recoment_to) ), %to make sure they are not already fr
66     Friend_to_recoment \= Recoment_to. %to not recoment the peson we are recomenting to.
67
```

The right pane shows the results of the query `friends(X, Y)`. The results are listed in a table:

X	Y
katerina	sara
sara	katerina
helen	maria
maria	helen
maria	george
george	maria
antoni	george
george	antoni
john	george
george	john
george	maria
maria	george
george	antoni
antoni	george

At the bottom of the right pane, there is a section for the query `?- friends(X, Y)` with tabs for Examples, History, and Solutions. A "Run!" button is visible at the bottom right.

4. Για να βρούμε τους ανδρες φίλους του george θα χρησιμοποιήσουμε την `male_friends/2` και σαν το πρώτο κατηγορημα δηνουμε το george και σαν δευτερο μια μεταβλητη  $\pi\chi$   $Y$  αρα περουμε στα τα  $Y$  για τα οποια ο george ειναι `male_friend`. Και περνοθμε  $Y=josh,antoni$  Που επιφειβαιωνεται και απο το σχεδιαγραμμα.



The screenshot shows the SWISH Prolog IDE interface. The left pane displays a Prolog program with the following code:

```
36 %A4
37 male_friends(Specific_person, Male_friend):-
38     friends(Specific_person, Male_friend),
39     male(Male_friend).
40
41 female_friends(Specific_person, Female_friend):-
42     friends(Specific_person, Female_friend),
43     female(Female_friend).
44
45 %A5
46 same_gender(Person_one, Person_two):-
47     male(Person_one),
48     male(Person_two).
49
50 same_gender(Person_one, Person_two):-
51     female(Person_one),
52     female(Person_two).
53
54 same_gender_friends(Friend_one, Friend_two):-
55     friends(Friend_one, Friend_two),
56     same_gender(Friend_one, Friend_two).
57
58 %A6
59 common_friend(Friend_one, Friend_two, Common_friend):-
60     friends(Friend_one, Common_friend),
61     friends(Friend_two, Common_friend).
62
63 recommend_new_friends(Friend_to_recoment , Recoment_to):-
64     common_friend(Friend_to_recoment , Recoment_to, _),
65     not( friends(Friend_to_recoment , Recoment_to) ), %to make sure they are not already fr
66     Friend_to_recoment \= Recoment_to. %to not recoment the peson we are recomenting to
67
```

The right pane shows the execution results for the query `male_friends(george, Y)`. The results are:

```
Y = john
Y = antoni
```

Below the results, there is a section for the query `?- male_friends(george, Y)` with a "table results" checkbox and a "Run" button.

5. θα χρησιμοποιησουμε την `same_gender_friends/2` και οπως το 3 θα δωσουμε δυο μεταβλητες για να παρουμε ολους τους συνδιασμους που ισχυει η σχεση. Οπως το 3 επισεις θα παρουμε δυο σχεσεις για καθε ζευγαρι για τον ιδιο λογο.  
Και περνουμε (katerina,sara), (helen, maria), (antoni,george), (josh,george) , και τα αντιστριφα  
Που επιβεβαιωνεται και απο το σχεδιαγραμμα.

