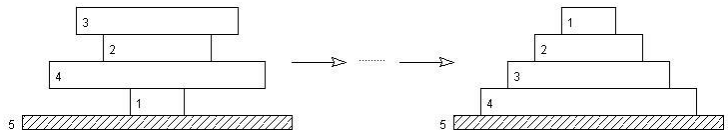
ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

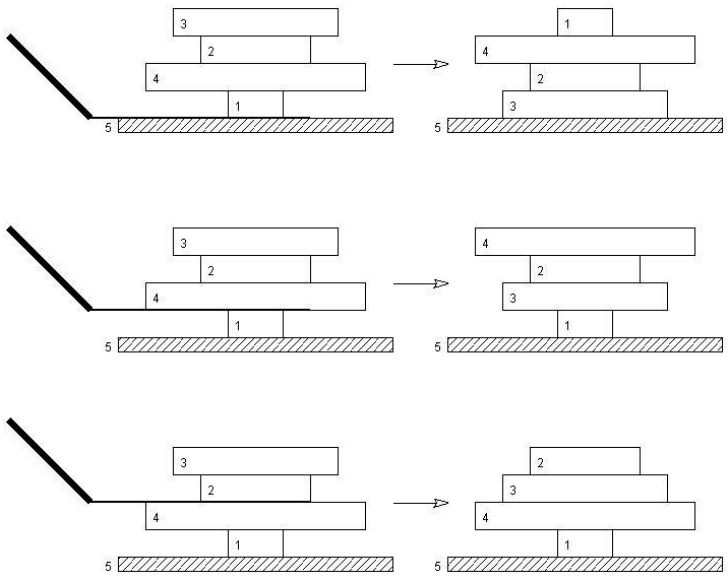
Α’ Ομάδα Ασκήσεων "Λογικού Προγραμματισμού" Ακαδημαϊκού Έτους 2022-23

Άσκηση 1

Έχουμε φτιάξει Ν πίτες, όλες κυκλικού σχήματος, αλλά διαφορετικών μεγεθών, έστω με διαμέτρους 1, 2, …, Ν. Οι πίτες είναι τοποθετημένες σε μία στήλη, η μία επάνω από την άλλη, με αυθαίρετη σειρά, επάνω σε ένα πιάτο, έστω διαμέτρου Ν+1. Θέλουμε να αλλάξουμε τη σειρά στις πίτες, έτσι ώστε κάτω-κάτω να είναι αυτή με το μεγαλύτερο μέγεθος (διαμέτρου Ν), από επάνω της αυτή με το αμέσως μικρότερο (διαμέτρου Ν-1), κοκ., μέχρι τη μικρότερη πίτα (διαμέτρου 1), που θα είναι στην κορυφή. Στο παρακάτω σχήμα, έχουμε Ν = 4 πίτες τοποθετημένες αρχικά όπως φαίνεται αριστερά και θέλουμε τελικά να τους αλλάξουμε τη σειρά και να είναι όπως φαίνεται δεξιά.



Ο μοναδικός τρόπος που έχουμε για να αλλάξουμε τη σειρά στις πίτες είναι να χρησιμοποιήσουμε μία σπάτουλα, να την βάλουμε κάτω από μία πίτα, να σηκώσουμε όλη τη στήλη από την πίτα αυτή και τις από πάνω της και με μία κίνηση (δύσκολο, αλλά όχι ακατόρθωτο για έναν σεφ!) να τοποθετήσουμε τις πίτες με αντεστραμμένη σειρά επάνω στις υπόλοιπες. Για να καταλάβετε καλύτερα τι μπορούμε να κάνουμε με τη σπάτουλα, δείτε το επόμενο σχήμα.



Στην 1η περίπτωση, βάζουμε τη σπάτουλα κάτω από την πίτα 1 και αντιστρέφουμε όλη τη στήλη από πάνω της, με αποτέλεσμα τη δεξιά διάταξη. Στη 2η περίπτωση, βάζουμε τη σπάτουλα κάτω από την πίτα 4 και αντιστρέφουμε όλη τη στήλη από πάνω της. Ομοίως, στην 3η περίπτωση, η σπάτουλα μπαίνει κάτω από την πίτα 2. Προφανώς, δεν έχει νόημα να βάλουμε τη σπάτουλα κάτω από την πίτα 3 και να αντιστρέψουμε τις πίτες επάνω από αυτήν, γιατί τελικά δεν θα αλλάξει κάτι στη διάταξη.[1]

