



ΕΛΛΗΝΙΚΟ ΑΝΟΙΚΤΟ ΠΑΝΕΠΙΣΤΗΜΙΟ

ΘΕΜΑΤΙΚΗ ΕΝΟΤΗΤΑ

Εισαγωγή στην Πληροφορική

ПЛН-10: XXXXXX

5^η Ομαδική Συμβουλευτική Συνάντηση (Ο.Σ.Σ.)

11 / 05 / 2019

Σ.Ε.Π.: XXXXXXXXXXXXXXXXXXXXXXXXXXXX



Θέματα 5ης Ο.Σ.Σ.

- Γενική συζήτηση (4^η γραπτή εργασία, 3^{ος} Τόμος κλπ.).
- Ενημέρωση για διαδικαστικά θέματα και συζήτηση επί των δηλώσεων μαθημάτων νέου έτους.
 - Εσωτερική αξιολόγηση
- Συζήτηση για τις τελικές εξετάσεις (ύλη, θέματα, τρόπος κλπ.).
- Παρουσίαση επιλεγμένων θεμάτων τελικών και επαναληπτικών εξετάσεων και των ενδεικτικών λύσεών τους.
- Συζήτηση - Απορίες επί όλων των Τόμων.

Επίσης δίνεται για διευκόλυνσή σας

Παράρτημα: Επανάληψη ύλης όλων των τόμων.

Τελικές και Επαναληπτικές Εξετάσεις

Τελικές Εξετάσεις	Επαναληπτικές εξετάσεις
Σάββατο 08/06/2019 13:30 – 17:00	Παρασκευή 28/06/2019 18:00 - 21:30

ΠΛΗ 10
Εισαγωγή στην
Πληροφορική

ΠΛΗ 11
Αρχές Τεχνολογίας
Λογισμικού

ΠΛΗ 12
Μαθηματικά για
Πληροφορική

1ο ΕΤΟΣ

ΠΛΗ 21
Ψηφιακά Συστήματα

ΠΛΗ 22
Βασικά Ζητήματα
Δικτύων Η/Υ

ΠΛΗ 20
Διακριτά Μαθηματικά
& Μαθηματική Λογική

2ο ΕΤΟΣ

ΠΛΗ 24
Σχεδιασμός
Λογισμικού

ΠΛΗ 31
Τεχνητή Νοημοσύνη

ΠΛΗ 30
Θεμελιώσεις
Επιστήμης Η/Υ

3ο ΕΤΟΣ

ΠΛΗ 23
Τηλεματική, Διαδίκτυο
& Κοινωνία

ΠΛΗ 35
Προστασία &
Ασφάλεια Συστ.Η/Υ

ΠΛΗ 36
Σύγχρονα Δίκτυα
& Υπηρεσίες

4ο ΕΤΟΣ
(επιλογή 3/7)

ΠΛΗ 37
Πληροφορική
& Εκπαίδευση

ΠΛΗ 42
Ειδικά Θέματα
Τεχνολ. Λογισμικού

ΠΛΗ 44
Σήματα & Επε-
ξεργασία Εικόνας

ΠΛΗ 40
Πρακτική Εξάσκηση
σε Θέματα Λογισμικού
ΠΤΥΧΙΑΚΗ
(μόνο ως 12η)

Κανονισμός εξετάσεων (1/3)

- Δικαίωμα συμμετοχής στην τελική εξέταση έχουν εκείνοι οι φοιτητές που έχουν παραδώσει κατά τη διάρκεια της ακαδημαϊκής χρονιάς 3 ή 4 γραπτές εργασίες και έχουν συγκεντρώσει τουλάχιστον 20 βαθμούς στις εργασίες που παρέδωσαν. Σε περίπτωση αμφιβολίας παρακαλείστε να επικοινωνήσετε με το Σύμβουλο - Καθηγητή σας.
- Οι τελικές γραπτές εξετάσεις της ΠΛΗ10 πραγματοποιούνται με **κλειστά βιβλία**.
- Κάθε φοιτητής για να μπορέσει να συμμετάσχει στην εξέταση πρέπει να φέρει μαζί του τη φοιτητική του ταυτότητα **και** την αστυνομική του ταυτότητα ή στρατιωτική του ταυτότητα ή το διαβατήριό του (**προσοχή όχι μόνο τη φοιτητική ταυτότητα**).
- Οι φοιτητές είναι υποχρεωτικό να βρίσκονται στις αίθουσες των εξετάσεων το **αργότερο μισή ώρα πριν** την έναρξη της εξέτασης.
- Σε όλες τις κόλλες (θέματα, κόλλες και τυχόν επιπλέον κόλλες που θα ζητήσετε) είστε υποχρεωμένοι να αναγράψετε το **ονοματεπώνυμό σας και να υπογράψετε**.

Κανονισμός εξετάσεων (2/3)

- Δεν επιτρέπεται η έξοδος από την αίθουσα κατά τη διάρκεια της εξέτασης, παρά μόνο εάν πρόκειται για **απόλυτη** ανάγκη και μόνο με τη συνοδεία επιτηρητή.
- Κατά τη διάρκεια της εξέτασης, **δεν δίνονται διευκρινίσεις** πάνω στα θέματα των εξετάσεων.

(Σημαντικό !!!) Σε περίπτωση που έχετε αμφιβολία σε κάποιο θέμα καταγράψτε την και στη συνέχεια, αφού καταγράψετε και την παραδοχή που κάνατε, επιλύστε το θέμα.

- Απαγορεύεται το **κάπνισμα**. Απαγορεύεται η χρήση **κινητών τηλεφώνων**. Απαγορεύεται η χρήση οποιουδήποτε είδους **ηλεκτρονικών υπολογιστικών συσκευών**.
- Απαγορεύονται οι **συνομιλίες** με οποιονδήποτε τρόπο με συν-εξεταζόμενους. Για οποιοδήποτε τυχόν πρόβλημα οι φοιτητές θα πρέπει να απευθύνονται αποκλειστικά και μόνο στους επιτηρητές.
- Δεν επιτρέπεται να αποχωρήσει φοιτητής από την αίθουσα, εάν δεν έχουν παρέλθει τουλάχιστον **εξήντα λεπτά (60')** από την ώρα έναρξης της εξέτασης.

Κανονισμός εξετάσεων (3/3)

- Παρακαλείστε να αποφεύγετε την καθ' οποιονδήποτε τρόπο **επικοινωνία** (προφορική, οπτική, ανταλλαγή υλικού, κλπ.) μεταξύ σας κατά τη διάρκεια της εξέτασης. Κάτι τέτοιο θα μπορούσε να θεωρηθεί ότι συνιστά αντιγραφή και να έχει ως συνέπεια το μηδενισμό όλων των εμπλεκομένων, κατά την κρίση των επιτηρητών.
- Για την αναγραφή των απαντήσεων στις κόλλες των εξετάσεων θα πρέπει να χρησιμοποιηθεί **στυλό διαρκείας κατά προτίμηση χρώματος μπλε**. Δεν επιτρέπεται η αναγραφή των απαντήσεων με μολύβι.
- Με την παράδοση των γραπτών, οι φοιτητές πρέπει να **παραδώσουν και την εκφώνηση των θεμάτων**, πάνω στην οποία θα έχουν σημειώσει απαραίτητα το όνομά τους και την υπογραφή τους καθώς και κάθε άλλο έντυπο που έχουν πάρει (κόλλες για πρόχειρο κλπ.).

ΣΗΜΑΝΤΙΚΟ!

Να γράψετε το όνομα του Συμβούλου - Καθηγητή σας στο γραπτό σας

Χρήσιμες και σημαντικές συμβουλές

- Σύμφωνα με τον κανονισμό **«Οι φοιτητές είναι υποχρεωτικό να βρίσκονται στις αίθουσες των εξετάσεων το αργότερο μισή ώρα πριν την έναρξη της εξέτασης»**. Είναι σημαντικό να φθάσετε έγκαιρα στο εξεταστικό κέντρο και να αποφύγετε απρόβλεπτες καταστάσεις.
- Οι εξετάσεις έχουν διάρκεια 3 ώρες. Αυτό θα τηρηθεί αυστηρά σε όλες τις πόλεις και σε όλα τα εξεταστικά κέντρα. Οι εξετάσεις συντονίζονται κεντρικά και ξεκινούν ταυτόχρονα.
- Η οργάνωση της αίθουσας γίνεται για να διευκολύνει τη διαδικασία των εξετάσεων. Συνεργαστείτε με τους επιτηρητές ώστε να τελειώσει σύντομα και πειθαρχημένα.
- Εάν κριθεί απαραίτητο, ο επιτηρητής μπορεί να σας αλλάξει θέση. Σίγουρα δεν είναι κάτι προσωπικό.

Χρήσιμες και σημαντικές συμβουλές

- Όταν αρχίσουν οι εξετάσεις, σεβαστείτε τους συναδέλφους σας και αποφύγετε τη δημιουργία φασαρίας (ειδικά σε μεγάλα αμφιθέατρα).
- Αν υπάρχει κάποιο πρόβλημα, απευθυνθείτε χαμηλόφωνα στον επιτηρητή.
- Μόλις τελειώσετε, σκεφθείτε ότι κάποιοι άλλοι μπορεί να γράφουν ακόμα. **ΣΗΜΑΝΤΙΚΟ!**
- Όταν τελειώσει ο χρόνος της εξέτασης, παραμείνετε στη θέση σας και παραδώστε το γραπτό σας στον επιτηρητή.
- Οι επιτηρητές είναι καθηγητές είτε στο δικό σας τμήμα, είτε σε άλλα τμήματα της ΠΛΗ-10. Η παρουσία των επιτηρητών έχει σκοπό να διασφαλίσει το αδιάβλητο των εξετάσεων, αλλά και τη δίκαια διαδικασία των εξετάσεων. Μην τους φέρετε σε δύσκολη θέση.

Άλλα ζητήματα εξετάσεων

- Πως διαβάζω; Πότε;
- Τι μπορείτε να έχετε μαζί σας;
- Μην πάρετε μαζί σας βιβλία - σημειώσεις - calculator.
- Τι χρόνο θα έχετε;
- Τι ισχύει με τις 2 εξεταστικές (Τελικές - Επαναληπτικές);
- Τι σημαίνει <κλειστά βιβλία> και τι χρειάζεται να μάθω «απ' έξω»;

Άλλα ζητήματα

- Διαβάστε πολύ **προσεκτικά** τα θέματα.
- Απαντήστε **πρώτα τα εύκολα** (κατά τη γνώμη σας) θέματα.
- **Δεν** χρειάζεται να απαντήσετε τα θέματα με τη **σειρά** που δίνονται.
- **Μη σπαταλάτε πολύ χρόνο** σε θέμα που σας έχει μπερδέψει (αφήστε το για το τέλος και ασχοληθείτε με κάποιο άλλο).
- Τα θέματα και ότι άλλο έντυπο δοθεί **θα επιστραφούν**.

Ύλη εξετάσεων

- **Τόμος 1: «Εισαγωγή στην Επιστήμη των Υπολογιστών»** Όλη η ύλη του τόμου είναι ύλη που μπορεί να εξεταστεί, αλλά θα πρέπει να δοθεί **μεγαλύτερη προσοχή** στις ενότητες:
2.1, 2.2, 2.3, 2.4, 2.5, 2.6
5.1, 5.2, 5.3, 5.4
10.1 και 10.3
- **Τόμος 2: «Τεχνικές Προγραμματισμού»** Από την ύλη του 2^{ου} Τόμου **αφαιρούνται** οι παρακάτω ενότητες:
5.2, 5.3, 5.4, 5.5, 5.6, 5.7, 5.8 και 8.2. και
τα Κεφάλαια 9, 10 και 11
- **Τόμος 3: «Δομές Δεδομένων»**, διδάχθηκε 4^{ος} χρονικά μετά τη γλώσσα C. Από την ύλη του 3ου Τόμου **αφαιρούνται** οι ενότητες:
3.4, 4.4 και 5.5
- **Τόμος 4: «Γλώσσες Προγραμματισμού»**, διδάχθηκε 3^{ος} χρονικά. Από την ύλη του 4ου Τόμου **αφαιρούνται** τα παρακάτω κεφάλαια και ενότητες:
1.3.2, 1.3.3, 5.2.4 και
το Κεφάλαιο 10

Σημαντική ύλη

- Συστήματα Αρίθμησης
- Σχεδίαση Λογικών Κυκλωμάτων
- Διάγραμμα Ροής
- Αμυντικός Προγραμματισμός
- Αναζήτηση (σειριακή - δυαδική)
- Ταξινόμηση (bubble & selection sort)
- Λίστα (απλή)
- Στοίβα - Ουρά
- Δυαδικά Δέντρα (Διαπεράσεις)
- Δυαδικά Δέντρα Αναζήτησης (Πράξεις)
- Συνεχόμενη αναπαράσταση (Δένδρου)
- Δέντρο σωρού (Πράξεις)

**Προγραμματισμός σε C
& Ψευδοκώδικας Τόμου Β'**

Χρονοδιάγραμμα

- 12 Μαΐου → **5η ΟΣΣ** (επανάληψη για τις εξετάσεις)
↓ **Επανάληψη**
- 8 Ιουνίου → **Εξετάσεις**
↓
- 28 Ιουνίου → **Επαναληπτικές εξετάσεις**

Έμφαση επί της ύλης του 1^{ου} Τόμου Μελέτης

Πού πρέπει να δοθεί έμφαση:

- Αριθμητικά συστήματα
 - Μετατροπές σε αριθμητικά συστήματα και πράξεις
 - Αφαίρεση με χρήση συμπληρώματος ως προς 2
- Κυκλώματα
 - Πύλες
 - Υπολογισμός συνάρτησης εξόδου σε κύκλωμα
 - Πίνακας Αληθείας
- Διαγράμματα Ροής Προγράμματος (ΔΡΠ)

Έμφαση επί της ύλης του 2^{ου} Τόμου Μελέτης

Πού πρέπει να δοθεί έμφαση:

- ΔΡΠ - Ψευδοκώδικας
 - Μετατροπή ΔΡΠ σε Ψευδοκώδικα και αντιστρόφως
 - Εφαρμογές σε Ψευδοκώδικα
 - Χρήση αμυντικού προγραμματισμού
- Αλγόριθμοι
 - Αναζήτηση (σειριακή - δυαδική αναζήτηση)
 - Ταξινόμηση (bubble & selection sort)

Έμφαση επί της ύλης του 3^{ου} Τόμου Μελέτης (4^{ου} τόμου)

Πού πρέπει να δοθεί έμφαση:

- Ανάπτυξη εφαρμογών σε C (σε απλά προγράμματα)
- Χρήση αμυντικού προγραμματισμού
- Προσοχή στη σύνταξη
- Εύρεση συντακτικών λαθών σε κώδικα C
- Χρήση αναδρομής
- Υλοποίηση εφαρμογών με χρήση Πινάκων, Λιστών
- Χρήση και κλήση συναρτήσεων

Έμφαση επί της ύλης του 4^{ου} Τόμου Μελέτης (3^{ου} Τόμου)

Πού πρέπει να δοθεί έμφαση:

- Δέντρα
 - Δυαδικό δέντρο αναζήτησης (διαπέραση, συνεχόμενη αναπαράσταση δέντρου)
 - Δέντρο - σωρός
 - Εισαγωγή/Διαγραφή κόμβου σε δέντρα - σωρούς και δυαδικά δέντρα αναζήτησης
- Λίστες
 - Διαπέραση
 - Εισαγωγή κόμβου
 - Διαγραφή κόμβου
- Στοιίβες
 - Εισαγωγή/Διαγραφή στοιχείου
- Ουρές
 - Εισαγωγή/Διαγραφή στοιχείου

Παρουσίαση Θεμάτων Τελικών Εξετάσεων 2013



ΕΛΛΗΝΙΚΟ ΑΝΟΙΚΤΟ ΠΑΝΕΠΙΣΤΗΜΙΟ

ΘΕΜΑΤΙΚΗ ΕΝΟΤΗΤΑ

ΕΙΣΑΓΩΓΗ ΣΤΗΝ ΠΛΗΡΟΦΟΡΙΚΗ - ΠΛΗ10

ΤΕΛΙΚΕΣ ΕΞΕΤΑΣΕΙΣ - 2 ΙΟΥΝΙΟΥ 2013

ΟΔΗΓΙΕΣ:

- Τα θέματα που έχετε στα χέρια σας είναι **τέσσερις (4) σελίδες**. Επιβεβαιώστε το και αν λείπει κάποια σελίδα ή δεν είναι καλά φωτοτυπημένη ενημερώστε άμεσα τον επιτηρητή.
- Η άριστη απάντηση σε κάθε θέμα βαθμολογείται με **δύο (2) μονάδες**. Τα ερωτήματα κάθε θέματος (α, β, κλπ.) **δεν είναι απαραίτητα ισοδύναμα** μεταξύ τους.
- Απαντήστε **και στα πέντε (5) θέματα** που ακολουθούν.
- Γράψτε στο γραπτό σας **ΚΑΘΑΡΑ ΚΑΙ ΜΕ ΕΥΚΡΙΝΕΙΑ** το **θέμα** και το **ερώτημα** στο οποίο απαντάτε κάθε φορά. Προσπαθήστε να απαντάτε όλα τα ερωτήματα κάθε θέματος μαζί.
- Η αναγραφή σχολίων όπου απαιτείται (προγράμματα σε C, ψευδοκώδικας) είναι υποχρεωτική.
- Ο συνολικός χρόνος εξέτασης είναι **τρεις (3) ώρες** ακριβώς.

Θέμα 1 (2,0 μονάδες)

Ο πίνακας που ακολουθεί αποτελεί τη συνεχόμενη αναπαράσταση ενός δένδρου, του οποίου οι κόμβοι έχουν ακέραιες τιμές, διαφορετικές μεταξύ τους.

1	2	3	4	5	6	7	8	9	10	11	12
14	X	4	10	8	2	3	9	5	7	6	1

- α)** Για ποιες τιμές του X το δυαδικό δένδρο είναι δένδρο - σωρός; Αιτιολογήστε την απάντησή σας.
- β)** Με ποια σειρά θα επισκεφθούμε τους κόμβους του δένδρου, χρησιμοποιώντας: i) την προ-διατεταγμένη, ii) την ενδο-διατεταγμένη και iii) τη μετα-διατεταγμένη διαπέραση;
- γ)** Διαγράφουμε διαδοχικά τη ρίζα του δένδρου - σωρού και κάθε στοιχείο που διαγράφεται το εισάγουμε σε ένα (αρχικά κενό) Δυαδικό Δένδρο Αναζήτησης (ΔΔΑ). Ποιο είναι το ύψος του ΔΔΑ που προκύπτει;
- δ)** Με ποια σειρά θα επισκεφθούμε τους κόμβους του Δυαδικού Δένδρου Αναζήτησης που προέκυψε από το ερώτημα (γ), χρησιμοποιώντας: i) την προ-διατεταγμένη και ii) τη μετα-διατεταγμένη διαπέραση;

Θέμα 2 (2,0 μονάδες)

α1) Η συνάρτηση `pylh` που ακολουθεί έχει ως παραμέτρους τα `a` και `b` που παίρνουν τιμές 0 ή 1 και υλοποιεί τη λειτουργία μιας πύλης.

```
int pylh(int a, int b) {  
    return ((a || b) && !(a && b));  
}
```

- i. Σχεδιάστε τον πίνακα αληθείας βάσει της συνάρτησης.
- ii. Από τον πίνακα αληθείας προσδιορίστε ποια πύλη υλοποιεί η συνάρτηση `pylh`.

α2) Ποια θα είναι η έξοδος του παρακάτω προγράμματος που καλεί τη συνάρτηση `pylh` του ερωτήματος (α1);

```
main () {  
    printf("%d", pylh(1, 0));  
    printf("%d", pylh(1, 1));  
    printf("%d", pylh(0, 1));  
    printf("%d", pylh(0, 0));  
}
```

β) Να γραφεί πρόγραμμα στη γλώσσα προγραμματισμού C, το οποίο θα διαβάζει ένα αλφαριθμητικό το πολύ 20 χαρακτήρων και θα το εμφανίζει με αντίστροφη σειρά (από τον τελευταίο χαρακτήρα προς τον πρώτο).

Υπόδειξη: Μπορείτε να χρησιμοποιήσετε τη συνάρτηση βιβλιοθήκης `strlen()` που επιστρέφει το μήκος ενός αλφαριθμητικού.

Θέμα 3 (2,0 μονάδες)

Να γραφεί πρόγραμμα στη γλώσσα προγραμματισμού C, το οποίο θα διαβάσει με αμυντικό προγραμματισμό μια χρονική στιγμή, η οποία προσδιορίζεται από την ώρα, τα λεπτά και τα δευτερόλεπτα, θα τα αποθηκεύει σε δομή `time`, που δίνεται στη συνέχεια, και θα εμφανίζει στην οθόνη τη χρονική στιγμή μετά από 30 δευτερόλεπτα. Θεωρήστε ότι η ώρα κωδικοποιείται σε 24ωρη βάση (από 0 έως 23).

```
struct time {  
    int hour;  
    int minutes;  
    int seconds;  
};
```

Θέμα 4 (2,0 μονάδες)

α) Γράψτε πρόγραμμα στη γλώσσα προγραμματισμού C, το οποίο θα διαβάζει συνεχώς ακέραιους αριθμούς (θετικούς και αρνητικούς) από το πληκτρολόγιο μέχρι ο χρήστης να δώσει τον αριθμό 0. Μετά από κάθε νέα εισαγωγή αριθμού θα πρέπει να τυπώνει μια γραμμή που θα γράφει κατά σειρά:

- τον αύξοντα αριθμό (A/A) της εισαγωγής,
- τον αριθμό που εισήχθη,
- τον μικρότερο από όλους τους αριθμούς που έχουν εισαχθεί μέχρι εκείνη τη στιγμή,
- τον μεγαλύτερο από όλους τους αριθμούς που έχουν εισαχθεί μέχρι εκείνη τη στιγμή,
- το μέσο όρο όλων των αριθμών που έχουν εισαχθεί μέχρι εκείνη τη στιγμή.

Με την εισαγωγή του αριθμού 0 το πρόγραμμα να εμφανίζει 'Τέλος' και να τερματίζεται.

Θέμα 4 (παράδειγμα εκτέλεσης)

Δώσε αριθμό: 4

A/A=1 Αριθμός = 4 Μικρότερος = 4 Μεγαλύτερος = 4 Μέσος Όρος = 4.00

Δώσε αριθμό: 2

A/A=2 Αριθμός = 2 Μικρότερος = 2 Μεγαλύτερος = 4 Μέσος Όρος = 3.00

Δώσε αριθμό: 13

A/A=3 Αριθμός = 13 Μικρότερος = 2 Μεγαλύτερος = 13 Μέσος Όρος = 6.33

Δώσε αριθμό: -2

A/A=4 Αριθμός = -2 Μικρότερος = -2 Μεγαλύτερος = 13 Μέσος Όρος = 4.25

Δώσε αριθμό: -7

A/A=5 Αριθμός = -7 Μικρότερος = -7 Μεγαλύτερος = 13 Μέσος Όρος = 2.00

Δώσε αριθμό: 0

Τέλος

Θέμα 4 (2,0 μονάδες)/Συνέχεια

β) Να γραφεί σε ψευδοκώδικα ο αλγόριθμος ΥΠΟΛΟΓΙΣΜΟΣ-ΑΘΡΟΙΣΜΑΤΟΣ-ΔΥΝΑΜΕΩΝ που θα υπολογίζει την παρακάτω παράσταση Σ χωρίς τη χρήση τελεστή ύψωσης σε δύναμη.

$$\Sigma = 1^1 + 3^3 + 5^5 + \dots + N^N$$

όπου το N είναι θετικός περιττός ακέραιος αριθμός. Να εφαρμόσετε αμυντικό προγραμματισμό κατά την ανάγνωση του N , ώστε να είναι περιττός και θετικός αριθμός.

Θέμα 5 (2,0 μονάδες)

α) Υπολογίστε το συμπλήρωμα ως προς 2 των παρακάτω δυαδικών αριθμών, όταν το μήκος λέξης είναι αυτό που φαίνεται σε κάθε ερώτημα (δηλαδή, μην συμπληρώσετε επιπλέον μηδενικά στην αρχή των αριθμών):

i) 111

ii) 000111

β) Μετατρέψτε τον αριθμό $29,35_{<10>}$ από το δεκαδικό σύστημα στο δυαδικό, με ακρίβεια 5 κλασματικών ψηφίων, και στη συνέχεια από το δυαδικό σύστημα στο δεκαεξαδικό.

