ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ
UNIVERSITY OF PATRAS

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ Η/Υ ΚΑΙ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΤΕΧΝΗΤΗ ΝΟΗΜΟΣΥΝΗ

Ακαδημαϊκό Έτος 2022-2023

ΤΕΧΝΙΚΗ ΑΝΑΦΟΡΑ

3η ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ

< ΠΡΟΒΛΗΜΑ ΤΡΙΩΝ ΔΟΧΕΊΩΝ ΝΕΡΟΎ>

<ΔΑΜΙΑΝΟΣ ΔΙΑΣΑΚΟΣ>
Α.Μ.: <1084632>

Πάτρα 2022-23

Πρόβλημα τριών δοχείων νερού.

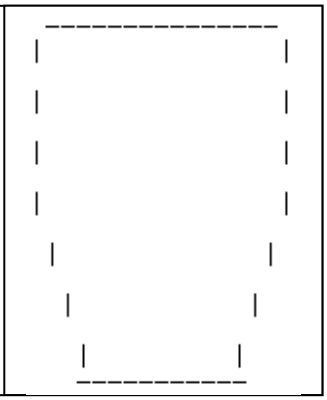
α)Τα κατηγορήματα για το πρόβλημα είναι τα εξής:

```
:-dynamic (doxeio_a)/3.
:-dynamic (doxeio_b)/3.
:-dynamic (doxeio_c)/3.

doxeio_a(0,7,yes).
doxeio_a(1,7,data).
doxeio_a(2,7,data).
doxeio_a(3,7,data).
doxeio_a(4,7,data).
doxeio_a(5,7,data).
doxeio_a(6,7,data).
doxeio_a(7,7,data).

doxeio_b(0,4,yes).
doxeio_b(1,4,data).
doxeio_b(2,4,data).
doxeio_b(3,4,data).
doxeio_b(4,4,data).

doxeio_c(0,2,yes).
doxeio_c(1,2,data).
doxeio_c(2,2,data).
```

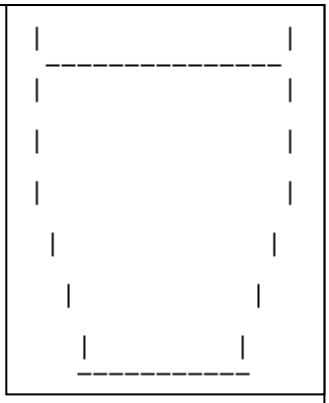
Για κάθε δοχείο έχει οριστεί η κάθε κατάσταση που μπορεί να βρίσκεται. Το δοχείο a μπορεί να γεμίσει με νερό από 0 έως 7, το δοχείο b από 0 έως 4 και το c από 0 έως 2.

β)Όσον αφορά τον τρόπο απεικόνισης έχουν δημιουργηθεί 16 συναρτήσεις, όσες και όλες οι καταστάσεις, που θα εκτυπώνουν το συγκεκριμένο ποτήρι, στο συγκεκριμένο επίπεδο :

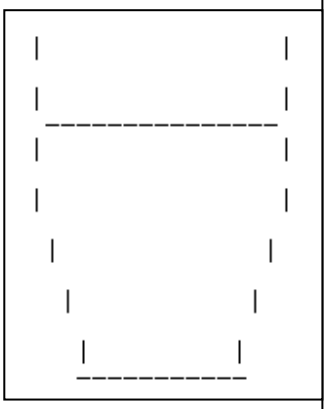
```
make_line() :- write(' ----------').
make_a7() :- make_line(),
        nl, write('|'), tab(15), write('|'),nl,
        nl, write('|'), tab(15), write('|'),nl,
        nl, write('|'), tab(15), write('|'),nl,
        nl, write('|'), tab(15), write('|'),nl,
        nl, write(' |'), tab(13), write('|'),nl,
        nl, write('  |'), tab(11), write('|'),nl,
        nl, write('   |'), tab(9), write('|'),nl,
        tab(3),write('-----------').
```



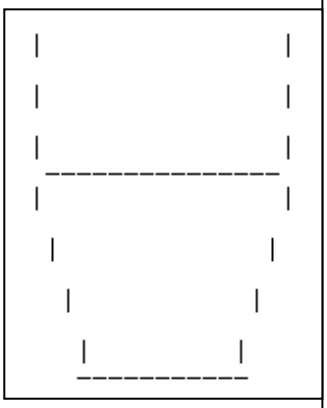
```
make_a6() :-
        nl, write('|'), tab(15), write('|'),nl,
        make_line(),
        nl, write('|'), tab(15), write('|'),nl,
        nl, write('|'), tab(15), write('|'),nl,
        nl, write('|'), tab(15), write('|'),nl,
        nl, write(' |'), tab(13), write('|'),nl,
        nl, write('  |'), tab(11), write('|'),nl,
        nl, write('   |'), tab(9), write('|'),nl,
        tab(3),write('-----------').
```



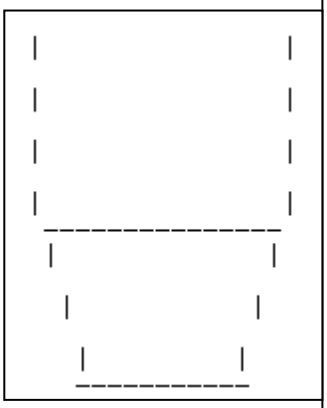
```
make_a5() :-
        nl, write('|'), tab(15), write('|'),nl,
        nl, write('|'), tab(15), write('|'),nl,
        make_line(),
        nl, write('|'), tab(15), write('|'),nl,
        nl, write('|'), tab(15), write('|'),nl,
        nl, write(' |'), tab(13), write('|'),nl,
        nl, write('  |'), tab(11), write('|'),nl,
        nl, write('   |'), tab(9), write('|'),nl,
        tab(3),write('-----------').
```



