ΠΡΟΓΡΑΜΜΑ ΣΠΟΥΔΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ	
Θεματική Ενότητα	**ΠΛΗ 31: Τεχνητή Νοημοσύνη - Εφαρμογές**
Ακαδημαϊκό Έτος	**2021 – 2022**
Γραπτή Εργασία	**#1**
Ημερομηνία Παράδοσης	**Παρασκευή, 10.12.2021, ώρα 23:59:59**

Παρατηρήσεις

Η καταληκτική ημερομηνία υποβολής της εργασίας είναι η Τετάρτη, *15 Δεκεμβρίου 2021*, ώρα 23:59:59, μετά από την οποία οποιαδήποτε υποβολή εργασίας θεωρείται ως εκπρόθεσμη.

Η παραπάνω ημερομηνία συνυπολογίζει την καθολική προκαταβολική έγκριση τυχόν αιτήματος παράτασης, κατά τον κανονισμό του ΕΑΠ (https://www.eap.gr/wp-content/uploads/2020/11/eap_kanonismos_spoydwn.pdf), με την υπόθεση πως η αρχικά οριζόμενη ημερομηνία παράδοσης θα ήταν η ακριβώς προηγούμενη Παρασκευή.

Την *Παρασκευή, 17 Δεκεμβρίου 2021, ώρα 11:59:59* (το αργότερο), θα δημοσιευθεί ενδεικτική επίλυση της εργασίας στο διαδίκτυο.

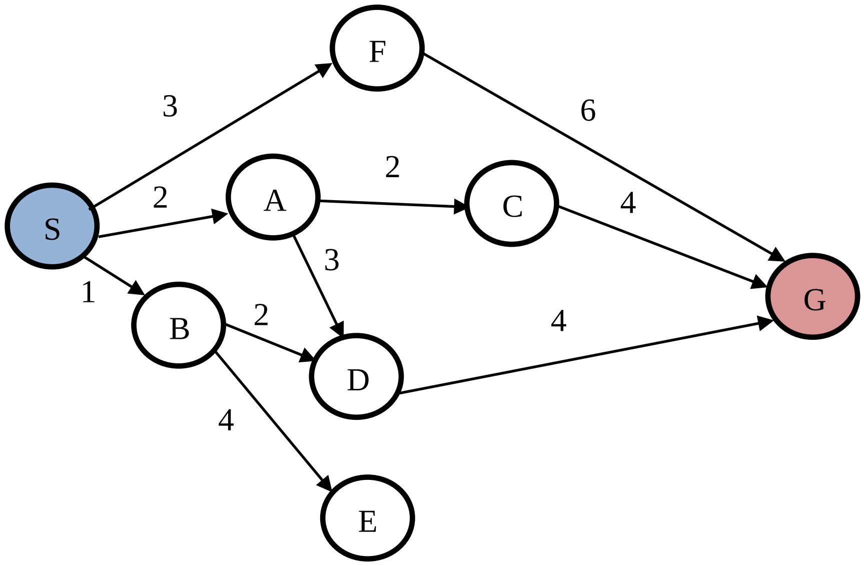
Περιμένουμε όλες οι εργασίες να υποβληθούν μέσω του συστήματος study. Πρέπει να υποβάλετε ΕΝΑ μόνο αρχείο (συμπιεσμένο), στο οποίο θα περιλαμβάνονται το πρωτότυπο κείμενο, σε οποιοδήποτε επεξεργαστή κειμένου έχετε χρησιμοποιήσει, και το παραγόμενο PDF, καθώς και όποια άλλα συνοδευτικά αρχεία. Αν υποβάλετε τμήματα κώδικα, θα πρέπει να βρίσκονται σε ξεχωριστά αρχεία, μέσα στο συμπιεσμένο αρχείο, και θα πρέπει ν' αναφέρονται στο κείμενο της εργασίας.

Αν υπάρχουν διευκρινιστικά ερωτήματα που θεωρείτε ότι είναι αναγκαία για την επίλυση της εργασίας και τα οποία δεν έχουν απαντηθεί, μπορείτε στην επίλυσή σας να τα αναφέρετε μαζί με τις παραδοχές που θα κάνετε και τις οποίες θα πρέπει να ακολουθήσετε στην επίλυσή σας.

ΘΕΜΑ	ΜΟΝΑΔΕΣ
ΠΕ1.Α	10
ΠΕ1.Β	10
1ο	30 (10 + 5+5 + 5+5)
2ο	15 (2 + 3 + 3 + 5 + 2)
3ο	15 (3 + 5 + 7)
4ο	10 (2 + 3 + 1 + 1+1+1)
5ο	10 (2 + 8)
ΣΥΝΟΛΟ	**100**

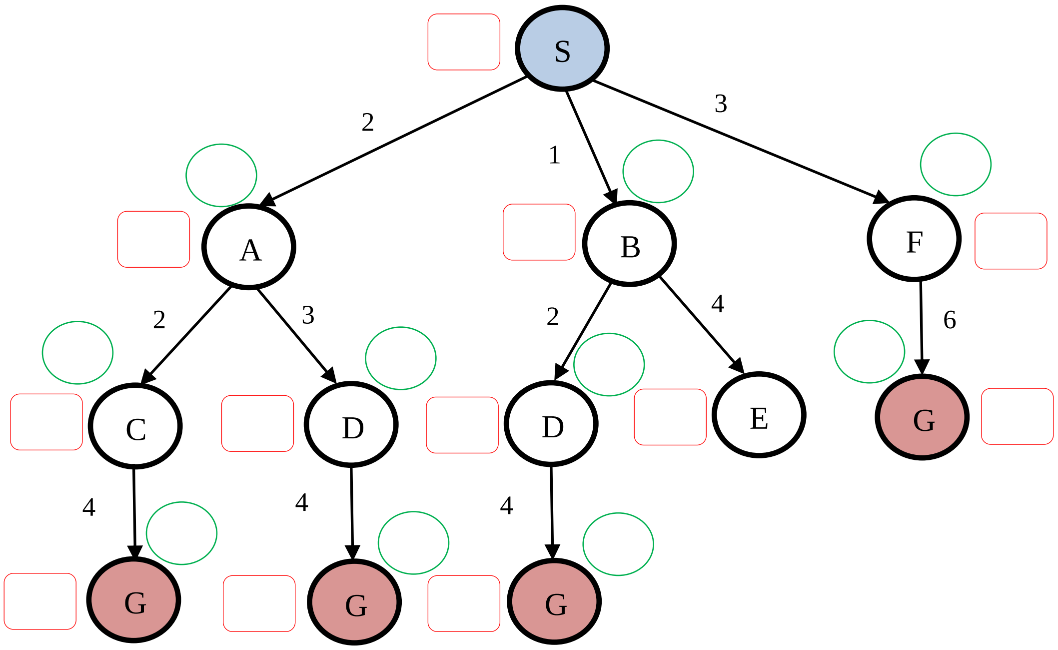
Θέμα 1: Ευρετική Αναζήτηση [*30 μονάδες*]

Θεωρήστε τον παρακάτω γράφο, ο οποίος αναπαριστά ένα δίκτυο πόλεων (S, A, B, C, D, E, F, G), που αποτελούν τους κόμβους του. Οι αριθμοί πάνω στις ακμές του γράφου αποτυπώνουν τις αποστάσεις μεταξύ των πόλεων σε κάποια μονάδα μέτρησης.



Το πρόβλημά μας είναι να μετακινηθούμε από την πόλη S στην πόλη G μέσω της βέλτιστης διαδρομής. Γι' αυτό προτείνεται να χρησιμοποιήσουμε τους ευρετικούς αλγορίθμους αναζήτησης **Best-First (Greedy Search)** και **A*** για να την βρούμε.

Προς διευκόλυνση, μας δίνεται το παρακάτω δέντρο, το οποίο είναι το **δέντρο αναζήτησης του προβλήματός** μας, δηλαδή όλες οι δυνατές διαδρομές/προσβάσεις από τον κόμβο S (ρίζα) στους υπόλοιπους κόμβους και στον κόμβο-στόχο G. Οι τιμές στους κλάδους είναι οι (πραγματικές) αποστάσεις μεταξύ των κόμβων-πόλεων. Τα κόκκινα ορθογώνια και οι πράσινοι κύκλοι είναι για να συμπληρώσετε μέσα τους στοιχεία της πορείας των αλγορίθμων.



Για τους συγκεκριμένους αλγόριθμους προτάθηκαν τα παρακάτω τέσσερα ευρετικά:

h1

S	A	B	C	D	E	F	G
6	4	5	1	2	8	4	0

h2

S	A	B	C	D	E	F	G
6	5	5	4	3	8	5	0

h3

S	A	B	C	D	E	F	G
6	6	5	4	4	8	5	0

h4

S	A	B	C	D	E	F	G
6	5	8	3	5	8	4	0

A. [10 *μονάδες*] Εξετάστε αν τα παραπάνω ευρετικά είναι παραδεκτά. Επίσης, συγκρίνετέ τα σε σχέση με την κυριαρχία (ποιο είναι πιο πληροφορημένο).

Απάντηση:

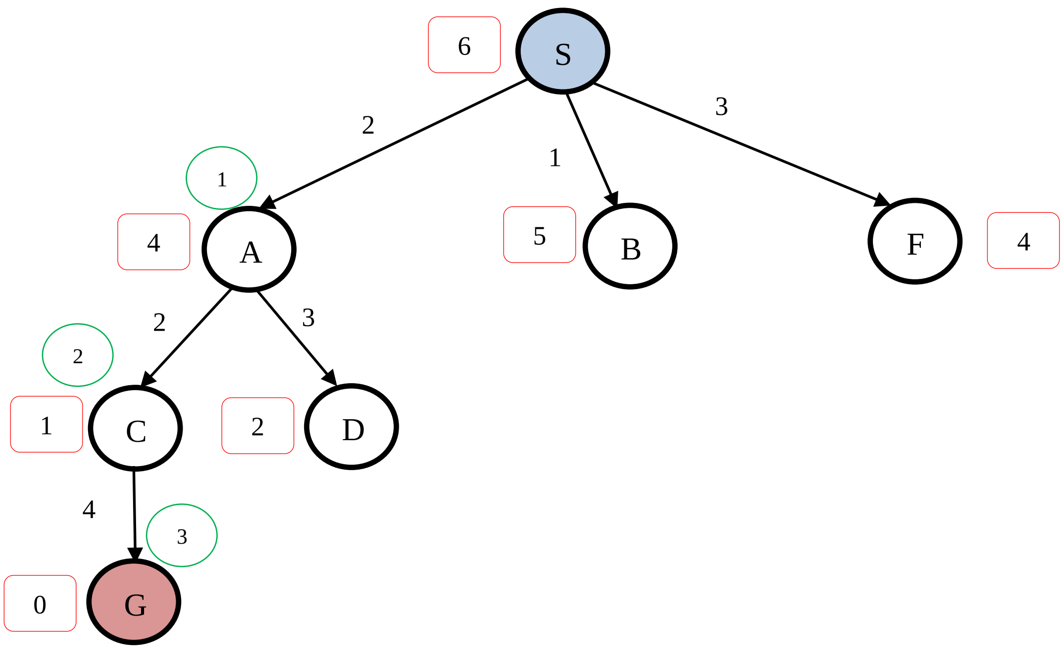
κόμβος	h*	h1	h2	h3	h4
S	**7**	**6**	**6**	**6**	**6**
A	**6**	**4**	**5**	**6**	**5**
B	**6**	**5**	**5**	**5**	**8**
C	**4**	**1**	**4**	**4**	**3**
D	**4**	**2**	**3**	**4**	**5**
E	**-**	**8**	**8**	**8**	**8**
F	**6**	**4**	**5**	**5**	**4**
G	**0**	**0**	**0**	**0**	**0**

Έπειτα από έλεγχο του δέντρου αναζήτησης ώστε να συγκρίνω την συνάρτηση πραγματικού κόστους με την ευρετική συνάρτηση παρατηρώ ότι παραδεκτό ευρετικό είναι το h1, h2 , h3 διότι για να είναι παραδεκτό πρέπει να ισχύει $0 \leq h \leq h^*$ για κάθε κόμβο. Το h4 δεν είναι παραδεκτό λόγω υπερεκτίμησης του κόμβου B είναι μεγαλύτερος από τη συνάρτηση πραγματικού κόστους.

Η πιο πληροφορημένη από τις τρείς παραδεκτές h1, h2, h3 είναι το h3 λόγο του ότι όλοι οι κόμβοι από και από τις τρει ευρετικές εκτιμήσεις είναι μεγαλύτεροι και συγκεκριμένα ο κόμβος A και D μεταξύ του h2 και του h3. Το h3 έχει μεγαλύτερες τιμές οι οποίες είναι πολύ πιο κοντά στη συναρτηση πραγματικού κόστους.