The screenshot shows the SWISH Prolog IDE interface. The left pane contains a Prolog program with the following code:

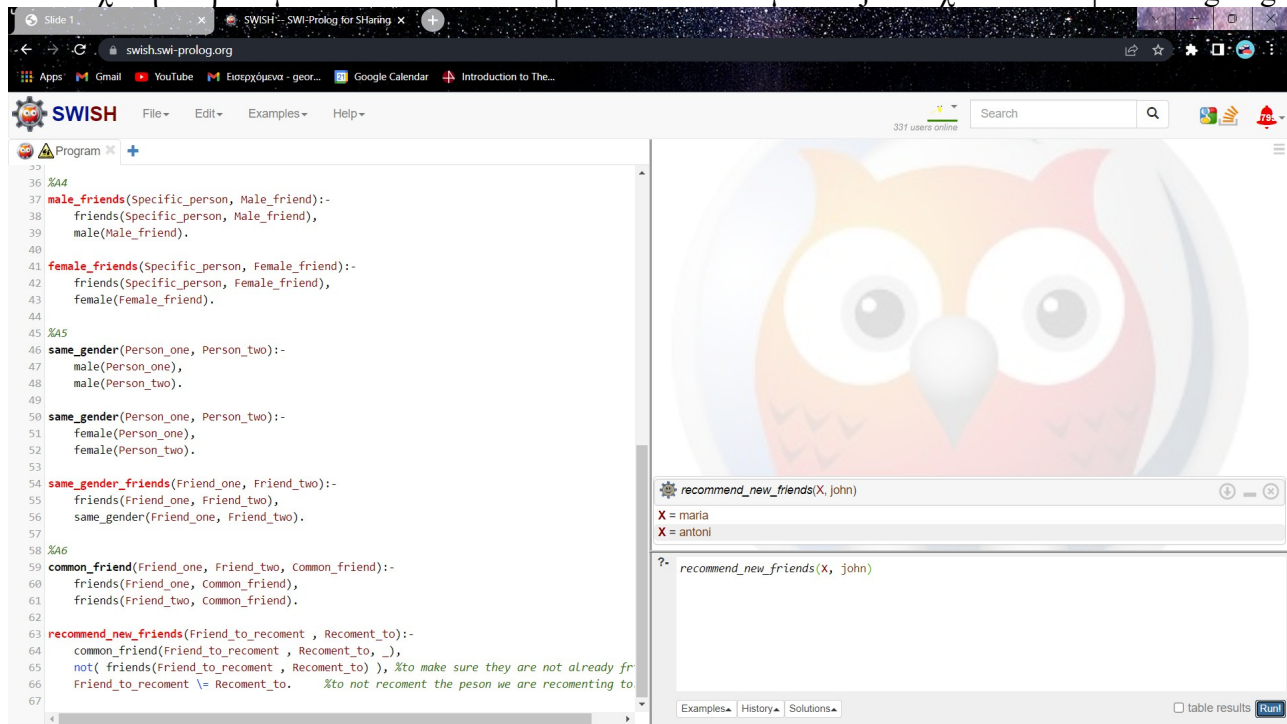
```
33
34 %44
35 male_friends(Specific_person, Male_friend):-
36   friends(Specific_person, Male_friend),
37   male(Male_friend).
38
39 %45
40 female_friends(Specific_person, Female_friend):-
41   friends(Specific_person, Female_friend),
42   female(Female_friend).
43
44 %46
45 same_gender(Person_one, Person_two):-
46   male(Person_one),
47   male(Person_two).
48
49 %47
50 same_gender(Person_one, Person_two):-
51   female(Person_one),
52   female(Person_two).
53
54 %48
55 same_gender_friends(Friend_one, Friend_two):-
56   friends(Friend_one, Friend_two),
57   same_gender(Friend_one, Friend_two).
58
59 %49
60 common_friend(Friend_one, Friend_two, Common_friend):-
61   friends(Friend_one, Common_friend),
62   friends(Friend_two, Common_friend).
63
64 %50
65 recommend_new_friends(Friend_to_recoment , Recoment_to):-
66   common_friend(Friend_to_recoment , Recoment_to, _),
67   not( friends(Friend_to_recoment , Recoment_to) ), %to make sure they are not already fr
68   Friend_to_recoment \= Recoment_to. %to not recoment the peson we are recomenting to.
```

The right pane shows the execution results for the query `same_gender_friends(X,Y)`. The results are displayed in a table format:

X	Y
katerina	sara
sara	katerina
helen	maria
maria	helen
antoni	george
george	antoni
josh	george
george	josh

Below the table, there is a section for the query `?- same_gender_friends(X,Y)` with a search bar and a `Run!` button.

6. θα χρησιμοποιήσουμε την `recommend_new_friends/2` και σαν ένα κατηγοριμα θα δώσουμε τον `john` και σαν το δεύτερο μια μεταβλητή εστώ `X` ώστε να πάρουμε όλα τα `X` για τα οποία ισχύει η σχέση. Περνούμε `maria` και `antoni` αφού αυτοί οι δύο με τον `john` έχουν κοινό φίλο τον `george`.



The screenshot shows the SWISH Prolog IDE interface. The left pane contains a Prolog program with the following code:

```
33
34 %44
35 male_friends(Specific_person, Male_friend):-
36   friends(Specific_person, Male_friend),
37   male(Male_friend).