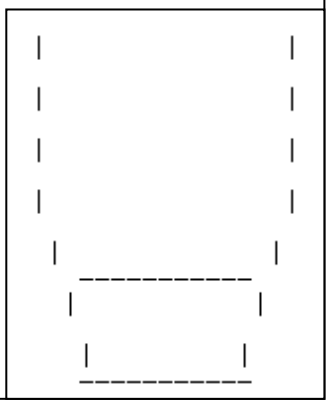
```
make_a4() :-
        nl, write('|'), tab(15), write('|'),nl,
        nl, write('|'), tab(15), write('|'),nl,
        nl, write('|'), tab(15), write('|'),nl,
        make_line(),
        nl, write('|'), tab(15), write('|'),nl,
        nl, write(' |'), tab(13), write('|'),nl,
        nl, write('  |'), tab(11), write('|'),nl,
        nl, write('   |'), tab(9), write('|'),nl,
        tab(3),write('-----------').
```



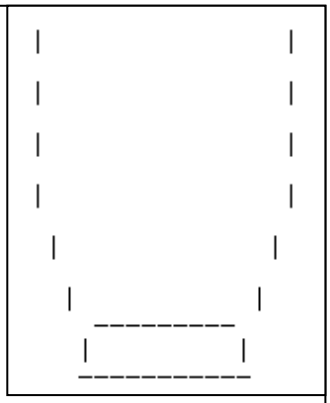
```
make_a3() :-
        nl, write('|'), tab(15), write('|'),nl,
        nl, write('|'), tab(15), write('|'),nl,
        nl, write('|'), tab(15), write('|'),nl,
        nl, write('|'), tab(15), write('|'),nl,
        make_line(),
        nl, write(' |'), tab(13), write('|'),nl,
        nl, write('  |'), tab(11), write('|'),nl,
        nl, write('   |'), tab(9), write('|'),nl,
        tab(3),write('-----------').
```



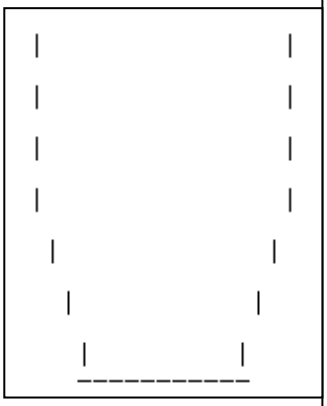
```
make_a2() :-
        nl, write('|'), tab(15), write('|'),nl,
        nl, write('|'), tab(15), write('|'),nl,
        nl, write('|'), tab(15), write('|'),nl,
        nl, write('|'), tab(15), write('|'),nl,
        nl, write(' |'), tab(13), write('|'),nl,
        tab(3),write('-----------'),
        nl, write('  |'), tab(11), write('|'),nl,
        nl, write('   |'), tab(9), write('|'),nl,
        tab(3),write('-----------').
```



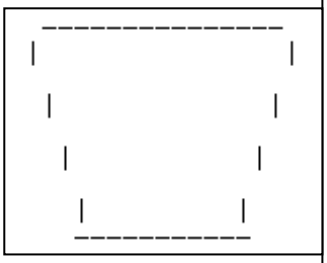
```
make_a1() :-
        nl, write('|'), tab(15), write('|'),nl,
        nl, write('|'), tab(15), write('|'),nl,
        nl, write('|'), tab(15), write('|'),nl,
        nl, write('|'), tab(15), write('|'),nl,
        nl, write(' |'), tab(13), write('|'),nl,
        nl, write('  |'), tab(11), write('|'),nl,
        tab(4),write('---------'),
        nl, write('   |'), tab(9), write('|'),nl,
        tab(3),write('-----------').
```



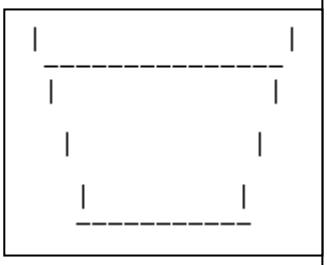
```
make_a0() :-
        nl, write('|'), tab(15), write('|'),nl,
        nl, write('|'), tab(15), write('|'),nl,
        nl, write('|'), tab(15), write('|'),nl,
        nl, write('|'), tab(15), write('|'),nl,
        nl, write(' |'), tab(13), write('|'),nl,
        nl, write('  |'), tab(11), write('|'),nl,
        nl, write('   |'), tab(9), write('|'),nl,
        tab(3),write('-----------').
```



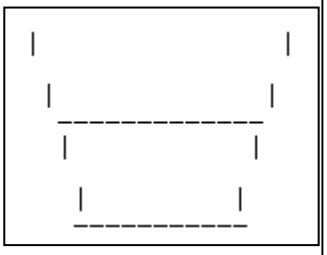
```
make_b4() :- make_line(),
        nl, write('|'), tab(15), write('|'),nl,
        nl, write(' |'), tab(13), write('|'),nl,
        nl, write('  |'), tab(11), write('|'),nl,
        nl, write('   |'), tab(9), write('|'),nl,
        tab(3),write('-----------').
```