Β. [10 *μονάδες*] Εφαρμόστε τον αλγόριθμο **Best-First** χρησιμοποιώντας και τα τέσσερα ευρετικά.
(α) (5/10) Σχεδιάστε τα τέσσερα δέντρα εφαρμογής του αλγορίθμου (ένα για κάθε ευρετικό), χρησιμοποιώντας το δέντρο αναζήτησης που σας δόθηκε. Αφαιρέστε όσους κόμβους και όσους κλάδους δεν χρησιμοποιούνται και στα άδεια κόκκινα ορθογώνια καταγράψτε την τιμή της συνάρτησης αξιολόγησης. Επίσης, καταγράψτε τη σειρά ανάπτυξης των κόμβων (πλην του κόμβου στόχου) χρησιμοποιώντας τον μικρό πράσινο κύκλο πάνω από τον αντίστοιχο κόμβο. Προτεραιοποιήστε την παραγωγή παιδιών κόμβου λεξικογραφικά (Α→Ζ). Σε περίπτωση ισότιμων κόμβων, δώστε προτεραιότητα στον «παλιότερο» (δηλ. λεξικογραφικά πρώτο). Υπολογίστε το κόστος της διαδρομής σε κάθε δέντρο.

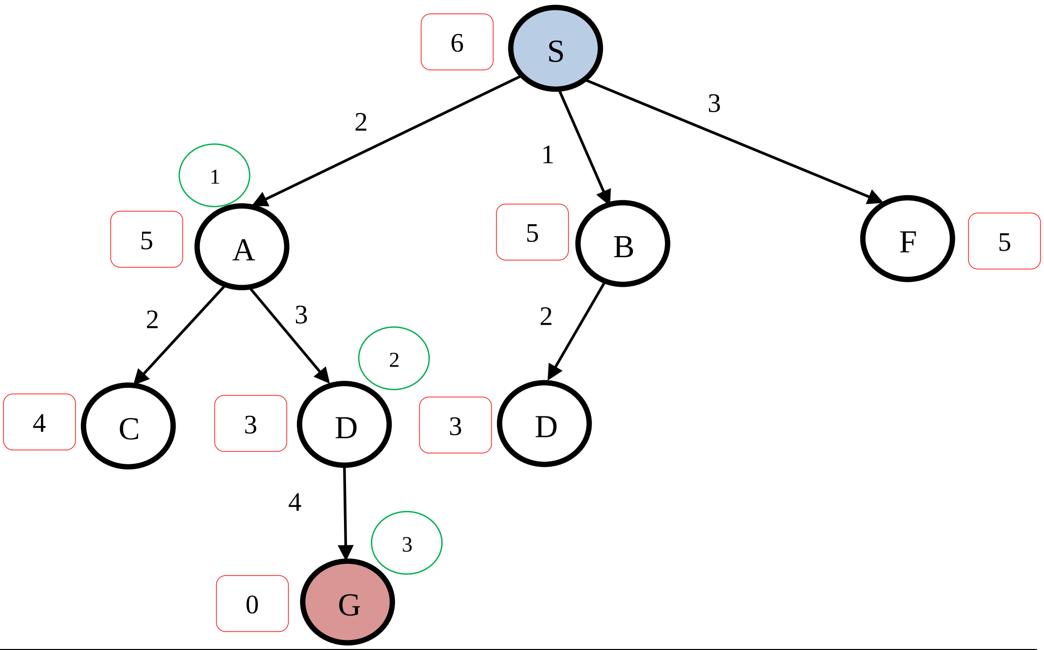
Απάντηση:
Για αλγόριθμο Best-First με ευρετικό h1:



Βήμα	Ανοιχτές	Κλειστές	Τρέχων
0	{S6}		
1	{A4,F4,B5}	{S6}	S6
2	{C1,D2,F4,B5}	{S6,A4}	A4
3	{G0,D2,F4,B5}	{S6,A4,C1}	C1
4	{D2,F4,B5}	{S6,A4,C1,G0}	G0

Μονοπάτι: S-A-C-G
Κόστος μονοπατιού: 8
Σειρά επίσκεψης: S-A-C-G

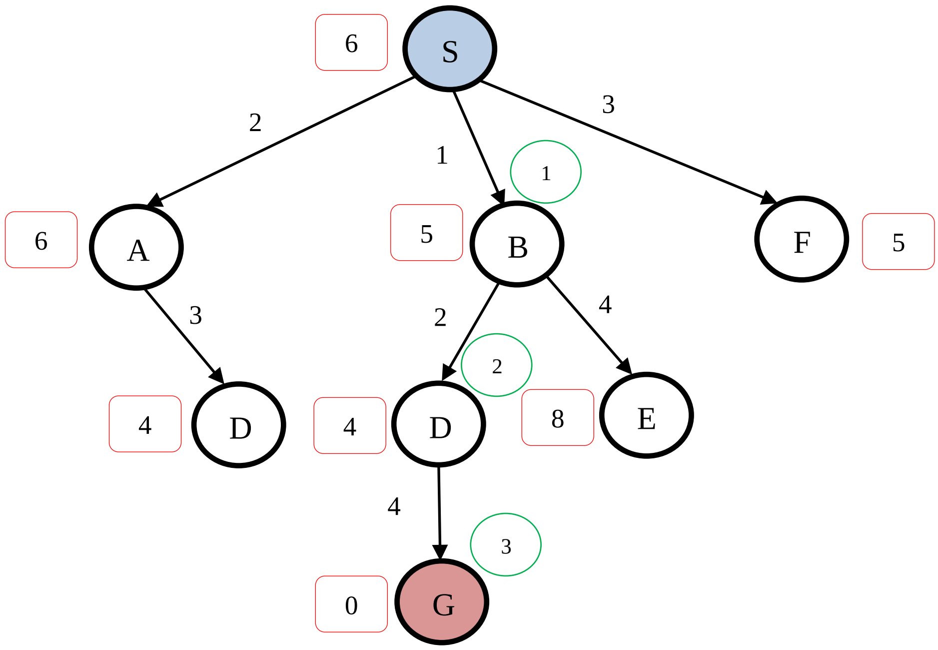
Για αλγόριθμο Best-First με ευρετικό h2:



Βήμα	Ανοιχτές	Κλειστές	Τρέχων
0	{S6}		
1	{A5,B5,F5}	{S6}	S6
2	{D3,C4,B5,F5}	{S6,A5}	A5
3	{G0,C4,B5,F5}	{S6,A5,D3}	D2
4	{C4,B5,F5}	{S6,A5,D3,G0}	G0

Μονοπάτι: S-A-D-G
Κόστος μονοπατιού: 9
Σειρά επίσκεψης: S-A-D-G

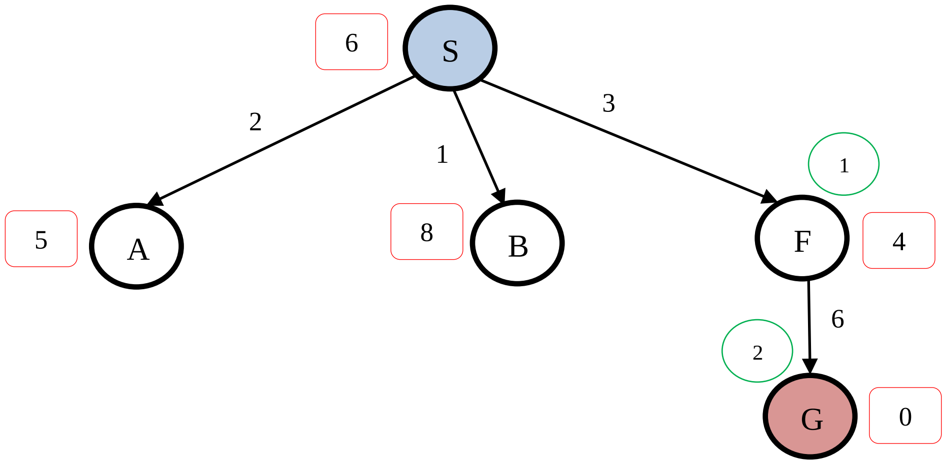
Για αλγόριθμο Best-First με ευρετικό h3:



Βήμα	Ανοιχτές	Κλειστές	Τρέχων
0	{S6}		
1	{B5,F5,A6}	{S6}	S6
2	{4D,F5,A6,E8}	{S6,B5}	B5
3	{G0,F5,A6,E8}	{S6,B5,4D}	D4
4	{F5,A6,E8}	{S6,B5,4D,G0}	G0

Μονοπάτι: S-B-D-G
Κόστος μονοπατιού: 7
Σειρά επίσκεψης: S-B-D-G

Για αλγόριθμο Best-First με ευρετικό h4:



Βήμα	Ανοιχτές	Κλειστές	Τρέχων
0	{S6}		
1	{F4,A5,B8}	{S6}	S6
2	{G0,A5,B8}	{S6,F4}	F4
3	{A5,B8}	{S6,F4,G0}	G0
4			

Μονοπάτι: S-F-G
Κόστος μονοπατιού: 9
Σειρά επίσκεψης: S-F-G

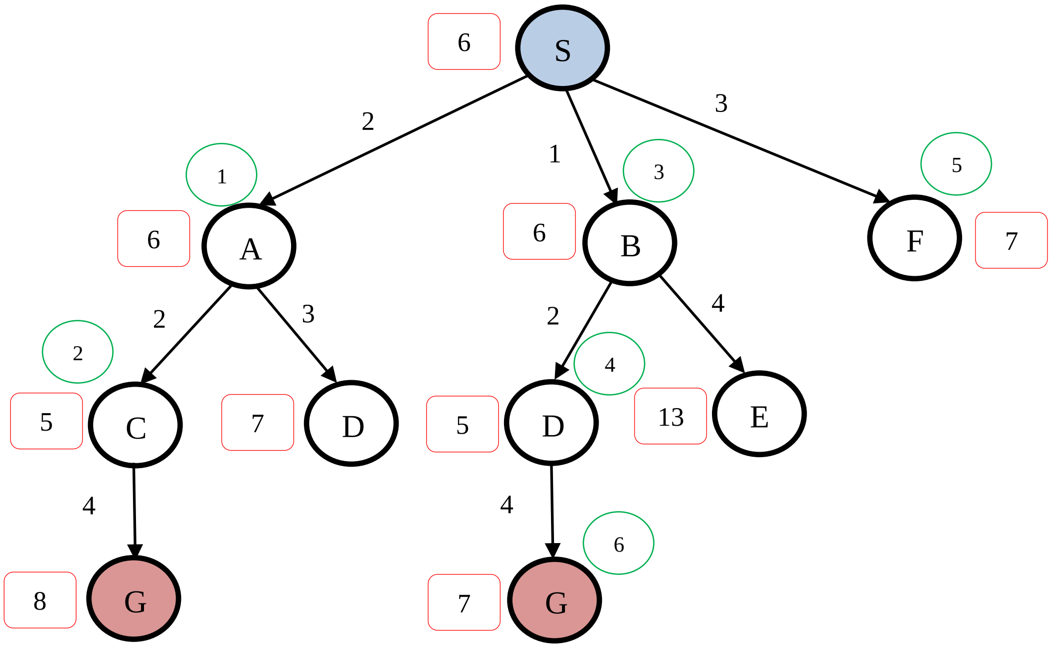
(β) (5/10) Συγκρίνετε τα τέσσερα δέντρα αναζήτησης ως προς την εύρεση της βέλτιστης διαδρομής και συζητήστε τις διαφορές τους με βάση τα ευρετικά.

Απάντηση:

Ο αλγόριθμος Best-First δεν είναι βέλτιστος αλγόριθμος αναζήτησης ουσιαστικά δεν εγγυάται τη βέλτιση λύση. Αυτό που παρατηρώ είναι ότι βρήκε την βέλτιστη διαδρομή επειδή το ευρετικό δεν είναι απλά παραδεκτό αλλά σε σχέση με τη κυριαρχία είναι το πιο πληροφορημένο.

Γ. [10 *μονάδες*] Εφαρμόστε τον αλγόριθμο **Α*** χρησιμοποιώντας και τα τέσσερα ευρετικά.
(α) (5/10) Σχεδιάστε τα τέσσερα δέντρα εφαρμογής του αλγορίθμου (ένα για κάθε ευρετικό), χρησιμοποιώντας το δέντρο αναζήτησης που σας δόθηκε, με τον τρόπο που περιγράφηκε παραπάνω. Υπολογίστε το κόστος της διαδρομής σε κάθε δέντρο.