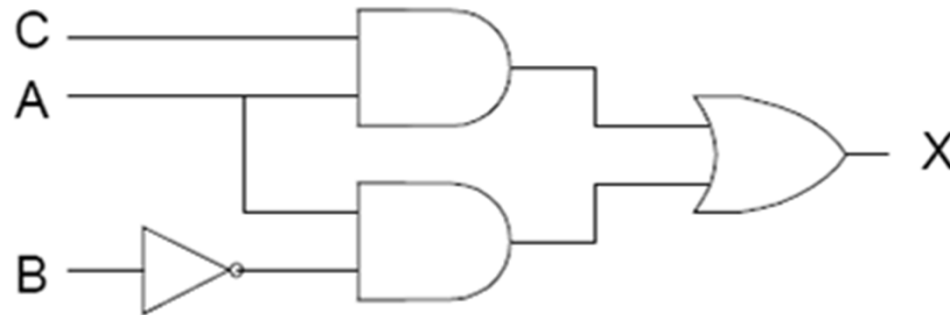
γ) Εκτελέστε τις ακόλουθες πράξεις στο σύστημα αρίθμησης στο οποίο δίνονται οι αριθμοί (δηλαδή, μην τους μετατρέψετε σε άλλο σύστημα για να κάνετε τις πράξεις):

i) $42574,1_{<8>} + 57421,7_{<8>}$

ii) $A1F1,1_{<16>} + D6F,E_{<16>}$

Θέμα 5 (2,0 μονάδες)/Συνέχεια

δ) Να δοθεί ο πίνακας αληθείας καθώς και η λογική συνάρτηση που υλοποιείται από το κύκλωμα του κάτωθι σχήματος:



Επίλυση θεμάτων περασμένων ετών

Τελικές Εξετάσεις 2005: Θέμα 1α

Περιγράψτε τη διαδικασία μετατροπής ενός οκταδικού αριθμού σε δεκαδικό. Στη συνέχεια μετατρέψτε τον αριθμό 57301_8 σε δεκαδικό. Στην απάντησή σας να φαίνονται αναλυτικά τα βήματα της μετατροπής.

Τελικές Εξετάσεις 2005: Θέμα 1α – Λύση

Ξεκινάμε από το λιγότερο σημαντικό ψηφίο (στο τέλος του αριθμού, δεξιά), το οποίο πολλαπλασιάζουμε επί 8 υψωμένο στη μηδενική (δηλαδή επί 1). Πολλαπλασιάζουμε το αμέσως πιο σημαντικό ψηφίο (το δεύτερο από δεξιά) επί 8 υψωμένο στην πρώτη (δηλαδή επί 8), το αμέσως πιο σημαντικό επί 8 υψωμένο στη δεύτερα (δηλαδή επί 64) κ.ο.κ.

Στο τέλος προσθέτουμε αυτά τα αποτελέσματα.

Για παράδειγμα:

$$\begin{aligned} 57301_8 &= 1 \times 8^0 + 0 \times 8^1 + 3 \times 8^2 + 7 \times 8^3 + 5 \times 8^4 = \\ &1 + 0 + 192 + 3584 + 20480 = 24257 \end{aligned}$$

Τελικές Εξετάσεις 2005: Θέμα 1β

Γράψτε σε γλώσσα προγραμματισμού C τη συνάρτηση **void octal_to_decimal(int octal[10])** που μετατρέπει έναν οκταδικό αριθμό στον αντίστοιχο δεκαδικό και στη συνέχεια τον εκτυπώνει. Ο οκταδικός αριθμός είναι είσοδος στη συνάρτηση ως πίνακας 10 θέσεων, κάθε στοιχείο του οποίου είναι ένα οκταδικό ψηφίο. Στην πρώτη θέση του πίνακα βρίσκεται το περισσότερο σημαντικό ψηφίο (MSB).

Για παράδειγμα, ο οκταδικός αριθμός 57301 αποθηκεύεται στον πίνακα με την παρακάτω μορφή:

0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	5	7	3	0	1

Τελικές Εξετάσεις 2005: Θέμα 1β – Λύση

```
void octal_to_decimal(int octal[10]){
    int i, coef; /* i: μετρητής για διάσχιση του πίνακα, coef:
                  συντελεστής με τον οποίο θα πολλαπλασιάζεται κάθε οκταδικό
                  ψηφίο. Αρχικά παίρνει τιμή 1 και στη συνέχεια
                  πολλαπλασιάζεται επί 8 σε κάθε επανάληψη */
    int s = 0; /* s: αθροίζει τα γινόμενα για να υπολογιστεί το
                  τελικό αποτέλεσμα */
    coef = 1;

    for (i = 9; i >= 0; --i) /* Ξεκινάμε από την τελευταία θέση
                              του πίνακα, όπου βρίσκεται το λιγότερο σημαντικό
                              ψηφίο, το οποίο θα πολλαπλασιαστεί επί 1 */
    {
        s += octal[i] * coef; /* το οκταδικό ψηφίο πολλαπλα-
                              σιάζεται επί coef και το αποτέλεσμα προστίθεται στον
                              αθροιστή sum */
        coef = 8 * coef; /* ο συντελεστής coef πολλαπλασιάζεται επί
                           8 σε κάθε επανάληψη (τη δεύτερη φορά γίνεται 8*1=8, την
                           τρίτη 8*8=64, κ.ο.κ.) */
    }
    printf("Ο αντίστοιχος δεκαδικός είναι ο %d", s);
}
```


Τελικές Εξετάσεις 2009: Θέμα 4

- α)** Έστω ένας υπολογιστής με μήκος λέξης 8 δυαδικών ψηφίων, στον οποίο οι πράξεις γίνονται αναπαριστώντας όλους τους αριθμούς με οκτώ δυαδικά ψηφία. Οι αρνητικοί αριθμοί αναπαρίστανται με παράσταση συμπληρώματος ως προς 2. Να εκτελεστεί στον υπολογιστή αυτό η αφαίρεση $(1000)_2 - (1001)_2$, χρησιμοποιώντας τη μέθοδο συμπληρώματος ως προς 2.
- β)** Μετατρέψτε τον δεκαδικό αριθμό $(110)_{10}$ σε δυαδικό, δείχνοντας όλα τα βήματα της μετατροπής.

Τελικές Εξετάσεις 2009: Θέμα 4α – Λύση

- Για την εκτέλεση της αφαίρεσης στο συγκεκριμένο υπολογιστή αναπαριστούμε τους αριθμούς που δόθηκαν με οκτώ δυαδικά ψηφία.
 - Μειωτέος: 0000 1000
 - Αφαιρετέος: 0000 1001
 - Συμπλήρωμα ως προς 1: 1111 0110
 - Συμπλήρωμα ως προς 2: 1111 0111
 - Μειωτέος: 0000 1000
 - Αφαιρετέος: 1111 0111
 - Αποτέλεσμα: 1111 1111
 - (με πρόσθεση ψηφίου προς ψηφίο)
- Το αποτέλεσμα είναι αρνητικός αριθμός, ο οποίος αναπαριστά τον δεκαδικό αριθμό -1.

Τελικές Εξετάσεις 2009: Θέμα 4β – Λύση

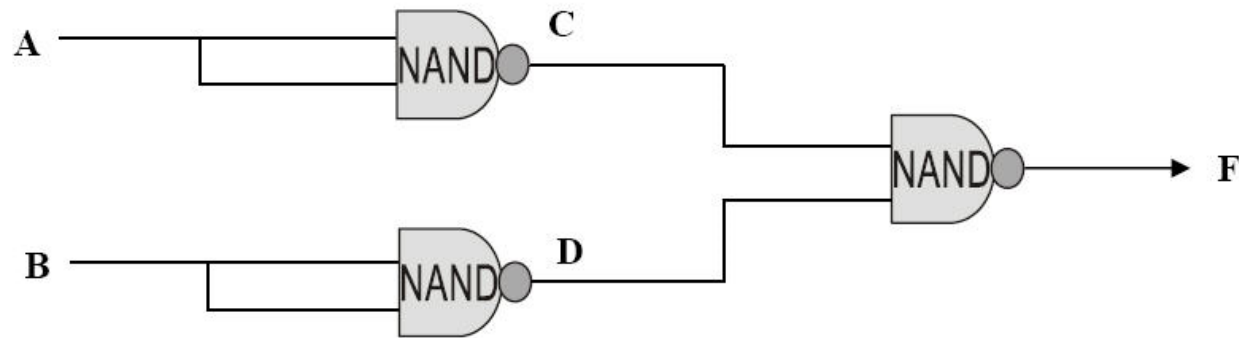
<u>Βάση</u>		<u>Πηλίκο</u>	<u>Υπόλοιπο</u>
110:2	=	55	0
55:2	=	27	1
27:2	=	13	1
13:2	=	6	1
6:2	=	3	0
3:2	=	1	1
1:2	=	0	1

1 1 0 1 1 1 0

Έτσι ο αντίστοιχος δυαδικός αριθμός του 110_{10} είναι ο 1101110_2 .

Τελικές Εξετάσεις 2006: Θέμα 1

α) Δίνεται το ακόλουθο λογικό κύκλωμα:



1. Γράψτε τον πίνακα αληθείας της F ως συνάρτηση των δύο εισόδων του κυκλώματος, A και B , παρουσιάζοντας και τα ενδιάμεσα αποτελέσματα στα σημεία C και D .
 2. Γράψτε πρόγραμμα σε γλώσσα προγραμματισμού C το οποίο να διαβάζει τις τιμές των εισόδων, A και B , του παραπάνω λογικού κυκλώματος και να εμφανίζει την τιμή εξόδου F . Το πρόγραμμα να ελέγχει αν οι τιμές των A και B είναι 0 ή 1 και σε αντίθετη περίπτωση να επαναλαμβάνει την ανάγνωση μέχρι η είσοδος να είναι σωστή.
- β) Να γίνει η αφαίρεση $(38)_{10} - (17)_{10}$ στο δυαδικό σύστημα χρησιμοποιώντας την τεχνική του συμπληρώματος ως προς 2.

Τελικές Εξετάσεις 2006: Θέμα 1 – Λύση (α)

Πίνακας αληθείας της F:

A	B	C	D	F
0	0	1	1	0
0	1	1	0	1
1	0	0	1	1
1	1	0	0	1

```
#include <stdio.h>
int a, b, f;
void main()
{
    do
    {
        printf("\nType value of A (0 - 1): ");
        scanf("%d",&a);
    } while ((a!=0) && (a!=1));
    do
    {
        printf("\nType value for B (0 - 1): ");
        scanf("%d",&b);
    } while ((b!=0) && (b!=1));
    if (a + b == 0) f = 0; else f = 1;
    printf("\nh exodos einai f = %d", f);
}
```

Τελικές Εξετάσεις 2006: Θέμα 1 – Λύση (β)

$$38_{10} = 00100110_2 \text{ και } 17_{10} = 00010001_2$$

Η αφαίρεση των δύο δυαδικών αριθμών υλοποιείται με πρόσθεση μέσω της μετατροπής του αφαιρετέου σε μορφή συμπληρώματος ως προς 2. Για να μετατρέψουμε έναν αριθμό σε μορφή συμπληρώματος ως προς 2, τον μετατρέπουμε σε μορφή συμπληρώματος ως προς 1 και στη συνέχεια προσθέτουμε στο αποτέλεσμα 1. Στη συνέχεια, προσθέτουμε τον μειωτέο και το συμπλήρωμα ως προς 2 του αφαιρετέου.

Συμπλήρωμα ως προς 1 του αφαιρετέου: 11101110

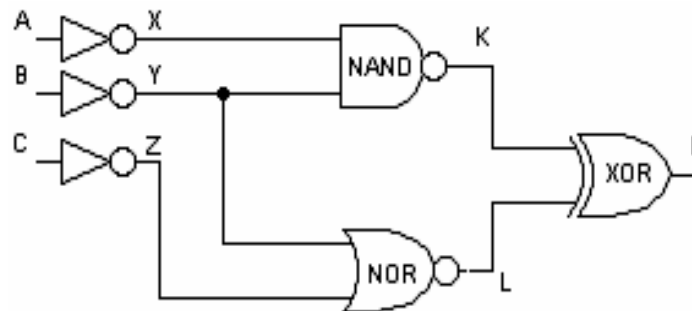
Συμπλήρωμα ως προς 2 του αφαιρετέου: 11101111

00100110		00100110		00100110
- 00010001	→	+ 11101111	→	+ 11101111
				100010101

Από το αποτέλεσμα αφαιρείται το πρώτο ψηφίο (που υπερβαίνει τα 8 bits) και το τελικό αποτέλεσμα είναι 00010101 (ο αντίστοιχος δεκαδικός είναι ο 21).

Επαναληπτικές Εξετάσεις 2006: Θέμα 1

Δίνεται το ακόλουθο λογικό κύκλωμα:



- α) 1. Γράψτε τον αναλυτικό τύπο της F ως συνάρτηση μόνο των τριών εισόδων, A, B και C, του λογικού κυκλώματος.
2. Σχεδιάστε τον πίνακα αληθείας της F, συμπεριλαμβάνοντας τις εισόδους A, B, C, την έξοδο F και τα ενδιάμεσα αποτελέσματα X, Y, Z, K και L.
- β) Γράψτε πρόγραμμα σε γλώσσα C, το οποίο να διαβάζει τις τιμές των εισόδων A, B και C του λογικού κυκλώματος και να εμφανίζει την τιμή εξόδου F του παραπάνω λογικού κυκλώματος. Το πρόγραμμα να ελέγχει αν οι τιμές των A, B και C είναι 0 ή 1 και σε αντίθετη περίπτωση να επαναλαμβάνει την ανάγνωση μέχρι η είσοδος να είναι σωστή.

Επαναληπτικές Εξετάσεις 2006: Θέμα 1 – Λύση (α)

α) 1. $F = ((\text{NOT } A) \text{ NAND } (\text{NOT } B)) \text{ XOR } ((\text{NOT } B) \text{ NOR } (\text{NOT } C))$

2.

A	B	C	X	Y	Z	K	L	F
0	0	0	1	1	1	0	0	0
0	0	1	1	1	0	0	0	0
0	1	0	1	0	1	1	0	1
0	1	1	1	0	0	1	1	0
1	0	0	0	1	1	1	0	1
1	0	1	0	1	0	1	0	1
1	1	0	0	0	1	1	0	1
1	1	1	0	0	0	1	1	0

Επαναληπτικές Εξετάσεις 2006: Θέμα 1 – Λύση (β)

```
#include <stdio.h>
int a, b, c, k, l, f;
void main()
{
    do
    {
        printf("\nType value of A (0 - 1): ");
        scanf("%d",&a);
    } while ((a!=0) && (a!=1));
    do
    {
        printf("\nType value for B (0 - 1): ");
        scanf("%d",&b);
    } while ((b!=0) && (b!=1));
    do
    {
        printf("\nType value of C (0 - 1): ");
        scanf("%d",&c);
    } while ((c!=0) && (c!=1));
    k = !((!a)&&(!b));
    l = !((!b)||(!c));
    f = (k && (l)) || ((!k) && l);
    printf("\nh exodos einai f = %d", f);
}
```

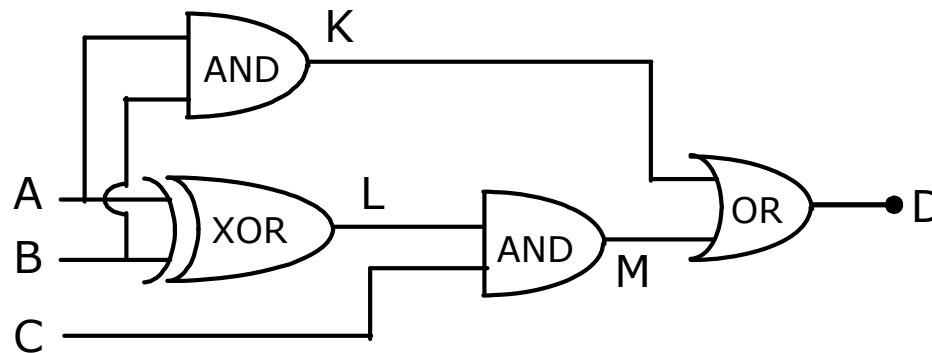
**A NAND B =
!(a && b)**

**A NOR B =
!(a || b)**

**A XOR B =
(a && (!b)) || ((!a) && b)**

Τελικές Εξετάσεις 2005: Θέμα 4β

Δίνεται το ακόλουθο λογικό κύκλωμα:



1. Να γραφεί η λογική συνάρτηση D που αντιστοιχεί στο λογικό κύκλωμα, ως συνάρτηση των εισόδων A, B και C.
2. Να γραφεί ο πίνακας αληθείας του λογικού κυκλώματος, συμπεριλαμβάνοντας και τα ενδιάμεσα αποτελέσματα K, L και M.

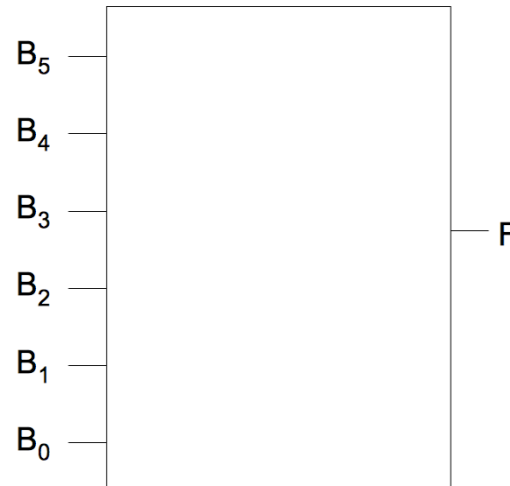
Τελικές Εξετάσεις 2005: Θέμα 4β – Λύση

1. $D = (A \oplus B) \cdot C + A \cdot B$, ή
 $D = ((A \text{ XOR } B) \text{ AND } C) \text{ OR } (A \text{ AND } B)$
- 2.

A	B	C	K=A.B	L=A⊕B	M=L.C	D=K+M
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	1	0	0
0	1	1	0	1	1	1
1	0	0	0	1	0	0
1	0	1	0	1	1	1
1	1	0	1	0	0	1
1	1	1	1	0	0	1

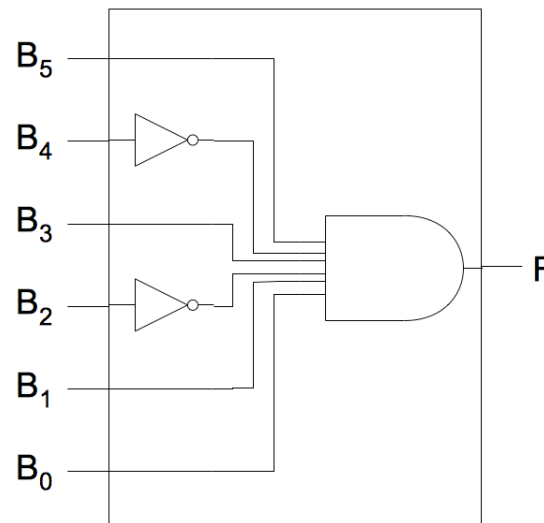
Τελικές Εξετάσεις 2010: Θέμα 4α

Ηλεκτρονική Κλειδαριά: Τοποθετήστε και συνδέστε τις κατάλληλες πύλες στο εσωτερικό του παρακάτω κουτιού, ώστε η έξοδος του κυκλώματος να γίνεται 1 μόνο όταν στην είσοδο εμφανίζεται ο συνδυασμός 101011 (θεωρήστε το B5 ως το πιο αριστερό στοιχείο). Εξηγήστε περιληπτικά το συλλογισμό σας.



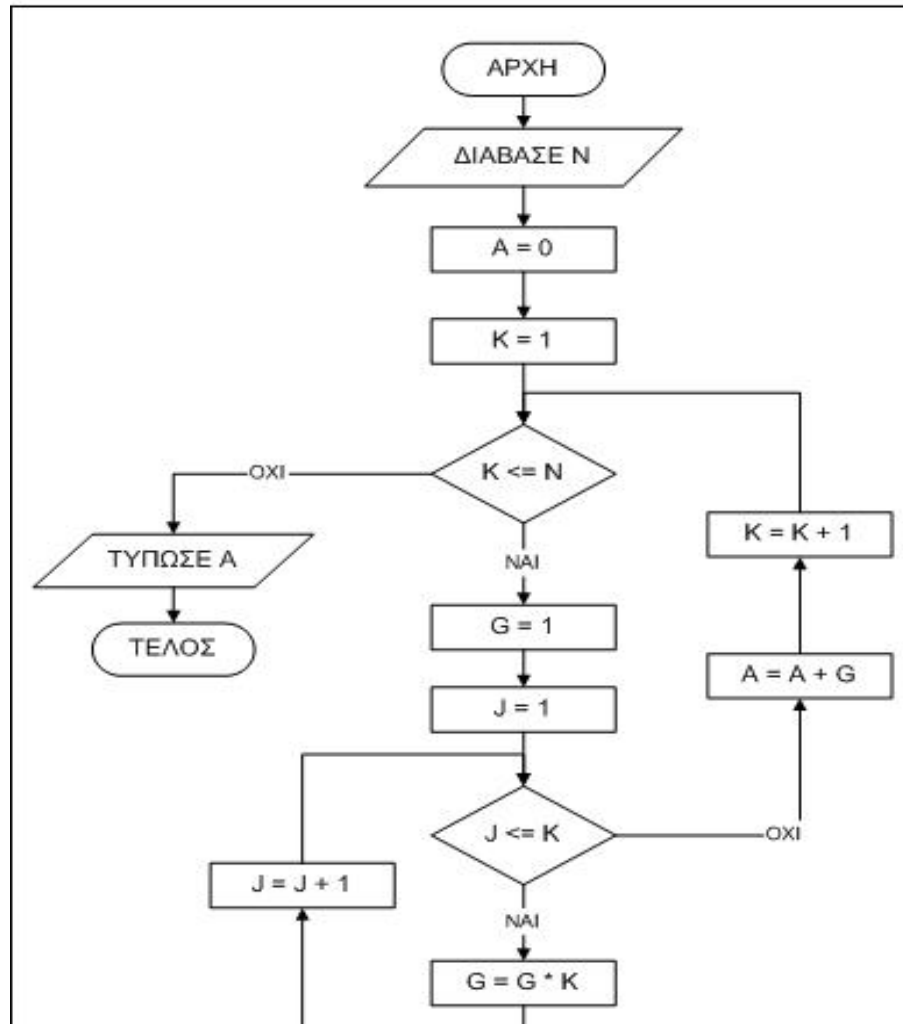
Τελικές Εξετάσεις 2010: Θέμα 4α – Λύση

Αν ο συνδυασμός ήταν όλο άσσοι (111111) θα αρκούσε μια AND. Τώρα θα πρέπει να αλλάξουμε με NOT τις εισόδους που είναι 0 πριν τις οδηγήσουμε στην AND.



Τελικές Εξετάσεις 2006: Θέμα 2

Δίνεται το παρακάτω Διάγραμμα Ροής Προγράμματος - ΔΡΠ (flowchart), όπου το N είναι θετικός ακέραιος αριθμός μεγαλύτερος του μηδενός ($N > 0$).



- α) Τι εκτυπώνει ο αλγόριθμος αν στην είσοδο δώσουμε την τιμή:
- i) $N = 1$,
 - ii) $N = 2$, και
 - iii) $N = 3$.
- β) Γράψτε πρόγραμμα σε γλώσσα προγραμματισμού C που να έχει την ίδια λειτουργία με το ΔΡΠ, εφαρμόζοντας επιπρόσθετα και αμυντικό προγραμματισμό στην εισαγωγή του N , έτσι ώστε να είναι ακέραιος αριθμός μεγαλύτερος του μηδενός ($N > 0$).