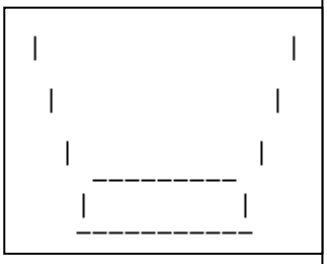
```
make_b3() :- nl, write('|'), tab(15), write('|'),nl,
        make_line(),
        nl, write(' |'), tab(13), write('|'),nl,
        nl, write('  |'), tab(11), write('|'),nl,
        nl, write('   |'), tab(9), write('|'),nl,
        tab(3),write('-----------').
```



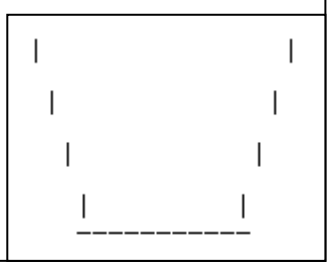
```
make_b2() :-
        nl, write('|'), tab(15), write('|'),nl,
        nl, write(' |'), tab(13), write('|'),nl,
        tab(2),write('-------------'),
        nl, write('  |'), tab(11), write('|'),nl,
        nl, write('   |'), tab(9), write('|'),nl,
        tab(3),write('-----------').
```



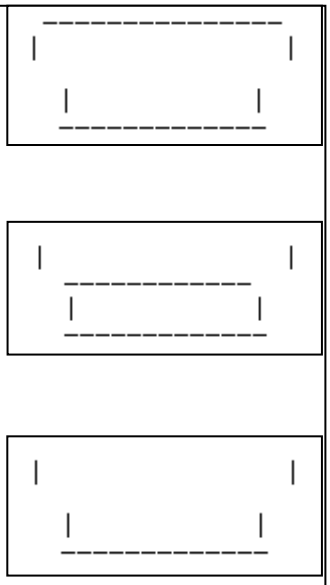
```
make_b1() :-
        nl, write('|'), tab(15), write('|'),nl,
        nl, write(' |'), tab(13), write('|'),nl,
        nl, write('  |'), tab(11), write('|'),nl,
        tab(4),write('---------'),
        nl, write('   |'), tab(9), write('|'),nl,
        tab(3),write('-----------').
```



```
make_b0() :-
        nl, write('|'), tab(15), write('|'),nl,
        nl, write(' |'), tab(13), write('|'),nl,
        nl, write('  |'), tab(11), write('|'),nl,
        nl, write('   |'), tab(9), write('|'),nl,
        tab(3),write('-----------').
```



```
make_c2() :- make_line(),
        nl, write('|'), tab(15), write('|'),nl,
        nl, write('  |'), tab(11), write('|'),nl,
        tab(2),write('-------------').

make_c1() :-
        nl, write('|'), tab(15), write('|'),nl,
        tab(2),write('-------------'),
        nl, write('  |'), tab(11), write('|'),nl,
        tab(2),write('-------------').

make_c0() :-
        nl, write('|'), tab(15), write('|'),nl,
        nl, write('  |'), tab(11), write('|'),nl,
        tab(2),write('-------------').

make_a() :- doxeio_a(Y,7,yes),
       (Y==7,make_a7(),nl;
        Y==6,make_a6(),nl;
        Y==5,make_a5(),nl;
        Y==4,make_a4(),nl;
        Y==3,make_a3(),nl;
        Y==2,make_a2(),nl;
        Y==1,make_a1(),nl;
        Y==0,make_a0(),nl).

make_b() :- doxeio_b(B,4,yes),
       (B==4,make_b4(),nl;
        B==3,make_b3(),nl;
        B==2,make_b2(),nl;
        B==1,make_b1(),nl;
        B==0,make_b0(),nl).

make_c() :- doxeio_c(W,2,yes),
       (W==2,make_c2(),nl;
        W==1,make_c1(),nl;
        W==0,make_c0(),nl).
```



Επιπλέον, έχουν δημιουργηθεί 3 συναρτήσεις (make_a(),make_b() ,make_c()) που προσπαθούν να βρούν την πραγματική τιμή του δοχείου που αντιπροσωπεύει η κάθε μια και να καλέσει την κατάλληλη συνάρτηση για το εκτυπώσει.

γ1) Το πρόβλημα περιέχει 3 ενέργειες (γέμισμα, άδειασμα ,μεταφορά). Για το γέμισμα του κάθε ποτηριού έχει κατασκευαστεί η συνάρτηση sink(). Αντίστοιχα για το άδειασμα η συνάρτηση empty() και για την μετακίνηση από δοχείο σε δοχείο η συνάρτηση move().