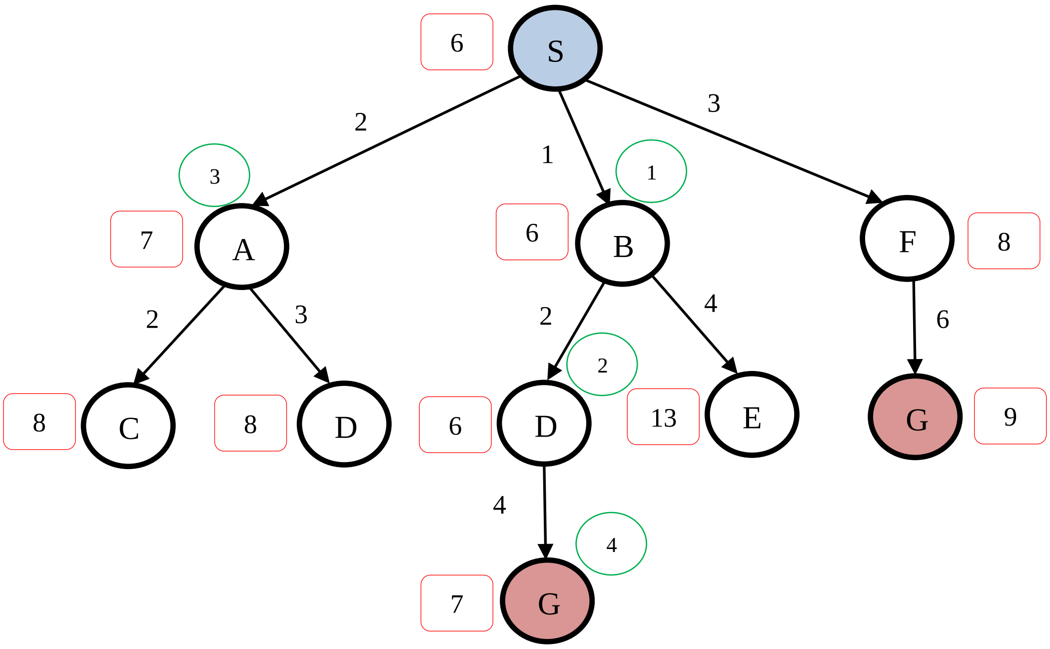
Απάντηση:
Για αλγόριθμο Α* με ευρετικό h1:



Βήμα	Ανοιχτές	Κλειστές	Τρέχων
0	S6		
1	{A6,B6,F7}	{S6}	S6
2	{C5,B6,D7,F7}	{S6,A6}	A6
3	{B6,D7,F7,G8}	{S6,A6,C5}	C5
4	{D5,F7,G8,E13}	{S6,A6,C5,B6}	B6
5	{F7,G7,E13}	{S6,A6,C5,B6,D5}	D5
6	{G7,E13}	{S6,A6,C5,B6,D5,F7}	F7
7	{E13}	{S6,A6,C5,B6,D5,F7,G7}	G7

Μονοπάτι: S-B-D-G
Κόστος μονοπατιού: 7
Σειρά επίσκεψης: S-A-C-B-D-F-G

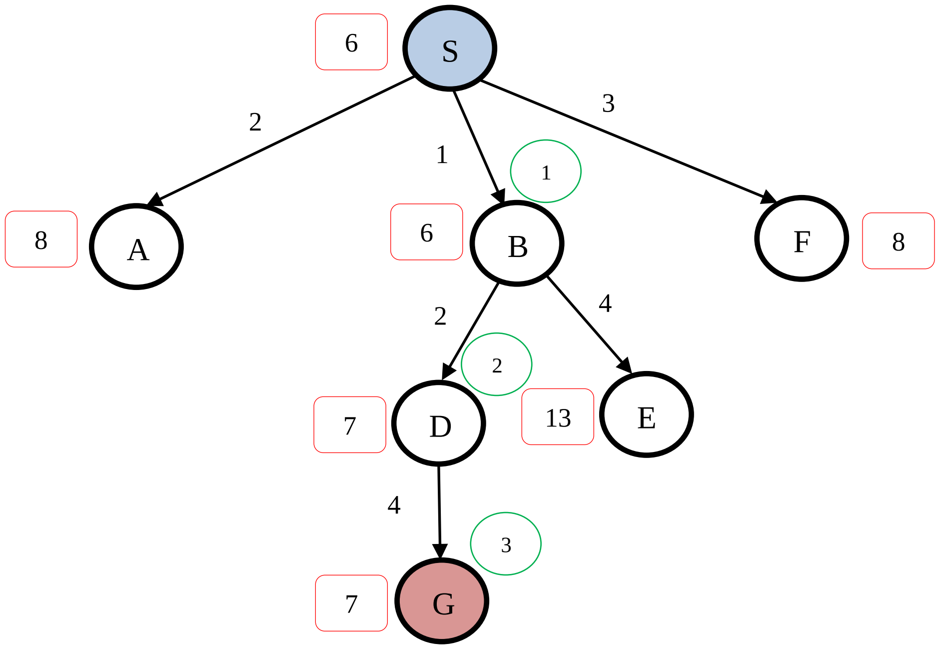
Για αλγόριθμο A* με ευρετικό h2:



Βήμα	Ανοιχτές	Κλειστές	Τρέχων
0	{S6}		
1	{B6,A7,F8}	{S6}	S6
2	{D6,A7,F8,E13}	{S6,B6}	B6
3	{A7,G7,F8,E13}	{S6,B6,D6}	D6
4	{G7,F8,G9,E13}	{S6,B6,D6,A7}	A7
5	{F8,E13}	{S6,B6,D6,A7,G7}	G7

Μονοπάτι: S-B-D-G
Κόστος μονοπατιού: 7
Σειρά επίσκεψης: S-B-D-A-G

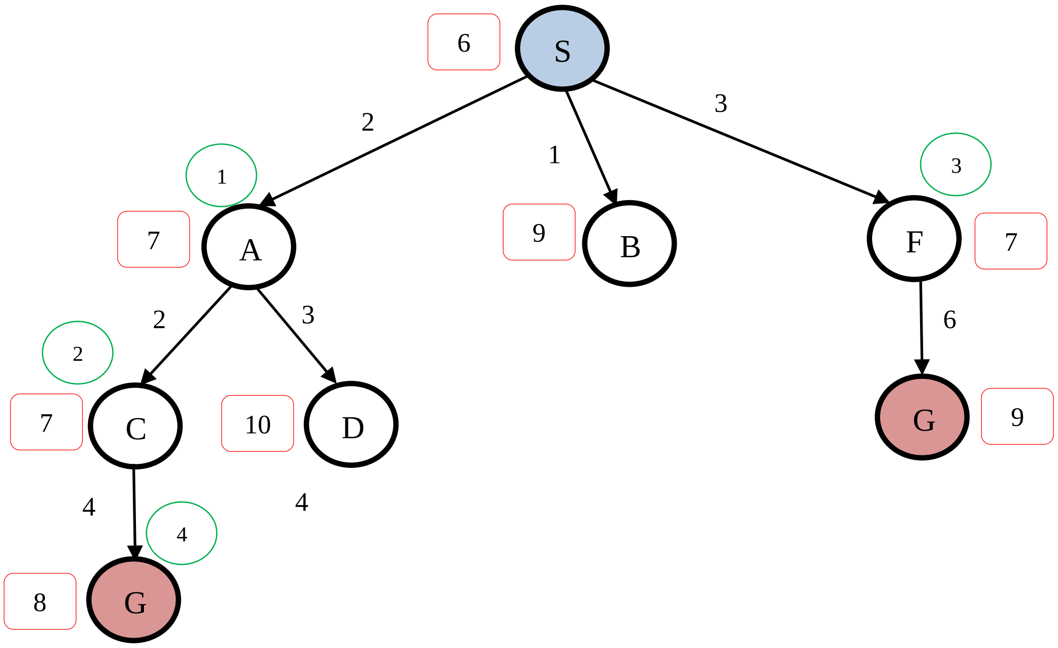
Για αλγόριθμο A* με ευρετικό h3:



Βήμα	Ανοιχτές	Κλειστές	Τρέχων
0	{S6}		
1	{B6,A8,F8}	{S6}	S6
2	{D7,A8,F8,E13}	{S6,B6}	B6
3	{G7,A8,F8,E13}	{S6,B6,D7}	D7
4	{A8,F8,E13}	{S6,B6,D7,G7}	G7

Μονοπάτι: S-B-D-G
Κόστος μονοπατιού: 7
Σειρά επίσκεψης: S-B-D-G

Για αλγόριθμο Α* με ευρετικό h4:



Βήμα	Ανοιχτές	Κλειστές	Τρέχων
0	{S6}		
1	{A7,F7,B9}	{S6}	S6
2	{C7,F7,B9,D10}	{S6,A7}	A7
3	{F7,G8,B9,D10}	{S6,A7,C7}	C7
4	{G8,B9,D10}	{S6,A7,C7,F7}	F7
5	{B9,D10}	{S6,A7,C7,F7,G8}	G8

Μονοπάτι: S-A-C-G
Κόστος μονοπατιού: 8
Σειρά επίσκεψης: S-A-C-F-G

(β) (5/10) Συγκρίνετε τα τέσσερα δέντρα αναζήτησης ως προς την εύρεση της βέλτιστης διαδρομής και συζητήστε τις διαφορές τους με βάση τα ευρετικά.

Απάντηση:

Γνωρίζοντας ότι ο αλγόριθμος Α* βρίσκει πάντα την βέλτιστη λύση όταν υπάρχει παραδεκτή ευρετική συνάρτηση. Αυτό που παρατηρώ όπως και έγινε και αντίστοιχη ανάπτυξη στο στο υπό ερώτημα α της παρούσας άσκησης: παραδεκτά ευρετικά είναι ο h1,h2,h3 όπου ο Α* βρήκε και την βέλτιστη λύση και συγκεκριμένα στο ευρετικό h3 βρήκε τη λύση απευθείας με τα λιγότερα δυνατά βήματα διότι ο h3 είναι πιο πληροφορημένος από τους υπόλοιπους δύο.
Το ευρετικό h4 δεν βρήκε την βέλτιση λύση διότι δεν είναι ούτε πληροφορημένο ούτε παραδεκτό λόγω υπερεκτίμησης του κόμβου Β ο οποίος είναι μεγαλύτερος από την πραγματική συνάρτηση.

Θέμα 2: Αναζήτηση υπό Περιορισμούς [*15 μονάδες*]

Έστω ότι έχετε αναλάβει να κατασκευάσετε το πρόγραμμα διαλέξεων μίας ημερίδας με έξι ομιλητές, τους (Α)λίκη, (Β)ασίλη, (Γ)ιώργο, (Δ)ήμητρα, (Ε)λένη και (Ζ)αχαρία, δηλαδή να καθορίσετε τη σειρά των ομιλητών. Οι διαλέξεις είναι ωριαίες και πρέπει να γίνουν μέσα σε ένα συνεχές εξάωρο, μόνο που πρέπει να γίνουν σεβαστοί οι περιορισμοί που έχουν θέσει κάποιοι ομιλητές:

Η Αλίκη **δεν** είναι διαθέσιμη την 1η και την 2η ώρα της ημερίδας. Ο Βασίλης και ο Γιώργος είναι διαθέσιμοι μόνο την 3η και την 4η ώρα. Η Δήμητρα **δεν** είναι διαθέσιμη την 1η και την 6η ώρα. Η Ελένη είναι διαθέσιμη μόνο την 2η, την 3η και την 4η ώρα. Ο Ζαχαρίας είναι διαθέσιμος όλες τις ώρες.

Έστω ότι αποφασίζουμε να επιλύσουμε το πρόβλημα μοντελοποιώντας το ως πρόβλημα ικανοποίησης περιορισμών και εφαρμόζοντας τις κατάλληλες μεθόδους που γνωρίζουμε από τη θεωρία. Για το σκοπό αυτό, ορίζουμε έξι μεταβλητές, τις Α, Β, Γ, Δ, Ε, Ζ (από τα αρχικά των ονομάτων των ομιλητών), οι οποίες έχουν ως αρχικό πεδίο το $\{1,2,3,4,5,6\}$. Η ερμηνεία της μοντελοποίησης αυτής είναι ότι αν τελικά κάποια από τις μεταβλητές αυτές πάρει την τιμή v μέσα από το πεδίο της, ο αντίστοιχος ομιλητής έχει προγραμματισθεί για την v-οστή ώρα της ημερίδας.

Α. Διατύπωση περιορισμών [2 *μονάδες*]
Χρησιμοποιώντας μόνο το σύμβολο $\neq$ (διάφορο), διατυπώστε όλους τους περιορισμούς του προβλήματος, συμπεριλαμβανομένων τόσο των περιορισμών διαθεσιμότητας κάθε ομιλητή, όσο και των προφανών περιορισμών που εκφράζουν την απαίτηση να μην προγραμματισθούν δύο ομιλίες την ίδια ώρα. Για παράδειγμα:

- $Α \neq 1$, $Α \neq 2$ (η Αλίκη δεν είναι διαθέσιμη την 1η και την 2η ώρα)
- ………………………………..