38
39 %45
40 female_friends(Specific_person, Female_friend):-
41   friends(Specific_person, Female_friend),
42   female(Female_friend).
43
44
45 same_gender(Person_one, Person_two):-
46   male(Person_one),
47   male(Person_two).
48
49
50 same_gender(Person_one, Person_two):-
51   female(Person_one),
52   female(Person_two).
53
54 same_gender_friends(Friend_one, Friend_two):-
55   friends(Friend_one, Friend_two),
56   same_gender(Friend_one, Friend_two).
57
58 %46
59 common_friend(Friend_one, Friend_two, Common_friend):-
60   friends(Friend_one, Common_friend),
61   friends(Friend_two, Common_friend).
62
63 recommend_new_friends(Friend_to_recoment , Recoment_to):-
64   common_friend(Friend_to_recoment , Recoment_to, _),
65   not( friends(Friend_to_recoment , Recoment_to) ), %to make sure they are not already fr
66   Friend_to_recoment \= Recoment_to. %to not recoment the peson we are recomenting to.
67
```

The right pane shows the execution of the query `recommend_new_friends(X, john)`. The results are:

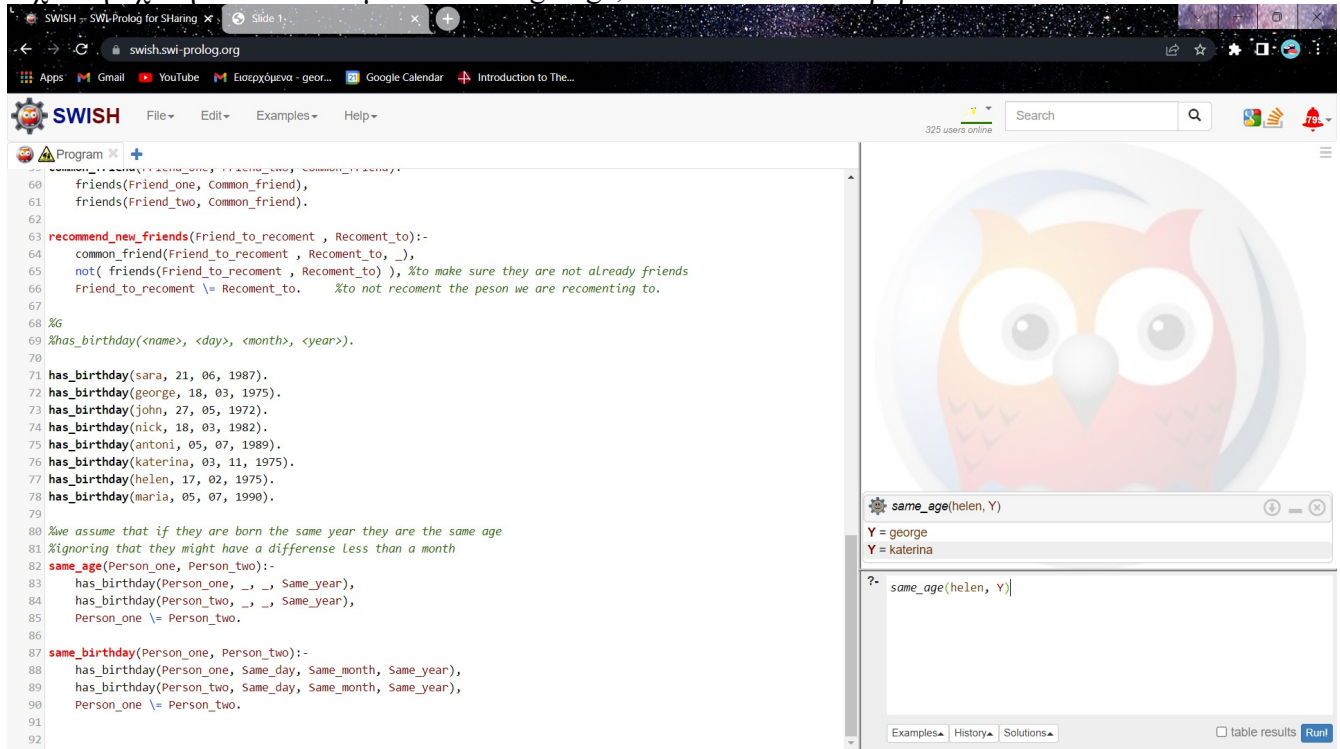
```
X = maria
X = antoni
```

Below the results, there is a search bar with the query `? recommend_new_friends(X, john)` and buttons for Examples, History, Solutions, and a Run button.