Τελικές Εξετάσεις 2006: Θέμα 2 – Λύση

α) Εκτέλεση του αλγορίθμου:

i) για $N = 1$: Εκτυπώνεται η τιμή 1

ii) για $N = 2$: Εκτυπώνεται η τιμή 5

iii) για $N = 3$: Εκτυπώνεται η τιμή 32

Στη γενική περίπτωση, το πρόγραμμα εκτυπώνει την παράσταση $A = 1^1 + 2^2 + 3^3 + \dots + N^N$

(β) Πρόγραμμα σε γλώσσα C:

```
#include <stdio.h>

int n, a, k, g, j;

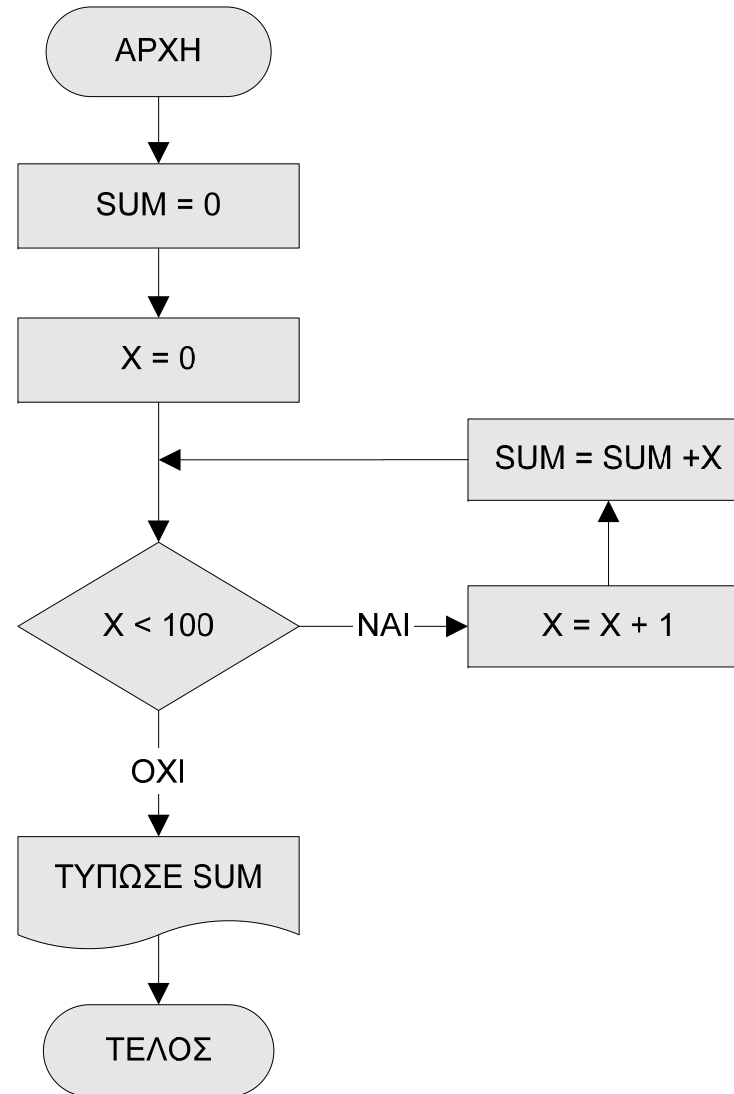
void main()
{
    do
    {
        printf("Dwse timh gia to N (akeraios thetikos arithmos): ");
        scanf("%d", &n);
    } while (n <= 0);

    a = 0;
    for (k = 1; k <= n; ++k)
    {
        g = 1;
        for (j = 1; j <= k; ++j)
            g = g * k;
        a = a + g;
    }

    printf("to apotelesma einai a = %d", a);    // εμφάνιση της εξόδου k
```

Τελικές Εξετάσεις 2005: Θέμα 2γ

Δίνεται το επόμενο Διάγραμμα Ροής Προγράμματος (flowchart) και ζητείται να εξηγήσετε (σε 1-2 γραμμές) τι κάνει και να γράψετε πρόγραμμα σε C που να υλοποιεί την ίδια λειτουργία.



Τελικές Εξετάσεις 2005: Θέμα 2γ – Λύση

Η λειτουργία που εκτελείται είναι η άθροιση των ακεραίων αριθμών από το 1 μέχρι το 100 ($SUM = 1+2+3+\dots+100$).

Το αντίστοιχο πρόγραμμα σε C είναι:

```
#include <stdio.h>
void main()
{
    int x, SUM = 0; /* x: μετρητής από 1 έως 100, SUM:
                     αθροίζει τα x */

    for (x=1; x<=100; x++)
        SUM += x; /* κάθε φορά στο sum προστίθεται το νέο x
                     που αρχίζει από 1 και αυξάνεται μέσω της for κατά 1
                     μέχρι να φτάσει το 100 */
    printf("%d", SUM);
}
```

Επαναληπτικές Εξετάσεις 2007: Θέμα 3

Να γραφεί αλγόριθμος σε ψευδοκώδικα, ο οποίος:

- α)** Να ζητά από τον χρήστη την εισαγωγή των στοιχείων ενός μονοδιάστατου πίνακα A , 35 θέσεων, που περιέχει θετικούς ακέραιους αριθμούς. Σε περίπτωση που ο χρήστης πληκτρολογήσει μη θετικό αριθμό, να ζητά την επανάληψη της πληκτρολόγησης μέχρις ότου ο αριθμός που θα πληκτρολογηθεί να είναι θετικός.
- β)** Να μεταφέρει τα στοιχεία του πίνακα A στον δισδιάστατο πίνακα B , διαστάσεων 7×5 , με τέτοιο τρόπο ώστε πρώτα να συμπληρωθεί η πρώτη γραμμή του πίνακα B , μετά η δεύτερη γραμμή του κ.ο.κ. Στη συνέχεια να εκτυπώνει τα στοιχεία του πίνακα B .

Επαναληπτικές Εξετάσεις 2007: Θέμα 3 – Λύση (α)

**ΑΛΓΟΡΙΘΜΟΣ ΕΙΣΑΓΩΓΗ-ΜΕΤΑΦΟΡΑ
ΣΤΑΘΕΡΕΣ**

M = 7;
N = 5;

ΔΕΔΟΜΕΝΑ

A: **ARRAY** [1..M*N] **OF INTEGER**;
B: **ARRAY** [1..M, 1..N] **OF INTEGER**;
I, J, K: **INTEGER**;

ΑΡΧΗ

/ Βρόχος για ανάγνωση του πίνακα A */*

ΓΙΑ I := 1 **ΕΩΣ** M*N **ΕΠΑΝΑΛΑΒΕ**

ΕΠΑΝΑΛΑΒΕ */* Αμυντικός προγραμματισμός */*

ΔΙΑΒΑΣΕ (A[I]);

ΕΑΝ (A[I] <= 0) **ΤΟΤΕ**

ΤΥΠΩΣΕ ("Ο ΑΡΙΘΜΟΣ ΠΡΕΠΕΙ ΝΑ ΕΙΝΑΙ ΘΕΤΙΚΟΣ")

ΕΑΝ-ΤΕΛΟΣ

ΜΕΧΡΙ (A[I] > 0)

ΓΙΑ-ΤΕΛΟΣ;

Επαναληπτικές Εξετάσεις 2007: Θέμα 3 – Λύση (β)

```
K:= 1;  
/* Διπλός βρόχος για διάσχιση του B κατά γραμμές */  
ΓΙΑ I:= 1 ΕΩΣ M ΕΠΑΝΑΛΑΒΕ  
    ΓΙΑ J:= 1 ΕΩΣ N ΕΠΑΝΑΛΑΒΕ  
        B[I, J] = A[K];  
/* Μετά από την ενημέρωση για τη θέση K, το K αυξάνεται κατά 1 */  
        K:= K + 1  
    ΓΙΑ-ΤΕΛΟΣ  
ΓΙΑ-ΤΕΛΟΣ;  
  
/* Εκτύπωση του πίνακα B */  
ΓΙΑ I:= 1 ΕΩΣ M ΕΠΑΝΑΛΑΒΕ  
    ΓΙΑ J:= 1 ΕΩΣ N ΕΠΑΝΑΛΑΒΕ  
        ΤΥΠΩΣΕ (B[I, J])  
    ΓΙΑ-ΤΕΛΟΣ;  
    ΤΥΠΩΣΕ (ΕΟΛΝ) /* Αλλαγή γραμμής */  
ΓΙΑ-ΤΕΛΟΣ  
ΤΕΛΟΣ
```

Επαναληπτικές Εξετάσεις 2005: Θέμα 3α

Έστω ότι μια απλά συνδεδεμένη λίστα με στοιχεία ακέραιους αριθμούς λειτουργεί με τη λογική της στοίβας, δηλαδή κάθε στοιχείο εισάγεται στην αρχή της λίστας και κάθε στοιχείο εξάγεται από την αρχή της λίστας. Σχεδιάστε σε ψευδοκώδικα τον αλγόριθμο εξαγωγής ενός στοιχείου από την απλά συνδεδεμένη λίστα - στοίβα. Χρησιμοποιήστε οποιονδήποτε ψευδοκώδικα επιθυμείτε.

Επαναληπτικές Εξετάσεις 2005: Θέμα 3α – Λύση

Είσοδος: Ο δείκτης αρχής της συνδεδεμένης λίστας (ARXH)

Έξοδος: Νέος δείκτης αρχής της συνδ. λίστας και τα δεδομένα του κόμβου που διαγράφηκε

DS-DIAGRAFH(ARXH)

if (ARXH = NIL) **then**

print 'Δεν υπάρχει στοιχείο για διαγραφή'

else

 Data \leftarrow STOIXEIO(KOMBOS(ARXH));

 ARXH \leftarrow DEIKTHS(KOMBOS(ARXH));

endif

TELOS DS-DIAGRAFH

Τελικές Εξετάσεις 2005: Θέμα 2α

Πόσες φορές θα εκτυπωθεί το Χ στα παρακάτω τμήματα κώδικα; Δικαιολογήστε την απάντησή σας.

1.

```
for (i=0; i<100; i++)  
{  
    printf("X");  
    i++;  
}
```

2.

```
for (i=0; i<100; i++)  
    printf("X");  
for (j=i; j<200; j++)  
    printf("X");
```

Τελικές Εξετάσεις 2005: Θέμα 2α – Λύση

1. **50** φορές, καθώς το i αυξάνεται κατά 1 τόσο στη for όσο και μέσα στο βρόγχο, συνεπώς σε κάθε επανάληψη αυξάνεται κατά 2.
2. **200** φορές, 100 από τον πρώτο βρόγχο και 100 από τον δεύτερο (το j αρχικοποιείται σε i στην αρχή του δεύτερου βρόγχου και το i έχει τιμή 100 μετά την εκτέλεση του προηγούμενου).

Τελικές Εξετάσεις 2005: Θέμα 4α

```
func(int k)
{
    int i;

    if (k==0)
        return;
    printf("%d", k);
    for (i=1;i<k;i++)
        func(i);
}
```

Τι θα εκτυπωθεί από την κλήση της func(4);
Προσοχή! Η func είναι αναδρομική συνάρτηση!

Τελικές Εξετάσεις 2005: Θέμα 4α – Λύση

Η **func(0)** δεν εκτυπώνει τίποτα και επιστρέφει.

Η **func(1)** θα εκτυπώσει “1” και θα επιστρέψει.

Η **func(2)** θα εκτυπώσει “2” και θα καλέσει τη **func(1)**, οπότε συνολικά θα εκτυπώσει “21”.

Η **func(3)** θα εκτυπώσει “3” και θα καλέσει τη **func(1)** και τη **func(2)**, οπότε συνολικά θα εκτυπώσει “3121”.

Η **func(4)** θα εκτυπώσει “4” και θα καλέσει την **func(1)**, την **func(2)** και την **func(3)**, οπότε θα εκτυπώσει συνολικά “41213121”.

Τελικές Εξετάσεις 2005: Θέμα 5

Να γραφεί πρόγραμμα σε C το οποίο:

(α) Να ζητά από το χρήστη την εισαγωγή των στοιχείων ενός πίνακα θετικών ακεραίων αριθμών διαστάσεων 4×5 . Σε περίπτωση που ο χρήστης πληκτρολογήσει μη θετικό αριθμό, να ζητά την επανάληψη της πληκτρολόγησης, μέχρις ότου ο αριθμός που θα πληκτρολογηθεί να είναι θετικός.

(β) Να μεταφέρει τα στοιχεία του δισδιάστατου πίνακα σε μονοδιάστατο πίνακα, με τέτοιο τρόπο ώστε πρώτα να μεταφερθεί η πρώτη στήλη του αρχικού πίνακα, μετά η δεύτερη κ.ο.κ. Στη συνέχεια να εκτυπώνει τα στοιχεία του νέου πίνακα.

Τελικές Εξετάσεις 2005: Θέμα 5 – Λύση

```
#include <stdio.h>

void main()
{
    int i, j, k, l, a[4][5], b[20];          /* i, j: μετρητές για διάσχιση του δισδιάστατου
        πίνακα a, k, l: μετρητές για το μονοδιάστατο πίνακα b */
    for (i=0; i<4; i++) /* Διπλός βρόχος για ανάγνωση του πίνακα a */
        for (j=0; j<5; j++)
            do{
                printf("Dose to stoixeio a[%d][%d]\n", i, j);
                scanf("%d", &a[i][j]);
            } while (a[i][j]<=0); /* Αμυντικός προγραμματισμός */
    k = 0;
```

Τελικές Εξετάσεις 2005: Θέμα 5 – Λύση

```
for (j=0; j<5; j++) /* Διπλός βρόχος για διάσχιση του a κατά στήλες */
    for (i=0; i<4; i++){
        b[k] = a[i][j];
        k++;      /* Μετά από την ενημέρωση της θέσης k του πίνακα b, ο μετρητής k
                     αυξάνεται κατά 1 */
    }
for (l=0; l<20; l++)
    printf("b[%d]=%d\n", l, b[l]); /* Εκτύπωση του πίνακα b */
}
```

Τελικές Εξετάσεις 2005: Θέμα 6β

Δώστε σε C τη συνάρτηση:

```
float conditional_mean(unsigned int *array, int N, int val)
```

η οποία δέχεται ως είσοδο:

- έναν δείκτη σε πίνακα μη προσημασμένων ακεραίων (`unsigned int * array`),
- το μέγεθος του πίνακα (`int N`), και
- έναν ακέραιο (`int val`),

και επιστρέφει το μέσο όρο των στοιχείων του πίνακα που έχουν τιμή μικρότερη του `val`.

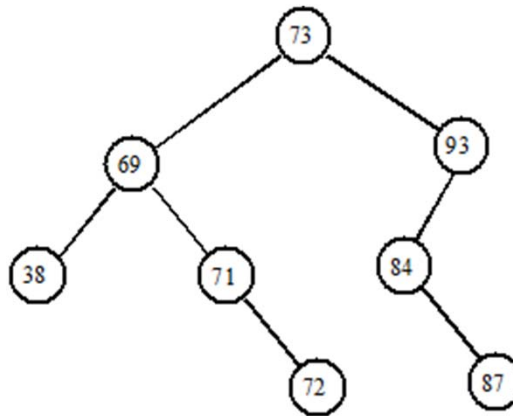
Σε περίπτωση που δεν βρεθεί στοιχείο μικρότερο του `val`, η συνάρτηση επιστρέφει την τιμή -1.

Τελικές Εξετάσεις 2005: Θέμα 6β – Λύση

```
float conditional_mean(unsigned int *array, int N, int val)
{
    int sum = 0;          /* sum: αθροιστής των στοιχείων του πίνακα a */
    int count = 0;        /* count: μετρητής πλήθους στοιχείων με τιμή < val */
    int i;                /* i: μετρητής για διάσχιση του πίνακα array */

    for (i=0; i<N; i++) /* διάσχιση του πίνακα array */
    {
        if (array[i] < val) /* Αν το στοιχείο του πίνακα array είναι <
            val*/
        {
            sum += array[i]; /* Προσθέτουμε το στοιχείο στο sum ... */
            count ++;        /* ... και αυξάνουμε το πλήθος των στοιχείων κατά 1
            */
        }
    }
    if (count == 0) return (-1.0); /* Αμυντικός προγραμματισμός: αν δεν
        υπήρχε στοιχείο του πίνακα array με τιμή < val, δεν
        επιτρέπουμε διαίρεση με 0 */
    else return ((float) sum / count); /* Διαφορετικά,
        επιστρέφεται ο μέσος όρος */
}
```

Τελικές Εξετάσεις 2005: Θέμα 3



Δίνεται το εξής Δυαδικό Δέντρο Αναζήτησης:

- (α) Με ποια σειρά θα επισκεφθούμε τους κόμβους του δέντρου αν χρησιμοποιήσουμε την προ-διατεταγμένη (**pre-order**) διαπέραση;
- (β) Δημιουργήστε **δέντρο – σωρό** με τα ίδια στοιχεία, τοποθετώντας τα με τη σειρά που προέκυψε από την προ-διατεταγμένη διαπέραση στο ερώτημα (α). Στο γραπτό σας να φαίνεται η μορφή του δέντρου μετά από την ολοκλήρωση της εισαγωγής κάθε στοιχείου.
- (γ) **Διαγράψτε** τη ρίζα του δέντρου – σωρού που προκύπτει από το ερώτημα (β). Τα βήματα της διαγραφής της ρίζας να φαίνονται αναλυτικά στο γραπτό σας.
- (δ) Με ποια σειρά θα επισκεφθούμε τους κόμβους του δέντρου – σωρού που προκύπτει από το ερώτημα (γ) αν χρησιμοποιήσουμε την ενδο-διατεταγμένη (**in-order**) διαπέραση;

Τελικές Εξετάσεις 2005: Θέμα 3α – Λύση

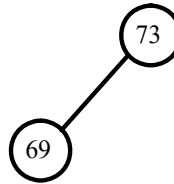
Η προ-διατεταγμένη διαπέραση μας δίνει:

73, 69, 38, 71, 72, 93, 84, 87

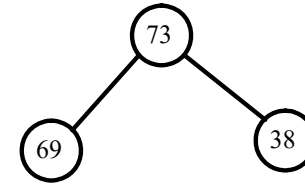
Τελικές Εξετάσεις 2005: Θέμα 3β – Λύση



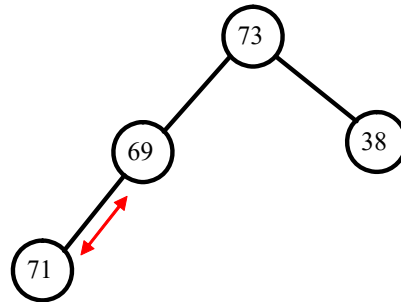
i. Εισαγωγή του 73



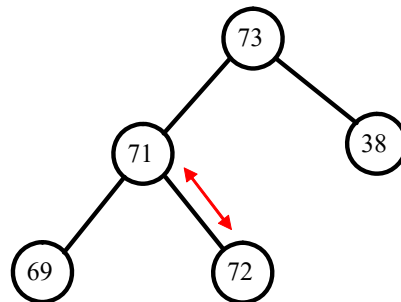
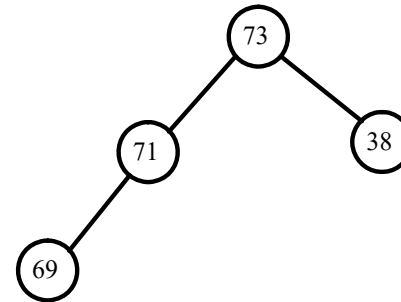
ii. Εισαγωγή του 69



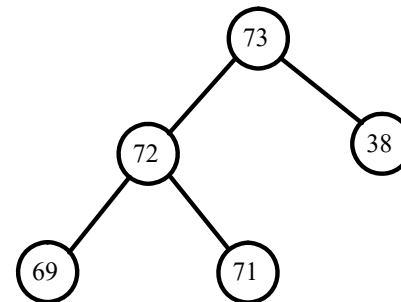
iii. Εισαγωγή του 38



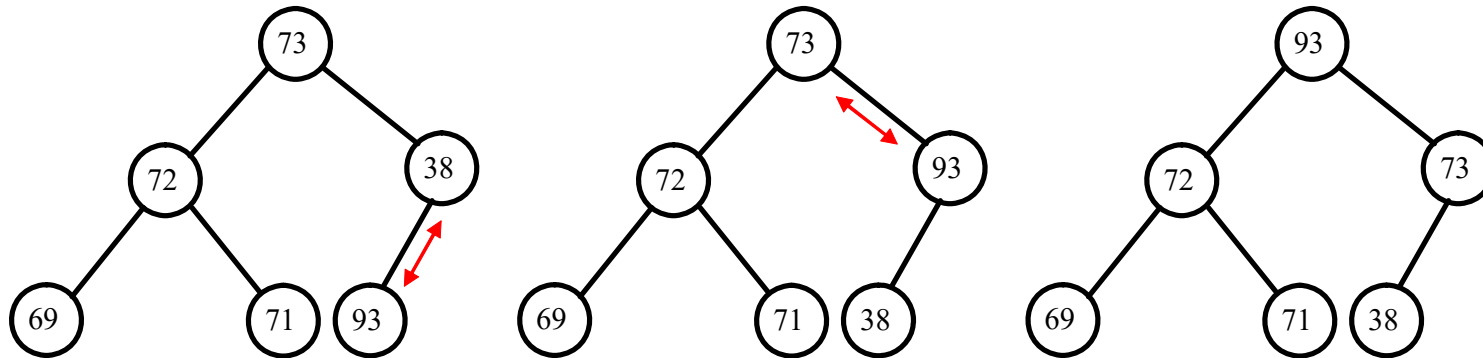
iv. Εισαγωγή του 71 (αντιμετάθεση με το 69)



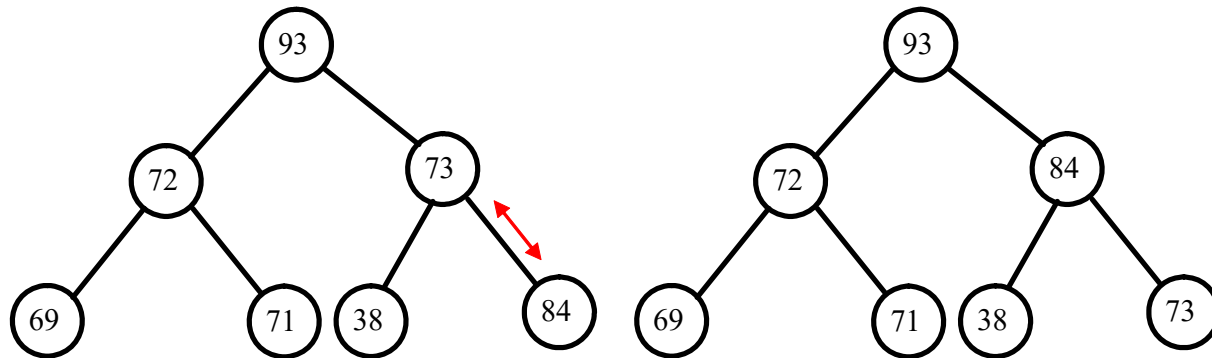
v. Εισαγωγή του 72 (αντιμετάθεση με το 71)



Τελικές Εξετάσεις 2005: Θέμα 3β – Λύση (συν.)