Απάντηση:

Περιορισμοί διαθεσιμότητας κάθε ομιλητή:

- $Α \neq 1$, $Α \neq 2$
- $Β \neq 1$, $Β \neq 2$, $Β \neq 5$, $Β \neq 6$
- $Γ \neq 1$, $Γ \neq 2$, $Γ \neq 5$, $Γ \neq 6$
- $Δ \neq 1$, $Δ \neq 6$
- $Ε \neq 1$, $Ε \neq 5$, $Ε \neq 6$

Περιορισμοί ώστε να μην προγραμματιστούν δύο ομιλίες την ίδια ώρα:

- $Α \neq Β$, $Α \neq Γ$, $Α \neq Δ$, $Α \neq Ε$, $Α \neq Ζ$
- $Β \neq Γ$, $Β \neq Δ$, $Β \neq Ε$, $Β \neq Ζ$
- $Γ \neq Δ$, $Γ \neq Ε$, $Γ \neq Ζ$
- $Δ \neq Ε$, $Δ \neq Ζ$
- $Ε \neq Ζ$

Β. Αρχική επιβολή συνέπειας τόξου [3 *μονάδες*]
Για την επίλυση του προβλήματος, σε ένα πρώτο βήμα, επιβάλλουμε συνέπεια τόξου (μέσω του αλγορίθμου AC-3) στην αρχική του κατάσταση. Χωρίς να δείξετε τα ενδιάμεσα βήματα εφαρμογής του AC-3, συμπληρώστε στον παρακάτω πίνακα τη στήλη «Βήμα 1» με τα νέα πεδία.

Απάντηση:

Μεταβλητή	Αρχική κατάσταση	Βήμα 1	Βημα 2	Βημα 3	Βημα 4	Βημα 5	Βημα 6
	Πεδίο						
Α	{1,2,3,4,5,6}	{3,4,5,6}					
Β	{1,2,3,4,5,6}	{3,4}					
Γ	{1,2,3,4,5,6}	{3,4}					
Δ	{1,2,3,4,5,6}	{2,3,4,5}					
Ε	{1,2,3,4,5,6}	{2,3,4}					
Ζ	{1,2,3,4,5,6}	{1,2,3,4,5,6}					

Γ. Έναρξη αναζήτησης [3 *μονάδες*]
Στη συνέχεια, πρέπει να επιλέξουμε μία μεταβλητή για την απόδοση τιμής σε αυτή μέσα από το πεδίο της. Έστω ότι αποφασίζουμε να ακολουθήσουμε τον ευρετικό κανόνα των *ελάχιστων απομενουσών τιμών* (minimum remaining values – MRV) ή της *συντομότερης αποτυχίας* (first fail – FF), όπως αλλιώς ονομάζεται. Σύμφωνα με τον κανόνα αυτό, πρέπει να επιλέξουμε τη μεταβλητή με τις λιγότερες δυνατές τιμές στο πεδίο της και να της αποδώσουμε μία τιμή μέσα από το πεδίο αυτό. Παρατηρήστε ότι στην κατάσταση των πεδίων μετά το βήμα 1, υπάρχουν δύο μεταβλητές με ελάχιστο πλήθος τιμών στα πεδία τους. Επιλέξτε εκείνη από τις δύο που προηγείται λεξικογραφικά και δώστε της τη μικρότερη τιμή μέσα από το πεδίο της. Αμέσως μετά, εφαρμόστε ~~πάλι τον αλγόριθμο επιβολής συνέπειας AC-3 και, χωρίς να δείξετε τα ενδιάμεσα βήματα του αλγορίθμου,~~ τον αλγόριθμο του προοπτικού/πρώιμου ελέγχου (forward checking – FC) και καταγράψτε στη στήλη «Βήμα 2» του πίνακα του ερωτήματος Β τη νέα κατάσταση των πεδίων.

Απάντηση:

Μεταβλητή	Αρχική κατάσταση	Βήμα 1	Βημα 2	Βημα 3	Βημα 4	Βημα 5	Βημα 6
	Πεδίο						
Α	{1,2,3,4,5,6}	{3,4,5,6}	{4,5,6}				
Β	{1,2,3,4,5,6}	{3,4}	{3}				
Γ	{1,2,3,4,5,6}	{3,4}	{4}				
Δ	{1,2,3,4,5,6}	{2,3,4,5}	{2,4,5}				
Ε	{1,2,3,4,5,6}	{2,3,4}	{2,4}				
Ζ	{1,2,3,4,5,6}	{1,2,3,4,5,6}	{1,2,4,5,6}				

Γ. Ολοκλήρωση αναζήτησης [5 *μονάδες*]
Συνεχίστε την αναζήτηση με την ίδια λογική (επαναλαμβανόμενες εφαρμογές MRV/FF και ~~AC-3~~ FC), συμπληρώνοντας κατάλληλα τις στήλες των βημάτων 3 έως 6. Ποια είναι η τελική λύση του προβλήματος;

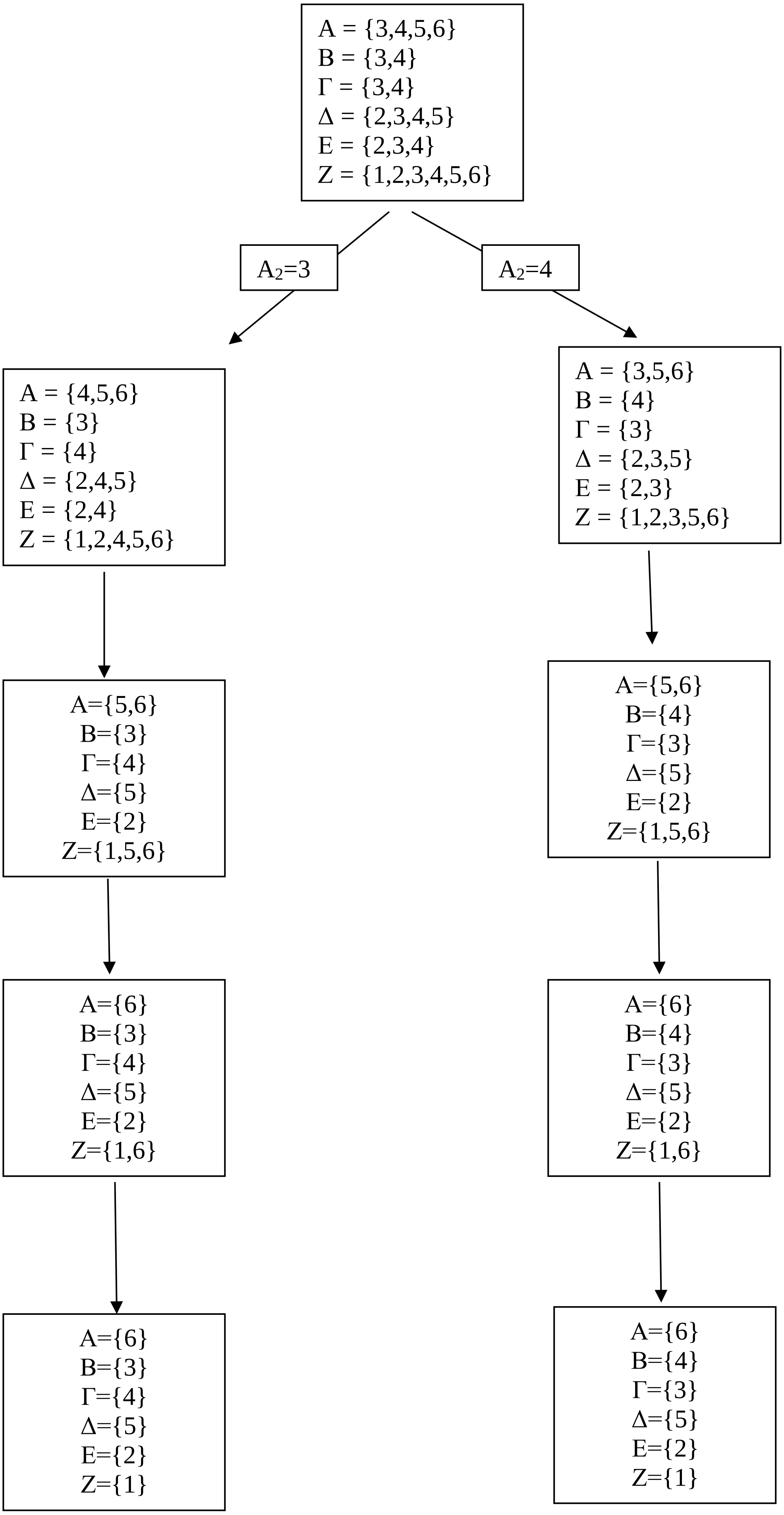
Απάντηση:

Μεταβλητή	Αρχική κατάσταση	Βήμα 1	Βημα 2	Βημα 3	Βημα 4	Βημα 5	Βημα 6
	Πεδίο						
Α	{1,2,3,4,5,6}	{3,4,5,6}	{4,5,6}	{5,6}	{5,6}	{6}	{6}
Β	{1,2,3,4,5,6}	{3,4}	{3}	{3}	{3}	{3}	{3}
Γ	{1,2,3,4,5,6}	{3,4}	{4}	{4}	{4}	{4}	{4}
Δ	{1,2,3,4,5,6}	{2,3,4,5}	{2,4,5}	{2,5}	{5}	{5}	{5}
Ε	{1,2,3,4,5,6}	{2,3,4}	{2,4}	{2}	{2}	{2}	{2}
Ζ	{1,2,3,4,5,6}	{1,2,3,4,5,6}	{1,2,4,5,6}	{1,2,5,6}	{1,5,6}	{1,6}	{1}

Δ. Άλλες πιθανές λύσεις [2 *μονάδες*]
Στη μεταβλητή που επιλέξατε στο ερώτημα Γ, αποδώσατε τη μικρότερη τιμή μέσα από το πεδίο της. Αν είχατε επιλέξει τη μεγαλύτερη τιμή, σε ποια λύση του προβλήματος θα καταλήγατε; Τελικά, πόσες δυνατές λύσεις έχει το συγκεκριμένο πρόβλημα;

Μεταβλητή	Αρχική κατάσταση	Βήμα 1	Βημα 2	Βημα 3	Βημα 4	Βημα 5	Βημα 6
	Πεδίο						
Α	{1,2,3,4,5,6}	{3,4,5,6}	{3,5,6}	{5,6}	{5,6}	{6}	{6}
Β	{1,2,3,4,5,6}	{3,4}	{4}	{4}	{4}	{4}	{4}
Γ	{1,2,3,4,5,6}	{3,4}	{3}	{3}	{3}	{3}	{3}
Δ	{1,2,3,4,5,6}	{2,3,4,5}	{2,3,5}	{2,5}	{5}	{5}	{5}
Ε	{1,2,3,4,5,6}	{2,3,4}	{2,3}	{2}	{2}	{2}	{2}
Ζ	{1,2,3,4,5,6}	{1,2,3,4,5,6}	{1,2,3,5,6}	{1,2,5,6}	{1,5,6}	{1,6}	{1}

Το πρόβλημα έχει δύο (2) δυνατές λύσεις όσες είναι και οι ελάχιστες τιμές των μεταβλητών Β και Γ. Μόλις επιλεχθούν οι τιμές αυτών των μεταβλητών αφαιρούνται από το πεδίο τιμών όλων των υπολοίπων μεταβλητών και δεν μας δίνεται άλλη δυνατή λύση σε οποιαδήποτε παραλλαγή.
Όπως και παρουσιάζω την ανάπτυξη των επιλογών στο επόμενο σχέδιο:


A = {3,4,5,6}
B = {3,4}
Γ = {3,4}
Δ = {2,3,4,5}
Ε = {2,3,4}
Ζ = {1,2,3,4,5,6}
A_2=3
A_2=4
A = {4,5,6}
B = {3}
Γ = {4}
Δ = {2,4,5}
Ε = {2,4}
Ζ = {1,2,4,5,6}
A = {3,5,6}
B = {4}
Γ = {3}
Δ = {2,3,5}
Ε = {2,3}
Ζ = {1,2,3,5,6}
A={5,6}
B={3}
Γ={4}
Δ={5}
Ε={2}
Ζ={1,5,6}
A={5,6}
B={4}
Γ={3}
Δ={5}
Ε={2}
Ζ={1,5,6}
A={6}
B={3}
Γ={4}
Δ={5}
Ε={2}
Ζ={1,6}
A={6}
B={4}
Γ={3}
Δ={5}
Ε={2}
Ζ={1,6}
A={6}
B={3}
Γ={4}
Δ={5}
Ε={2}
Ζ={1}
A={6}
B={4}
Γ={3}
Δ={5}
Ε={2}
Ζ={1}

Θέμα 3: Αναζήτηση σε Παίγνια [*15 μονάδες*]

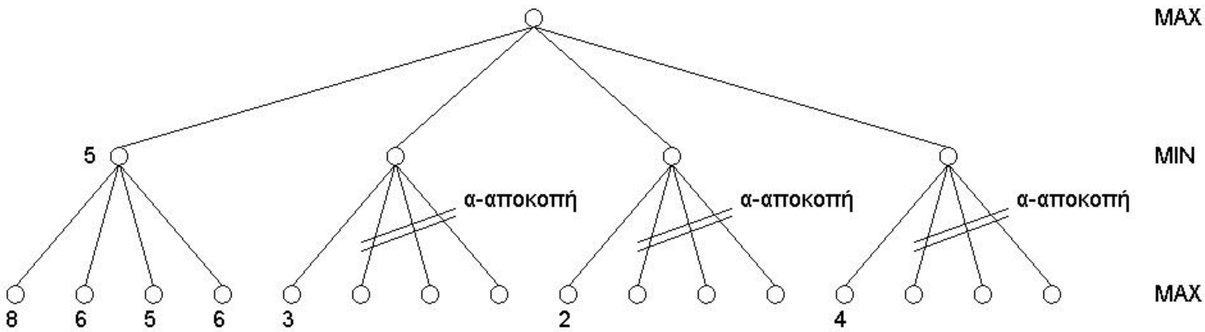
Είναι γνωστό ότι σε ένα δέντρο αναζήτησης παιγνιδιού που έχει παράγοντα διακλάδωσης b και έχει αναπτυχθεί μέχρι βάθους d, εφαρμόζοντας τη μέθοδο minimax, η ευρετική συνάρτηση αξιολόγησης πρέπει να εφαρμοσθεί σε όλους του κόμβους του επιπέδου βάθους d, οι οποίοι είναι πλήθους b^d.