Γ

1. we will use the `same_age/2` and ask `same_age(helen, Y)` ώστε να παρούμε όλα τα  $Y$  για τα οποία ισχυει η σχεση το αποτελεσμα ειναι  $Y = \text{george, katerina}$ . Οπου επιβεβαιωνεται και απο τον πινακα.



The screenshot displays the SWISH Prolog IDE interface. The left pane shows a Prolog program with the following code:

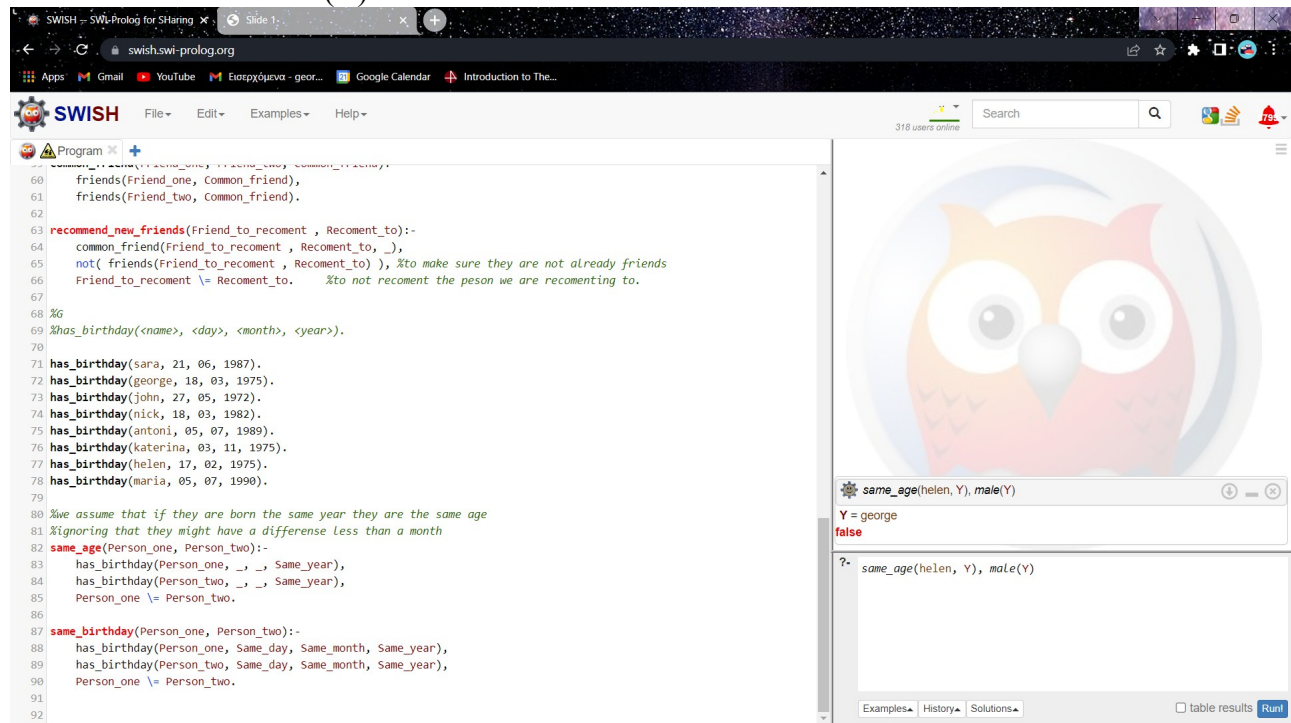
```
60 friends(Friend_one, Common_friend),
61 friends(Friend_two, Common_friend).
62
63 recommend_new_friends(Friend_to_recoment , Recoment_to):-
64 common_friend(Friend_to_recoment , Recoment_to, _),
65 not( friends(Friend_to_recoment , Recoment_to) ), %to make sure they are not already friends
66 Friend_to_recoment \= Recoment_to. %to not recoment the peson we are recomenting to.
67
68 %G
69 %has_birthday(<name>, <day>, <month>, <year>).
70
71 has_birthday(sara, 21, 06, 1987).