Παρακάτω είναι ο κώδικας της κάθε συνάρτησης και μια εξήγηση για τον τρόπο που υλοποιήθηκε :

```
sink() :- doxeio_a(Y,7,yes),doxeio_b(B,4,yes),doxeio_c(W,2,yes),write('What
glass(a,b,c) do you want to fill?: '),
        read(X),
                    (X==a,(Y<7, write('Gemizw to a'),Y1 is 7,nl,nl,
                        retract(doxeio_a(Y,7,yes)),assert(doxeio_a(Y,7,data)),
                        retract(doxeio_a(Y1,7,data)),assert(doxeio_a(Y1,7,yes)));

                     X==b,(B<4, write('Gemizw to b'),B1 is 4,nl,nl,
                       retract(doxeio_b(B,4,yes)),assert(doxeio_b(B,4,data)),
                       retract(doxeio_b(B1,4,data)),assert(doxeio_b(B1,4,yes)));

                     X==c,(W<2, write('Gemizw to c'),W1 is 2,nl,nl,
                        retract(doxeio_c(W,2,yes)),assert(doxeio_c(W,2,data)),
                        retract(doxeio_c(W1,2,data)),assert(doxeio_c(W1,2,yes)));
                    nl,write('Wrong Input')),nl,nl,
                    showglass().
```

Η παραπάνω συνάρτηση αρχικά ψάχνει να βρεί αν υπάρχουν τα δεδομένα doxeio_a(Y,7,yes),doxeio_b(B,4,yes),doxeio_c(W,2,yes). Αν υπάρχουν, καλεί τον χρήστη να γράψει ποιο ποτήρι θέλει να γεμίσει. Ανάλογα ποιο γράμμα απαντήσει θα πάμε σε αυτό το δοχείο και θα κάνουμε retract ό,τι είχε πριν,'yes',και θα το κάνουμε assert απλά σαν δεδομένο,'data'. Έπειτα, θα κάνουμε retract τον αριθμό του μέγιστου level σαν δεδομένο,'data',και θα το κάνουμε assert σαν κάτι πραγματικό,'yes'.

```
empty() :- doxeio_a(Y,7,yes),doxeio_b(B,4,yes),doxeio_c(W,2,yes),write('What
glass(a,b,c) do you want to empty?: '),
        read(H),
                    (H==a,(Y>0, write('Adeiazw to a'),Y1 is 0,nl,nl,
                        retract(doxeio_a(Y,7,yes)),assert(doxeio_a(Y,7,data)),
                        retract(doxeio_a(Y1,7,data)),assert(doxeio_a(Y1,7,yes)));

                    H==b,(B>0, write('Adeiazw to b'),B1 is 0,nl,nl,
                       retract(doxeio_b(B,4,yes)),assert(doxeio_b(B,4,data)),
                       retract(doxeio_b(B1,4,data)),assert(doxeio_b(B1,4,yes)));

                     H==c,(W>0, write('Adeiazw to c'),W1 is 0,nl,nl,
                        retract(doxeio_c(W,2,yes)),assert(doxeio_c(W,2,data)),
                        retract(doxeio_c(W1,2,data)),assert(doxeio_c(W1,2,yes)));
                    nl,write('Wrong Input')),nl,nl,
                    showglass().
```