Στο βιβλίο «Τεχνητή Νοημοσύνη», Β΄ έκδοση, Βλαχάβας Ι., κ.ά., αναφέρεται (παράγραφος 5.4, σελ. 79-80) ότι στην περίπτωση που το δέντρο είναι τέλεια διατεταγμένο, δηλαδή ο αριστερότερος απόγονος κάθε κόμβου είναι ο καλύτερος, τότε, αν εφαρμόσουμε τη μέθοδο alpha-beta, το πλήθος των κόμβων του επιπέδου βάθους d στο οποίο πρέπει να εφαρμοσθεί η ευρετική συνάρτηση αξιολόγησης ισούται με:

$$2 \cdot b^{\frac{d}{2}} - 1, \qquad \alpha\nu\ d\ \acute{\alpha}\rho\tau\iota o\varsigma \qquad (1)$$

$$b^{\frac{d+1}{2}} + b^{\frac{d-1}{2}} - 1, \quad \alpha\nu\ d\ \pi\varepsilon\rho\iota\tau\tau\acute{o}\varsigma \qquad (2)$$

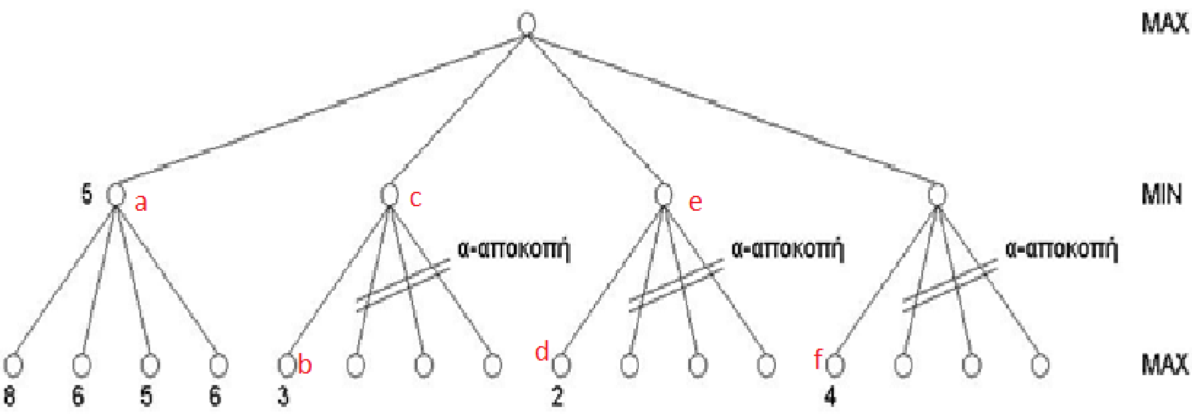
Μπορούμε να επιβεβαιώσουμε τον τύπο (1) για την περίπτωση ενός δέντρου παιγνιδιού με $b = 4$ και $d = 2$, όπως φαίνεται στο επόμενο σχήμα.



Παρατηρήστε ότι στο δέντρο αυτό, η ευρετική συνάρτηση αξιολόγησης εφαρμόστηκε σε $2 \cdot 4^{\frac{2}{2}} - 1 =$ 7 κόμβους, δηλαδή ο τύπος (1) παραπάνω δίνει σωστό αποτέλεσμα.

Α. Συνθήκες α-αποκοπών [3 *μονάδες*]
Για ποιους λόγους έγιναν οι α-αποκοπές στο προηγούμενο δέντρο; Δηλαδή, ποιες συνθήκες ελέγχθηκαν σε κάθε περίπτωση και επιβεβαιώθηκαν ως αληθείς;

<u>Απάντηση:</u>



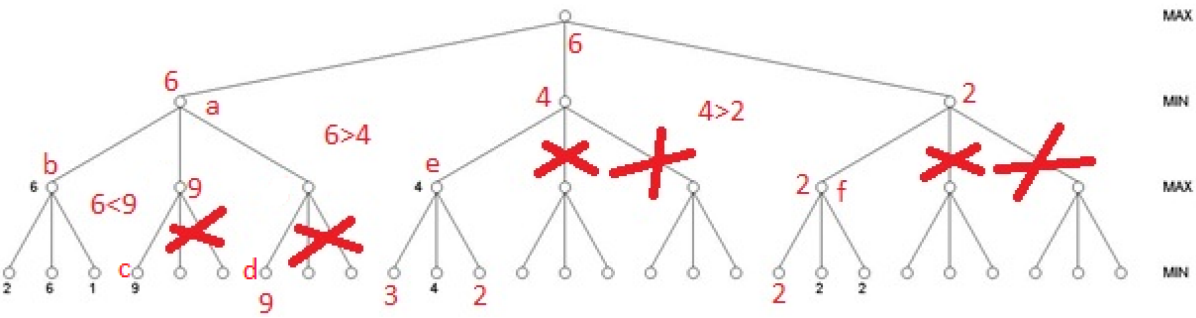
Για τις τρεις αποκοπές του συγκεκριμένου σχήματος ονόμασα τους κόμβους για περισσότερη ευκολία στην επεξήγηση:

1 . έγινε α-αποκοπή διότι ο κόμβος a παίρνοντας την μικρότερη τιμή από τα παιδιά του και συγκρίνοντας το με το παιδί του αδελφικού του κόμβου δηλαδή τον κόμβο b ο οποίος ο κόμβος a είναι μεγαλύτερος ή ίσος από τον κόμβο b. Άρα a > b δηλαδή 5 > 3

2.έγινε α-αποκοπή διότι ο κόμβος c παίρνοντας την μικρότερη και μοναδική τιμή το 3 από τα παιδιά του και συγκρίνοντας το με το παιδί του αδελφικού του κόμβου δηλαδή τον κόμβο d ο οποίος ο κόμβος c είναι μεγαλύτερος ή ίσο από τον κόμβο d. Άρα c > d δηλαδή 3 > 2.

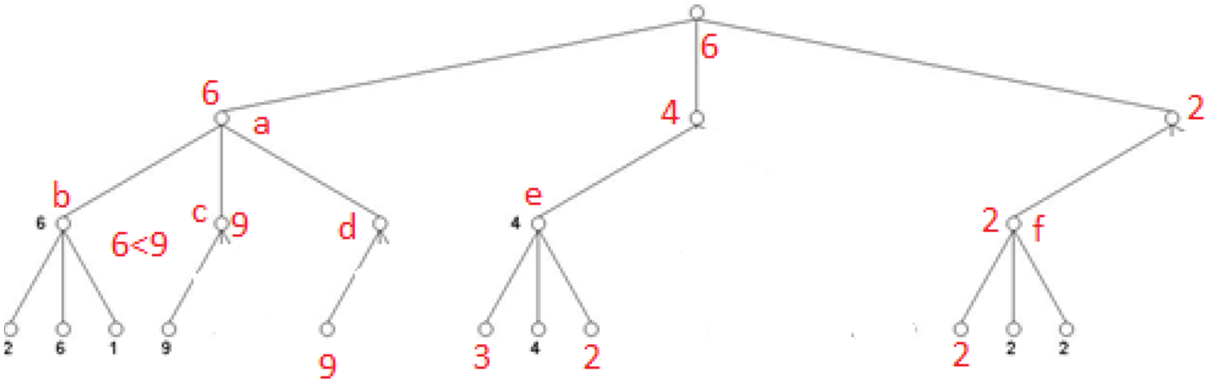
3.έγινε α-αποκοπή διότι ο κόμβος a παίρνοντας την μικρότερη τιμή τι 5 από τα παιδιά του και συγκρίνοντας το με το παιδί του αδελφικού του κόμβου δηλαδή τον κόμβο f ο οποίος ο κόμβος a είναι μεγαλύτερος ή ίσο από τον κόμβο f. Άρα a > f δηλαδή 5 > 4.

Β. Δέντρο με $b = 3, d = 3$ [5 *μονάδες*]
Συμπληρώστε στο επόμενο δέντρο παιγνιδιού κατάλληλες τιμές της ευρετικής συνάρτησης αξιολόγησης στους κόμβους του τελευταίου επιπέδου, **όπου απαιτείται**, ώστε η μέθοδος alpha-beta να κάνει τις περισσότερες δυνατές αποκοπές, δείξτε ποιες αποκοπές και τι τύπου είναι αυτές (α- ή β-) και επιβεβαιώστε την ισχύ του τύπου (2) προηγουμένως ως προς το πλήθος των κόμβων του τελευταίου επιπέδου που πρέπει να εφαρμοσθεί η ευρετική συνάρτηση αξιολόγησης, σε κάθε περίπτωση, από τη μέθοδο alpha-beta.



Θα γίνουν 4 αποκοπές
1. αποκοπή περίπτωση B διότι ο κόμβος b είναι μικρότερος ή ίσος από τον κόμβο c
2. αποκοπή περίπτωση B διότι ο κόμβος b είναι μικρότερος ή ίσος από τον κόμβο d
3. αποκοπή περίπτωση A διότι ο κόμβος a είναι μεγαλύτερος ή ίσος από τον κόμβο e
4. αποκοπή περίπτωση A διότι ο κόμβος a είναι μεγαλύτερος ή ίσος από τον κόμβο f

Άρα θα είναι το εξής σχήμα:

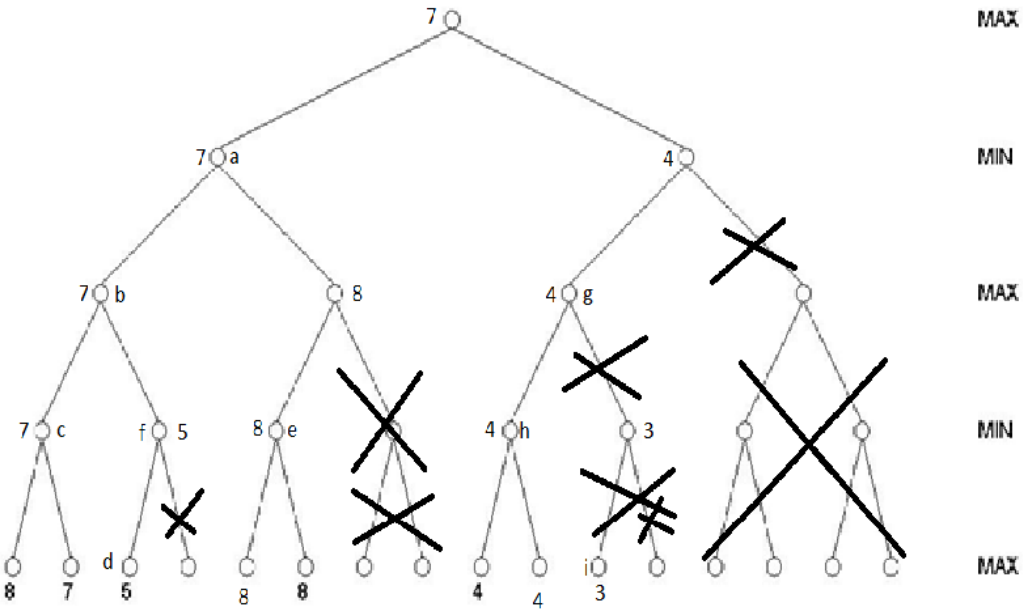


Μετρώντας με τα εναπομείναντα φύλα και επαληθεύω με βάση τον τύπο για d= περιττός αριθμός:

Άρα επαληθεύω με τον τύπο ότι ισχύει.

$$b^{\frac{d+1}{2}} + b^{\frac{d-1}{2}} - 1 = 3^{\frac{3+1}{2}} + 3^{\frac{3-1}{2}} - 1 = 3^2 + 3 - 1 = 9 + 3 - 1 = 12 - 1 = 11$$

Γ. Δέντρο με $b = 2, d = 4$ [7 *μονάδες*]
Ομοίως με το προηγούμενο ερώτημα για το επόμενο δέντρο παιγνιδιού. Στην περίπτωση αυτή ζητείται, προφανώς, η επιβεβαίωση του τύπου (1), αφού το d είναι άρτιος.