72 has_birthday(george, 18, 03, 1975).
73 has_birthday(john, 27, 05, 1972).
74 has_birthday(nick, 18, 03, 1982).
75 has_birthday(antoni, 05, 07, 1989).
76 has_birthday(katerina, 03, 11, 1975).
77 has_birthday(helen, 17, 02, 1975).
78 has_birthday(maria, 05, 07, 1990).
79
80 %we assume that if they are born the same year they are the same age
81 %ignoring that they might have a differense less than a month
82 same_age(Person_one, Person_two):-
83 has_birthday(Person_one, _, _, Same_year),
84 has_birthday(Person_two, _, _, Same_year),
85 Person_one \= Person_two.
86
87 same_birthday(Person_one, Person_two):-
88 has_birthday(Person_one, Same_day, Same_month, Same_year),
89 has_birthday(Person_two, Same_day, Same_month, Same_year),
90 Person_one \= Person_two.
91
92
```

The right pane shows the execution results for the query `same_age(helen, Y)`. The results are:

```
Y = george
Y = katerina
```

Below the results, there is a text input field containing the query `?- same_age(helen, Y)|`. At the bottom right, there are buttons for "Examples", "History", "Solutions", and a "Run!" button.

2. Για να βρούμε αυτούς που έχουν την ίδια ηλικία και είναι και άντρες θα κάνουμε ακριβώς την ίδια ερώτηση με το 1 απλά θα βάλουμε σαν extra συνθήκη και το Y να είναι male  
δ.λ.δ. `same_age(helen, Y), male(Y)`  
βελπθμε οτι σαν αποτελεσμα περνουμε μονο τον george δλδ φιλτραραμε την katerina ποθ δεν ικανοποιει το `male(Y)`



The screenshot shows the SWISH Prolog IDE interface. The left pane displays a Prolog program with the following code:

```
60 friends(Friend_one, Common_friend),
61 friends(Friend_two, Common_friend).
62
63 recommend_new_friends(Friend_to_recoment , Recoment_to):-
64   common_friend(Friend_to_recoment , Recoment_to, _),
65   not( friends(Friend_to_recoment , Recoment_to) ), %to make sure they are not already friends
66   Friend_to_recoment \= Recoment_to. %to not recoment the peson we are recomenting to.
67
68 %G
69 %has_birthday(<name>, <day>, <month>, <year>).
70
71 has_birthday(sara, 21, 06, 1987).
72 has_birthday(george, 18, 03, 1975).
73 has_birthday(john, 27, 05, 1972).
74 has_birthday(nick, 18, 03, 1982).
75 has_birthday(antoni, 05, 07, 1989).
76 has_birthday(katerina, 03, 11, 1975).
77 has_birthday(helen, 17, 02, 1975).
78 has_birthday(maria, 05, 07, 1990).