Ας θεωρήσουμε ότι η αναπαράσταση μίας κατάστασης του προβλήματος γίνεται από μία λίστα που τα στοιχεία της είναι οι διάμετροι των πιτών από επάνω προς τα κάτω. Δηλαδή, η αρχική κατάσταση του προηγούμενου στιγμιότυπου προβλήματος είναι η `[3,2,4,1]`. Επίσης, ας θεωρήσουμε ότι η αναπαράσταση μίας ενέργειας κατά την οποία τοποθετούμε την σπάτουλα κάτω από την πίτα διαμέτρου N και αντιστρέφουμε όλη τη στήλη από πίτες από επάνω μέχρι και την πίτα αυτή γίνεται από τον ακέραιο N. Οπότε, μία αλληλουχία τέτοιων ενεργειών, άρα και μία λύση του προβλήματος, μπορεί να αναπαρασταθεί από μία λίστα ακεραίων μεταξύ 1 και N. Για παράδειγμα, μπορείτε να επιβεβαιώσετε ότι η λίστα `[4,3,2,4,1]` είναι μία λύση του προβλήματος με αρχική κατάσταση την `[3,2,4,1]`.

Ορίστε σε Prolog ένα κατηγόρημα `pancakes_dfs/3`, το οποίο όταν καλείται ως `pancakes_dfs(InitialState, Operators, States)`, για αρχική κατάσταση του προβλήματος την `InitialState`, να επιστρέφει μέσω οπισθοδρόμησης όλες τις πιθανές λύσεις σαν λίστα ενεργειών `Operators`, αλλά και τις καταστάσεις από τις οποίες περνά η επίλυσή του σαν λίστα `States`.

Υπόδειξη: Μελετήστε τα προβλήματα και τις λύσεις τους στις σελίδες 139-141 και 148-152 των διαφανειών του μαθήματος, όπου χρησιμοποιείται η μέθοδος *αναζήτησης πρώτα-κατά-βάθος* (depth-first-search – dfs).

Ενδεικτικές εκτελέσεις:

```
?- pancakes_dfs([3, 2, 4, 1], Operators, States).
Operators = [1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 4, 1]
States = [[3, 2, 4, 1], [1, 4, 2, 3], [2, 4, 1, 3],
          [3, 1, 4, 2], [1, 3, 4, 2], [2, 4, 3, 1],
          [3, 4, 2, 1], [1, 2, 4, 3], [2, 1, 4, 3],
          [3, 4, 1, 2], [1, 4, 3, 2], [2, 3, 4, 1],
          [4, 3, 2, 1], [1, 2, 3, 4]]                --> ;
Operators = [1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 4, 2, 1, 3, 2, 1]
States = [[3, 2, 4, 1], [1, 4, 2, 3], [2, 4, 1, 3],
          [3, 1, 4, 2], [1, 3, 4, 2], [2, 4, 3, 1],
          [3, 4, 2, 1], [1, 2, 4, 3], [2, 1, 4, 3],
          [3, 4, 1, 2], [1, 4, 3, 2], [4, 1, 3, 2],
          [2, 3, 1, 4], [1, 3, 2, 4], [3, 1, 2, 4],
          [2, 1, 3, 4], [1, 2, 3, 4]]                --> ;
..........................
```

[1] Σε μία παραλλαγή του προβλήματος, οι πίτες είναι καμένες από τη μία πλευρά και ο σεφ θέλει να τις αναδιατάξει έτσι ώστε τελικά να είναι με την επιθυμητή σειρά μεγεθών, όπως έχει ήδη περιγραφεί, αλλά με την καμένη επιφάνειά τους από κάτω. Τότε, έχει νόημα να αντιστρέψουμε μόνο την πίτα της κορυφής, αλλά στη συγκεκριμένη άσκηση, δεν θα ασχοληθούμε με αυτή την παραλλαγή του προβλήματος.

```
?- findall(Operators,
           pancakes_dfs([3, 2, 4, 1], Operators, _),
           Solutions),
   length(Solutions, N).
..........................
N = 2936
```