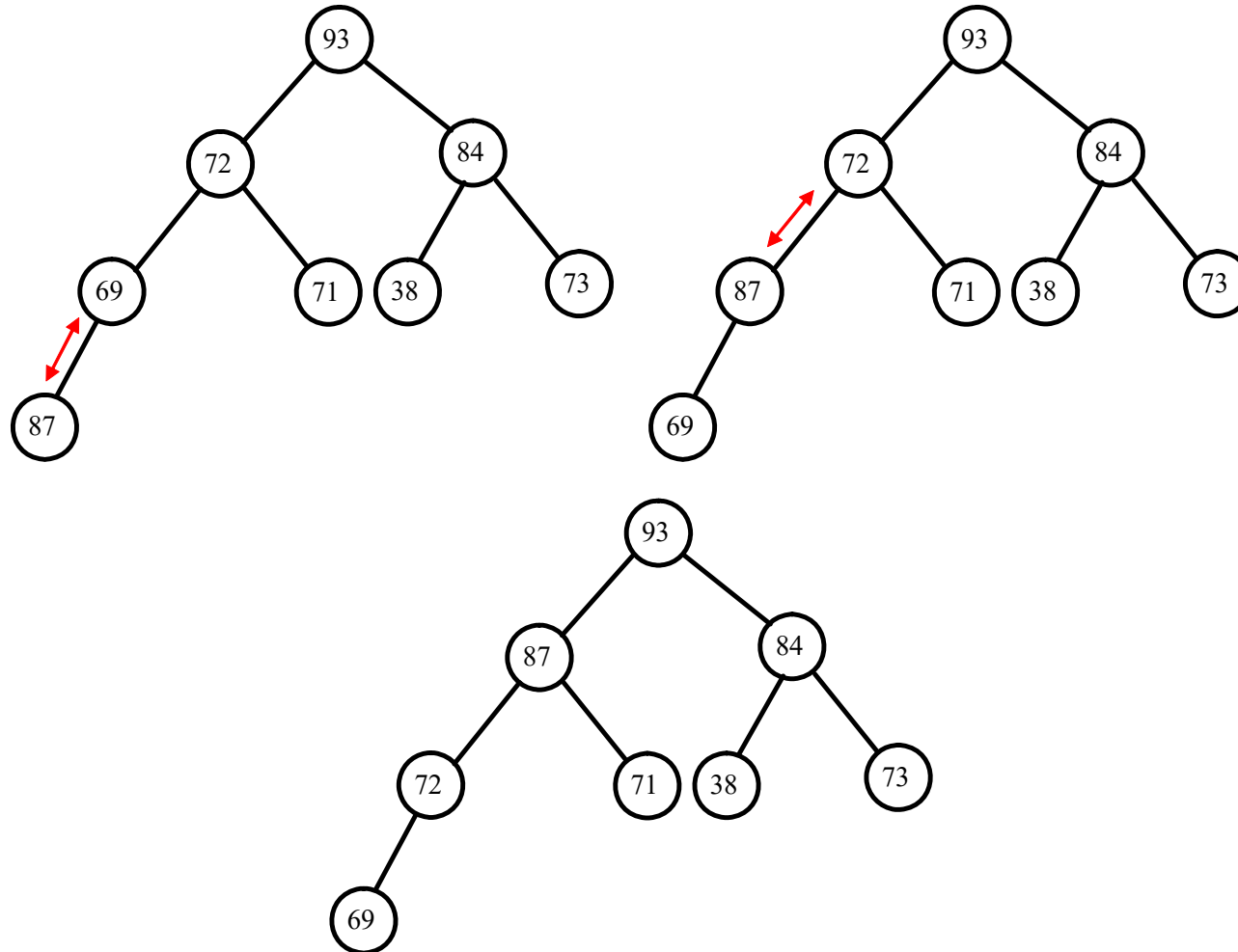


vi. Εισαγωγή του 93 (αντιμετάθεση με το 38 και στη συνέχεια με το 73)



vii. Εισαγωγή του 84 (αντιμετάθεση με το 73)

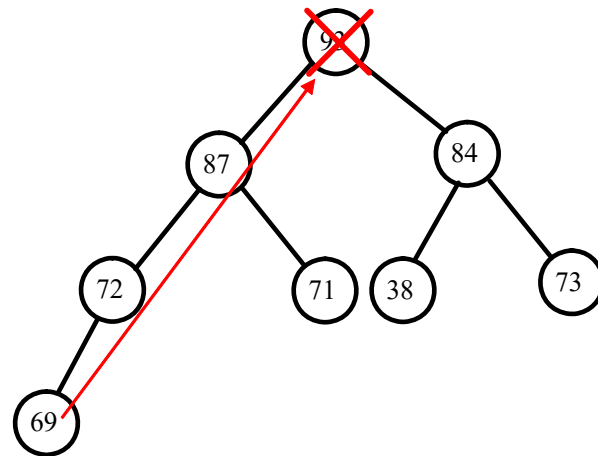
Τελικές Εξετάσεις 2005: Θέμα 3β – Λύση (συν.)



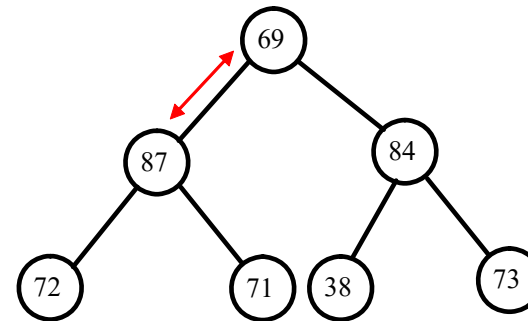
viii. Εισαγωγή του 87 (αντιμετάθεση με το 69 και στη συνέχεια με το 72)

Τελικές Εξετάσεις 2005: Θέμα 3γ – Λύση

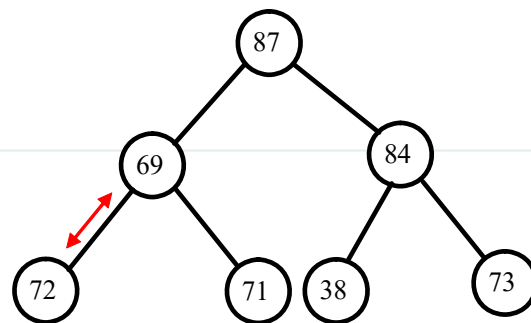
Η διαδικασία διαγραφής της ρίζας (93) του δέντρου - σωρού είναι:



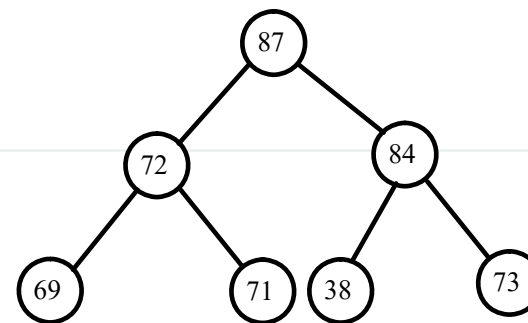
i. Διαγραφή του 93 (τοποθέτηση του 69 (τελευταίου κόμβου) στη θέση του



ii. Αντιμετάθεση του 69 με τα μεγαλύτερο από τα παιδιά του (87)



iii. Αντιμετάθεση του 69 με τα μεγαλύτερο από τα παιδιά του (72)



iv. Αντιμετάθεση του 69 με τα μεγαλύτερο από τα παιδιά του (87)

Τελικές Εξετάσεις 2005: Θέμα 3δ – Λύση

Η ενδο-διατεταγμένη διαπέραση μας δίνει:

69, 72, 71, 87, 38, 84, 73

Επαναληπτικές Εξετάσεις 2004: Θέμα 4β

Δώστε σε C τη συνάρτηση:

```
float list_mean(struct node * ptr)
```

η οποία δέχεται ως είσοδο μία διασυνδεδεμένη λίστα στοιχείων τύπου node με ορισμό:

```
struct node {  
    int value;  
    struct node * next;  
}
```

και επιστρέφει το μέσο όρο των τιμών των πεδίων value των κόμβων της λίστας.

Επαναληπτικές Εξετάσεις 2004: Θέμα 4β – Λύση

```
float list_mean(struct node * ptr) /*Δήλωση της συνάρτησης
list_mean */
{
    unsigned long count = 0; /*Δήλωση και αρχικοποίηση της τοπικής
μεταβλητής - μετρητή count (μετρά το πλήθος των στοιχείων της λίστας) */
    long total_value = 0; /*Δήλωση και αρχικοποίηση της τοπικής
μεταβλητής - αθροιστή total_value (αθροίζει τα στοιχεία της λίστας) */
    while (ptr != NULL) /*Όσο η λίστα δεν έχει φτάσει στο τέλος */
    {
        count++; /*μετράμε το πλήθος των στοιχείων της */
        total_value += ptr->value; /*αθροίζουμε τις τιμές */
        ptr = ptr->next; /*και μεταβαίνουμε στον επόμενο κόμβο */
    }
    return ((float) total_value / (float) count);
    /* Στο τέλος, η συνάρτηση υπολογίζει και επιστρέφει το μέσο όρο */
}
```


Επαναληπτικές Εξετάσεις 2004: Θέμα 5

Στον πίνακα που ακολουθεί είναι αποθηκευμένοι οι κόμβοι ενός δένδρου - σωρού (συνεχόμενη αναπαράσταση δένδρου).

1	2	3	4	5	6	7	8	9	10	11	12
A	B	Γ	Δ	E	Z	H	Θ	I	K	Λ	M

α) Κατασκευάστε το δένδρο - σωρό.

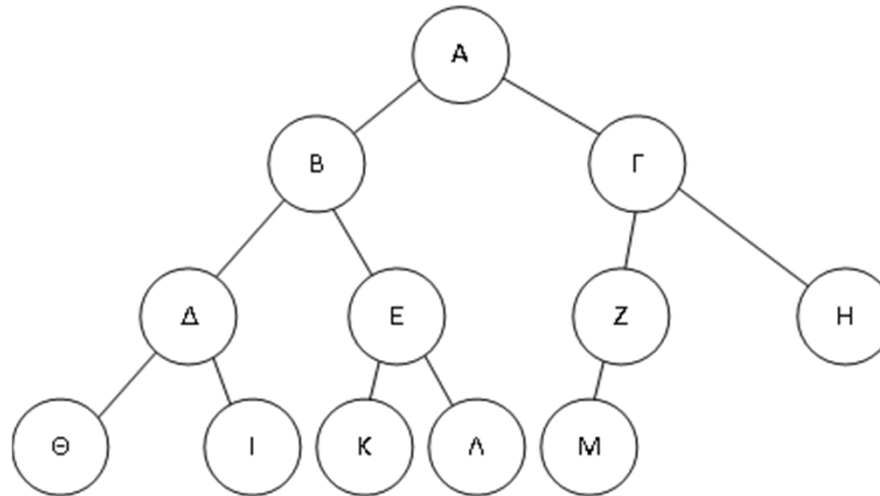
β) Προσδιορίστε τις σχέσεις (μικρότερο, μεγαλύτερο ή απροσδιόριστο) που ισχύουν μεταξύ των στοιχείων B, Γ, Δ, E και Z ανά δύο, με βάση τη θέση των στοιχείων στο δένδρο. Αιτιολογήστε την απάντησή σας.

Υπόδειξη: Προσδιορίστε τις σχέσεις και για τους 10 διαφορετικούς συνδυασμούς των B, Γ, Δ, E, Z.

γ) Με ποια σειρά θα επισκεφθούμε τους κόμβους του δένδρου χρησιμοποιώντας τη μετα-διατεταγμένη διαπέραση;

Επαναληπτικές Εξετάσεις 2004: Θέμα 5 – Λύση

α)



β)

Ισχύει: $B > \Delta$, $B > E$ και $\Gamma > Z$ λόγω σχέσης γονέα - παιδιών σε δένδρο - σωρό.

Οι σχέσεις μεταξύ B και Γ , μεταξύ B και Z , μεταξύ Γ και Δ , μεταξύ Γ και E , μεταξύ Δ και E , μεταξύ Δ και Z και μεταξύ E και Z είναι απροσδιόριστες.

γ)

Μετα-διατεταγμένη διαπέραση: $\Theta - \text{Ι} - \Delta - \text{Κ} - \Lambda - E - B - \text{Μ} - Z - \text{Η} - \Gamma - A$

Τελικές Εξετάσεις 2004: Θέμα 1α

- i. Σχεδιάστε ένα Δυαδικό Δένδρο Αναζήτησης με 4 στοιχεία και ύψος 3.

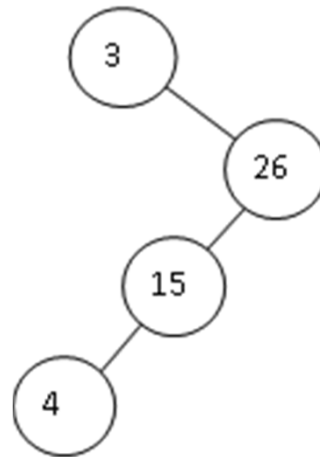
Υποδείξεις:

1. Επιλέξτε τα στοιχεία κατά βούληση (διαλέξτε εσείς τους τέσσερις αριθμούς του δένδρου).
2. Θεωρήστε ότι η ρίζα του δένδρου έχει ύψος 0 (όπως ακριβώς αναφέρεται και στο βιβλίο σας).

- ii. Ποιο είναι το ύψος ενός Δένδρου - Σωρού με 10 στοιχεία; Αιτιολογήστε την απάντησή σας.

Τελικές Εξετάσεις 2004: Θέμα 1α - Λύση

i.



- ii. Το δένδρο - σωρός είναι πάντα πλήρες δένδρο. Για το λόγο αυτό το ύψος του δένδρου είναι 3 (1 κόμβος στο επίπεδο 0, 2 κόμβοι στο επίπεδο 1, 4 κόμβοι στο επίπεδο 2 και 3 κόμβοι στο επίπεδο 3).

Τελικές Εξετάσεις 2004: Θέμα 1β

Στο παρακάτω τμήμα κώδικα C πόσες φορές θα τυπωθεί το X;

```
for (i=0; i<7; i++)  
{  
    printf("X");  
    for (j=0; j<5; j++)  
        printf("X");  
}
```

Τελικές Εξετάσεις 2004: Θέμα 1β – Λύση

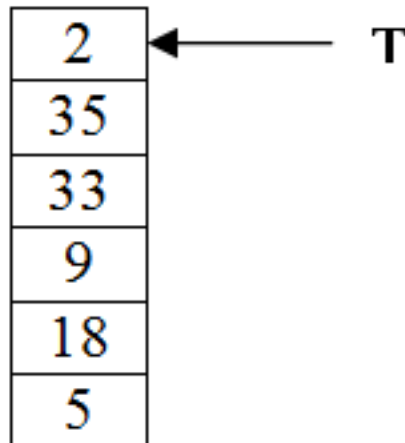
Η πρώτη `printf()` θα εκτελεστεί 7 φορές, όσες και ο εξωτερικός βρόγχος, και κατά συνέπεια θα εμφανίσει 7 φορές το X.

Η δεύτερη `printf()` θα εκτελεστεί $7 \cdot 5 = 35$ φορές, όσες και ο εσωτερικός βρόγχος, και κατά συνέπεια θα εμφανίσει 35 φορές το X.

Συνολικά το X θα εμφανιστεί $35 + 7 = 42$ φορές.

Τελικές Εξετάσεις 2004: Θέμα 1γ

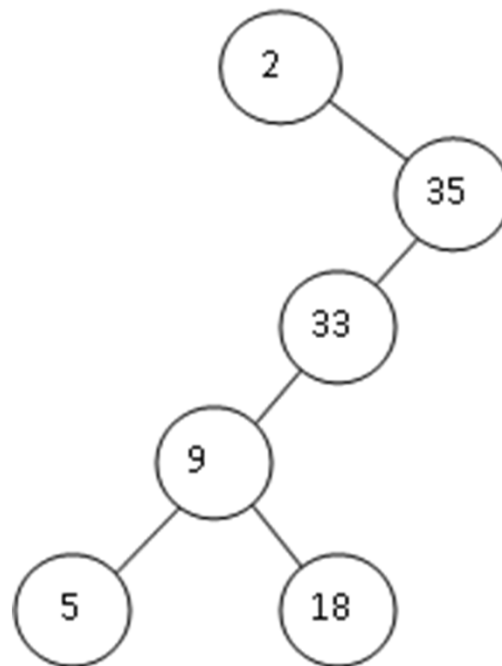
Έστω στοίβα S με τα εξής έξι στοιχεία:



Δώστε το δέντρο που σχηματίζεται διαγράφοντας ένα προς ένα τα στοιχεία από τη στοίβα και εισάγοντάς τα (με τη σειρά που διαγράφονται) σε ένα Δυαδικό Δένδρο Αναζήτησης. Στην απάντησή σας δεν απαιτείται αλγόριθμος, ούτε περιγραφή, δώστε μόνο το τελικό δένδρο που προκύπτει.

Τελικές Εξετάσεις 2004: Θέμα 1β – Λύση

Τα στοιχεία διαγράφονται από τη στοίβα (και αντίστοιχα εισάγονται στο ΔΔΑ) με την εξής σειρά: {2, 35, 33, 9, 18, 5}. Άρα, το ΔΔΑ αναζήτησης που προκύπτει είναι το εξής:




Επαναληπτικές Εξετάσεις 2006: Θέμα 6β

Δίνεται ο πίνακας A με τα εξής στοιχεία:

1	2	3	4	5	6
61	49	23	32	35	12

Δείξτε τις διαδοχικές φάσεις του πίνακα A κατά τη διάρκεια διάταξης των στοιχείων του σε αύξουσα σειρά με τον αλγόριθμο ταξινόμησης φυσαλίδας (bubble sort). Σχεδιάστε για το λόγο αυτό στο γραπτό σας έναν πίνακα σαν τον παρακάτω (με όσες γραμμές χρειάζεται) και συμπληρώστε τις μεταβολές στον πίνακα βήμα προς βήμα, συμπληρώνοντας στον μικρό πίνακα δεξιά τα στοιχεία που αλλάζουν θέση σε κάθε μεταβολή.

1	2	3	4	5	6
61	49	23	32	12	35

	
35	12

Επαναληπτικές Εξετάσεις 2006: Θέμα 6β – Λύση

1	2	3	4	5	6		↔	
61	49	23	32	12	35		35	12
61	49	23	12	32	35		32	12
61	49	12	23	32	35		23	12
61	12	49	23	32	35		49	12
12	61	49	23	32	35		61	12
12	61	23	49	32	35		49	23
12	23	61	49	32	35		61	23
12	23	61	32	49	35		49	32
12	23	32	61	49	35		61	32
12	23	32	61	35	49		49	35
12	23	32	35	61	49		61	35
12	23	32	35	49	61		61	49

Τελικές Εξετάσεις 2007: Θέμα 4

α) Δίνονται οι εξής ακέραιοι αριθμοί: 43, 44, 70, 58, 11, 59, 61, 76. Δημιουργήστε το Δυαδικό Δέντρο Αναζήτησης που προκύπτει αν οι αριθμοί εισάγονται με την παραπάνω σειρά (από αριστερά προς τα δεξιά). Στην απάντησή σας δείξτε τη μορφή του δέντρου μετά από κάθε εισαγωγή.

β) Διαγράψτε 2 φορές τη ρίζα του Δυαδικού Δέντρου Αναζήτησης που θα προκύψει. Στην απάντησή σας δείξτε τη μορφή του δέντρου μετά από κάθε διαγραφή.

γ) Για ένα Δυαδικό Δέντρο η προ-διατεταγμένη και η ενδο-διατεταγμένη διαπέραση έχουν τα εξής αποτελέσματα:

ΠΡΟ-ΔΙΑΤΕΤΑΓΜΕΝΗ: Ε, Γ, Β, Ζ, Α, Δ, Η

ΕΝΔΟ-ΔΙΑΤΕΤΑΓΜΕΝΗ: Γ, Ε, Δ, Α, Ζ, Β, Η

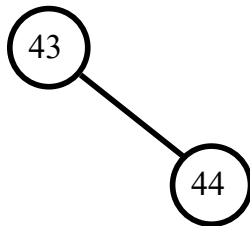
Σχεδιάστε το Δυαδικό Δέντρο.

Τελικές Εξετάσεις 2007: Θέμα 4α – Λύση

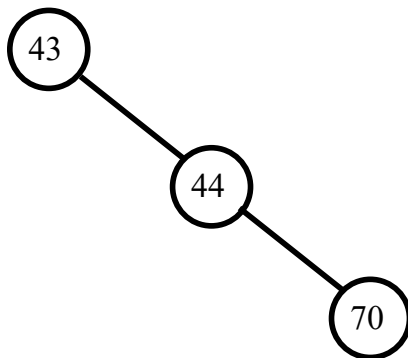
1^{ος} αριθμός



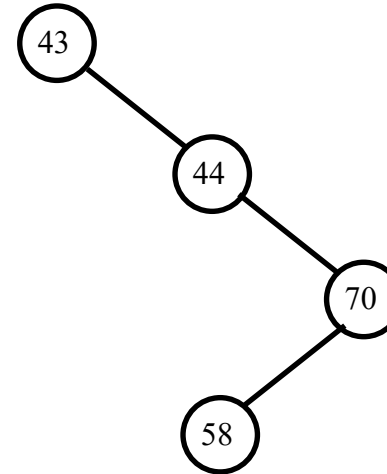
2^{ος} αριθμός



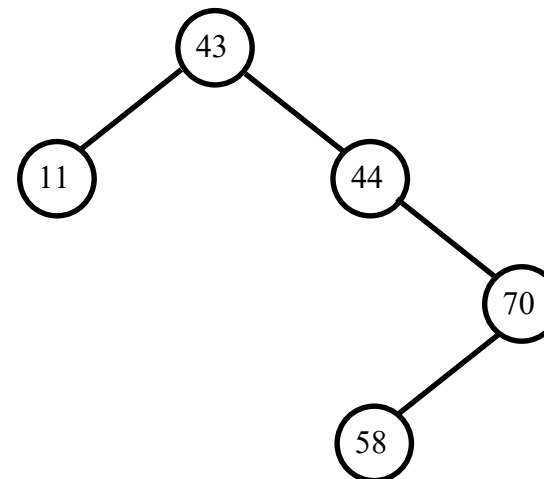
3^{ος} αριθμός



4^{ος} αριθμός

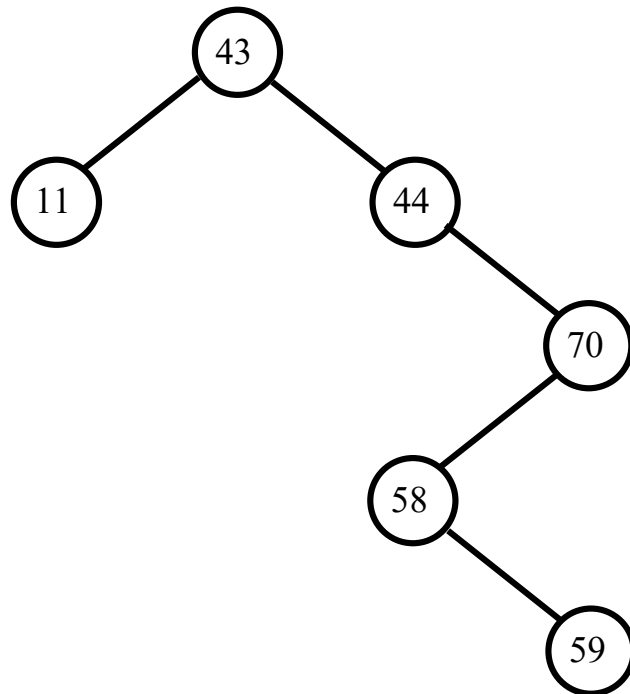


5^{ος} αριθμός

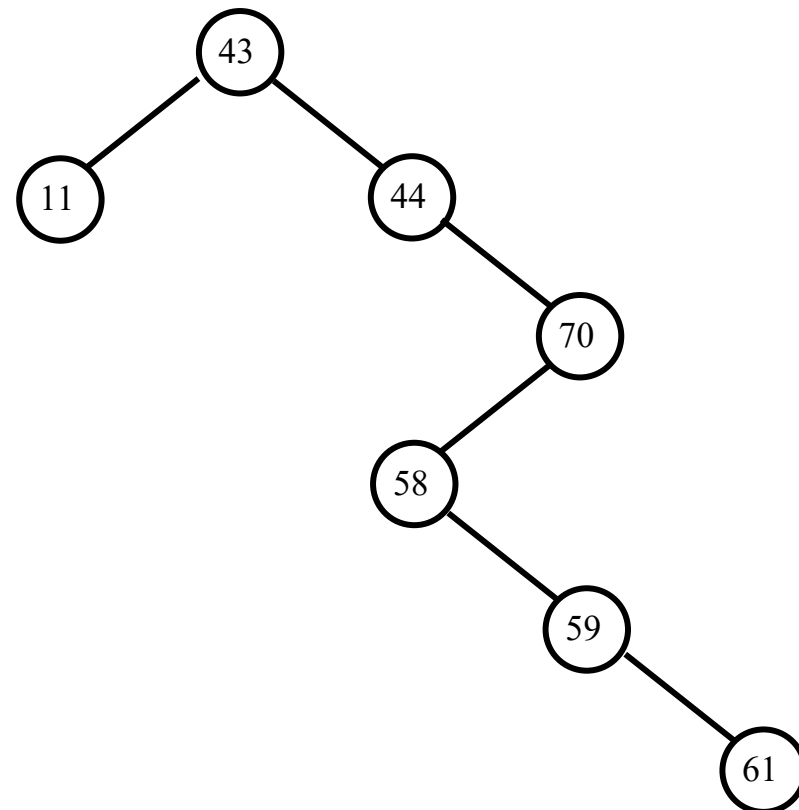


Τελικές Εξετάσεις 2007: Θέμα 4α – Λύση (συν.)