Ακριβώς με τον ίδιο τρόπο λειτουργεί και η συνάρτηση empty(). Το μόνο που αλλάζει είναι πως σαν πραγματική τιμή του δοχείου που απαντάει ο χρήστης θα πάρει το 0 και ότι είχε προηγουμένως θα γίνει 'data'.

```
move() :- doxeio_a(Y,7,yes),doxeio_b(B,4,yes),doxeio_c(W,2,yes),write('What glass(a,b,c) do you want to move from?: '),
        read(F),
                        (F==a, (
                            Y>0,write('What glass(a,b,c) do you want to move to?: '),read(T),
                                        (
                                         T==b,(    B<4,
                                                write('move a to b'),nl,
                                                (B+Y>=4,B1 is 4, Y1 is Y-(4-B) ;B+Y<4, B1 is B+Y, Y1 is 0),
                                                                write('glass a is: '),write(Y1),nl,
                                                                write('glass b is: '),write(B1),nl,
                                                                write('glass c is: '),write(W),nl,
                                                retract(doxeio_a(Y,7,yes)),assert(doxeio_a(Y,7,data)),
                                                                retract(doxeio_a(Y1,7,data)),assert(doxeio_a(Y1,7,yes)),
                                                                retract(doxeio_b(B,4,yes)),assert(doxeio_b(B,4,data)),
                                                                retract(doxeio_b(B1,4,data)),assert(doxeio_b(B1,4,yes))
                                                        );

                                         T==c,(   W<2,
                                                 write('move a to c'),nl,
                                                (W+Y>=2,W1 is 2, Y1 is Y-(2-W) ;W+Y<2, W1 is W+Y, Y1 is 0),
                                                                write('glass a is: '),write(Y1),nl,
                                                                write('glass b is: '),write(B),nl,
                                                                write('glass c is: '),write(W1),nl,
                                                retract(doxeio_a(Y,7,yes)),assert(doxeio_a(Y,7,data)),
                                                                retract(doxeio_a(Y1,7,data)),assert(doxeio_a(Y1,7,yes)),
                                                                retract(doxeio_c(W,2,yes)),assert(doxeio_c(W,2,data)),
                                                                retract(doxeio_c(W1,2,data)),assert(doxeio_c(W1,2,yes))
                                                )

                                        )
                                        );

                        F==b,(
                            B>0,write('What glass(a,b,c) do you want to move to?: '),read(T),
                                        (
                                         T==a,(    Y<7,
                                                write('move b to a'),nl,
                                                (Y+B>=7,Y1 is 7, B1 is B-(7-Y) ;Y+B<7, Y1 is Y+B, B1 is 0),
                                                                write('glass a is: '),write(Y1),nl,
                                                                write('glass b is: '),write(B1),nl,
                                                                write('glass c is: '),write(W),nl,
                                                retract(doxeio_a(Y,7,yes)),assert(doxeio_a(Y,7,data)),
                                                                retract(doxeio_a(Y1,7,data)),assert(doxeio_a(Y1,7,yes)),
                                                                retract(doxeio_b(B,4,yes)),assert(doxeio_b(B,4,data)),
                                                                retract(doxeio_b(B1,4,data)),assert(doxeio_b(B1,4,yes))
                                                        );

                                         T==c,(   W<2,
                                                 write('move b to c'),nl,
                                                (W+B>=2,W1 is 2, B1 is B-(2-W) ;W+B<2, W1 is W+B, B1 is 0),
                                                                write('glass a is: '),write(Y),nl,
                                                                write('glass b is: '),write(B1),nl,
                                                                write('glass c is: '),write(W1),nl,
                                                retract(doxeio_b(B,4,yes)),assert(doxeio_b(B,4,data)),
                                                                retract(doxeio_b(B1,4,data)),assert(doxeio_b(B1,4,yes)),
                                                                retract(doxeio_c(W,2,yes)),assert(doxeio_c(W,2,data)),
                                                                retract(doxeio_c(W1,2,data)),assert(doxeio_c(W1,2,yes))
                                                )

                                        )
                                        );

                        F==c,(
                            W>0,write('What glass(a,b,c) do you want to move to?: '),read(T),
                                        (
                                         T==b,(    B<4,
                                                write('move c to b'),nl,
                                                (B+W>=4,B1 is 4, W1 is W-(4-B) ;B+W<4, B1 is B+W, W1 is 0),
                                                                write('glass a is: '),write(Y),nl,
                                                                write('glass b is: '),write(B1),nl,
                                                                write('glass c is: '),write(W1),nl,
                                                retract(doxeio_c(W,2,yes)),assert(doxeio_c(W,2,data)),
                                                                retract(doxeio_c(W1,2,data)),assert(doxeio_c(W1,2,yes)),
                                                                retract(doxeio_b(B,4,yes)),assert(doxeio_b(B,4,data)),
                                                                retract(doxeio_b(B1,4,data)),assert(doxeio_b(B1,4,yes))
                                                        );

                                         T==a,(   Y<7,
                                                write('move c to a'),nl,
                                                (Y+W>=7,Y1 is 7, W1 is W-(7-Y) ;Y+W<7, Y1 is Y+W, W1 is 0),
                                                                write('glass a is: '),write(Y1),nl,
                                                                write('glass b is: '),write(B),nl,
                                                                write('glass c is: '),write(W1),nl,
                                                retract(doxeio_a(Y,7,yes)),assert(doxeio_a(Y,7,data)),
                                                                retract(doxeio_a(Y1,7,data)),assert(doxeio_a(Y1,7,yes)),
                                                                retract(doxeio_c(W,2,yes)),assert(doxeio_c(W,2,data)),
                                                                retract(doxeio_c(W1,2,data)),assert(doxeio_c(W1,2,yes))
                                                        )

                                        )
                                        );
                            nl,write('Wrong Input')

                                        )
                                        ,nl,nl,showglass().
```

Η παραπάνω συνάρτηση είναι η move(). Στην αρχή προσπαθεί να διαπιστώσει αν υπάρχουν οι πραγματικές τιμές των ποτηριών και έπειτα ρωτά το χρήστη από ποιο ποτήρι θέλει να μεταφέρει νερό. Αν αυτό είναι άδειο επιστρέφει 'Wrong Input'.

Μετά, ρωτά τον χρήστη σε ποιο ποτήρι θέλει να μεταφερθεί. Αν αυτό είναι γεμάτο απαντάει 'Wrong Input'.

Ανάλογα το ποτήρι που θα επιλέξουμε θα πρέπει να λάβουμε υπόψιν δύο περιπτώσεις για κάθε έγκυρη μεταφορά:

1. Το άθροισμα του νερού από το ερχόμενο ποτήρι μαζί με το ποτήρι που θα γεμίσουμε δεν θα προκαλέσει ξεχείλισμα.
Τότε, θα προσθέσουμε τα 2 επίπεδα και θα τα κάνουμε assert στο ποτήρι που έγινε η μεταφορά καθώς και θα «μηδενίσουμε» το ποτήρι που έκανε την μεταφορά.

2. Η μεταφορά θα προκαλέσει ξεχείλισμα.
Σε αυτήν την περίπτωση θα πρέπει το ποτήρι που μεταφέρθηκε η ποσότητα (cup1) να γεμίσει μέχρι πάνω και το ποτήρι μεταφοράς (cup2) να βρίσκεται στο επίπεδο που υπολογίζεται από τον τύπο: cup2_new=cup2-(cup1_full - cup1).