Bonus ερώτημα (+100%): Έστω, όμως, ότι από τον πολύ μεγάλο αριθμό λύσεων στο προηγούμενο στιγμιότυπο θέλουμε να βρούμε λύση με τον ελάχιστο αριθμό ενεργειών. Αυτό θα μπορούσε υλοποιηθεί με την εξής λογική. «Προσπάθησε να βρεις λύση με 0 ενέργειες. Αν δεν είναι δυνατόν, προσπάθησε να βρεις λύση με 1 ενέργεια. Αν ούτε αυτό γίνεται, δοκίμασε με 2 ενέργειες, κοκ.». Έτσι, η πρώτη λύση που θα βρεθεί θα είναι βέλτιστη, δηλαδή με τον ελάχιστο αριθμό ενεργειών. Αυτή η μέθοδος είναι η *αναζήτηση επαναληπτικής εμβάθυνσης* (iterative deepening search – ids). Ορίστε σε Prolog ένα κατηγόρημα `pancakes_ids/3`, με ίδια ορίσματα με το `pancakes_dfs/3`, το οποίο, εκμεταλλευόμενο τη φιλοσοφία της ids, να βρίσκει **όλες τις εξ ίσου βέλτιστες λύσεις** του προβλήματος. Παραδείγματα εκτέλεσης:

```
?- pancakes_ids([3, 2, 4, 1], Operators, States).
Operators = [2, 4, 1]
States = [[3, 2, 4, 1], [2, 3, 4, 1], [4, 3, 2, 1],
          [1, 2, 3, 4]]                                --> ;
no

?- pancakes_ids([3, 5, 6, 2, 4, 1], Operators, States).
Operators = [2, 4, 3, 6, 1]
States = [[3, 5, 6, 2, 4, 1], [2, 6, 5, 3, 4, 1],
          [4, 3, 5, 6, 2, 1], [3, 4, 5, 6, 2, 1],
          [6, 5, 4, 3, 2, 1], [1, 2, 3, 4, 5, 6]]    --> ;
Operators = [6, 1, 4, 3, 1]
States = [[3, 5, 6, 2, 4, 1], [6, 5, 3, 2, 4, 1],
          [1, 4, 2, 3, 5, 6], [4, 1, 2, 3, 5, 6],
          [3, 2, 1, 4, 5, 6], [1, 2, 3, 4, 5, 6]]    --> ;
no

?- findall(Operators,
           pancakes_ids([5, 9, 1, 4, 8, 6, 7, 3, 2],
                        Operators, _),
           Solutions),
   length(Solutions, N).
Operators = Operators
Solutions =
       [[1, 2, 9, 4, 7, 6, 9, 1], [1, 4, 7, 6, 9, 2, 8, 1],
        [2, 4, 7, 6, 9, 5, 8, 1], [4, 9, 2, 4, 7, 6, 8, 1],
        [7, 6, 9, 2, 4, 5, 8, 1], [8, 1, 4, 9, 2, 4, 7, 1],
        [8, 7, 4, 9, 2, 4, 8, 1], [8, 9, 2, 4, 7, 5, 8, 1],
        [9, 4, 7, 6, 9, 2, 4, 1]]
N = 9
```

Παραδοτέο για την άσκηση είναι **<u>ένα</u> πηγαίο αρχείο Prolog** με όνομα **`pancakes.pl`**.

Άσκηση[2] 2

Έστω ότι δίνονται τα παρακάτω γεγονότα Prolog[3]:

```
activity(a01, act(0,3)).
activity(a02, act(0,4)).
activity(a03, act(1,5)).
activity(a04, act(4,6)).
activity(a05, act(6,8)).
activity(a06, act(6,9)).
activity(a07, act(9,10)).
activity(a08, act(9,13)).
activity(a09, act(11,14)).
activity(a10, act(12,15)).
activity(a11, act(14,17)).
activity(a12, act(16,18)).
activity(a13, act(17,19)).
activity(a14, act(18,20)).
activity(a15, act(19,20)).
```