Θα γίνουν 4 αποκοπές
1. α-αποκοπή περίπτωση A διότι ο κόμβος c είναι μεγαλύτερος ή ίσος από τον κόμβο d
2. β-αποκοπή περίπτωση B διότι ο κόμβος b είναι μικρότερος ή ίσος από τον κόμβο e

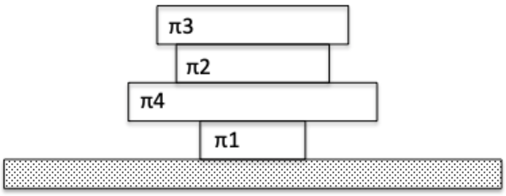
3. α-αποκοπή περίπτωση Α διότι ο κόμβος h είναι μεγαλύτερος ή ίσος από τον κόμβο i
4. αποκοπή περίπτωση Α διότι ο κόμβος a είναι μεγαλύτερος ή ίσος από τον κόμβο g

Μετρώντας με τα εναπομείναντα φύλα και επαληθεύω με βάση τον τύπο για d= ζυγός αριθμός:

Επαληθεύω τη σχέση με τον τύπο: $2 \cdot b^{\frac{d}{2}} - 1 = 2 \cdot 2^{\frac{4}{2}} - 1 = 2 \cdot 2^2 - 1 = 8 \cdot 1 = 7$

Θέμα 4: Αναπαράσταση Γνώσης και Συμπερασματολογία [*10 μονάδες*]

Οι παρακάτω προτάσεις κατηγορηματικής λογικής αφορούν μια στοίβα από 4 πίτες π1, π2, π3, π4, κάθε μια από τις οποίες έχει διάμετρο 1, 2, 3, και 4 αντίστοιχα. Η στοίβα με τις πίτες βρίσκεται πάνω σε ένα τραπέζι (Σχήμα 1).[1]



Σχήμα 1

1. ¬(π1 = π2) ∧ ¬(π1 = π3) ∧ ¬(π1 = π4) ∧ ¬(π2 = π3) ∧ ¬(π2 = π4) ∧ ¬(π3 = π4)
2. ∀x(pie(x) ⇔ (x=π1 ∨ x=π2 ∨ x=π3 ∨ x=π4))
3. ∀x (pie(x) ⇒ ¬(x = table))
4. ∀x∀y (on(x, y) ⇒ ¬(x = y) ∧ pie(x) ∧ (pie(y) ∨ y=table))
5. ∀x (pie(x) ⇒ ∃y(on(x, y)))
6. ∀x∀y∀z (on(x, y) ∧ on(x, z) ⇒ y=z)
7. ∀x∀y∀z (on(x, z) ∧ on(y, z) ⇒ (x=y ∨ z=table))
8. ∀x (clear(x) ⇔ ¬∃y (on(y, x)))

(Σημείωση: Σε πολλές προτάσεις κάνουμε χρήση του κατηγορήματος ισότητας.)
Οι πρώτες τρεις προτάσεις προσδιορίζουν τα αντικείμενα του “μικρόκοσμου” με τις πίτες. Συγκεκριμένα δηλώνουν πως υπάρχουν 4 διακριτές πίτες που αναπαριστώνται από τις σταθερές π1, π2, π3, π4, και ένα τραπέζι (που δεν είναι πίτα) και αναπαριστάται από την σταθερά table.

Οι υπόλοιπες προτάσεις περιγράφουν την “φυσική” του μικρόκοσμου. Συγκεκριμένα,
- η (4) δηλώνει πως μια πίτα μπορεί να τοποθετηθεί είτε πάνω σε άλλη πίτα ή πάνω στο τραπέζι.
- η (5) δηλώνει πώς κάθε πίτα είναι πάνω σε κάποιο αντικείμενο.
- η (6) δηλώνει πώς μια πίτα δεν μπορεί να βρίσκεται ταυτόχρονα πάνω σε δύο διαφορετικά αντικείμενα.
- η (7) δηλώνει πώς πάνω σε μια πίτα μπορούμε να τοποθετήσουμε το πολύ ένα αντικείμενο.[2]
- η (8) ορίζει πως ένα αντικείμενο θεωρείται “καθαρό” (clear) όταν δεν έχει τίποτε άλλο από πάνω του.

1.Α. [2 *μονάδες*] Να μετατρέψετε τις προτάσεις (1) – (8) σε Συζευκτική Κανονική Μορφή (ΣΚΜ). Για να δεικτοδοτήσετε τις προτάσεις χρησιμοποιήστε το συμβολισμό 1.α, 1.β, 2.α, κλπ (αν για κάποια από τις προτάσεις (1) – (8) προκύπτουν >1 προτάσεις σε ΣΚΜ). Επισημαίνεται πως κατά την μετατροπή των προτάσεων (1) – (8) σε ΣΚΜ θα χρειαστεί σε δύο περιπτώσεις να εφαρμόσετε

[1] Στο θέμα 5 θα δούμε μια παραλλαγή του “μικρόκοσμου” με τις πίτες που επιτρέπει και την αναδιάταξη μιας στοίβας με την χρήση σπάτουλας.
[2] Το κατηγόρημα on(x, y) αφορά στο αντικείμενο x που βρίσκεται *ακριβώς* πάνω από το y (δηλαδή, που ακουμπάει το y). Για παράδειγμα για το Σχήμα 1 ισχύουν τα on(π2,π4) και on(π4,π1), αλλά δεν ισχύει το on(π2,π1).

Skolemization απ' όπου θα προκύψουν δύο νέες συναρτήσεις. Τις δύο νέες αυτές συναρτήσεις θα πρέπει να τις ονομάσετε *f* και *g*.

Οι ακόλουθες εκφράσεις δίνονται έτοιμες για να διευκολύνουν την προσέγγισή σας:

1.α $\{ \neg(\pi1 = \pi2) \}$
1.β $\{ \neg(\pi1 = \pi3) \}$
1.γ $\{\neg(\pi1 = \pi4) \}$
1.δ. $\{ \neg(\pi2 = \pi3) \}$
1.ε $\{ \neg(\pi2 = \pi4) \}$
1.ζ $\{ \neg(\pi3 = \pi4) \}$

4.α $\{ \neg on(x, y), \neg(x = y) \}$
4.β $\{ \neg on(x, y), pie(x) \}$
4.γ $\{ \neg on(x, y), pie(y), y=table \}$

1.Β. [8 *μονάδες*] Τα αξιώματα της ισότητας για την παραπάνω αναπαράσταση του μικρόκοσμου με τις πίτες φαίνονται παρακάτω, σε ΚΛ και σε Συζευκτική Κανονική Μορφή (ΣΚΜ). Σημειώνεται πως τα αξιώματα ισότητας (12) και (13) αφορούν τα νέα συναρτησιακά σύμβολα που προκύπτουν μέσω Skolemization από την μετατροπή των προτάσεων (1) – (8) σε ΣΚΜ.

Σε ΚΛ:
9. $\forall x(x=x)$
10. $\forall x\forall y (x=y \Rightarrow y=x)$
11. $\forall x\forall y\forall z (x=y \wedge y=z \Rightarrow x=z)$

12. $\forall x\forall y (x=y \Rightarrow f(x) = f(y))$
13. $\forall x\forall y (x=y \Rightarrow g(x) =g(y))$
14. $\forall x\forall y (x=y \Rightarrow (pie(x) \Leftrightarrow pie(y)))$
15. $\forall x\forall y (x=y \Rightarrow (clear(x) \Leftrightarrow clear(y)))$
16. $\forall x\forall y (x_1=y_1 \wedge x_2=y_2 \Rightarrow (on(x_1,x_2) \Leftrightarrow on(y_1,y_2)))$

Σε ΣΚΜ:
9. $\{ x=x \}$
10. $\{ \neg(x=y), y=x \}$
11. $\{ \neg(x=y), \neg(y=z), x=z \}$
12. $\{ \neg(x=y), f(x) = f(y) \}$
13. $\{ \neg(x=y), g(x) = g(y) \}$
14.α $\{ \neg(x=y), \neg pie(x), pie(y) \}$
14.β $\{ \neg(x=y), pie(x), \neg pie(y) \}$
15.α $\{ \neg(x=y), \neg clear(x), clear (y) \}$
15.β $\{ \neg(x=y), clear(x), \neg clear (y) \}$
16.α $\{ \neg(x_1=y_1), \neg(x_2=y_2), \neg on(x_1,x_2), on(y_1,y_2) \}$
16.β $\{ \neg(x_1=y_1), \neg(x_2=y_2), on(x_1,x_2), \neg on(y_1,y_2) \}$

Να αποδείξετε με την μέθοδο της αναγωγής μέσω αντίκρουσης της άρνησης πως από τις προτάσεις (1) – (16), προκύπτει ταυτολογικά η πρόταση $on(\pi4,\pi1) \Rightarrow \neg on(\pi4,table)$.

Απαντήσεις:
Υπό ερώτημα 1) Α)

Πρόταση 1:

Από την εκφώνηση της άσκησης παίρνω τα εξής για την 1η πρόταση

1α)$\neg(\Pi 1 = \Pi 2)$

1β) $\neg(\Pi 1 = \Pi 3)$

1γ) $\neg(\Pi 1 = \Pi 4)$

1δ) $\neg(\Pi 2 = \Pi 3)$

1ε) $\neg(\Pi 2 = \Pi 4)$

1ζ) $\neg(\Pi 3 = \Pi 4)$

Πρόταση 2:

Για την απαλοιφή της ισοδυναμίας θα χρησιμοποιήσω από τον νόμο αντικατάστασης: $(\varphi \leftrightarrow \psi) \equiv (\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)$. Άρα:

$\forall$x(pie(x)$\leftrightarrow$(x=π1$\vee$ x=π2 $\vee$ x=π3 $\vee$ x=π4) ≡ $\forall$x((pie(x)$\rightarrow$(x=π1$\vee$ x=π2$\vee$ x=π3$\vee$ x=π4)$\wedge$(x=π1$\vee$ x=π2 $\vee$ x=π3 $\vee$ x=π4)$\rightarrow$pie(x))

Έπειτα θα διώξω τις συνεπαγωγές:

$\forall$x[$\neg$pie(x)$\vee$ x=π1$\vee$ x=π2 $\vee$ x=π3 $\vee$ x=π4 $\wedge$ ($\neg$(x=π1$\vee$ x=π2 $\vee$ x=π3 $\vee$ x=π4)$\vee$pie(x))]

Κανόνας De Morgan:

$\forall$x[$\neg$pie(x)$\vee$(x=π1$\vee$ x=π2 $\vee$ x=π3 $\vee$ x=π4$\wedge$($\neg$(x=π1)$\wedge$ $\neg$(x=π2) $\wedge$ $\neg$(x=π3) $\wedge$$\neg$(x=π4)$\vee$pie(x)]

Επιμερισμός A$\vee$(B$\wedge$C)≡(A$\vee$B)$\wedge$(A$\vee$C):

$\forall$x[$\neg$pie(x)$\vee$ x=π1$\vee$ x=π2 $\vee$ x=π3 $\vee$ x=π4$\wedge$(pie(x)$\vee$$\neg$(x=π1))$\wedge$ (pie(x)$\vee$$\neg$(x=π2))$\wedge$ (pie(x)$\vee$$\neg$(x=π3))$\wedge$ (pie(x)$\vee$$\neg$(x=π4))]

Απαλοιφή καθολικών ποσοδεικτών και μετονομασία μεταβλητών και σταθερών:

$\neg$pie(x1)$\vee$ x1=Π1$\vee$ x1=Π2 $\vee$ x1=Π3 $\vee$ x1=Π4

pie(x2)$\vee$$\neg$(x2 = Π1)

pie(x3)$\vee$$\neg$(x3 = Π2)

pie(x4)$\vee$$\neg$(x4 = Π3)

pie(x5)$\vee$$\neg$(x5 = Π4)