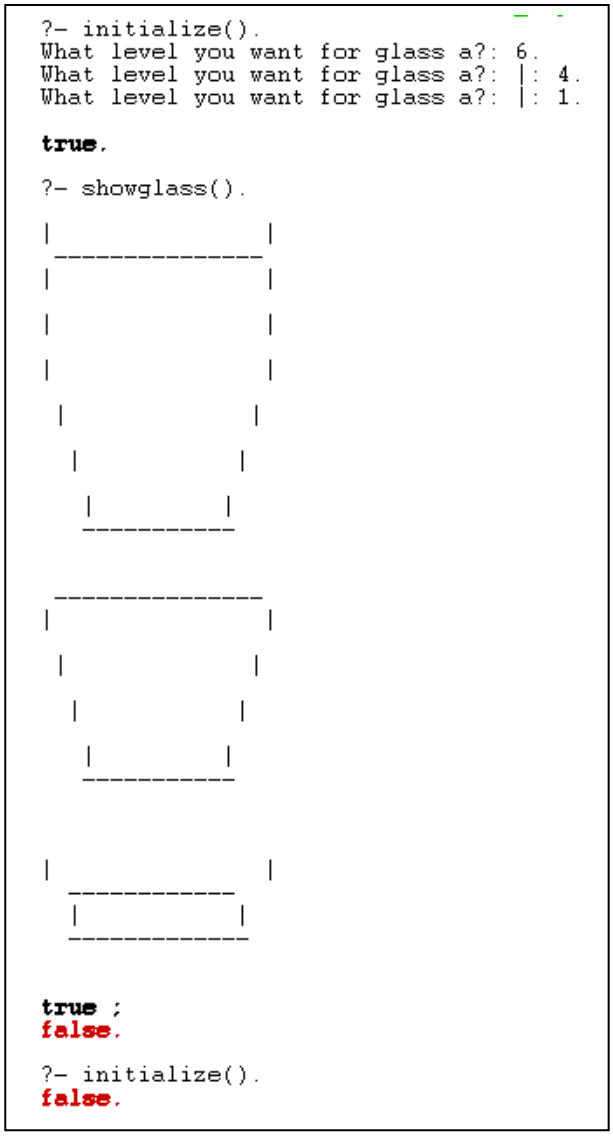
γ2)
Όσον αφορά την εκτύπωση της κατάστασης των ποτηριών έχει δημιουργηθεί η συνάρτηση showglass() που καλεί τις make_a(),make_b(),make_c() που έχουν αναφερθεί παραπάνω.

```
showglass() :- make_a(),nl,nl,make_b(),nl,nl,make_c(),nl,nl.
```

δ)
Για αυτό το ερώτημα θα χρειαστεί να τροποποιήσουμε τα δεδομένα μας και να αλλάξουμε τα doxeio_a(0,7,yes), doxeio_b(0,4,yes), doxeio_c(0,2,yes) σε doxeio_a(0,7,data), doxeio_b(0,4,data), doxeio_c(0,2,data).
Έπειτα, με την συνάρτηση initialize() θα αρχικοποιήσουμε τα ποτήρια.

```
initialize() :- \+doxeio_a(_,7,yes),\+doxeio_b(_,4,yes),\+doxeio_c(_,2,yes),
        write('What level you want for glass a?: '),read(L),L=<7,
        write('What level you want for glass a?: '),read(M),M=<4,
        write('What level you want for glass a?: '),read(N),N=<2,
        retract(doxeio_a(L,7,data)),assert(doxeio_a(L,7,yes)),
        retract(doxeio_b(M,4,data)),assert(doxeio_b(M,4,yes)),
        retract(doxeio_c(N,2,data)),assert(doxeio_c(N,2,yes)).
```

Αρχικά, ψάχνουμε να βρούμε αν έχουν ήδη αρχικοποιήθει οπότε δεν θέλουμε '\+' κανένα να έχει την τιμή 'yes' σε οποιαδήποτε κατάσταση που συμβολίζουμε με '-'.
Έπειτα, ρωτάμε τον χρήστη τα levels που θέλει και μετά, με την retract και την assert αλλάζουμε δυναμικά τα level μόνο 1 φορά. Αν ξανακαλέσουμε την initialize() θα μας απαντήσει με false. Παρακάτω είναι ένα παράδειγμα(screenshot).



ε) Τώρα είμαστε σε θέση να απομονώσουμε στο δοχείο b 1lt νερό.
Οι κινήσεις που πρέπει να κάνουμε είναι οι εξής:
1.Αρχικοποίηση των ποτηριών σε άδεια.
2.Γέμισμα του a→7 (a=7,b=0,c=0)
3.Μετακίνηση του a→b (a=3,b=4,c=0)
4.Μετακίνηση του a→c (a=1,b=4,c=2)
5.Άδειασμα του b→0 (a=1,b=0,c=2)
6.Μετακίνηση του a→b(a=0,**b=1**,c=2)

Ενέργειες:

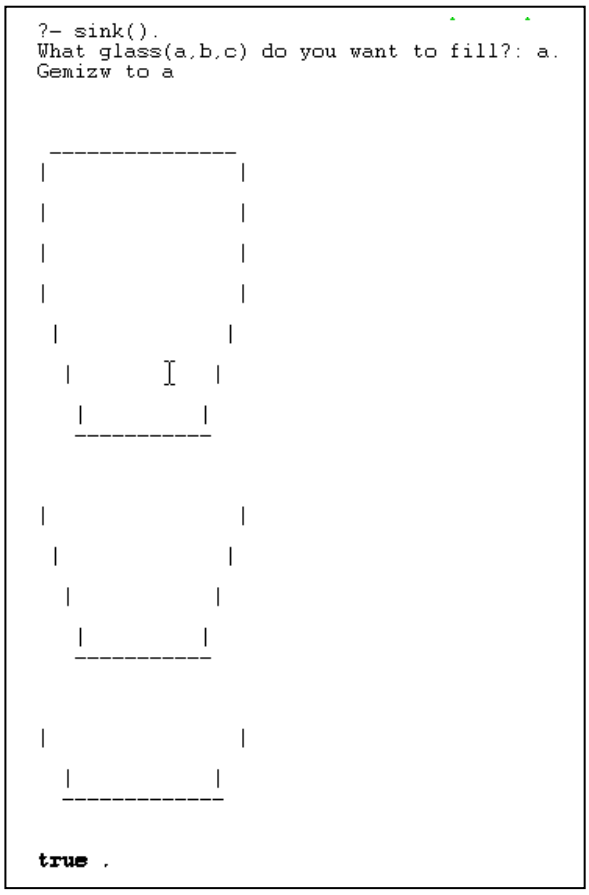
1.