Τα γεγονότα `activity/2` κωδικοποιούν δραστηριότητες που θα πρέπει να στελεχωθούν από άτομα. Κάθε γεγονός `activity(A, act(S,E))` σημαίνει ότι η δραστηριότητα `A` αρχίζει τη χρονική στιγμή `S` και τελειώνει τη χρονική στιγμή `E`. Υποθέστε ότι κάθε δραστηριότητα πρέπει να στελεχωθεί από ένα ακριβώς άτομο και ότι κάθε άτομο μπορεί να αναλάβει όσες δραστηριότητες απαιτούνται, αρκεί ο συνολικός χρόνος εργασίας του να μην υπερβαίνει κάποιο καθορισμένο όριο. Επίσης, υπάρχει και ο περιορισμός ότι όχι μόνο δεν μπορεί ένα άτομο να αναλάβει δύο δραστηριότητες που επικαλύπτονται χρονικά, αλλά θα πρέπει μεταξύ δύο οποιωνδήποτε διαδοχικών δραστηριοτήτων κάθε ατόμου να μεσολαβεί τουλάχιστον μία μονάδα χρόνου. Με τα δεδομένα αυτά, υλοποιήστε σε Prolog ένα κατηγόρημα `assignment/4`, το οποίο, όταν καλείται ως `assignment(NP, MT, ASP, ASA)`, όπου `NP` είναι το πλήθος των διαθέσιμων ατόμων και `MT` είναι ο μέγιστος συνολικός χρόνος των δραστηριοτήτων που μπορεί να αναλάβει ένα άτομο, να αναθέτει τις δοθείσες δραστηριότητες στα δοθέντα άτομα με εφικτό τρόπο, επιστρέφοντας στις μεταβλητές `ASP` και `ASA` δύο ισοδύναμες αναπαραστάσεις της ανάθεσης που πραγματοποιήθηκε. Συγκεκριμένα, η μεταβλητή `ASP` να είναι μία λίστα από στοιχεία της μορφής `N-As-T`, όπου `N` είναι ο αύξων αριθμός ενός ατόμου (`1`, `2`, …, `NP`), `As` είναι η λίστα των δραστηριοτήτων που ανατέθηκαν στο άτομο αυτό και `T` είναι η συνολική διάρκεια των δραστηριοτήτων που του ανατέθηκαν. Η μεταβλητή `ASA` να είναι μία λίστα από στοιχεία της μορφής `A-N`, που σημαίνουν ότι η δραστηριότητα `A` ανατίθεται στο άτομο `N`. Παραδείγματα ερωτήσεων και των αντίστοιχων απαντήσεων στο πρόγραμμα:

```
?- assignment(3, 14, ASP, ASA).

ASP = [1 - [a15, a12, a09, a07, a05, a03] - 13,
       2 - [a14, a11, a08, a04, a01] - 14,
       3 - [a13, a10, a06, a02] - 12]
```

[2] Για την άσκηση αυτή πρέπει να χρησιμοποιήσετε απλή Prolog και όχι επεκτάσεις της που υποστηρίζουν προγραμματισμό με περιορισμούς.