Πρόταση 3:

$\forall$x(pie(x)$\rightarrow$$\neg$(x=table))

Απαλοιφή συνεπαγωγής:

$\forall$x($\neg$pie(x)$\vee$$\neg$(x=table)

Απαλοιφή καθολικών ποσοδεικτών και μετονομασία μεταβλητών και σταθερών:

$\neg$pie(x)$\vee$$\neg$(x=TABLE)

Πρόταση 4:

Από την εκφώνηση της άσκησης παίρνω τα εξής για την 1η πρόταση:

$\neg$on(x1,y1)$\vee$$\neg$(x1=y1)

$\neg$on(x2,y2)$\vee$pie(x2)

$\neg$on(x3,y3)$\vee$pie(y3) $\vee$ y3=TABLE

Πρόταση 5:
∀x[pie(x)→∃y(on(x,y))]
Απαλοιφή συνεπαγωγής:
∀x[¬pie(x) ∨ ∃y(on(x,y))]
Σκολεμοποίηση λόγω του ότι ο υπαρξιακός ποσοδείκτης είναι σε εμβέλεια ενός καθολικού ποσοδείκτη. Θα χρησιμοποιήσω την μεταβλητή f όπως και αναφέρει η εκφώνηση:
∀x[¬pie(x) ∨ on(x,f(x)]
Απαλοιφή καθολικού ποσοδείκτη και μετονομασία μεταβλητών και σταθερών:
¬pie(x) ∨ on(x,f(x))

Πρόταση 6:
∀x∀y∀z(on(x,y)∧on(x,z)→y=z)
Απαλοιφή συνεπαγωγής:
∀x∀y∀z(¬(on(x,y)∧on(x,z)) ∨ y=z)
De Morgan:
∀x∀y∀z(¬on(x,y) ∨ ¬on(x,z) ∨ y=z)
Απαλοιφή καθολικού ποσοδείκτη και μετονομασία μεταβλητών και σταθερών:
¬on(x,y) ∨ ¬on(x,z) ∨ y=z

Πρόταση 7:
∀x∀y∀z[on(x,z)∧on(y,z)→(x=y ∨ z=table)]
Απαλοιφή συνεπαγωγής:
∀x∀y∀z[¬(on(x,z)∧on(y,z))∨(x=y ∨ z=table)]
De Morgan:
∀x∀y∀z[¬on(x,z)∨¬on(y,z) ∨ x=y ∨ z=table]
Απαλοιφή καθολικού ποσοδείκτη και μετονομασία μεταβλητών και σταθερών:
¬on(x,z) ∨ ¬on(y,z) ∨ x=y ∨ z=TABLE

Πρόταση 8:
∀x(clear(x)↔¬∃y(on(y,x)))
Για την απαλοιφή της ισοδυναμίας θα χρησιμοποιήσω από τον νόμο αντικατάστασης: $(\varphi \leftrightarrow \psi) \equiv (\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)$. Άρα:
∀x[(clear(x)→¬∃y(on(y,x)))∧ (¬∃y(on(y,x)→clear(x))]
Απαλοιφή συνεπαγωγής:
∀x[(¬clear(x)∨¬∃y(on(y,x))) ∧ (¬(¬∃y(on(y,x)))∨clear(x))]
De Morgan:
∀x[(¬clear(x)∨∀y(¬on(y,x))) ∧ ∃y(on(y,x))∨clear(x)]
Σκολεμοποίηση λόγω του ότι ο υπαρξιακός ποσοδείκτης είναι σε εμβέλεια ενός καθολικού ποσοδείκτη. Θα χρησιμοποιήσω την μεταβλητή g όπως και αναφέρει η εκφώνηση:
∀x[(¬clear(x)∨∀y(¬on(y,x))) ∧ (on(g(x),x)∨clear(x))]
Μετακίνηση ποσοδεικτών:
∀x∀y[(¬clear(x)∨¬on(y,x)) ∧ (on(g(x),x)∨clear(x))]
Απαλοιφή καθολικών ποσοδεικτών και μετονομασία μεταβλητών και σταθερών:
(¬clear(x)∨¬on(y,x)) ∧ (on(g(x),x)∨clear(x))
Άρα:
¬clear(x1)∨¬on(y1,x1)
on(g(x2),x2)∨clear(x2)

Συνοπτικά:
1.α ¬(Π1=Π2)
1.β ¬(Π1=Π3)
1.γ ¬(Π1=Π4)
1.δ ¬(Π2=Π3)
1.ε ¬(Π2=Π4)
1.ζ ¬(Π3=Π4)
2.α ¬pie(x1)∨ x1=Π1∨ x1=Π2 ∨ x1=Π3 ∨ x1=Π4
2.β pie(x2)∨¬(x2 = Π1)
2.γ pie(x3)∨¬(x3 = Π2)
2.δ pie(x4)∨¬(x4 = Π3)
2.ε pie(x5)∨¬(x5 = Π4)
3. ¬pie(x)∨¬(x=TABLE)
4.α ¬on(x1,y1)∨¬(x1=y1)
4.β ¬on(x2,y2)∨pie(x2)
4.γ ¬on(x3,y3)∨pie(y3) ∨ y3=TABLE
5. ¬pie(x) ∨ on(x,f(x))
6. ¬on(x,y) ∨ ¬on(x,z) ∨ y=z
7. ¬on(x,z) ∨ ¬on(y,z) ∨ x=y ∨ z=TABLE
8.α ¬clear(x1)∨¬on(y1,x1)
8.β on(g(x2),x2)∨clear(x2)

Υπό ερώτημα 1) Β)

Για την απόδειξη της πρότασης on(π1,π4)→¬on(π4,table) η άρνηση της πρότασης είναι:
¬(on(π1,π4)→¬on(π4,table))
Τη μετατρέπω σε ΣΚΜ όπου:
¬(on(π1,π4)→¬on(π4,table))
<u>Απαλοιφή συνεπαγωγής</u>:
¬(¬on(π1,π4)∨¬on(π4,table))
<u>DeMorgan:</u>
¬¬on(π1,π4)∧¬¬on(π4,table))
Απαλοιφή άρνησης:
on(π1,π4)∧on(π4,table))
άρα:
on(Π1, Π 4)
on(Π4, TABLE))

Τέλος την προσθέτω στη βάση γνώσης την πρόταση:

Βάση Γνώσης:	
Τύπος	Πρόταση:
1.α	¬(Π1=Π2)
1.β	¬(Π1=Π3)
1.γ	¬(Π1=Π4)
1.δ	¬(Π2=Π3)
1.ε	¬(Π2=Π4)
1.ζ	¬(Π3=Π4)
2.α	¬pie(x1)∨ x1=Π1∨ x1=Π2 ∨ x1=Π3 ∨ x1=Π4
2.β	pie(x2)∨¬(x2 = Π1)
2.γ	pie(x3)∨¬(x3 = Π2)
2.δ	pie(x4)∨¬(x4 = Π3)
2.ε	pie(x5)∨¬(x5 = Π4)
3.	¬pie(x)∨¬(x=TABLE)
4.α	¬on(x1,y1)∨¬(x1=y1)
4.β	¬on(x2,y2)∨pie(x2)
4.γ	¬on(x3,y3)∨pie(y3) ∨ y3=TABLE
5.	¬pie(x) ∨ on(x,f(x))
6.	¬on(x,y) ∨ ¬on(x,z) ∨ y=z
7.	¬on(x,z) ∨ ¬on(y,z) ∨ x=y ∨ z=TABLE
8.α	¬clear(x1)∨¬on(y1,x1)
8.β	on(g(x2),x2)∨clear(x2)
9.	x = x
10.	¬x=y ∨ y=x
11.	¬(x=y) ∨ ¬(y=z) ∨ x=z
12.	¬(x=y) ∨ f(x) = f(y)
13.	¬(x=y) ∨ g(x) = g(y)
14.α	¬(x=y) ∨ ¬pie(x) ∨ pie(y)
14.β	¬(x=y) ∨ pie(x) ∨ ¬pie(y)
15.α	¬(x=y) ∨ ¬clear(x) ∨ clear(y)
15.β	¬(x=y) ∨ clear(x) ∨ ¬clear(y)
16.α	¬(x1=y1) ∨ ¬(x2=y2) ∨ ¬on(x1,x2) ∨ on(y1,y2)
16.β	¬(x1=y1) ∨ ¬(x2=y2) ∨ on(x1,x2) ∨ on(y1,y2)
17.α	on(Π1, Π 4)
17.β	on(Π4, TABLE))

on(Π1,Π4) ¬on(x,y)∨¬(x1=y1) on(Π1,TABLE) ¬on(x1,y1)∨(x1=y1)

ενοποίηση:{Π1|χ1,Π4|y1} ενοποίηση{Π1|x1,TABLE|y1}

¬(Π1=Π4) ¬on(x2,y2)∨pie(x2) (Π1=TABLE)

ενοποίηση:{Π1|χ2,Π4|y2}

pie(Π1) ¬pie(x)∨¬(x1=y1)

¬(Π1=TABLE)

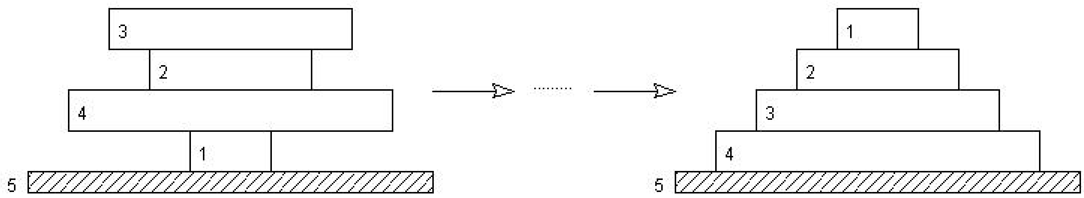
{}

Θέμα 5: Προγραμματισμός σε Prolog [*10 μονάδες*]

Στην 1η γραπτή εργασία της ΠΛΗ31 το 2020-2021, ζητήθηκε από τους φοιτητές να επιλύσουν το πρόβλημα με τις πίτες (δείτε παρακάτω), τόσο μέσω αναζήτησης κατά βάθος (dfs), όσο και μέσω υλοποίησης της dfs σε Prolog.

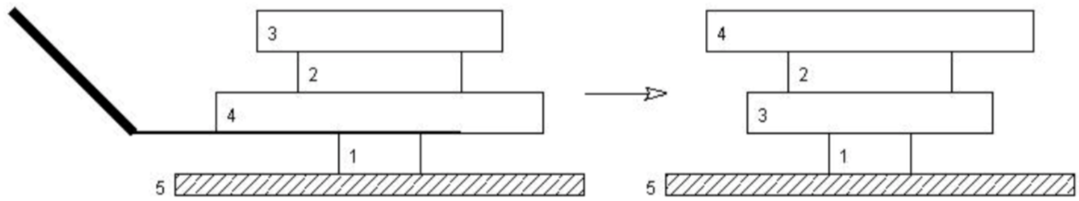
Εδώ καλείστε να επιλύσετε το ίδιο πρόβλημα (με τις πίτες) με Prolog, χρησιμοποιώντας μια διαφορετική προσέγγιση, γνωστής ως *generate-and-test*.

Το πρόβλημα με τις πίτες μπορεί να περιγραφεί ως εξής. Δίνονται N πίτες με διαμέτρους 1, 2, . . . , N, οι οποίες είναι τοποθετημένες σε μια στοίβα με τυχαία σειρά. Θέλουμε να ταξινομήσουμε τις πίτες στην στοίβα με αύξουσα σειρά (με την πίτα διαμέτρου 1 στην κορυφή της στοίβας) – βλ. Σχήμα 1 για παράδειγμα με N=4.



Σχήμα 1

Ο μοναδικός τρόπος για να αλλάξει η διάταξη της στοίβας είναι με την χρήση μια σπάτουλας. Τοποθετώντας την σπάτουλα κάτω από μια πίτα μπορεί κανείς να αναποδογυρίσει όλες της πίτες που βρίσκονται πάνω στην σπάτουλα και να τις τοποθετήσει στην στοίβα με ανεστραμμένη σειρά (βλ. Σχήμα 2).