?- **initialize().**
What level you want for glass a?: **0.**
What level you want for glass a?: |: **0.**
What level you want for glass a?: |: **0.**

true.

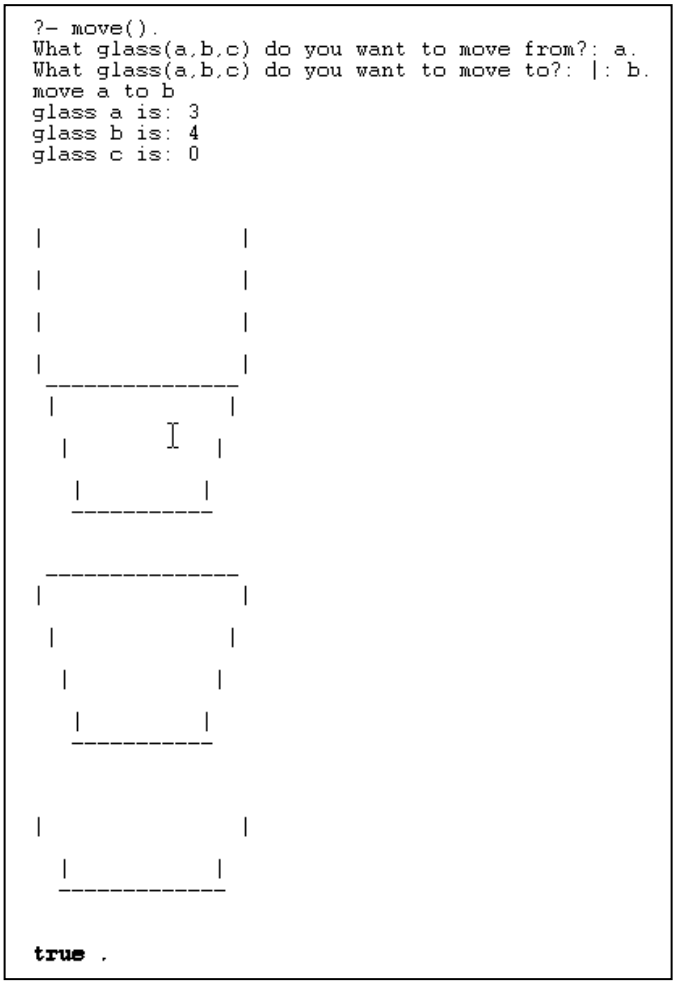
2.

?- **sink().**
What glass(a,b,c) do you want to fill?: **a.**



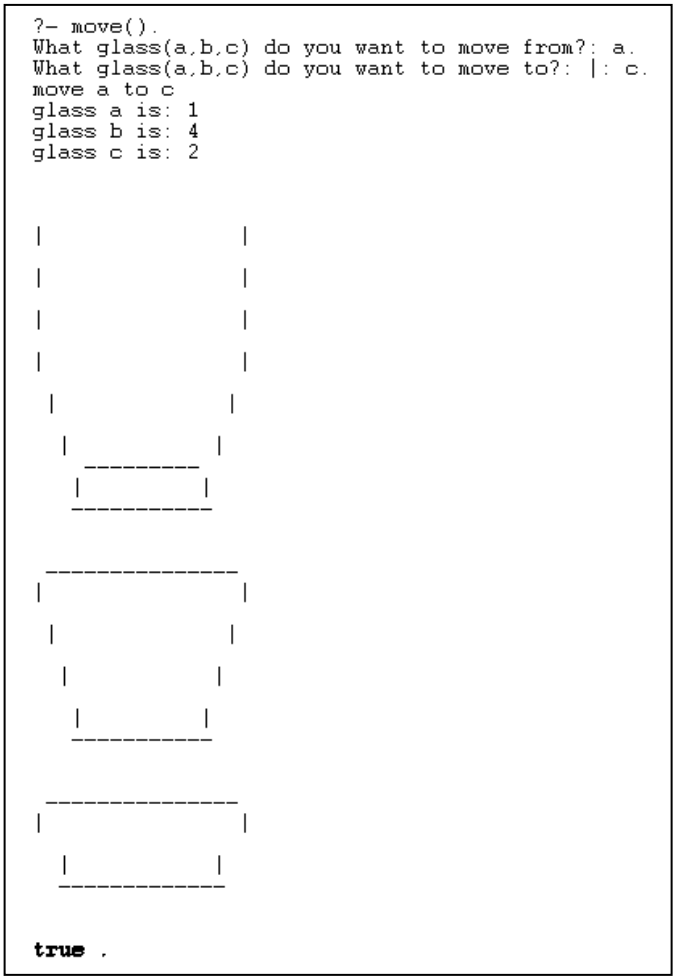
3.