[3] Βρίσκονται στο αρχείο: http://www.di.uoa.gr/~takis/activity.pl.

```
ASA = [a01 - 2, a02 - 3, a03 - 1, a04 - 2, a05 - 1, a06 - 3,
       a07 - 1, a08 - 2, a09 - 1, a10 - 3, a11 - 2, a12 - 1,
       a13 - 3, a14 - 2, a15 - 1]                    --> ;
ASP = [1 - [a15, a12, a10, a07, a05, a03] - 13,
       2 - [a14, a11, a08, a04, a01] - 14,
       3 - [a13, a09, a06, a02] - 12]
ASA = [a01 - 2, a02 - 3, a03 - 1, a04 - 2, a05 - 1, a06 - 3,
       a07 - 1, a08 - 2, a09 - 3, a10 - 1, a11 - 2, a12 - 1,
       a13 - 3, a14 - 2, a15 - 1]                    --> ;
ASP = [1 - [a15, a12, a10, a06, a03] - 13,
       2 - [a14, a11, a08, a04, a01] - 14,
       3 - [a13, a09, a07, a05, a02] - 12]
ASA = [a01 - 2, a02 - 3, a03 - 1, a04 - 2, a05 - 3, a06 - 1,
       a07 - 3, a08 - 2, a09 - 3, a10 - 1, a11 - 2, a12 - 1,
       a13 - 3, a14 - 2, a15 - 1]                    --> ;
ASP = [1 - [a15, a12, a09, a06, a03] - 13,
       2 - [a14, a11, a08, a04, a01] - 14,
       3 - [a13, a10, a07, a05, a02] - 12]
ASA = [a01 - 2, a02 - 3, a03 - 1, a04 - 2, a05 - 3, a06 - 1,
       a07 - 3, a08 - 2, a09 - 1, a10 - 3, a11 - 2, a12 - 1,
       a13 - 3, a14 - 2, a15 - 1]                    --> ;
ASP = [1 - [a15, a12, a10, a06, a02] - 13,
       2 - [a14, a11, a08, a04, a01] - 14,
       3 - [a13, a09, a07, a05, a03] - 12]
ASA = [a01 - 2, a02 - 1, a03 - 3, a04 - 2, a05 - 3, a06 - 1,
       a07 - 3, a08 - 2, a09 - 3, a10 - 1, a11 - 2, a12 - 1,
       a13 - 3, a14 - 2, a15 - 1]                    --> ;
ASP = [1 - [a15, a12, a09, a06, a02] - 13,
       2 - [a14, a11, a08, a04, a01] - 14,
       3 - [a13, a10, a07, a05, a03] - 12]
ASA = [a01 - 2, a02 - 1, a03 - 3, a04 - 2, a05 - 3, a06 - 1,
       a07 - 3, a08 - 2, a09 - 1, a10 - 3, a11 - 2, a12 - 1,
       a13 - 3, a14 - 2, a15 - 1]                    --> ;
ASP = [1 - [a15, a12, a09, a07, a05, a02] - 13,
       2 - [a14, a11, a08, a04, a01] - 14,
       3 - [a13, a10, a06, a03] - 12]
ASA = [a01 - 2, a02 - 1, a03 - 3, a04 - 2, a05 - 1, a06 - 3,
       a07 - 1, a08 - 2, a09 - 1, a10 - 3, a11 - 2, a12 - 1,
       a13 - 3, a14 - 2, a15 - 1]                    --> ;
ASP = [1 - [a15, a12, a10, a07, a05, a02] - 13,
       2 - [a14, a11, a08, a04, a01] - 14,
       3 - [a13, a09, a06, a03] - 12]
ASA = [a01 - 2, a02 - 1, a03 - 3, a04 - 2, a05 - 1, a06 - 3,
       a07 - 1, a08 - 2, a09 - 3, a10 - 1, a11 - 2, a12 - 1,
       a13 - 3, a14 - 2, a15 - 1]                    --> ;
no

?- assignment(2, 40, ASP, ASA).
no

?- assignment(3, 13, ASP, ASA).
no
```

```
?- findall(sol, assignment(5, 8, _, _), Solutions),
   length(Solutions, N).
Solutions = [sol, sol, sol, sol, sol, sol, sol, ...]
N = 2544

?- findall(sol, assignment(5, 9, _, _), Solutions),
   length(Solutions, N).
Solutions = [sol, sol, sol, sol, sol, sol, sol, ...]
N = 63852
```

Το πρόγραμμα που θα γράψετε πρέπει να επιστρέφει όλες τις διαφορετικές λύσεις του προβλήματος, θεωρώντας ότι τα άτομα που συμμετέχουν στην ανάθεση είναι ισοδύναμα μεταξύ τους. Αυτό σημαίνει ότι για δεδομένη ανάθεση σε Ν άτομα, είναι ορθές και όλες οι Ν! αναθέσεις που προκύπτουν αν μεταθέσουμε τα άτομα με όλους τους δυνατούς τρόπους, αλλά, επί της ουσίας, είναι όλες ίδιες μεταξύ τους. Το πρόγραμμά σας πρέπει να επιστρέφει μόνο μία από αυτές. Για παράδειγμα, αν είχαμε

```
activity(a1, act(0,3)).
activity(a2, act(4,6)).
activity(a3, act(1,2)).
```

θα πρέπει η συμπεριφορά του προγράμματος να είναι κάπως έτσι:

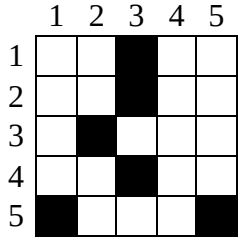
```
?- assignment(2, 10, ASP, ASA).
ASP = [1 - [a3] - 1, 2 - [a2, a1] - 5]
ASA = [a1 - 2, a2 - 2, a3 - 1]                    --> ;
ASP = [1 - [a3, a2] - 3, 2 - [a1] - 3]
ASA = [a1 - 2, a2 - 1, a3 - 1]                    --> ;
no
```

Δηλαδή, υπάρχουν ακριβώς δύο διακριτές λύσεις για το πρόβλημα αυτό και όχι τέσσερις που θα προέκυπταν αν ανταλλάσαμε τις αναθέσεις των δύο ατόμων.