Σχήμα 2

Συγκεκριμένα, αν η αρχική διάταξη στην στοίβα είναι ($\Sigma_1, \ldots, \Sigma_k, \Sigma_{k+1}, \ldots \Sigma_N$) και τοποθετήσουμε την σπάτουλα κάτω από την πίτα Σ_k, αναποδογυρίζοντας τις πίτες πάνω στην σπάτουλα παίρνουμε την στοίβα ($\Sigma_k, \ldots, \Sigma_1, \Sigma_{k+1}, \ldots \Sigma_N$).[3]

Επομένως για μια τυχαία αρχική διάταξη της στοίβας από πίτες, το ζητούμενο είναι να βρεθεί μια ακολουθία κινήσεων της σπάτουλας που ταξινομεί την στοίβα. Για παράδειγμα, μπορούμε να ταξινομήσουμε την στοίβα στα αριστερά του Σχήματος 1 με τις ακόλουθες κινήσεις:

1. Τοποθετούμε την σπάτουλα κάτω από την πίτα **2** και αναποδογυρίζουμε.
2. Τοποθετούμε την σπάτουλα κάτω από την πίτα **4** και αναποδογυρίζουμε.
3. Τοποθετούμε την σπάτουλα κάτω από την πίτα **1** και αναποδογυρίζουμε.

[3] Στην αναπαράσταση ($\Sigma_1, \ldots, \Sigma_N$) της στοίβας, θεωρούμε πως το Σ_1 είναι στην κορυφή της στοίβας.

Η παραπάνω ακολουθία κινήσεων μπορεί να αναπαρασταθεί απλά με την ακολουθία αριθμών (2, 4, 1), με κάθε αριθμό να αντιστοιχεί στην πίτα κάτω από την οποία τοποθετείται κάθε φορά η σπάτουλα.

Στην επίλυση σας σε Prolog στο πρόβλημα με τις πίτες, θα πρέπει αναπαραστήσετε τόσο την στοίβα, όσο και μια ακολουθία κινήσεων της σπάτουλας, ως **λίστες**.

Για παράδειγμα η αριστερή στοίβα στο Σχήμα 1 θα πρέπει να αναπαρασταθεί με την λίστα [3, 2, 4, 1], και η ακολουθία κινήσεων που περιγράψαμε νωρίτερα για την ταξινόμηση αυτής της στοίβας, θα πρέπει να αναπαρασταθεί με την λίστα [2, 4, 1].

Επισημαίνεται πως η ακολουθία κινήσεων που ταξινομεί μια στοίβα είναι πιθανό να περιέχει τον ίδιο αριθμό περισσότερες από μια φορά.[4]

Ο κώδικάς σας θα πρέπει να ξεκινάει με τις παρακάτω γραμμές:

```
input([3,2,4,1]).
solution(S,Max) :-  input(L), sort(L,L1),
                    belongs(S,L,Max),  turnSequence(S,L,L1).
```

Το κατηγόρημα `input/1` ορίζει την στοίβα που θέλουμε να ταξινομήσουμε. Αρχικά θα χρησιμοποιήσουμε την στοίβα [3, 2, 4, 1] στα αριστερά του Σχήματος 1. Αργότερα, όταν ολοκληρώσετε τον κώδικα, μπορείτε να δοκιμάσετε διαφορετικές στοίβες στο `input`.

Με το κατηγόρημα `solution/2` ελέγχουμε αν η λίστα στο `input/1` μπορεί να ταξινομηθεί με μια ακολουθία κινήσεων S με μήκος το πολύ μέχρι Max.

Συγκεκριμένα στο σώμα της `solution/2` αρχικά ορίζουμε την L ως την λίστα που δίνεται στο `input`, ταξινομούμε την L και αποθηκεύουμε το αποτέλεσμα στην λίστα L1 (το κατηγόρημα `sort/2` δίνεται από την SWI Prolog).

Στην συνέχεια με το κατηγόρημα `belongs/3` παράγουμε (generate) μια ακολουθία κινήσεων S που έχει μήκος το πολύ Max, και που αποτελείται από αριθμούς που βρίσκονται στην L.

Τέλος μέσω του `turnSequence/3` ελέγχουμε (test) αν ξεκινώντας από την L και με την διαδοχική εκτέλεση της ακολουθίας κινήσεων S, καταλήγουμε στην ταξινομημένη λίστα L1.

Τα κατηγορήματα `belongs/3` και `turnSequence/3` θα πρέπει να τα ορίσετε εσείς με τον τρόπο που περιγράφεται στην συνέχεια.

Α. [2 *μονάδες*] Συμπληρώστε τον παρακάτω κώδικα για να ορίσετε το κατηγόρημα `belongs/3` έτσι ώστε το `belongs(S,L,Max)` να αληθεύει όταν η λίστα S έχει το πολύ Max στοιχεία, και όλα τα στοιχεία της S ανήκουν στην λίστα L. [Υπόδειξη: Χρησιμοποιείστε αναδρομή και το κατηγόρημα `member/2` της SWI Prolog].

```
belongs([],_,Max) :- 0=<Max.
belongs([A|B],L,Max) :- .....................
```

Απάντηση:

[4] Μάλιστα υπάρχουν στοίβες για τις οποίες υπάρχουν μόνο τέτοιες λύσεις. Για παράδειγμα η στοίβα [4,1,3,5,2] δεν μπορεί να ταξινομηθεί αν δεν επιτρέπεται η τοποθέτηση της σπάτουλας κάτω από την ίδια πίτα πάνω από μια φορά. Οι μόνες ακολουθίες κινήσεων που ταξινομούν την στοίβα, όπως η [3,1,5,2,1], τοποθετούν την σπάτουλα *δύο (ή περισσότερες) φορές* κάτω από την ίδια πίτα.

% υπό ερώτημα Α

belongs([],_,Max).
belongs([A|B],L,Max):-member([A|B],L),length([A|B],S),S<Max,belongs([B,L,Max]).

%Η λογική σκέψη είναι να ελέγξω σε κάθε αναδρομή το στοιχείο της κεφαλής της λίστας S από το κατηγόρημα belongs με την λίστα L αν υπάρχει το στοιχείο
%έπειτα χρησιμοποιώ το κατηγόρημα length όπου υπάρχει στην βιβλιοθήκη της prologue ώστε να βρει η prolog το μέγεθος της λίστας S
%και να το συγκρίνω χρησιμοποιώντας μαθηματικό τελεστή ώστε να είναι το πολύ Max στοιχεία
%τέλος συνεχίζω την αναδρομή με την ουρά της λίστας μιας και το πρώτο στοιχείο αφαιρέθηκε από τον έλεγχο

Β. [3 *μονάδες*] Συμπληρώστε τον παρακάτω κώδικα για να ορίσετε ένα κατηγόρημα `turnOnce/3`, τέτοιο ώστε το `turnOnce(N,In,Out)` να αληθεύει όταν η λίστα Out προκύπτει από την λίστα In με *μια* κίνηση της σπάτουλας όταν αυτή τοποθετείται κάτω από την πίτα N. Επιπλέον, για να αποφεύγονται περιττές κινήσεις, θα πρέπει το N να είναι διαφορετικό από το πρώτο στοιχεία της In. Για παράδειγμα το `turnOnce(4,[2,4,1,3,5],[4,2,1,3,5])` είναι αληθές.

```
turnOnce(N, In, Out) :- append(A, [N|B], In), length(A,K), 0<K, ...
```

[Υπόδειξη:
Η βασική ιδέα στην υλοποίηση της `turnOnce(N, In, Out)` είναι η εξής:

1. Διαχωρίζεται την λίστα `In` σε δύο υπο-λίστες, εκ των οποίων η πρώτη υπο-λίστα, με το όνομα `A`, περιέχει όλα τα στοιχεία της `In` πριν από το N, και η δεύτερη υπο-λίστα, `[N|B]`, περιέχει όλα τα υπόλοιπα στοιχεία της `In`. Αυτός ο διαχωρισμός μπορεί να γίνει με το κατηγόρημα `append/3` της SWI Prolog όπως φαίνεται στον παραπάνω κανόνα.
2. Αναποδογυρίζετε τις πίτες της Α με το κατηγόρημα της SWI Prolog `reverse/2`.
3. Το αποτέλεσμα το ενώνετε με την λίστα `B` μέσω του κατηγορήματος `append/3`, τοποθετώντας παράλληλα την πίτα N στην κορυφή της στοίβας. Την νέα στοίβα την ονομάζεται `Out`.]

Επισημαίνεται ότι τα `length(A,K), 0<K` έχουν προστεθεί στο σώμα του παραπάνω κανόνα για να αποφευχθούν οι περιττές κινήσεις.[5] Συγκεκριμένα, το `length(A,K)` αποθηκεύει στο K το πλήθος των στοιχείων της λίστας A. Επομένως το `0<K` εξασφαλίζει πως η λίστα A δεν είναι κενή, ή αλλιώς πως η σπάτουλα δεν έχει τοποθετηθεί κάτω από την πρώτη πίτα στην κορυφή της λίστας (κάτι τέτοιο δεν θα επέφερε καμία μεταβολή στην στοίβα – είναι περιττή κίνηση).

Απάντηση:

% υπό ερώτημα Β

turnOnce(N,In,Out):-
append(A,[N|B],In),length(A,K),0<K,reverse(A,RevA),append(RevA,B,Out).

Γ. [2 *μονάδες*] Συμπληρώστε τον παρακάτω κώδικα για να ορίσετε ένα κατηγόρημα `turnSequence/3`, τέτοιο ώστε το `turnSequence(S,In,Out)` να αληθεύει όταν ξεκινώντας

[5] Το `length/2` είναι κατηγόρημα της SWI Prolog.

από την στοίβα In και εκτελώντας διαδοχικά τις κινήσεις στην λίστα S, καταλήγουμε στην στοίβα Out. Για παράδειγμα το `turnSequence([3,1],[2,4,1,3,5],[1,3,4,2,5])` είναι αληθές. [Υπόδειξη: Χρησιμοποιήστε αναδρομή και το κατηγόρημα `turnOnce/3` που ορίσατε προηγουμένως.]

```
turnSequence([],L,L).
turnSequence([N|L], In, Out) :- ...............
```

Απάντηση:

%υπό ερώτημα Γ

turnSequence([],L,L).
turnSequence([N|L],In,Out):-turnOnce(N,In,Out),append([],N|L,Out),turnSequence(L,In,Out).

%χρησιμοποίησα το κατηγόρημα turnOnce βάζοντας την κεφαλή της λίστας. Έπειτα χρησιμοποίησα append([],N|L,Out) ώστε να πάρω το tail της λίστας στη σταθερά out
%και να το ξαναγυρίσω στην αναδρομή ως κατά το ζητούμενο της άσκησης ως όταν τελειώσει η αναδρομή με το κάλεσμα του κατηγορήματος ελέγχου και τερματισμού turnSequence([],L,L).

Δ1. [1 *μονάδα*] Εκτελέστε το query `solution(S,3)` για να διαπιστώσετε αν υπάρχει λύση με το πολύ 3 κινήσεις για την στοίβα στα αριστερά του Σχήματος 1; Γράψτε την απάντηση που δίνει το πρόγραμμά σας.

Απάντηση:

Η λογική σκέψη που προσπάθησα να αναπτύξω γύρω από τον αναδρομικό τρόπο σκέψης δεν λειτούργησε και δυστυχώς δεν κατάφερα να κάνω compile το πρόγραμμα χωρίς να έχει σφάλματα. Δεν έχω απαντήσει σε αυτό το ερώτημα.

Δ2. [1 *μονάδα*] Ορίστε ως input την στοίβα [5, 3, 6, 2, 4, 1, 7]. Χρησιμοποιήστε το πρόγραμμά σας για να βρείτε τον ελάχιστο αριθμό κινήσεων που θα την ταξινομήσει. Ποια λύση δίνει το πρόγραμμά σας γι' αυτό τον αριθμό κινήσεων;

Απάντηση:

Η λογική σκέψη που προσπάθησα να αναπτύξω γύρω από τον αναδρομικό τρόπο σκέψης δεν λειτούργησε και δυστυχώς δεν κατάφερα να κάνω compile το πρόγραμμα χωρίς να έχει σφάλματα. Δεν έχω απαντήσει σε αυτό το ερώτημα.

Δ3. [1 *μονάδα*] Ορίστε ως input την στοίβα [5, 3, 6, 2, 9, 4, 1, 8, 7] και ελέγξτε αν αυτή η στοίβα μπορεί να ταξινομηθεί με το πολύ N κινήσεις. Ποιος είναι ο μέγιστος αριθμός κινήσεων N για τον οποίο το πρόγραμμά σας δίνει (θετική ή αρνητική) απάντηση σε λιγότερο από 2 λεπτά πραγματικού χρόνου; Βρήκατε κάποια ακολουθία κινήσεων που ταξινομεί την λίστα μέσα σ' αυτόν τον χρονικό περιορισμό;

Απάντηση:

Η λογική σκέψη που προσπάθησα να αναπτύξω γύρω από τον αναδρομικό τρόπο σκέψης δεν λειτούργησε και δυστυχώς δεν κατάφερα να κάνω compile το πρόγραμμα χωρίς να έχει σφάλματα. Δεν έχω απαντήσει σε αυτό το ερώτημα.