79
80 %we assume that if they are born the same year they are the same age
81 %ignoring that they might have a differense less than a month
82 same_age(Person_one, Person_two):-
83   has_birthday(Person_one, _, _, Same_year),
84   has_birthday(Person_two, _, _, Same_year),
85   Person_one \= Person_two.
86
87 same_birthday(Person_one, Person_two):-
88   has_birthday(Person_one, Same_day, Same_month, Same_year),
89   has_birthday(Person_two, Same_day, Same_month, Same_year),
90   Person_one \= Person_two.
91
92
```

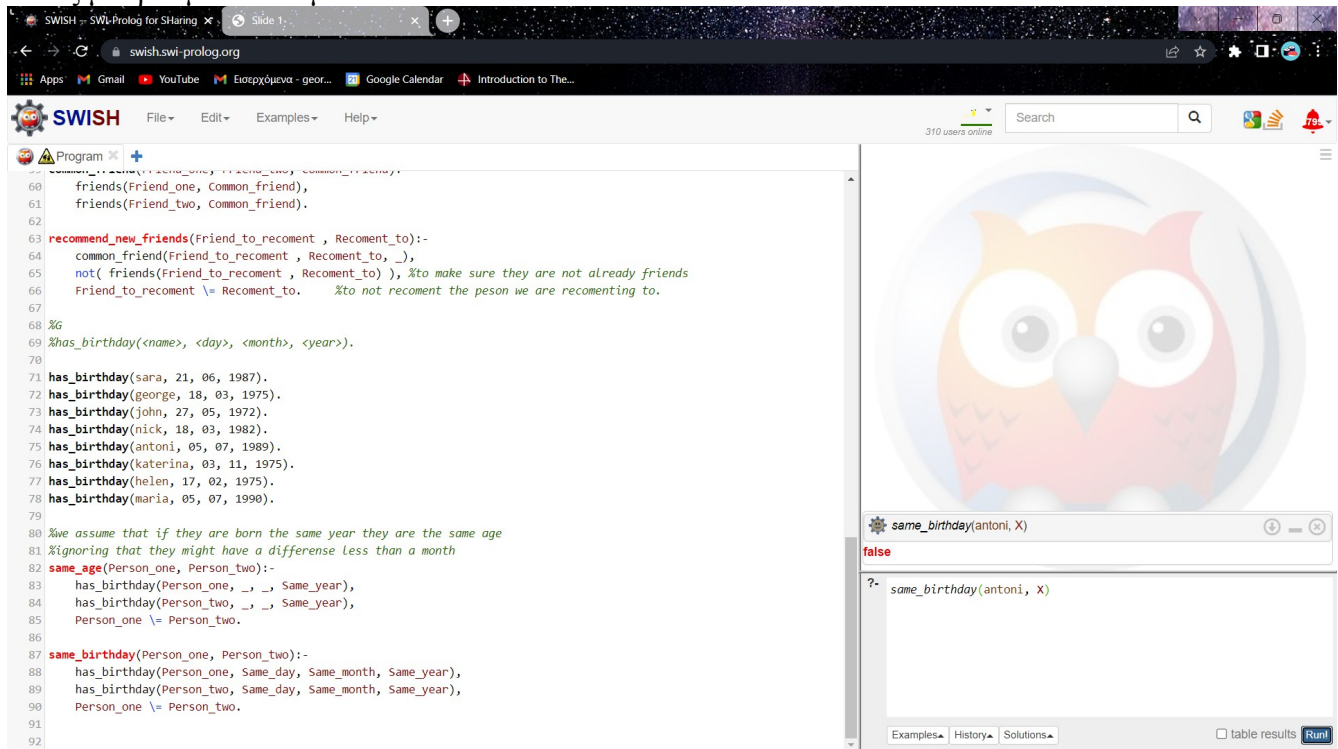
The right pane shows the execution results for the query `same_age(helen, Y), male(Y)`. The results are:

```
Y = george
false
```

Below the results, there is a section for the query `?- same_age(helen, Y), male(Y)` with tabs for Examples, History, and Solutions. At the bottom right, there is a checkbox for "table results" and a "Run!" button.



3.θα χρησιμοποιήσουμε την `same_birthday/2` περνώντας τον `antoni` και μια μεταβλητή εστώ `Y` για να δούμε ποιες τιμές του το ικανοποιούν. Παρατηρούμε ότι το αποτέλεσμα είναι `an false` που σημαίνει ότι δεν βρέθηκε κάποιος `Y` που να ικανοποιεί την σχέση άρα ο `antoni` δεν έχει τα ίδια γενέθλια με κανέναν. Οπως μπορούμε να δούμε και από τον πίνακα.



The screenshot shows the SWISH Prolog IDE interface. The left pane displays a Prolog program with the following code:

```
60 friends(Friend_one, Common_friend),
61 friends(Friend_two, Common_friend).
62
63 recommend_new_friends(Friend_to_recoment , Recoment_to):-
64     common_friend(Friend_to_recoment , Recoment_to, _),
65     not( friends(Friend_to_recoment , Recoment_to) ), %to make sure they are not already friends
66     Friend_to_recoment \= Recoment_to. %to not recoment the peson we are recomenting to.
67
68 %G
69 %has_birthday(<name>, <day>, <month>, <year>).
70
71 has_birthday(sara, 21, 06, 1987).
72 has_birthday(george, 18, 03, 1975).
73 has_birthday(john, 27, 05, 1972).
74 has_birthday(nick, 18, 03, 1982).
75 has_birthday(antoni, 05, 07, 1989).
76 has_birthday(katerina, 03, 11, 1975).
77 has_birthday(helen, 17, 02, 1975).
78 has_birthday(maria, 05, 07, 1990).