?- **move().**
What glass(a,b,c) do you want to move from?: **a.**
What glass(a,b,c) do you want to move to?: |: **b.**
move a to b
glass a is: 3
glass b is: 4
glass c is: 0



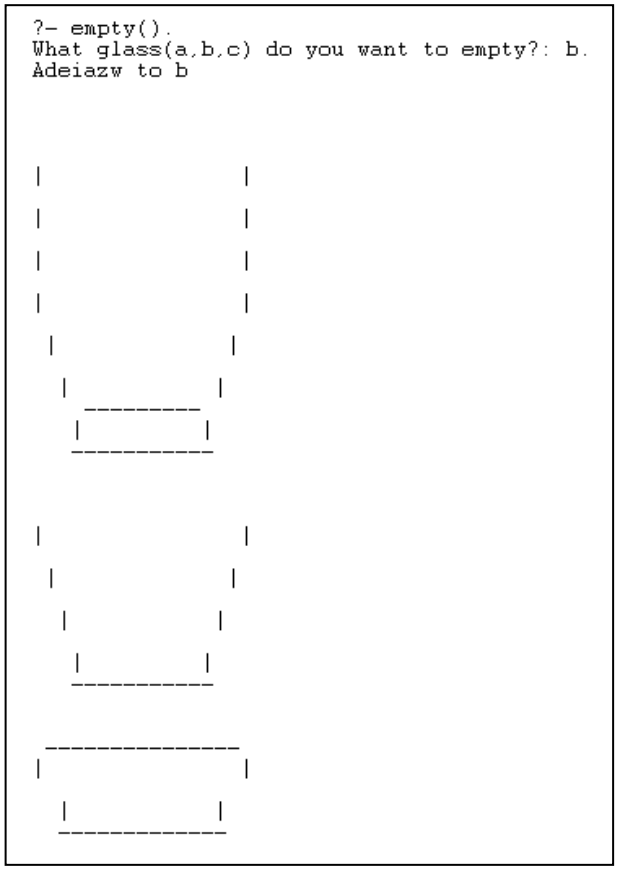
4.

?- **move().**
What glass(a,b,c) do you want to move from?: **a.**
What glass(a,b,c) do you want to move to?: |: **c.**
move a to c
glass a is: 1
glass b is: 4
glass c is: 2



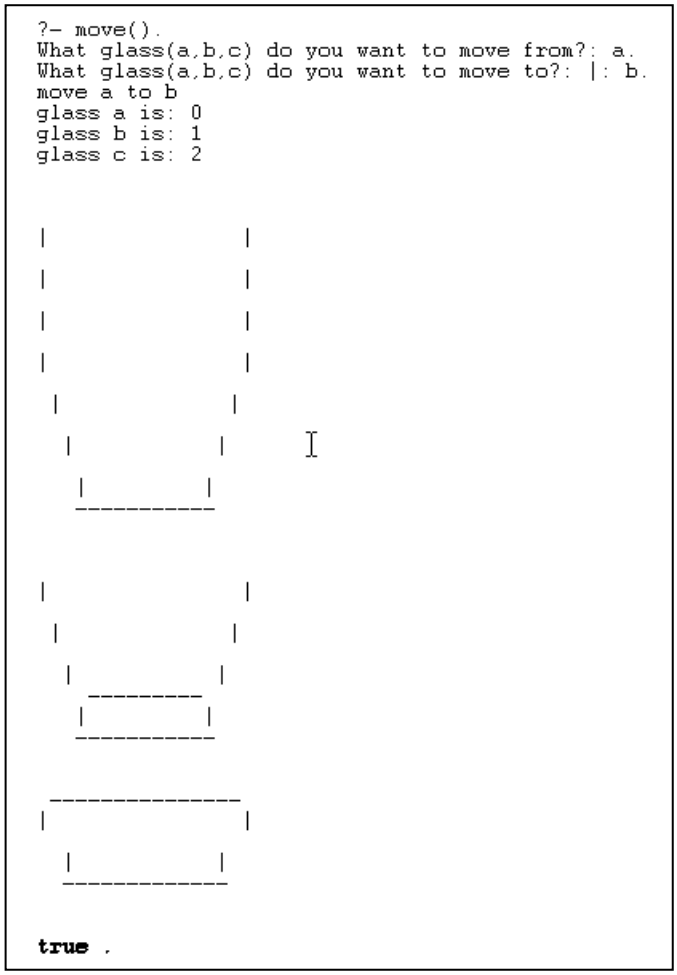
5.

?- **empty().**
What glass(a,b,c) do you want to empty?: **b.**
Adeiazw to b



6.

?- **move().**
What glass(a,b,c) do you want to move from?: **a.**
What glass(a,b,c) do you want to move to?: |: **b.**
move a to b
glass a is: 0
glass b is: 1
glass c is: 2



Υποσημείωση: Παρατήρησα πως το SWI-PROLOG για κάποιο λόγο μερικές φορές θυμάται προηγούμενες καταστάσεις όταν κάνεις CONSULT αν δεν το έχεις κλείσει πριν. Οπότε σας συμβουλεύω για να μην δημιουργηθούν μη αναμενόμενα αποτελέσματα στο πρόγραμμα, να κλείνετε το SWI-PROLOG όταν θέλετε να κάνετε πάλι CONSULT το πρόγραμμα. Ευχαριστώ.

-ΔΑΜΙΑΝΟΣ ΔΙΑΣΑΚΟΣ ΑΜ: 1084632 3ο έτος