Παραδοτέο για την άσκηση είναι **ένα πηγαίο αρχείο Prolog** με όνομα **`assignment.pl`**, μέσα στο οποίο **δεν** θα πρέπει να περιέχονται γεγονότα `activity/2`.

Άσκηση 3

Ένα σταυρόλεξο τετραγώνου σχήματος μπορεί να ορισθεί από τη διάστασή του και τις συντεταγμένες των μαύρων θέσεων. Για παράδειγμα, το σταυρόλεξο



μπορεί να περιγραφεί από τα γεγονότα Prolog:

```
dimension(5).

black(1,3).
black(2,3).
black(3,2).
black(4,3).
black(5,1).
black(5,5).
```

Δεδομένου και ενός γεγονότος με κατηγόρημα `words/1` μέσω του οποίου δίνονται οι λέξεις που πρέπει να τοποθετηθούν στο σταυρόλεξο, ορίστε ένα κατηγόρημα `crossword/1` που να επιστρέφει τη λίστα των διαθεσίμων λέξεων με τη σειρά που αυτές μπορούν να τοποθετηθούν στο σταυρόλεξο, πρώτα οι οριζόντιες και μετά οι κάθετες. Για παράδειγμα, αν δίνεται και το

```
words([adam,al,as,do,ik,lis,ma,oker,ore,pirus,po,so,ur]).
```

η σωστή απάντηση στο προηγούμενο σταυρόλεξο, δηλαδή αυτή που πρέπει να επιστρέψει το `crossword/1`, είναι η

```
[as,po,do,ik,ore,ma,ur,lis,adam,so,al,pirus,oker]
```

που αντιπροσωπεύει την εξής λύση:

	1	2	3	4	5
1	a	s	■	p	o
2	d	o	■	i	k
3	a	■	o	r	e
4	m	a	■	u	r
5	■	l	i	s	■

Θα πρέπει επίσης η λύση να εκτυπώνεται και περισσότερο παραστατικά, όπως παραπάνω (σε μορφή κειμένου, φυσικά).

Ενδεικτική εκτέλεση:

```
?- crossword(S).
 a  s ### p  o
 d  o ### i  k
 a ### o  r  e
 m  a ### u  r
### l  i  s ###

S = [as,po,do,ik,ore,ma,ur,lis,adam,so,al,pirus,oker]
```

Αν τυχόν υπάρχουν περισσότερες της μίας λύσεις, αρκεί το `crossword/1` να επιστρέφει μόνο μία από αυτές. Φυσικά, αν δεν υπάρχει λύση, πρέπει να αποτυγχάνει. Θεωρείται γνωστό, επίσης, ότι στα σταυρόλεξα δεν λογίζονται σαν λέξεις αυτές του ενός γράμματος.

Δοκιμάστε το πρόγραμμά σας και σε μεγαλύτερα σταυρόλεξα, όπως αυτά που βρίσκονται στο συμπιεσμένο αρχείο http://www.di.uoa.gr/~takis/crosswords.zip.