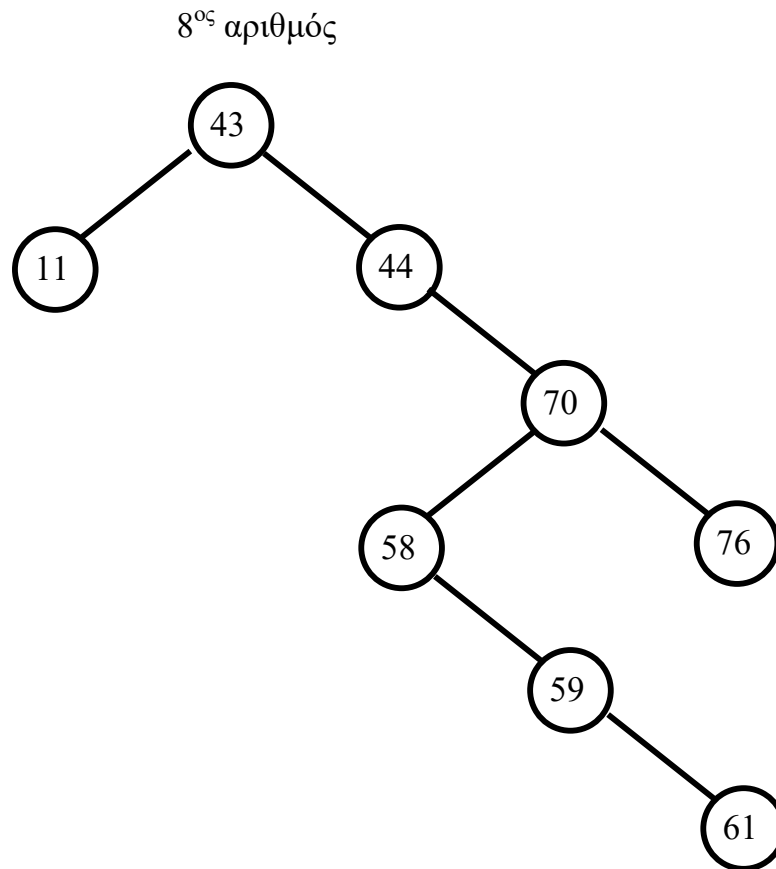
6^{ος} αριθμός



7^{ος} αριθμός

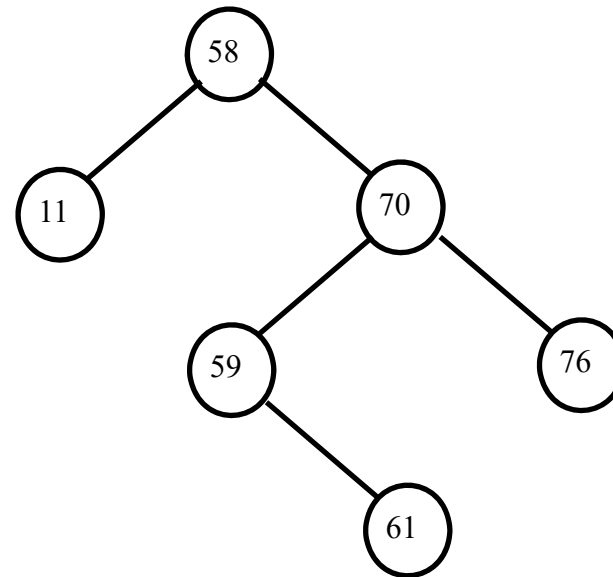
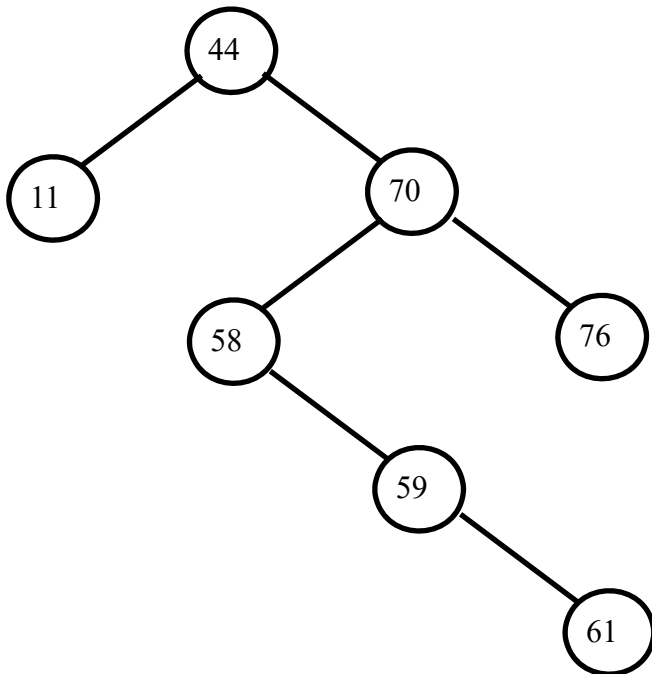


Τελικές Εξετάσεις 2007: Θέμα 4α – Λύση (συν.)



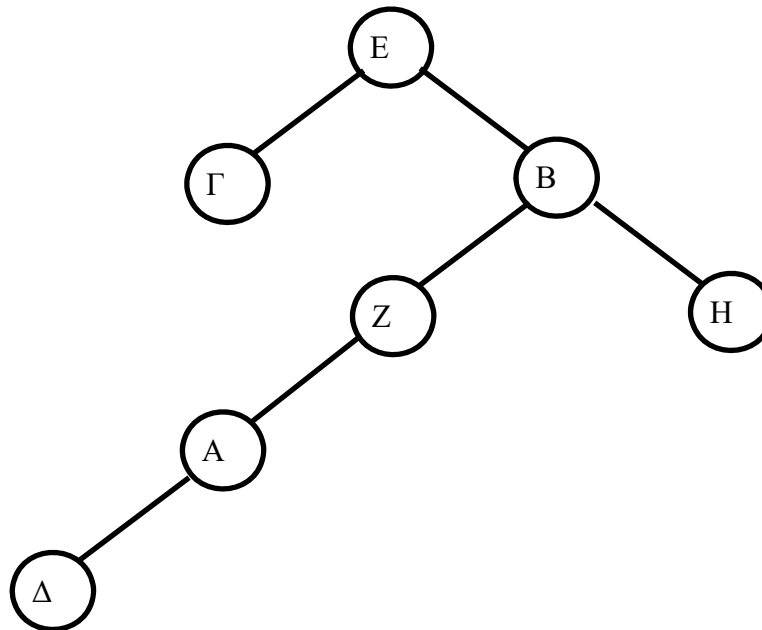
Τελικές Εξετάσεις 2007: Θέμα 4β – Λύση

Διαγραφή ρίζας (αντικατάσταση από τον αμέσως μεγαλύτερο κόμβο)



Τελικές Εξετάσεις 2007: Θέμα 4γ – Λύση

Κατασκευή Δυαδικού Δέντρου βάσει προ-διατεταγμένης και ενδο-διατεταγμένης διαπέρασης:



Τελικές Εξετάσεις 2006: Θέμα 4

Γράψτε σε γλώσσα προγραμματισμού C τη συνάρτηση:

```
float tmean(float a[9]);
```

η οποία υπολογίζει και επιστρέφει τον μέσο όρο των επτά αριθμών του πίνακα `a[]` που προκύπτουν αφού εξαιρεθούν από τους 9 αρχικούς ο πιο μεγάλος και ο πιο μικρός αριθμός.

Τελικές Εξετάσεις 2006: Θέμα 4 – Λύση

```
float tmean(float a[9])
{
    int i;
    float maxx = a[0];
    float minx = a[0];
    float sum = a[0];
    for (i = 1; i < 9; ++i)
    {
        if (a[i] > maxx) maxx = a[i];
        if (a[i] < minx) minx = a[i];
        sum = sum + a[i];
    }
    return((sum - maxx - minx) / 7.0);
}
```

Τελικές Εξετάσεις 2011: Θέμα 3α

Να γραφεί πρόγραμμα σε γλώσσα προγραμματισμού C, στο οποίο θα δίνονται από τον χρήστη τιμές σε δύο ακέραιες μεταβλητές και στη συνέχεια θα καλείται μία συνάρτηση τύπου void με παραμέτρους τις δύο αυτές μεταβλητές, η οποία συνάρτηση θα επιστρέφει στην πρώτη παράμετρο το άθροισμα των τιμών των δύο ακεραίων και στη δεύτερη παράμετρο τη διαφορά των τιμών των δύο ακεραίων.

Τελικές Εξετάσεις 2011: Θέμα 3α – Λύση

```
#include <stdio.h>

void abplus(int *a1, int *b1)
{
    int sum, diff;
    sum = *a1 + *b1;
    diff = *a1 - *b1;
    *a1 = sum;
    *b1 = diff;
}

void main()
{
    int a, b;
    printf("\nΔώσε τιμή για τον πρώτο ακέραιο: ");
    scanf("%d", &a);
    printf("\nΔώσε τιμή για τον δεύτερο ακέραιο: ");
    scanf("%d", &b);
    abplus(&a, &b);
    printf("\nΟι νέες τιμές των a και b είναι: %d και %d", a, b);
}
```

Επαναληπτικές Εξετάσεις 2006: Θέμα 6α

Έστω ότι μια απλά διασυνδεδεμένη λίστα περιέχει ονόματα και βαθμολογίες φοιτητών (οι βαθμολογίες είναι σε κλίμακα 0 έως 10). Η δομή κάθε κόμβου της λίστας δίνεται από τη δήλωση:

```
struct student {  
    char name[30];  
    float bathmos;  
    struct student *next;  
};
```

Γράψτε σε γλώσσα C τη συνάρτηση:

```
int count_pass(struct student *base);
```

η οποία δέχεται ως είσοδο τη διεύθυνση του πρώτου κόμβου της λίστας και επιστρέφει το πλήθος των φοιτητών που έχουν προβιβάσιμη βαθμολογία (δηλαδή βαθμό ≥ 5).

Επαναληπτικές Εξετάσεις 2006: Θέμα 6α – Λύση

```
int count_pass(struct student *base)
{
    struct student *p;
    int pass;
    pass = 0;
    p = base;
    while (p != NULL)
    {
        if (p->bathmos >= 5) pass++;
        p = p->next;
    }
    return (pass) ;
}
```

Τελικές Εξετάσεις 2010: Θέμα 6α

Να γράψετε στη γλώσσα προγραμματισμού C τη συνάρτηση:

```
int count(struct node *list)
```

η οποία δέχεται ως παράμετρο μια απλά συνδεδεμένη λίστα ακεραίων αριθμών και επιστρέφει το πλήθος των στοιχείων της συνδεδεμένης λίστας. Αν η λίστα είναι κενή η συνάρτηση θα επιστρέφει -1.

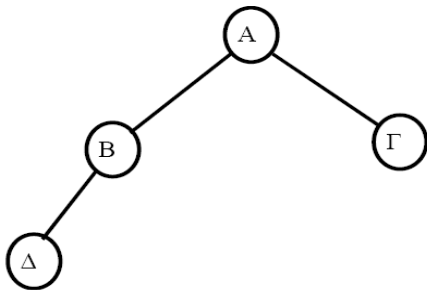
Η δομή των κόμβων της λίστας δίνεται ως:

```
struct node {  
    int key;  
    struct node *next;  
};
```

Τελικές Εξετάσεις 2010: Θέμα 6α – Λύση

```
int count(struct node *list)
{
    struct node *p = list;
    int cnt=1;
    if (p == NULL) return(-1);
    while (p->next != NULL)
    {
        cnt++;
        p = p->next;
    }
    return(cnt);
}
```


Επαναληπτικές Εξετάσεις 2006: Θέμα 1β



1β) Υπάρχει Δυαδικό Δέντρο με τέσσερα (4) στοιχεία, το οποίο να είναι ταυτόχρονα και Δυαδικό Δέντρο Αναζήτησης και Δέντρο - Σωρός; Αιτιολογήστε την απάντησή σας.

Απάντηση: Δεν υπάρχει δυαδικό δένδρο με τέσσερα (4) στοιχεία, το οποίο να είναι ταυτόχρονα και Δυαδικό Δέντρο Αναζήτησης και Δέντρο - Σωρός. Διότι το Δέντρο - Σωρός πρέπει να είναι πλήρες δέντρο, οπότε το δέντρο θα πρέπει να έχει την εξής μορφή:

Όμως, για να είναι Δυαδικό Δέντρο Αναζήτησης πρέπει $A < \Gamma$, ενώ για να είναι Σωρός πρέπει $A > \Gamma$.

ΠΑΡΑΡΤΗΜΑ

Επανάληψη ύλης όλων των τόμων

1^{ος} Τόμος Μελέτης

Σύντομη επισκόπηση

Αριθμητικά Συστήματα

Ο γενικός κανόνας παράστασης σε ένα αριθμητικό σύστημα έχει ως εξής:
Ο αριθμός:

$$\alpha_{n-1}r^{n-1} + \alpha_{n-2}r^{n-2} + \dots + \alpha_1r^1 + \alpha_0r^0 + \alpha_{-1}r^{-1} + \dots + \alpha_{-m}r^{-m}$$

συμβολίζεται ως: $\alpha_{n-1} \alpha_{n-2} \dots \alpha_1 \alpha_0 \alpha_{-1} \dots \alpha_{-m}$

Ως **βάση** ή **ρίζα** ενός αριθμητικού συστήματος ορίζεται το πλήθος των διαφορετικών ψηφίων που χρησιμοποιούνται για την παράσταση των αριθμών

- **Δεκαδικό:** 0 1 2 3 4 5 6 7 8 9
- **Δυαδικό:** 0 1
- **Οκταδικό :** 0 1 2 3 4 5 6 7
- **Δεκαεξαδικό :** 0 1 2 3 4 5 6 7 8 9 A B C D E F

Συστήματα

- Ένας δεκαδικός αριθμός αποτελείται από μία ακολουθία δεκαδικών ψηφίων και ίσως από μία υποδιαστολή. Οποιοσδήποτε αριθμός (ποσότητα) εκφράζεται ως :

$$(x)_b = \sum_{i=-m}^{n-1} a_i b^i$$

όπου: b είναι η βάση του συστήματος ($b \geq 2$) και a_i τα ψηφία του αριθμού αυτού με τιμές 0 έως $b-1$

- Π.χ. $3347.4 = 3 \times 10^3 + 3 \times 10^2 + 4 \times 10^1 + 7 \times 10^0 + 4 \times 10^{-1}$
- Το σύστημα αυτό ονομάζεται δεκαδικό, λόγω του ότι ως εκθετική βάση έχει επιλεγεί το 10.

Συστήματα

Δεκαδικό σύστημα αρίθμησης

Βάση 10, ψηφία 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

Δυαδικό σύστημα αρίθμησης

Βάση 2, ψηφία 0, 1

Οκταδικό σύστημα αρίθμησης

Βάση 8, ψηφία 0, 1, 2, 3, 4, 5, 6, 7

Δεκαεξαδικό σύστημα αρίθμησης

Βάση 16, ψηφία 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

Αντιστοιχίσεις σε συστήματα αρίθμησης

Δεκαδικός	Δυαδικός	Δεκαεξαδικός	Οκταδικός
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	8	10
9	1001	9	11
10	1010	A	12
11	1011	B	13
12	1100	C	14
13	1101	D	15
14	1110	E	16
15	1111	F	17

Μετατροπές (από οποιοδήποτε σύστημα στο δεκαδικό σύστημα)

Ο τύπος της γενικής αναπαράστασης ενός αριθμού σε οποιοδήποτε σύστημα μας δίνει και την αξία του στο δεκαδικό σύστημα αρίθμησης (αφού γίνουν οι πράξεις).

Πολλαπλασιάζω κάθε ψηφίο του αριθμού που δίνεται (σε κάποιο σύστημα) με τη βάση του ίδιου του συστήματος υψωμένη σε δύναμη που προσδιορίζεται από τη θέση του ψηφίου στον αριθμό.

Στο **ακέραιο μέρος** ξεκινώ από το τελευταίο ψηφίο του (μονάδες) το οποίο βρίσκεται στη θέση 0 και συνεχίζω προς την αρχή του (κινούμαι αριστερά) αυξάνοντας συνεχώς τον αριθμό της θέσης κατά ένα.

Στο **κλασματικό μέρος** κινούμαι δεξιά θεωρώντας ότι το πρώτο κλασματικό ψηφίο βρίσκεται στη θέση -1 και σε κάθε βήμα μειώνω κατά ένα τον αριθμό της θέσης (-2, -3 κ.ο.κ.).

$$\begin{aligned} A^0 &= 1, A^1 = A, \\ A^{-1} &= 1/A, A^{-2} = 1/A^2 \end{aligned}$$

Παραδείγματα

$$(1673,42)_{10} = 1 \times 10^3 + 6 \times 10^2 + 7 \times 10^1 + 3 \times 10^0 + 4 \times 10^{-1} + 2 \times 10^{-2}$$

$$(100110)_2 = 1 \times 2^5 + 0 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = (38)_{10}$$

$$(372)_8 = 3 \times 8^2 + 7 \times 8^1 + 2 \times 8^0 = (250)_{10}$$

$$(A34F,4)_{16} = A \times 16^3 + 3 \times 16^2 + 4 \times 16^1 + F \times 16^0 + 4 \times 16^{-1} = (41807,25)_{10}$$

Μετατροπές (του δεκαδικού συστήματος σε οποιοδήποτε σύστημα)

Έστω αριθμός με **ακέραιο και κλασματικό μέρος** (γγγγ,xxx).

Χωρίζω το ακέραιο από το κλασματικό μέρος γγγγ και 0,xxx

Το **ακέραιο μέρος** του αριθμού διαιρείται (ακέραια διαίρεση) συνεχώς με τη βάση του συστήματος στο οποίο θέλουμε να το μετατρέψουμε.

Η διαίρεση σταματάει όταν το πηλίκο γίνει 0.

Ο ζητούμενος αριθμός προκύπτει παίρνοντας ανάποδα όλα τα υπόλοιπα των διαιρέσεων.

Στη συνέχεια το **κλασματικό μέρος** πολλαπλασιάζεται συνεχώς με τη βάση του συστήματος στο οποίο θέλουμε να το μετατρέψουμε. Οποιοδήποτε ακέραιο μέρος παραχθεί κατά τον πολλαπλασιασμό τότε αυτό διαχωρίζεται πάλι από το κλασματικό μέρος (με βάση την παραπάνω λογική) και η διαδικασία επαναλαμβάνεται. Ο πολλαπλασιασμός σταματάει όταν φτάσουμε το πλήθος των δεκαδικών ψηφίων που θέλουμε.

Ο κλασματικός αριθμός στο νέο σύστημα σχηματίζεται παίρνοντας όλα τα ακέραια μέρη όλων των πολλαπλασιασμών που κάναμε με φορά από τον πρώτο προς τον τελευταίο.

Μετατροπές (δεκαδικό μέρος)

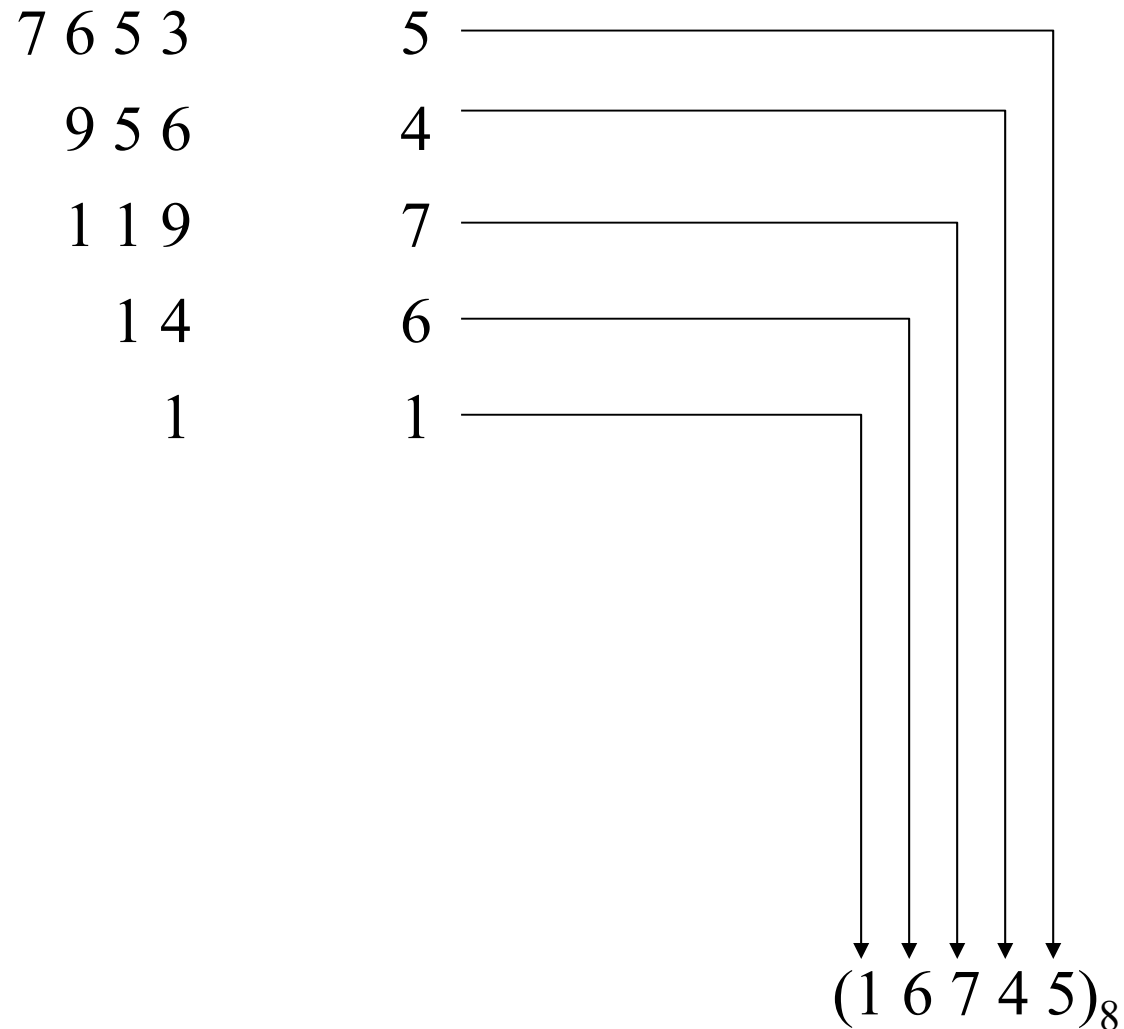
Από δεκαδικό σε δεκαεξαδικό του 0,23

(Διαδοχικοί πολλαπλασιασμοί με το 16):

0,23	16	3,68	3	
0,68	16	10,88	A	
0,88	16	14,08	E	
0,08	16	1,28	1	
0,28	16	4,48	4	
				= 0, 3 A E 1 4

Μετατροπές (ακέραιο μέρος)

Από δεκαδικό σε οκταδικό του 7653 (Διαδοχικές διαιρέσεις με το 8):



Μετατροπές - Παράγωγα συστήματα

Συστήματα που η βάση του ενός είναι η ύψωση σε δύναμη της βάσης ενός άλλου ονομάζονται **παράγωγα συστήματα αρίθμησης**.

Παράδειγμα το εννεαδικό σύστημα αρίθμησης είναι παράγωγο του τριαδικού γιατί $3^2=9$.

Παράδειγμα το δεκαεξαδικό σύστημα αρίθμησης είναι παράγωγο του δυαδικού γιατί $2^4=16$.

Κάθε ψηφίο του **δεκαεξαδικού αριθμού** μετατρέπεται αυτόνομα στο **δυαδικό σύστημα** (χρησιμοποιώντας 4 δυαδικά ψηφία) και έτσι προκύπτει ο δυαδικός αριθμός και αντίστροφα.

Αντίστοιχα, κάθε ψηφίο του **οκταδικού αριθμού** μετατρέπεται αυτόνομα στο **δυαδικό σύστημα** (χρησιμοποιώντας 3 δυαδικά) και έτσι προκύπτει ο δυαδικός αριθμός και αντίστροφα.