79
80 %we assume that if they are born the same year they are the same age
81 %ignoring that they might have a differense less than a month
82 same_age(Person_one, Person_two):-
83     has_birthday(Person_one, _ , _ , Same_year),
84     has_birthday(Person_two, _ , _ , Same_year),
85     Person_one \= Person_two.
86
87 same_birthday(Person_one, Person_two):-
88     has_birthday(Person_one, Same_day, Same_month, Same_year),
89     has_birthday(Person_two, Same_day, Same_month, Same_year),
90     Person_one \= Person_two.
91
92
```

The right pane shows the execution results for the query `same_birthday(antoni, X)`. The result is `false`, indicating that no person with the same birthday as `antoni` was found. Below the result, the query `?- same_birthday(antoni, X)` is shown. The interface also includes a search bar, a user count (310 users online), and buttons for Examples, History, Solutions, and a Run button.

4.Μπορούμε είτε να δημιουργήσουμε ένα κενουργίο κανονα που θα βρισκει αν εχουν τα ιδια γενεθεια πλιν του ετους :

```
same_birth_day_month(Person_one, Person_two):-  
    has_birthday(Person_one, Same_day, Same_month, _),  
    has_birthday(Person_two, Same_day, Same_month, _),  
    Person_one \= Person_two.
```

Ειτε θα κανουμε αυτους τους ελεγχους στην ερωτηση καθευθειαν

```
has_birthday(X, 18, 03, _),  
has_birthday(Y, 18, 03, _),  
X \= Y.
```

Ετσι βρισκουμε ενα X και ενα Y οπου εχουν γενιθει την ιδια μερα και μηνα ανεξαρτητα απο τον χρονο ( \_ ) και διαβεβαιωνουμε οτι δεν ειναι ιδιοι.

Σαν αποτελεσμα περνουμε μονο το (george, nick) οπου επιβεβαιωνεται και απο τον πινακα και προφανος και το αντιστροφο.

The screenshot displays the SWISH Prolog IDE. The main editor on the left contains the following Prolog code:

```
60 friends(Friend_one, Common_friend),  
61 friends(Friend_two, Common_friend).  
62  
63 recommend_new_friends(Friend_to_recoment , Recoment_to):-  
64     common_friend(Friend_to_recoment , Recoment_to, _),  
65     not( friends(Friend_to_recoment , Recoment_to) ), %to make sure they are not already friends  
66     Friend_to_recoment \= Recoment_to. %to not recoment the peson we are recomenting to.  
67  
68 %G  
69 %has_birthday(<name>, <day>, <month>, <year>).  
70  
71 has_birthday(sara, 21, 06, 1987).  
72 has_birthday(george, 18, 03, 1975).  
73 has_birthday(john, 27, 05, 1972).  
74 has_birthday(nick, 18, 03, 1982).  
75 has_birthday(antoni, 05, 07, 1989).  
76 has_birthday(katerina, 03, 11, 1975).  
77 has_birthday(helen, 17, 02, 1975).  
78 has_birthday(maria, 05, 07, 1990).  
79  
80 %we assume that if they are born the same year they are the same age  
81 %ignoring that they might have a differense less than a month  
82 same_age(Person_one, Person_two):-  
83     has_birthday(Person_one, _, _, Same_year),  
84     has_birthday(Person_two, _, _, Same_year),  
85     Person_one \= Person_two.  
86  
87 same_birthday(Person_one, Person_two):-  
88     has_birthday(Person_one, Same_day, Same_month, Same_year),  
89     has_birthday(Person_two, Same_day, Same_month, Same_year),  
90     Person_one \= Person_two.  
91  
92
```

The right pane features a large owl logo. Below it, a console window shows the query and results:

```
has_birthday(X, 18, 03, _), has_birthday(Y, 18, 03, _), X \= Y.  
X = george,  
Y = nick  
X = nick,  
Y = george
```

Below the console, there is a section for the query with the following text:

```
?- has_birthday(X, 18, 03, _),  
   has_birthday(Y, 18, 03, _),  
   X \= Y.
```

At the bottom right, there are buttons for 'Examples', 'History', 'Solutions', 'table results', and 'Run!'.