Και μία άλλη εκτέλεση (για το αρχείο `cross14.pl`):

```
?- crossword(S).
 u  l  t  r  a  i  s  t ### g  l  a  s  s  i  e  s ### d  o  m  i  c  i  l
 l  o  r  i  n  d  a ### p  r  o  d  u  c  t  s ### r  e  n  a  t  u  r  e
 t  w  i  s  t ### m  i  r  a  c  u  l  i  s  t  s ### t  e  l  e  r  a  n
 r  e  s  e  r  v  e  s ### y  o  l  k  s ### a  d  v  i  s  e  m  e  n  t
 a ### e ### o ### n ###### n ### t ### s ### b ###### n ### n ### t  i  e
 s ### c  u  r  s  e ### t  e  r  e  d  o ### l  u  a  u ### e ### t  a  n
 p  u  t ### s  u  s ###### s  i  r ### r  e  i  n  l  e  s  s ### e  n ###
 e ### i ### e  s  s  a  y  s ### a ### s ### s  r  i ###### s ### m ### t
 c  l  o  p ### p ### i  o ###### t ######### h  e  e ############ e ### h
 i  o  n  i  z  e ###### u  r  s  i  n  e ### m  e  n ### s  w  i  n  g  e
 a ############ n ###### r  o  a  n ### y ### e  v  e ### p ###### t ### r
 l  e  f  t  i  s  t ### s  u  n  g ### e ### n  e  e  d  l  e  s ###### m
 i ######### v  e  r  n  e ######### s  l  u  t ###### w  a  r  d ###### o
 z ### p ###### s  i  d  l  i  n  g ### e  n  s  o  l  i  t  e ######### m
 a  l  a  t  e ######### f ### e  a  r  s ###### w  i  n ###### p  l  e  a
 t  e  n  u  r  e ### u ### f  o  r  e  s  a  w ### a  d  h  e  r  i  n  g
 i  n  t  e  r  v  e  n  o  r ###### g ### i  r ### i  l ### l  o  g  a  n
 o ### h ### s  e  v  e  r  a  b  l  e ### r  o  s  s  i ### s  p  u  m  e
 n ### e  m ###### e  n ### t  o  o  n ### s  t  e  e  n ### i  s  l  e  t
### d  i  a  m  o  n  d  s ### t  a  t  t  i  e  r ### g ### e ### e  l  i
 r  e  s  c  i  s  s  i  o  n ### n  a  r  c ### e  h ######### u ###### c
 u ### m  a  d  m  e  n ###### t ### l  i  k  e  n  e  s  s  e  s ###### a
 s ###### w  r  o  n  g  d  o  e  r ### v ###### a  r  c  h  d  u  c  a  l
 t  a  d ### i  s ### l ### p  e  p  s  i ###### d ### c  o  n  r  a  i  l
 s  n  o  b  b  e  r  y ###### s  t  r  a  p  l  e  s  s ### a  p  t  l  y

S = [ultraist, glassies, domicil, lorinda, products, renature, twist,
     miraculists, teleran, reserves, yolks, advisement, tie, curse, teredo,
     luau, tan, put, sus, sir, reinless, en, essays, sri, clop, io, hee,
     ionize, ursine, men, swinge, roan, eve, leftist, sung, needles, verne,
     slut, ward, sidling, ensolite, alate, ears, win, plea, tenure, foresaw,
     adhering, intervenor, ir, il, logan, severable, rossi, spume, em, en,
     toon, steen, islet, diamonds, tattier, eli, rescission, narc, eh,
     madmen, likenesses, wrongdoer, archducal, tad, is, pepsi, conrail,
     snobbery, strapless, aptly, ultraspecialization, rusts, lowe, lo, len,
     de, an, trisection, pantheism, do, rise, pi, tue, macaw, antrorse, iv,
     errs, midrib, id, suspenses, eve, osmose, sameness, tri, evensen, is,
     ai, nd, unendingly, pr, yourself, or, so, grayness, rou, frat, op,
     loco, ri, san, neo, bot, tees, adulterating, gar, loan, rpt, sulk,
     regental, sr, scissors, eyeless, trivia, its, un, airsick,
     establishments, wrote, sd, unreeve, ow, serenade, alienee, liaise, her,
     detinue, dwindling, sccs, ones, splat, sho, maleness, ere, elsie, edna,
     item, sd, props, usurp, curettement, ligule, cat, iranian, enamel, ail,
     lenten, thermomagnetically]
```

Παραδοτέο για την άσκηση είναι **ένα πηγαίο αρχείο Prolog** με όνομα **`crossword.pl`**, μέσα στο οποίο **δεν** θα πρέπει να περιέχονται γεγονότα `dimension/1`, `black/2` και `words/1`.

Extra bonus 1 μονάδα στον τελικό βαθμό του μαθήματος: Στο password protected συμπιεσμένο αρχείο http://www.di.uoa.gr/~takis/crosshard.zip βρίσκονται τα δεδομένα για ένα σταυρόλεξο, αρκετά πιο δύσκολο από τα δοθέντα. Υπάρχει η δυνατότητα να αποσταλεί με e-mail στον διδάσκοντα, μέχρι τα μεσάνυχτα της ημέρας εξέτασης του μαθήματος τον Ιούνιο, βελτιωμένη έκδοση του προγράμματος που θα έχει υποβληθεί για αξιολόγηση εντός της αρχικής προθεσμίας. Ο συντάκτης του ταχύτερου προγράμματος που θα βρίσκει σωστή λύση για το σταυρόλεξο αυτό σε λιγότερο από ένα δευτερόλεπτο (σε Linux υπολογιστή του εργαστηρίου) θα λάβει το παραπάνω bonus στον τελικό βαθμό του. Το password θα ανακοινωθεί την επόμενη ημέρα της εξέτασης του μαθήματος.