Μετατροπές - Παράγωγα συστήματα

Δυαδικό **Οκταδικό**

0 0 0 \longleftrightarrow 0

0 0 1 \longleftrightarrow 1

0 1 0 \longleftrightarrow 2

0 1 1 \longleftrightarrow 3

Δυαδικό **Οκταδικό**

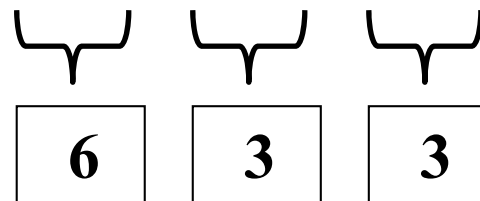
1 0 0 \longleftrightarrow 4

1 0 1 \longleftrightarrow 5

1 1 0 \longleftrightarrow 6

1 1 1 \longleftrightarrow 7

$$(633)_8 = (1\ 1\ 0\ 0\ 1\ 1\ 0\ 1\ 1)_2$$



Μετατροπές - Παράγωγα συστήματα

Δυαδικό Δεκαεξαδικό

0 0 0 0 ↔ 0

0 0 0 1 ↔ 1

0 0 1 0 ↔ 2

0 0 1 1 ↔ 3

0 1 0 0 ↔ 4

0 1 0 1 ↔ 5

0 1 1 0 ↔ 6

0 1 1 1 ↔ 7

Δυαδικό Δεκαεξαδικό

1 0 0 0 ↔ 8

1 0 0 1 ↔ 9

1 0 1 0 ↔ A

1 0 1 1 ↔ B

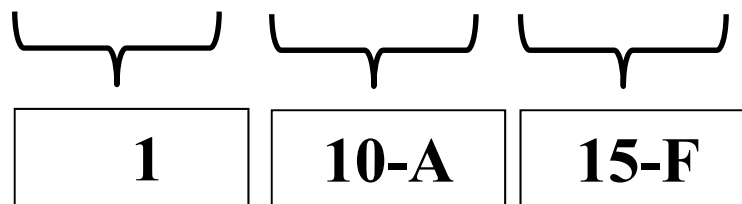
1 1 0 0 ↔ C

1 1 0 1 ↔ D

1 1 1 0 ↔ E

1 1 1 1 ↔ F

$$(1AF)_{16} = (0\ 0\ 0\ 1\ 1\ 0\ 1\ 0\ 1\ 1\ 1\ 1)_2$$



Μετατροπές - Παράγωγα συστήματα

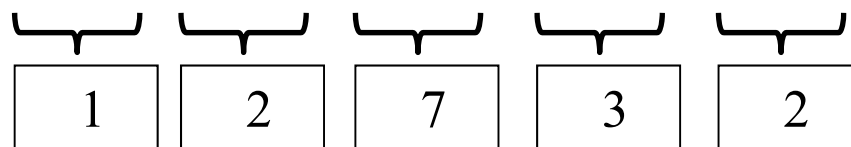
Από δυαδικό σε οκταδικό

$$(1\ 0\ 1\ 0\ 1\ 1\ 1, 0\ 1\ 1\ 0\ 1)_2$$

Χωρίζουμε σε τριάδες από υποδιαστολή προς τα αριστερά (ακέραιο μέρος) και από υποδιαστολή προς τα δεξιά (δεκαδικό μέρος)

(κατά περίπτωση προσθέτουμε μηδενικά)

$$(0\ 0\ 1\ 0\ 1\ 0\ 1\ 1\ 1, 0\ 1\ 1\ 0\ 1\ 0)_2$$



Μετατροπές - Παράγωγα συστήματα

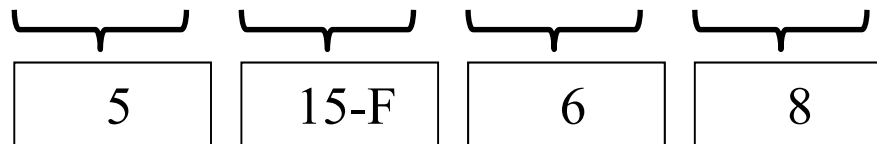
Από δυαδικό σε δεκαεξαδικό

$$(1\ 0\ 1\ 1\ 1\ 1\ 1,0\ 1\ 1\ 0\ 1)_2$$

Χωρίζουμε σε τετράδες από υποδιαστολή προς τα αριστερά (ακέραιο μέρος) και από υποδιαστολή προς τα δεξιά (δεκαδικό μέρος)

(κατά περίπτωση προσθέτουμε μηδενικά)

$$(0\ 1\ 0\ 1\ 1\ 1\ 1\ 1,0\ 1\ 1\ 0\ 1\ 0\ 0\ 0)_2$$



Πράξεις (πρόσθεση)

Η πρόσθεση ανεξάρτητα από το αριθμητικό σύστημα γίνεται θεωρώντας το κάθε ζεύγος (μονοψήφιων) αριθμών που προστίθεται ως δεκαδικό

αλλά

το αποτέλεσμα της πράξης γράφεται πάντα στο σύστημα στο οποίο γίνεται η πράξη (με τη λογική του αποτελέσματος και του κρατουμένου που ισχύει και στο δεκαδικό σύστημα).

Το **ίδιο ισχύει** και για τις υπόλοιπες πράξεις που όμως δεν μας ενδιαφέρουν άμεσα μιας και οι αριθμητικές πράξεις στον υπολογιστή υλοποιούνται με την πράξη της πρόσθεσης και γι' αυτό επικεντρωνόμαστε σε αυτή.

Πρόσθεση στο δυαδικό σύστημα

Πρόσθεση δυαδικών:

A	+	B	Αποτέλεσμα	Κρατούμενο
0		0	0	0
0		1	1	0
1		0	1	0
1		1	0	1

Παράδειγμα:

1ος Προσθετέος	0	0	1	1	1	$(7)_{10}$
2ος Προσθετέος :	0	1	0	1	0	$(10)_{10}$
Άθροισμα :	1	0	0	0	1	$(17)_{10}$
Κρατούμενο :	0	1	1	1	0	

Αφαίρεση (μέθοδος συμπληρώματος)

Η μέθοδος του συμπληρώματος χρησιμοποιείται για να εκφράσουμε **αρνητικούς αριθμούς** οπότε μπορούμε να μετατρέψουμε τις αφαιρέσεις που πρέπει να κάνει ο υπολογιστής σε προσθέσεις (πρόσθεση του αντιστρόφου - συμπληρώματος).

Εδώ πρέπει να θυμάστε ότι ο αριθμός των ψηφίων με τα οποία θα αναπαραστήσουμε τους αριθμούς είναι σημαντικός και παίζει ρόλο στη μέθοδο.

A. Το συμπλήρωμα ενός **αριθμού ως προς τη (βάση - 1) ενός συστήματος** υπολογίζεται αν τον αφαιρέσουμε από τον μεγαλύτερο αριθμό του συστήματος με το ίδιο πλήθος ψηφίων.

B. Το συμπλήρωμα ενός **αριθμού ως προς τη (βάση) ενός συστήματος** υπολογίζεται αν στο συμπλήρωμα του αριθμού ως προς τη (βάση - 1) προσθέσουμε τη μονάδα (+1).

Παράσταση συμπληρώματος ως προς ένα

Το **συμπλήρωμα ως προς ένα** μιας δυαδικής ακολουθίας λαμβάνεται αλλάζοντας όλα τα bit από μηδέν σε ένα και αντιστρόφως

1 0 0 1 0 0 1 1

0 1 1 0 1 1 0 0

Το πιο σημαντικό bit (MSB) χρησιμοποιείται για τον προσδιορισμό του προσήμου.

Στη γενική περίπτωση αναπαριστώνται οι $[-(2^{n-1}-1), +(2^{n-1}-1)]$ ($[-32767, 32767]$)

Θετικοί + 0

Αρνητικοί

0000	0	1000	-7
0001	+1	1001	-6
0010	+2	1010	-5
0011	+3	1011	-4
0100	+4	1100	-3
0101	+5	1101	-2
0110	+6	1110	-1
0111	+7	1111	-0

Αριθμητική Συμπληρώματος ως προς δύο

Το πιο σημαντικό bit (MSB) χρησιμοποιείται επίσης για τον προσδιορισμό του προσήμου

Όταν **MSB = 0**, τότε ο αριθμός είναι **θετικός ή μηδέν** και το μέτρο δίνεται από τα υπόλοιπα (n-1) ψηφία του αριθμού

Όταν **MSB = 1**, ο αριθμός είναι **αρνητικός** και το μέτρο του αριθμού δίνεται από το συμπλήρωμα ως προς 2 του συνόλου των ψηφίων του αριθμού

Το συμπλήρωμα ως προς 2 βρίσκεται αλλάζοντας όλα τα bit (0 σε 1 και 1 σε 0) και προσθέτοντας 1 στο αποτέλεσμα.

Αρνητικοί Αριθμοί

για $n=6$ bits, $X_{10} = -17 \Rightarrow$

$$X_2 = \boxed{101111}$$

Εύρεση συμπληρώματος ως προς 2 του αριθμού 17 :

$$X = 17 \rightarrow 010001$$

$$101110 \quad (\text{Αντιστροφή όλων των bit})$$

$$101111 \quad (\text{Πρόσθεση του 1})$$

Προσοχή: Πάντα πρέπει να θεωρήσουμε συγκεκριμένο αριθμό bits

Αριθμητική Συμπληρώματος του 2

Το αλγεβρικό άθροισμα δύο αριθμών στην παράσταση συμπληρώματος του 2 προκύπτει ως το δυαδικό άθροισμα των δύο αριθμών, αγνοώντας το τυχόν κρατούμενο:

+12	001100	+12	001100
+17	010001	- 17	101111
-----	-----	-----	-----
29	011101	- 5	111011
<hr/>			
- 12	110100	- 12	110100
+17	010001	- 17	101111
-----	-----	-----	-----
+5	000101	- 29	100011
Αγνοείται το κρατούμενο 1		Αγνοείται το κρατούμενο 1	

Συμπλήρωμα - Συχνά λάθη

Το συμπλήρωμα ως προς 2 του αποτελέσματος πρέπει να το υπολογίσουμε; ΌΧΙ μόνο για δική μας χρήση επαλήθευσης.

Συμπλήρωμα ως προς 2 με δεκαδικά ψηφία.

Αντιστροφή όλων των bits και πρόσθεση του 1 στη δεξιότερη θέση (σαν να μην υπάρχει η υποδιαστολή).

Βρείτε το συμπλήρωμα ως προς 2 του 010111,111

010111,111 \rightarrow 101000,000

και προσθέτουμε 1

101000 000

1 +

101000,001 (αποτέλεσμα)

Βασική λογική σχεδίαση και άλγεβρα Boole

Μαθηματική Λογική – Μαθηματική Συμπερασματολογία

Μαθηματικές προτάσεις – Λογικοί τελεστές

Οι βασικοί λογικοί (ή Boolean) τελεστές που χρησιμοποιούνται είναι οι: **NOT, AND, OR**.

Πύλες = Βασικά ψηφιακά λογικά στοιχεία που αντιστοιχούν στις βασικές λογικές συναρτήσεις.

AND: Λογικός πολ/σμός (σύζευξη) Boolean Έκφραση: $A \bullet B$

OR: Λογική πρόσθεση (διάζευξη) Boolean Έκφραση: $A + B$

NOT: Λογική αντιστροφή Boolean Έκφραση: \bar{A} ή $\sim A$ ή A'

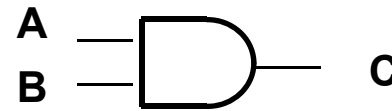
Λογικός πολλαπλασιασμός (AND)

Έχει αποτέλεσμα = 1, μόνο εάν και οι δύο είσοδοι είναι 1.

TRUE=1, FALSE=0

$$C = A \bullet B \text{ ή } C = A \text{ AND } B$$

Λογική Πύλη



Πίνακας αληθείας

A	B	C
0	0	0
0	1	0
1	0	0
1	1	1

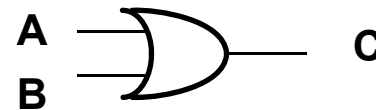
Λογική πρόσθεση (OR)

Έχει αποτέλεσμα = 1, εάν κάποια από τις εισόδους ισούται με 1.

TRUE=1, FALSE=0

$$C = A + B \text{ ή } C = A \text{ OR } B$$

Λογική Πύλη



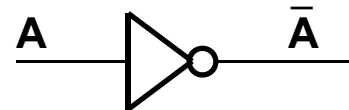
Πίνακας αληθείας

A	B	C
0	0	0
0	1	1
1	0	1
1	1	1

Λογική αντιστροφή (NOT)

Έξοδος αντίστροφη της εισόδου

Λογική Πύλη

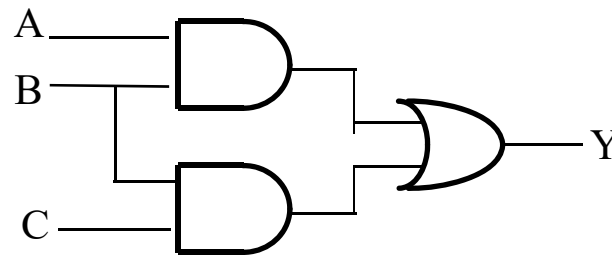


Πίνακας αληθείας

A	$\sim A$
0	1
1	0

Συνδυάζοντας Λογικές Πύλες

- Η έξοδος ενός συνδυαστικού κυκλώματος εξαρτάται μόνο από τις εκάστοτε εισόδους.

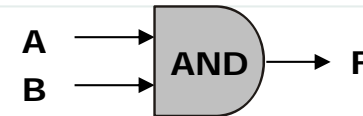


AND - OR Λογικό Κύκλωμα

Σύνοψη: Λογικά κυκλώματα - Πύλες

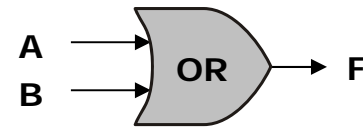
- AND: $F = A \cdot B = A \text{ AND } B$

Η έξοδος F είναι 1 όταν **όλες** οι είσοδοι (A, B) είναι 1



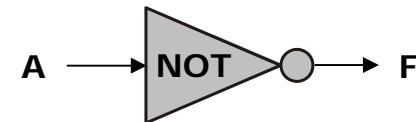
- OR: $F = A + B = A \text{ OR } B$

Η έξοδος F είναι 1 όταν **τουλάχιστον** μία είσοδος είναι 1



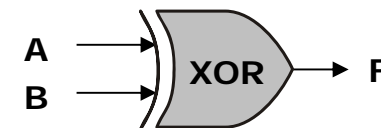
- NOT: $F = A'$

Η έξοδος είναι το αντίθετο της εισόδου



- XOR: $F = A' \cdot B + A \cdot B' =$
 $(\text{NOT}(A) \text{ AND } B) \text{ OR } (A \text{ AND } \text{NOT}(B))$

Η έξοδος F είναι 1 όταν **μία και μόνο μία** είσοδος είναι 1



- NAND: $F = (A \cdot B)' = \text{NOT}(A \text{ AND } B)$

- NOR: $F = (A + B)' = \text{NOT}(A \text{ OR } B)$

- XNOR: $F = A \cdot B + A' \cdot B' = (A \text{ AND } B) \text{ OR } (\text{NOT}(A) \text{ AND } (\text{NOT}(B)))$

Πύλες: Άσκηση

Έστω η λογική συνάρτηση:

$$F = A \cdot B' + A' \cdot B \cdot C$$

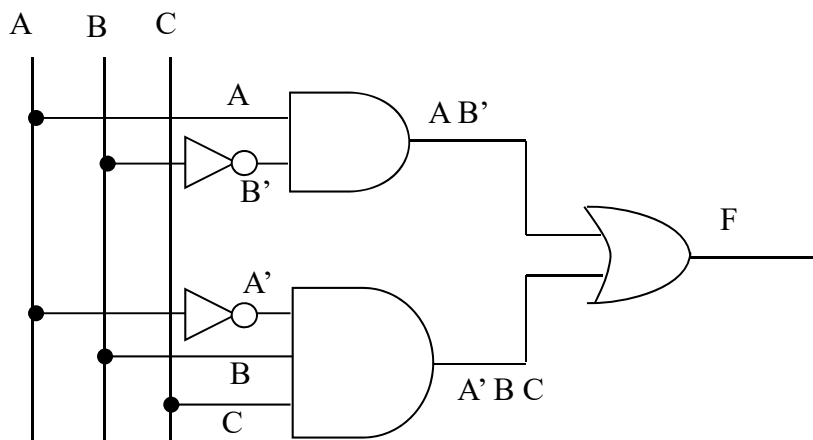
- 1) Γράψτε τη συνάρτηση, χρησιμοποιώντας τα ονόματα των πυλών (AND, OR, NOT)
- 2) Σχεδιάστε το αντίστοιχο λογικό κύκλωμα
- 3) Σχηματίστε τον πίνακα αληθείας της F

Πύλες: Άσκηση

$$F = A \cdot B' + A' \cdot B \cdot C$$

$$F = (A \text{ AND } (\text{NOT } B)) \text{ OR } ((\text{NOT } A) \text{ AND } B \text{ AND } C)$$

Λογικό κύκλωμα



Πίνακας αληθείας

A	B	C	A'	B'	K = AB'	L = A'BC	F = K+L
0	0	0	1	1	0	0	0
0	0	1	1	1	0	0	0
0	1	0	1	0	0	0	0
0	1	1	1	0	0	1	1
1	0	0	0	1	1	0	1
1	0	1	0	1	1	0	1
1	1	0	0	0	0	0	0
1	1	1	0	0	0	0	0

Διαγράμματα Ροής Προγράμματος

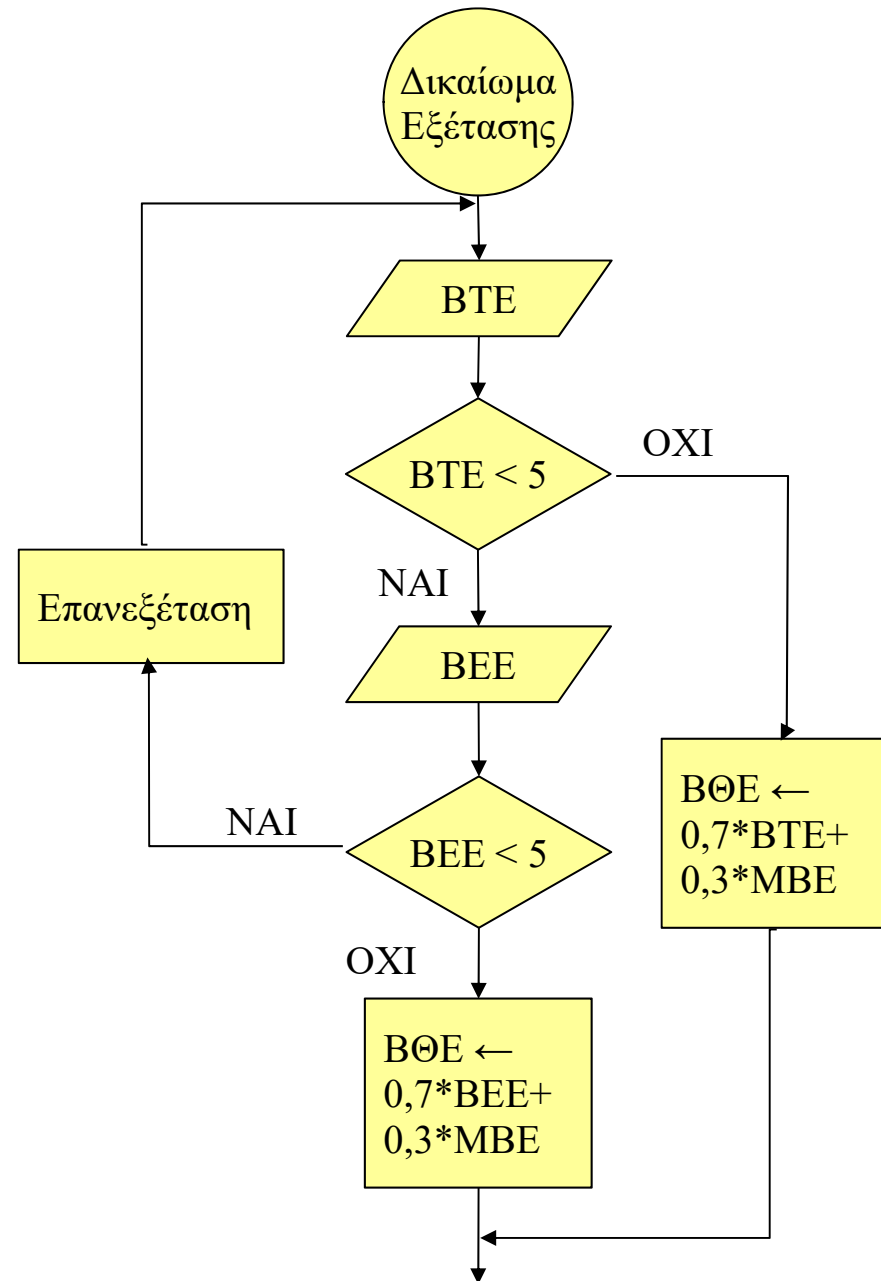
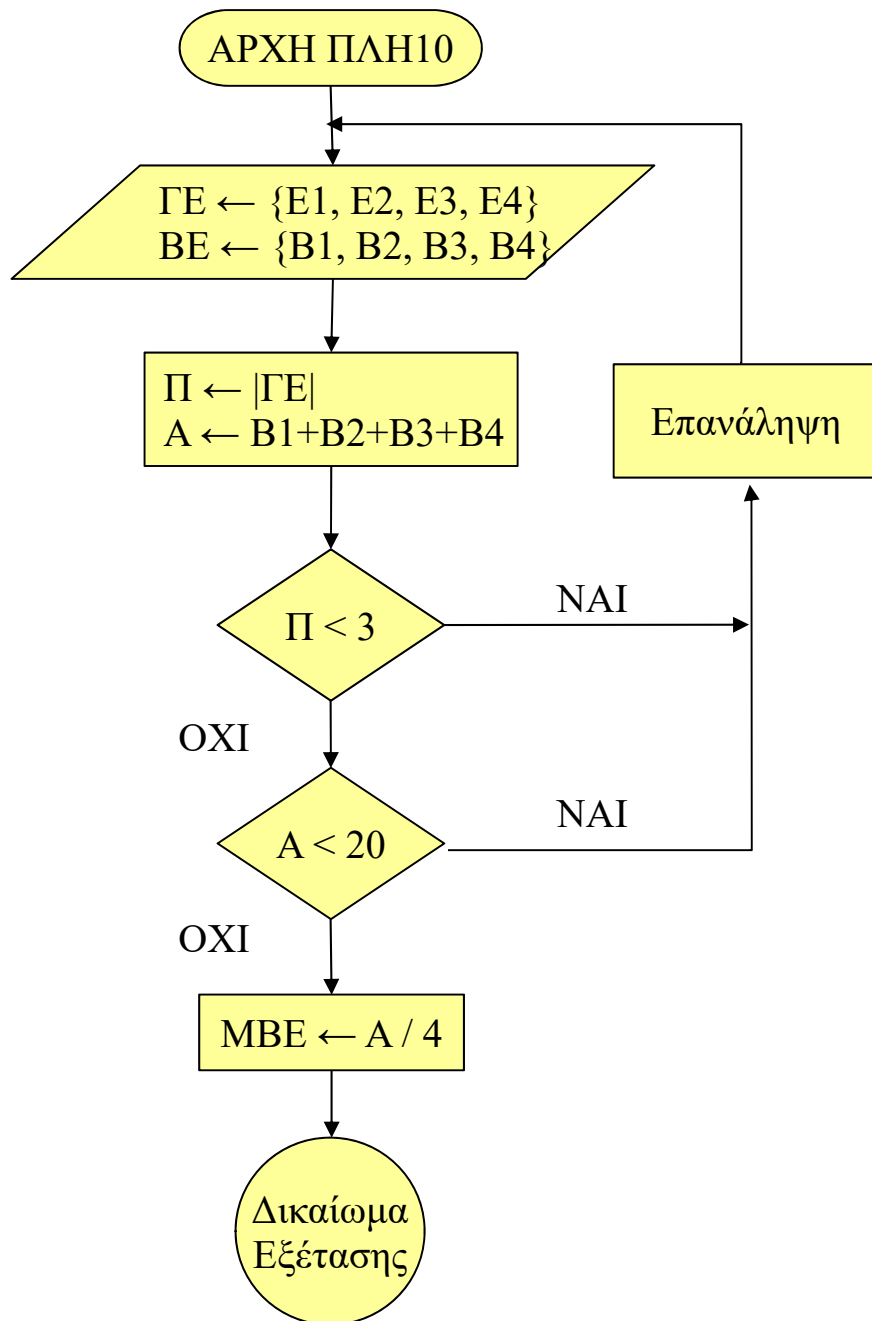
Αρχή - Τέλος: Έλλειψη

Είσοδος - Έξοδος: Παραλληλόγραμμο
Διάβασμα τιμής από πληκτρολόγιο, Εκτύπωση

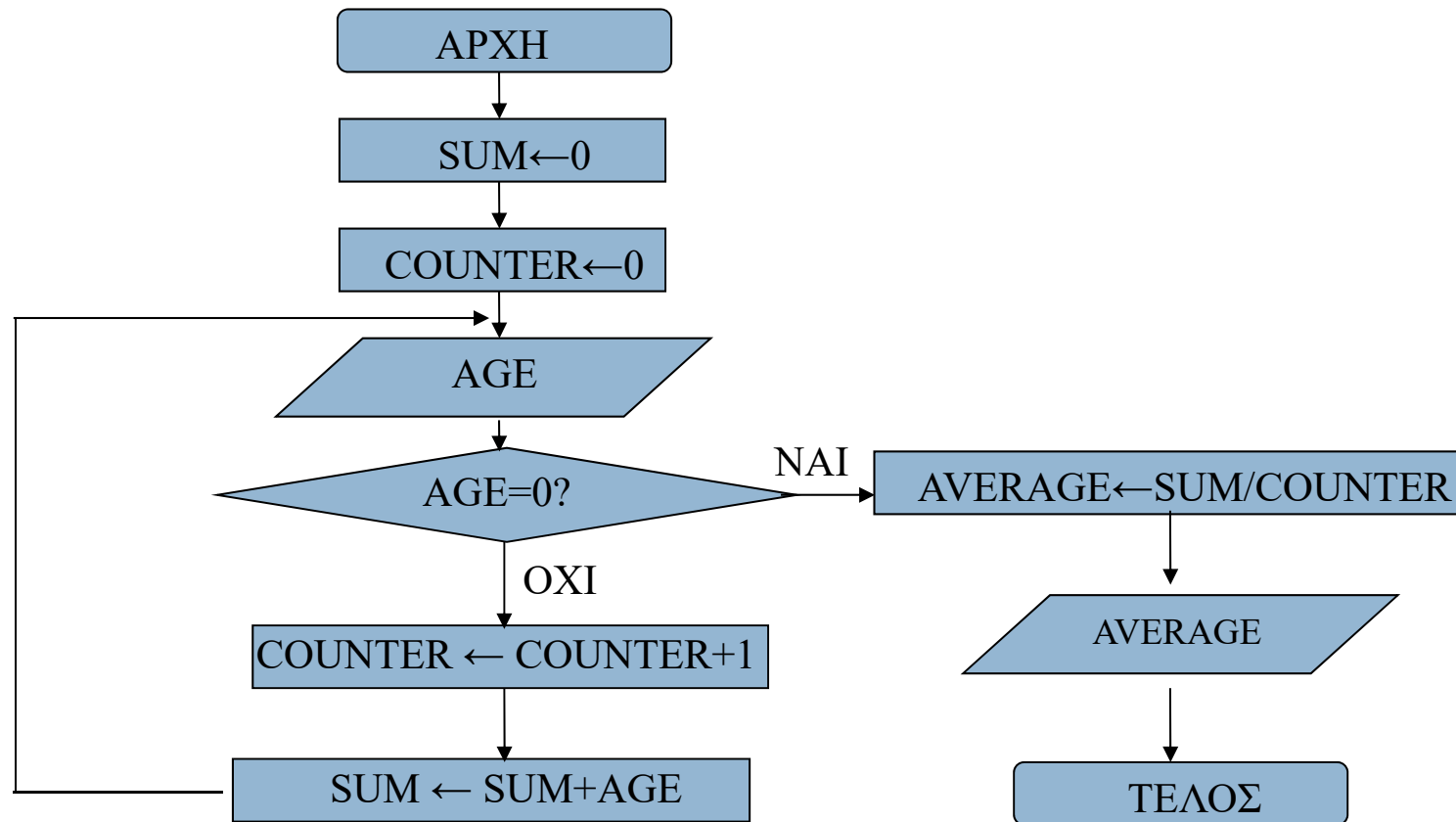
Ανάθεση τιμής: Ορθογώνιο

Απόφαση: Ρόμβος
Λογική συνθήκη με δύο εξόδους: Αληθής - Ψευδής
(Ναι - Όχι)

Σύνδεση με άλλο τμήμα του ΔΡΠ: Κύκλος
Σε πολύπλοκα ΔΡΠ (π.χ. όταν δεν χωράνε σε μία σελίδα)



Ποια η λειτουργία του;



2^{ος} Τόμος Μελέτης

Σύντομη επισκόπηση

Αμυντικός Προγραμματισμός

Όταν ζητάτε είσοδο από το χρήστη κάποιου δεδομένου, συνήθως αυτό υπόκειται σε κάποιους περιορισμούς.