## Ασκηση 2.

SWISH - SWI-Prolog for SHaring

HY180\_assignment2B\_2023.pdf | Course: HY-180 Λογική (Εαρινό) | Slide 1

swish.swi-prolog.org

Apps | Gmail | YouTube | Εισαγωγή - geor... | Google Calendar | Introduction to The... | Welcome To Colab...

SWISH File Edit Examples Help

365 users online

Search

Program

```
82 %we assume that if they are born the same year they are the same age
83 %ignoring that they might have a difference less than a month
84 same_age(Person_one, Person_two):-
85     has_birthday(Person_one, _, _, Same_year),
86     has_birthday(Person_two, _, _, Same_year),
87     Person_one \= Person_two.
88
89 same_birthday(Person_one, Person_two):-
90     has_birthday(Person_one, Same_day, Same_month, Same_year),
91     has_birthday(Person_two, Same_day, Same_month, Same_year),
92     Person_one \= Person_two.
93
94
95 %2.
96
97 is_even(Number):-
98     Number mod 2 =:= 0.
99
100 is_odd(Number):-
101     not( is_even(Number) ).
102
103
104 sum_odd([], 0).                %base condition if its empty the sum of its odd is 0
105
106 sum_odd([Head|Rest], Sum):-
107     sum_odd(Rest, Sum_of_odds_in_Rest), %recursively call the sum_odd/2 without the head
108     (is_odd(Head) ->
109         Sum is Head + Sum_of_odds_in_Rest %if head is odd add head and result of recursion
110     ;
111         Sum is Sum_of_odds_in_Rest      %else add only the result of the recursion
112     ).
113
```

sum\_odd([2,5,4,3,1], X)

X = 9

?- sum\_odd([2,5,4,3,1], X)

Examples History Solutions

table results Run

### Ασκηση 3.

Ενα παραδειγμα θα ηταν `add_at_the_end([4, 5, 6], [1, 2, 3], Result)`.

κανουμε appent to  $X = [4, 5, 6]$ , στην λιστα  $L = [1, 2, 3]$  και το αποτελεσμα ειναι  $R = [1, 2, 3, 4, 5, 6]$ .

The screenshot shows the SWISH Prolog IDE interface. The left pane contains a Prolog program with the following code:

```
95 %2.
96
97 is_even(Number):-
98     Number mod 2 =:= 0.
99
100 is_odd(Number):-
101     not( is_even(Number) ).
102
103
104 sum_odd([], 0).                %base condition if its empty the sum of its odd is 0
105
106 sum_odd([Head|Rest], Sum):-
107     sum_odd(Rest, sum_of_odds_in_Rest), %recursively call the sum_odd/2 without the head
108     (is_odd(Head) ->
109         Sum is Head + Sum_of_odds_in_Rest %if head is odd add head and result of recursion
110     ;
111         Sum is Sum_of_odds_in_Rest %else add only the result of the recursion
112     ).
113
114 %3.
115
116 add_at_the_end(X, [], X). %base case, appenting an element to an empty list results to the element
117
118 %basically adding the the elements of the second list (L) one by one
119 %in the third list (R) untill L is epty end the base case is trigered apenting the X after the L
120 %in the R
121
122 add_at_the_end(X, [Element_to_appent|L_rest], [Element_to_appent|R_rest]) :-
123     add_at_the_end(X, L_rest, R_rest).
124
125
126
127
```

The right pane shows the execution of the query `add_at_the_end([4, 5, 6], [1, 2, 3], Result).` with the result `Result = [1, 2, 3, 4, 5, 6]`. Below the result, there are buttons for 'Next', '10', '100', '1,000', and 'Stop'. At the bottom, there is a text input field with the query `?- add_at_the_end([4, 5, 6], [1, 2, 3], Result).` and a 'Run!' button.

Επισειξ οντως βλεπουμε οτι αν βαλουμε αυτο το αποτελεσμα σαν Result οντως η συνθηκη επιτυγχανει.

The screenshot shows the SWISH Prolog IDE interface. The top bar contains the query `add_at_the_end([4, 5, 6], [1, 2, 3], [1, 2, 3, 4, 5, 6]).` with buttons for 'true', 'Next', '10', '100', '1,000', and 'Stop'. Below the query, there is a text input field with the query `?- add_at_the_end([4, 5, 6], [1, 2, 3], [1, 2, 3, 4, 5, 6]).` and a 'Run!' button.