Είναι πολύ σημαντικό ο χρήστης να δώσει το απαιτούμενο δεδομένο σωστά και σε περίπτωση που αυτό δεν γίνει (θα πρέπει να γίνει έλεγχος της εισόδου) θα πρέπει να του ζητηθεί ξανά, μέχρι να δοθεί ξανά

Τρόπος υλοποίησης (ενδεικνυόμενη εντολή)

ΕΠΑΝΑΛΑΒΕ

Πάρε δεδομένο/α από το χρήστη

ΜΕΧΡΙ (να πληρούνται οι δοσμένες συνθήκες)

Αμυντικός Προγραμματισμός – Παράδειγμα

Χρήση αμυντικού προγραμματισμού για την εισαγωγή θετικού διψήφιου ακέραιου αριθμού ώστε να υπολογιστεί η τετραγωνική του ρίζα

ΕΠΑΝΑΛΑΒΕ

ΤΥΠΩΣΕ (“Δώστε διψήφιο ακέραιο για υπολογισμό της τετραγωνικής του ρίζας”);

ΔΙΑΒΑΣΕ (N)

ΜΕΧΡΙ ((N <= 99) **AND** (N >= 10));

Χρήση αμυντικού προγραμματισμού για την εισαγωγή αριθμού που αντιστοιχεί σε κάποιο μήνα

ΕΠΑΝΑΛΑΒΕ

ΤΥΠΩΣΕ (“Δώστε μήνα”);

ΔΙΑΒΑΣΕ (MONTH)

ΜΕΧΡΙ ((MONTH >= 1) **AND** (MONTH <=12));

Πέρασμα παραμέτρων σε διαδικασίες

Κατά την κλήση μίας διαδικασίας, οι παράμετροι ενδέχεται να:

- Εισέρχονται από το καλούν (υπο)πρόγραμμα στη διαδικασία (αλλαγή τιμής στη διαδικασία δεν αλλάζει την τιμή στο καλούν υποπρόγραμμα).
 - Πέρασμα με τιμή - pass by value
- Εισέρχονται από το καλούν (υπό)πρόγραμμα στη διαδικασία και αλλαγή τιμής στη διαδικασία αλλάζει την τιμή στο καλούν υποπρόγραμμα.
 - Πέρασμα με αναφορά - pass by reference
 - χρησιμοποιείται το % για τη δήλωση τέτοιων παραμέτρων τόσο στην κλήση όσο και στις παραμέτρους του υποπρογράμματος που καλείται

Πρόβλημα: Διάβασε έναν αριθμό από το πληκτρολόγιο και εκτύπωσε τον διπλάσιό του.

Περιορισμός: Ο διπλασιασμός του αριθμού να γίνει σε διαδικασία

Αλγόριθμος Διπλασιασμός-Αριθμού

ΔΙΑΔΙΚΑΣΙΑ ΔΙΠΛΑΣΙΑΣΕ(%A)

ΔΙΕΠΑΦΗ

ΕΙΣΟΔΟΣ A : INTEGER;

ΕΞΟΔΟΣ A : INTEGER;

ΑΡΧΗ

A := 2*A

ΤΕΛΟΣ-ΔΙΑΔΙΚΑΣΙΑΣ

ΑΛΓΟΡΙΘΜΟΣ ΔΙΠΛΑΣΙΑΣΜΟΣ

ΔΕΔΟΜΕΝΑ

X : INTEGER;

ΑΡΧΗ

ΔΙΑΒΑΣΕ(X);

ΥΠΟΛΟΓΙΣΕ ΔΙΠΛΑΣΙΑΣΕ(%X);

ΤΥΠΩΣΕ("Το διπλάσιο του αριθμού που δώσατε είναι ", X)

ΤΕΛΟΣ

Αναδρομή – ορισμός

Μια συνάρτηση είναι **αναδρομική** όταν **καλεί τον εαυτό της**.

Για κάποια συνθήκη λαμβάνει συγκεκριμένη τιμή χωρίς άλλη αναδρομή (τερματισμός αναδρομής).

Συχνά μπορούμε να κάνουμε τον ίδιο υπολογισμό με χρήση μιας επαναληπτικής διαδικασίας.

Η γενική μορφή της εφαρμογής αναδρομής είναι:

Υποπρόγραμμα $\Pi(N)$

Αν (συνθήκη τερματισμού) **τότε**

συγκεκριμένη τιμή

αλλιώς

εντολές που περιέχουν κλήση του **$\Pi(N-1)$** (ή **$\Pi(N-2)$** , κλπ.).

Αναδρομή - Υπολογισμός Παραγοντικού

Το παραγοντικό ενός ακέραιου αριθμού N , συμβολίζεται $N!$ και ορίζεται ως εξής: $N! = 1*2*3*...*N$.

Δηλαδή:

$$1! = 1$$

$$2! = 1*2 = 2$$

$$3! = 1*2*3 = 6, \text{ κλπ.}$$

Όμως μπορεί να γραφεί:

$$1! = 1 \quad 2! = 2*1! \quad 3! = 3*2! \quad 4! = 4*3!, \text{ κλπ.}$$

Και στη γενική μορφή:

$$N! = N*(N-1)! \quad \text{που είναι αναδρομικός τύπος}$$

Αναδρομικός υπολογισμός του παραγοντικού

Η επόμενη συνάρτηση υλοποιεί τον αναδρομικό υπολογισμό του παραγοντικού ($N! = N * (N-1)!$) για $N \geq 1$

ΣΥΝΑΡΤΗΣΗ Factorial(N): **INTEGER**

ΔΙΕΠΑΦΗ

ΕΙΣΟΔΟΣ

N: **INTEGER**;

ΕΞΟΔΟΣ

Factorial: **INTEGER**;

ΑΡΧΗ

ΕΑΝ (N=1) **ΤΟΤΕ**

Factorial:=1 /* τερματισμός αναδρομής */

ΑΛΛΙΩΣ

Factorial:= N*Factorial(N-1)

ΕΑΝ-ΤΕΛΟΣ

ΤΕΛΟΣ-ΣΥΝΑΡΤΗΣΗΣ

Αναδρομικός υπολογισμός του 5!

ΣΥΝΑΡΤΗΣΗ Factorial (N): **INTEGER**

ΔΙΕΠΑΦΗ

ΕΙΣΟΔΟΣ

N: **INTEGER**;

ΕΞΟΔΟΣ

Factorial: **INTEGER**;

ΑΡΧΗ

ΕΑΝ (N=1) **ΤΟΤΕ**

Factorial:=1

ΑΛΛΙΩΣ

Factorial:=N*Factorial(N-1)

ΕΑΝ-ΤΕΛΟΣ

ΤΕΛΟΣ-ΣΥΝΑΡΤΗΣΗΣ

N=5, Factorial=5 * Factorial(4)

Αναδρομικός υπολογισμός του 5!

ΣΥΝΑΡΤΗΣΗ Factorial (N): **INTEGER**

ΔΙΕΠΑΦΗ

ΕΙΣΟΔΟΣ

N: **INTEGER**;

ΕΞΟΔΟΣ

Factorial: **INTEGER**;

ΑΡΧΗ

ΕΑΝ (N=1) **ΤΟΤΕ**

Factorial:=1

ΑΛΛΙΩΣ

Factorial:=N*Factorial(N-1)

ΕΑΝ-ΤΕΛΟΣ

ΤΕΛΟΣ-ΣΥΝΑΡΤΗΣΗΣ

N=5, Factorial=5 * Factorial(4)

N=4, Factorial=4 * Factorial(3)

Αναδρομικός υπολογισμός του 5!

ΣΥΝΑΡΤΗΣΗ Factorial (N): **INTEGER**

ΔΙΕΠΑΦΗ

ΕΙΣΟΔΟΣ

N: **INTEGER**;

ΕΞΟΔΟΣ

Factorial: **INTEGER**;

ΑΡΧΗ

ΕΑΝ (N=1) **ΤΟΤΕ**

Factorial:=1

ΑΛΛΙΩΣ

Factorial:=N*Factorial(N-1)

ΕΑΝ-ΤΕΛΟΣ

ΤΕΛΟΣ-ΣΥΝΑΡΤΗΣΗΣ

N=5, Factorial=5 * Factorial(4)

N=4, Factorial=4 * Factorial(3)

N=3, Factorial=3 * Factorial(2)

Αναδρομικός υπολογισμός του 5!

ΣΥΝΑΡΤΗΣΗ Factorial (N): **INTEGER**

ΔΙΕΠΑΦΗ

ΕΙΣΟΔΟΣ

N: **INTEGER**;

ΕΞΟΔΟΣ

Factorial: **INTEGER**;

ΑΡΧΗ

ΕΑΝ (N=1) **ΤΟΤΕ**

Factorial:=1

ΑΛΛΙΩΣ

Factorial:=N*Factorial(N-1)

ΕΑΝ-ΤΕΛΟΣ

ΤΕΛΟΣ-ΣΥΝΑΡΤΗΣΗΣ

N=5, Factorial=5 * Factorial(4)

N=4, Factorial=4 * Factorial(3)

N=3, Factorial=3 * Factorial(2)

N=2, Factorial=2 * Factorial(1)

Αναδρομικός υπολογισμός του 5!

ΣΥΝΑΡΤΗΣΗ Factorial (N): **INTEGER**

ΔΙΕΠΑΦΗ

ΕΙΣΟΔΟΣ

N: **INTEGER**;

ΕΞΟΔΟΣ

Factorial: **INTEGER**;

ΑΡΧΗ

ΕΑΝ (N=1) **ΤΟΤΕ**

Factorial:=1

ΑΛΛΙΩΣ

Factorial:=N*Factorial(N-1)

ΕΑΝ-ΤΕΛΟΣ

ΤΕΛΟΣ-ΣΥΝΑΡΤΗΣΗΣ

N=5, Factorial=5 * Factorial(4)

N=4, Factorial=4 * Factorial(3)

N=3, Factorial=3 * Factorial(2)

N=2, Factorial=2 * Factorial(1)

N=1, Factorial=1

Αναδρομικός υπολογισμός του 5!

ΣΥΝΑΡΤΗΣΗ Factorial (N): **INTEGER**

ΔΙΕΠΑΦΗ

ΕΙΣΟΔΟΣ

N: **INTEGER**;

ΕΞΟΔΟΣ

Factorial: **INTEGER**;

ΑΡΧΗ

ΕΑΝ (N=1) **ΤΟΤΕ**

Factorial:=1

ΑΛΛΙΩΣ

Factorial:=N*Factorial(N-1)

ΕΑΝ-ΤΕΛΟΣ

ΤΕΛΟΣ-ΣΥΝΑΡΤΗΣΗΣ

N=5, Factorial=5 * Factorial(4)

N=4, Factorial=4 * Factorial(3)

N=3, Factorial=3 * Factorial(2)

N=2, Factorial=2 * 1 = 2

Αναδρομικός υπολογισμός του 5!

ΣΥΝΑΡΤΗΣΗ Factorial (N): **INTEGER**

ΔΙΕΠΑΦΗ

ΕΙΣΟΔΟΣ

N: **INTEGER**;

ΕΞΟΔΟΣ

Factorial: **INTEGER**;

ΑΡΧΗ

ΕΑΝ (N=1) **ΤΟΤΕ**

Factorial:=1

ΑΛΛΙΩΣ

Factorial:=N*Factorial(N-1)

ΕΑΝ-ΤΕΛΟΣ

ΤΕΛΟΣ-ΣΥΝΑΡΤΗΣΗΣ

N=5, Factorial=5 * Factorial(4)

N=4, Factorial=4 * Factorial(3)

N=3, Factorial=3 * 2 = 6

Αναδρομικός υπολογισμός του 5!

ΣΥΝΑΡΤΗΣΗ Factorial (N): **INTEGER**

ΔΙΕΠΑΦΗ

ΕΙΣΟΔΟΣ

N: **INTEGER**;

ΕΞΟΔΟΣ

Factorial: **INTEGER**;

ΑΡΧΗ

ΕΑΝ (N=1) **ΤΟΤΕ**

Factorial:=1

ΑΛΛΙΩΣ

Factorial:=N*Factorial(N-1)

ΕΑΝ-ΤΕΛΟΣ

ΤΕΛΟΣ-ΣΥΝΑΡΤΗΣΗΣ

N=5, Factorial=5 * Factorial(4)

N=4, Factorial=4 * 3 = 24

Αναδρομικός υπολογισμός του 5!

ΣΥΝΑΡΤΗΣΗ Factorial (N): **INTEGER**

ΔΙΕΠΑΦΗ

ΕΙΣΟΔΟΣ

N: **INTEGER**;

ΕΞΟΔΟΣ

Factorial: **INTEGER**;

ΑΡΧΗ

ΕΑΝ (N=1) **ΤΟΤΕ**

Factorial:=1

ΑΛΛΙΩΣ

Factorial:=N*Factorial(N-1)

ΕΑΝ-ΤΕΛΟΣ

ΤΕΛΟΣ-ΣΥΝΑΡΤΗΣΗΣ

N=5, Factorial=5 * 24 = 120

Ταξινόμηση

Είναι η διαδικασία τοποθέτησης αριθμητικών ή αλφαριθμητικών δεδομένων σε διάταξη (αύξουσα ή φθίνουσα).

Υπάρχουν πολλοί σχετικοί αλγόριθμοι:

- **Bubble sort - αλγόριθμος φυσαλίδας**

σύγκριση ανά δύο των στοιχείων και αντιμετάθεσή τους αν χρειαστεί

- **Ταξινόμηση με παρεμβολή**

κάθε στοιχείο τοποθετείται με παρεμβολή στη θέση που αρμόζει

Ταξινόμηση Φυσαλίδας

ΑΛΓΟΡΙΘΜΟΣ ΤΑΞΙΝΟΜΗΣΗ_ΦΥΣΑΛΙΔΑΣ

ΣΤΑΘΕΡΕΣ N=100;

ΔΕΔΟΜΕΝΑ

I, K, TEMP: INTEGER;

P: ARRAY [1..N] OF INTEGER;

ΑΡΧΗ

ΓΙΑ I:=1 ΕΩΣ N-1 ΕΠΑΝΑΛΑΒΕ

ΓΙΑ K := N ΕΩΣ I+1 ΕΠΑΝΑΛΑΒΕ

ΕΑΝ (P[K] < P[K-1]) ΤΟΤΕ

TEMP := P[K-1];

P[K-1] := P[K];

P[K] := TEMP

ΕΑΝ-ΤΕΛΟΣ

ΓΙΑ-ΤΕΛΟΣ

ΓΙΑ-ΤΕΛΟΣ

ΤΕΛΟΣ

Αναζήτηση

Διαδικασία ανεύρεσης ενός στοιχείου σε μια συλλογή (δομή) δεδομένων (π.χ. σε πίνακα)

Υπάρχουν πολλοί αλγόριθμοι αναζήτησης:

- **Σειριακή αναζήτηση**

Διαπέραση των στοιχείων το ένα μετά το άλλο και έλεγχος

- **Δυαδική αναζήτηση**

Ο πίνακας χωρίζεται διαδοχικά σε υποπίνακες του μισού μεγέθους και η διαδικασία συνεχίζεται μόνο στο ένα μισό

Προσοχή: απαιτείται ο πίνακας να είναι ταξινομημένος.

Ο αλγόριθμος επιστρέφει τη θέση του αναζητούμενου στοιχείου.

Προσοχή: αν το αναζητούμενο στοιχείο δεν υπάρχει, θα πρέπει να εμφανιστεί αντίστοιχο μήνυμα ή / και να επιστρέφεται ειδική τιμή (π.χ. η τιμή -1).

Δυαδική Αναζήτηση

Χρησιμοποιούνται δείκτες LEFT, RIGHT, MID

1^η περίπτωση: αναζήτηση του 16 (αριστερός υποπίνακας)

2^η περίπτωση: αναζήτηση του 60 (που δεν υπάρχει, οπότε κάποια στιγμή θα είναι $LEFT > RIGHT$)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
4	8	15	16	22	25	29	32	44	46	49	56	59	69	78

Αλγόριθμος Δυαδικής Αναζήτησης

ΔΙΑΔΙΚΑΣΙΑ BINSEARCH (KEY, AR, LEFT, RIGHT, %INDEX)

ΔΙΕΠΑΦΗ

ΕΙΣΟΔΟΣ

KEY, LEFT, RIGHT: INTEGER;
AR: ARRAY[1..N] OF INTEGER;

ΕΞΟΔΟΣ

INDEX: INTEGER;

ΔΕΔΟΜΕΝΑ

MID: INTEGER;

ΑΡΧΗ

MID := (LEFT + RIGHT) DIV 2;

ΕΑΝ (LEFT > RIGHT) **ΤΟΤΕ**

INDEX := -1

ΑΛΛΙΩΣ ΕΑΝ (KEY = AR[MID]) **ΤΟΤΕ**

INDEX := MID

ΑΛΛΙΩΣ ΕΑΝ (KEY < AR[MID]) **ΤΟΤΕ**

ΥΠΟΛΟΓΙΣΕ BINSEARCH (KEY, AR, LEFT, MID-1, %INDEX)

ΑΛΛΙΩΣ { KEY > AR[MID] }

ΥΠΟΛΟΓΙΣΕ BINSEARCH (KEY, AR, MID+1, RIGHT, %INDEX)

ΕΑΝ-ΤΕΛΟΣ

ΕΑΝ-ΤΕΛΟΣ

ΕΑΝ-ΤΕΛΟΣ

ΤΕΛΟΣ-ΔΙΑΔΙΚΑΣΙΑΣ

Πρόκειται για αναδρομικό αλγόριθμο που καλείται με την εντολή:
ΥΠΟΛΟΓΙΣΕ BINSEARCH (key, ΠΙΝΑΚΑΣ, 1, N, %pos)

Δείκτες

Δείκτης: μία μεταβλητή που δείχνει (αναφέρεται) σε μία θέση μνήμης και μας επιτρέπει με τον τρόπο αυτό να έχουμε έμμεση προσπέλαση στα περιεχόμενά της.

Παράδειγμα δήλωσης δείκτη:

N : **INTEGER**;

X : **POINTER(INTEGER)**;

Μεταβλητή	Διεύθυνση μνήμης	Τιμή
N	12364	57
X	22344	12364

Για την έμμεση προσπέλαση χρησιμοποιείται ο τελεστής ^

X^:=N;

Κατά σύμβαση, όταν θέλουμε ένας δείκτης να μην δείχνει πουθενά, εκχωρούμε σε αυτόν την τιμή **NIL**.

Ένας μη αρχικοποιημένος δείκτης μπορεί να περιέχει οποιαδήποτε τιμή, άρα να δείχνει σε οποιαδήποτε θέση μνήμης

Διασυνδεδεμένες λίστες

Λίστα: ένα σύνολο στοιχείων, που λέγονται **κόμβοι**. Γενικά, κάθε κόμβος περιέχει υποχρεωτικά ένα **δείκτη** (που λαμβάνει ως τιμή τη διεύθυνση αποθήκευσης του επόμενου κόμβου ή **NIL**) και **μεταβλητές** όπου αποθηκεύονται δεδομένα.

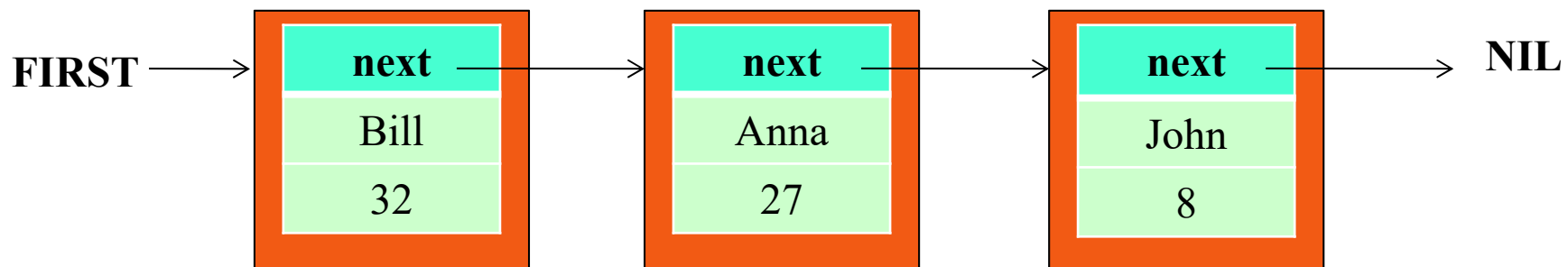
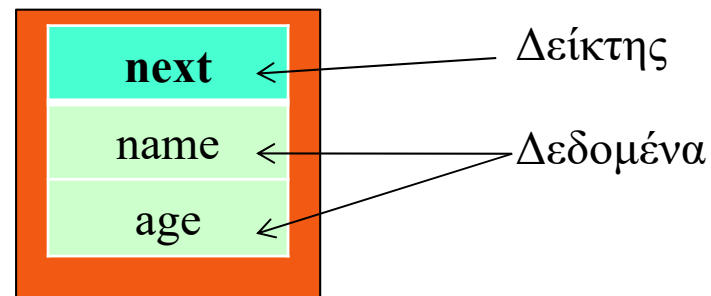
Για παράδειγμα, ένας κόμβος που περιλαμβάνει το όνομα και την ηλικία ενός ατόμου, μπορεί να δηλωθεί και να σχηματιστεί ως εξής:

ΚΟΜΒΟΣ :

next : **POINTER**[ΚΟΜΒΟΣ];

name : **CHAR**;

age : **INTEGER**;



Παράδειγμα λίστας 3 κόμβων

Μετατροπή ψευδοκώδικα σε Διάγραμμα Ροής Προγράμματος (ΔΡΠ)

- Κάποιες απλές τεχνικές για τη μετατροπή ενός αλγορίθμου από τη μορφή του ψευδοκώδικα στη μορφή του Διαγράμματος Ροής Προγράμματος.
- Ανατρέξτε στις διαφάνειες της 2ης ΟΣΣ για να δείτε το αντίστοιχο παράδειγμα αναλυτικά.

3ος Τόμος Μελέτης (4^{ος} Τόμος)

Σύντομη επισκόπηση

Αμυντικός προγραμματισμός

Παράδειγμα εισαγωγής θετικού ακεραίου:

```
do
{
    printf("Dose thetiko akeraio:\n");
    scanf("%d",&a);
    if (a<=0)
        printf("Parakalw eisagete thetiko arithmo\n");
}
while (a<=0);
```

ή εναλλακτικά

```
printf("Dose thetiko akeraio:\n");
scanf("%d",&a);
while (a<=0){
    printf("Parakalw eisagete thetiko arithmo\n");
    scanf("%d",&a);
}
```

Παράδειγμα - Συγχώνευση πινάκων

```
#include <stdio.h>
#define N 5
#define M 4

main()
{
    /* Βοηθητικές μεταβλητές */
    int i, j, k, c[9];
    /* Ορισμός πίνακα a */
    int a[N]={1, 4, 6, 33, 34};
    /* Ορισμός πίνακα b */
    int b[M]={2, 3, 7, 8};
    i = j = k = 0;
    while (i<N && j<M)
    {
        if(a[i]<b[j])
            c[k++]=a[i++];
        else
            c[k++]=b[j++];
    }
```

```
    if(i<N)
    {
        int t;
        for (t=i; t<N; t++)
            c[k++]=a[t++];
    }
    else
    {
        int t;
        for (t=j; t<M; t++)
            c[k++]=b[t++];
    }
    printf("Merged Array C:\n");
    /* Εκτύπωση συγχωνευμένου πίνακα */
    for (k=0;k<(N+M);k++)
        printf("\n %d ",c[k]);
}
```


Μεταβίβαση παραμέτρων

- **Κλήση με τιμή (call by value)**

- Κάθε τυπική παράμετρος παίρνει ένα αντίγραφο της τιμής της πραγματικής παραμέτρου.
- Η τιμή της πραγματικής παραμέτρου δεν επηρεάζεται.

- **Κλήση με αναφορά (call by reference)**

- Κάθε τυπική παράμετρος είναι δείκτης στη διεύθυνση της πραγματικής παραμέτρου.
- Η τιμή της πραγματικής παραμέτρου αλλάζει με βάση τις αλλαγές μέσα στο υποπρόγραμμα.

Παράδειγμα

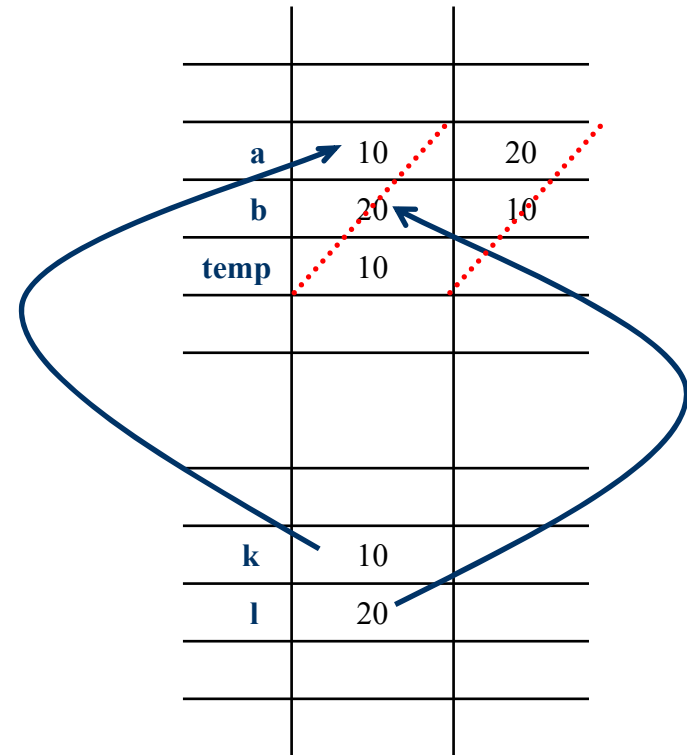
Κλήση με τιμή

```
void swap(int a, int b)
{
    int temp;
    temp = a;
    a = b;
    b = temp;
}

main() {
    k = 10;
    l = 20;
    swap(k,l)
    printf("%d,%d\n", k, l);
}
```

Αποτέλεσμα

10,20



Παράδειγμα

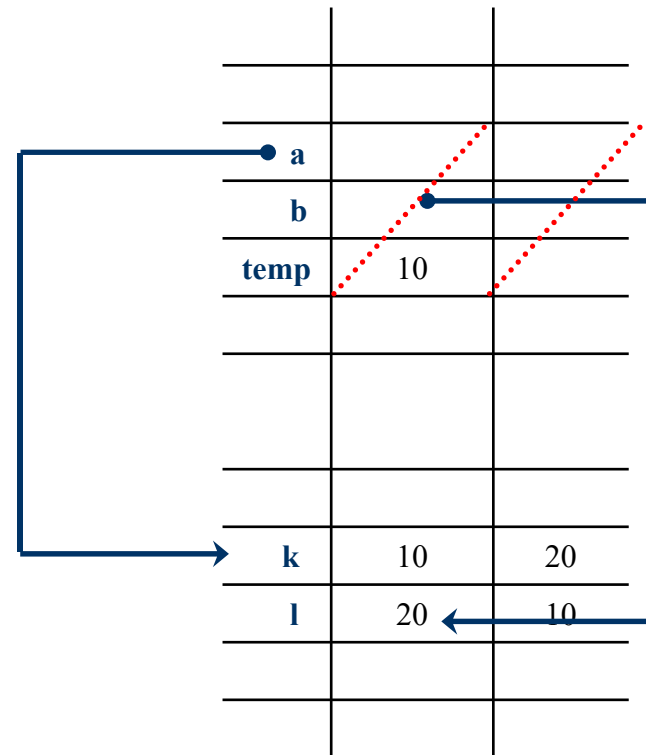
Κλήση με αναφορά

```
void swap(int *a, int *b)
{
    int temp;
    temp = *a;
    *a = *b;
    *b = temp;
}

main() {
    k = 10;
    l = 20;
    swap(&k, &l)
    printf("%d,%d\n", k, l);
}
```

Αποτέλεσμα

20,10



Παράδειγμα - Απόλυτη τιμή αριθμού

```
#include <stdio.h>
```

```
void absolute(int *a) /* Η παράμετρος ορίζεται ως δείκτης */  
{  
    if (*a < 0){  
        *a = -(*a);  
    }  
}
```

```
main()  
{  
    int b=-5;  
    printf("Value of b before invoking absolute is: %d\n", b);  
    absolute(&b); /* κλήση με αναφορά */  
    printf("Value of b after invoking absolute is:%d\n", b);  
}
```

Παράδειγμα αναδρομικής συνάρτησης – Υπολογισμός ακέραιας δύναμης

```
#include <stdio.h>
int power(int x, int y);
main()
{
    int x=4, y=3;
    printf("%d^%d equals to %d\n", x, y, power(x, y));
}
int power(int x, int y) /*Υπολογισμός δύναμης για ακέραιο θετικό εκθέτη*/
{
    if (y==1)
        return x; /*τερματισμός αναδρομής*/
    else
        return x*power(x, y-1); /*αναδρομή*/
}
```

Μεταβίβαση πινάκων σε συναρτήσεις

Πίνακες μιας διάστασης

```
#include <stdio.h>
```

```
double getAverage(int array[], int size);
```

```
/* Ισοδύναμα double getAverage(int *array, int size); */
```

```
main (){
```

```
    /* Ορισμός πίνακα */
```

```
    int balance[5] = {30, 15, 5, 0, 6};
```

```
    double avg;
```

```
    /* Μεταβιβάζουμε τον πίνακα στη συνάρτηση ως δείκτη */
```

```
    avg = getAverage(balance,5);
```

```
    printf("Average value is: %f ", avg);
```

```
}
```

```
double getAverage(int array[], int size){
```

```
    int i;
```

```
    double sum;
```

```
    for (i = 0; i < size; i++)
```

```
        sum += array[i];
```

```
    return sum/size;
```

```
}
```

Μεταβίβαση πινάκων σε συναρτήσεις

Πίνακες δύο διαστάσεων

```
#include <stdio.h>
```

```
double getAverage(int array[][5], int size);
```

```
/*      Η δεύτερη διάσταση του πίνακα δίνεται ρητά */
```

```
main (){
```

```
    int balance[2][5]={30, 15, 5, 0, 6},{2, 5, 6, 6, 10}}; /*Ορισμός πίνακα*/
```

```
    double avg;
```

```
    /* Μεταβιβάζουμε τον πίνακα στη συνάρτηση ως δείκτη, καθώς και τη 2η διάσταση */
```

```
    avg = getAverage(balance,2);
```

```
    printf("Average value is: %f ", avg);
```

```
}
```

```
double getAverage(int array[][5], int size){
```

```
    int i, j;
```

```
    double sum=0.0;
```

```
    for (i=0; i<size; i++)
```

```
        for (j=0; j<5; j++)
```

```
            sum += array[i][j];
```

```
    return sum/(size*5);
```

```
}
```

4^{ος} Τόμος Μελέτης (3^{ος} Τόμος)

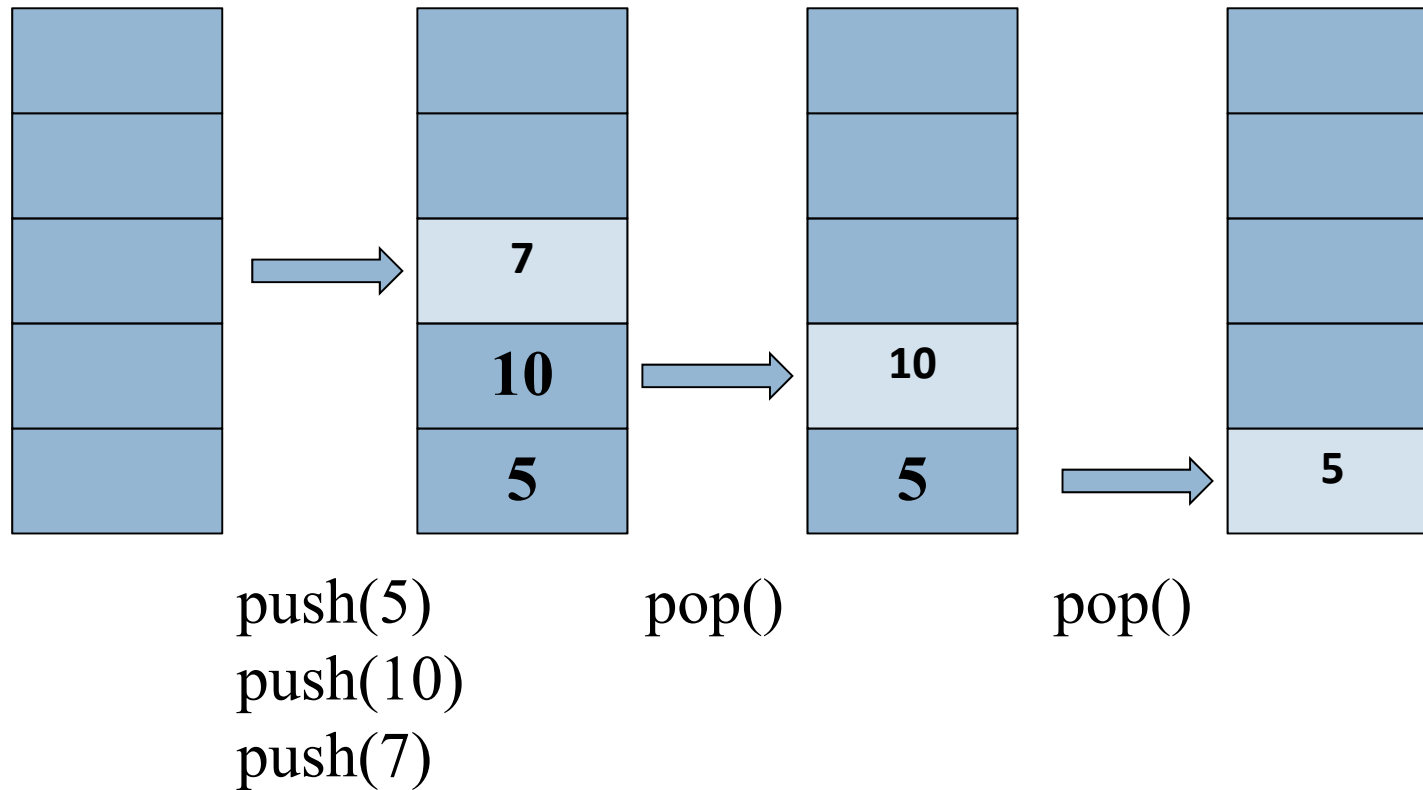
Σύντομη επισκόπηση

Στοιίβα

- Λογική **LIFO** (Last-In First-Out).
- Υποστηρίζονται δύο βασικές λειτουργίες:
 - εισαγωγή (**push**)
 - εξαγωγή (**pop**).
- Υλοποιείται συνήθως με πίνακα (όπου η κεφαλή T αυξομειώνεται κατά 1 σε κάθε εισαγωγή ή διαγραφή), αλλά και με συνδεδεμένη λίστα.
- Χρησιμοποιείται όταν θέλουμε να επεξεργαστούμε κάποια στοιχεία με αντίστροφη σειρά από αυτή που τα εισάγουμε (π.χ. στη μετατροπή δεκαδικού σε δυαδικό, όπου κάνουμε διαιρέσεις και παίρνουμε τα υπόλοιπα από το τέλος προς την αρχή).

Στοίβα

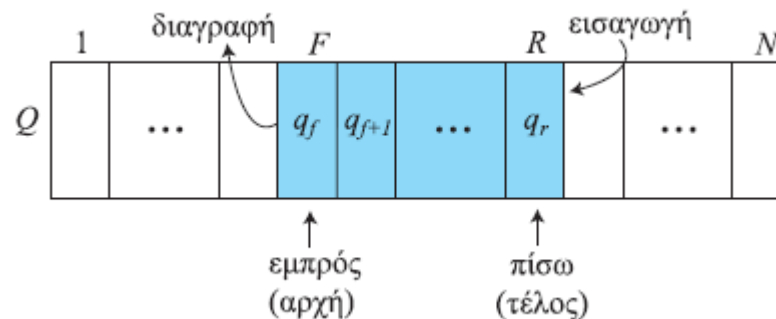
Παράδειγμα



Ουρά

- Λογική **FIFO** (First-In First-Out)
- Υποστηρίζονται δύο βασικές λειτουργίες:
 - εισαγωγή στο τέλος (**insert**)
 - διαγραφή από την αρχή (**delete**).
- Υλοποιείται συνήθως με πίνακα, όπου υπάρχει μετρητής - δείκτης της θέσης εξαγωγής (first) και μετρητής - δείκτης της θέσης εισαγωγής (last), αλλά και με συνδεδεμένη λίστα σε περίπτωση που δε θέλουμε να περιορίσουμε το πλήθος των στοιχείων που θα εισαχθούν στη δομή.

■ Μια ουρά είναι μια λίστα, στην οποία μπορούν να γίνονται εισαγωγές στοιχείων μόνο στο ένα άκρο της (πίσω ή τέλος) και διαγραφές στοιχείων μόνο από το άλλο άκρο της (εμπρός ή αρχή).



Δένδρα

Αποτελούνται από κόμβους και ακμές που τους συνδέουν.

Κάθε κόμβος έχει παιδιά (πλην των φύλλων) και έναν μόνο γονέα (πλην της ρίζας).

Ορολογία

Διαδρομή (path) από ένα κόμβο N_x σε ένα άλλο κόμβο N_y είναι μια ακολουθία κόμβων όπου πρώτος είναι ο N_x , τελευταίος ο N_y και κάθε άλλος κόμβος είναι παιδί του προηγούμενου.

Μήκος (length) μιας διαδρομής είναι ο αριθμός των ακμών που περιλαμβάνονται σ' αυτήν.

Επίπεδο (level) ή βάθος (depth) ενός κόμβου είναι το μήκος της μοναδικής διαδρομής από τη ρίζα στον κόμβο αυτό.

Ύψος (height) ενός δένδρου είναι το μέγιστο βάθος των (τερματικών) κόμβων του.

Βαθμός (degree) ενός κόμβου είναι ο αριθμός των παιδιών του.

Βαθμός ενός δένδρου είναι ο μέγιστος των βαθμών των κόμβων του.

Δυαδικά Δένδρα

Δυαδικά δένδρα: έχουν βαθμό 2.

Σε κάθε κόμβο: **Αριστερό υποδένδρο - δεξί υποδένδρο**

Πλήρες δυαδικό δένδρο: έχει το μέγιστο αριθμό κόμβων σε κάθε επίπεδο πλην ίσως του τελευταίου (αν στο τελευταίο δεν έχει το μέγιστο αριθμό κόμβων, οι κόμβοι πρέπει να είναι αριστερά).

Αναπαράσταση

Συνεχόμενη αναπαράσταση (με πίνακα)

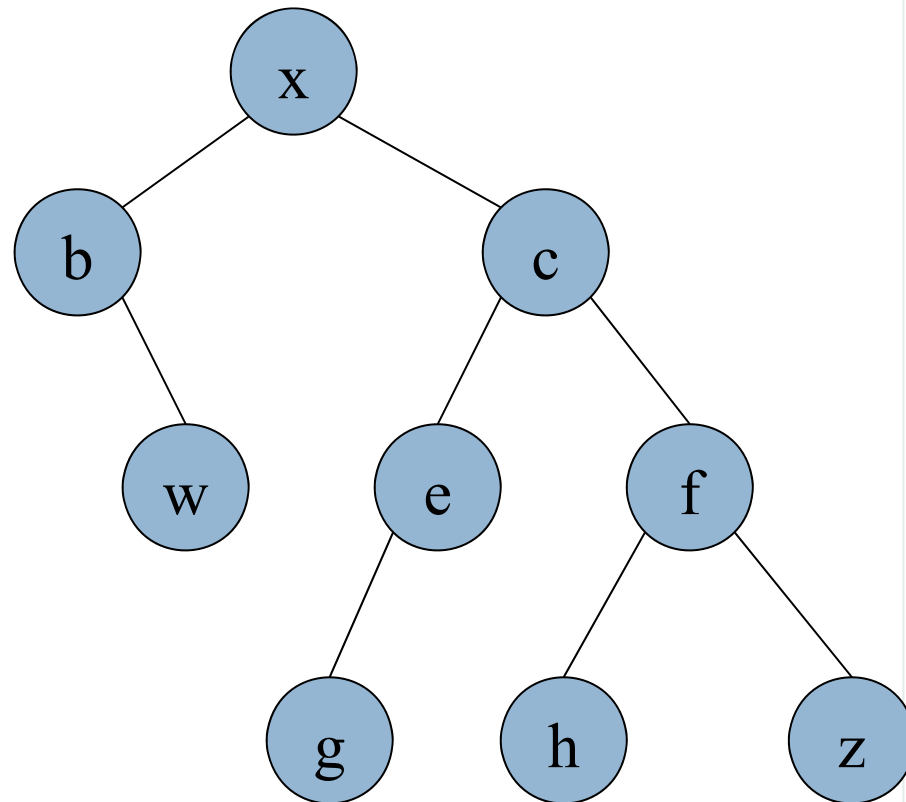
- Η ρίζα μπαίνει στη θέση 0.
- Αν ένας κόμβος είναι στη θέση k , το αριστερό παιδί του θα είναι στη θέση $2*k+1$ και το δεξί παιδί του θα είναι στη θέση $2*k+2$.

Συνδεδεμένη αναπαράσταση (η πιο συνηθής μορφή)

- Χρήση 2 δεικτών (left, right)

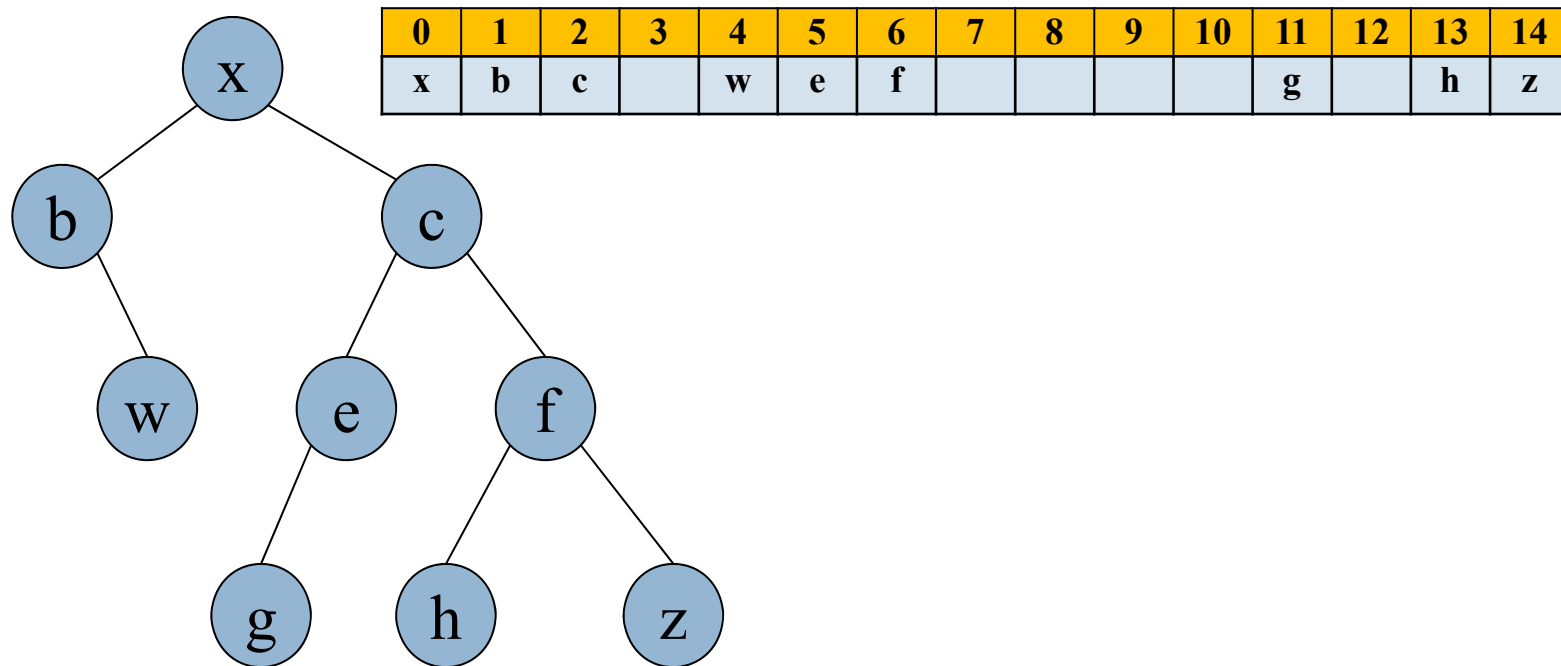
Διαπεράσεις Δυαδικών Δένδρων

- Προδιατεταγμένη (**PAΔ**):
x, b, w, c, e, g, f, h, z
- Μεταδιατεταγμένη (**ADP**):
w, b, g, e, h, z, f, c, x
- Ενδοδιατεταγμένη (**APΔ**):
b, w, x, g, e, c, h, f, z
- Κατά σειρά επιπέδων:
x, b, c, w, e, f, g, h, z



Συνεχόμενη αναπαράσταση δένδρου

- Η συνεχόμενη αναπαράσταση του δέντρου θα είναι ως εξής (ο πίνακας έχει **15** θέσεις λόγω του ύψους του δέντρου) (στις θέσεις $2*k+1$ και $2*k+2$ βρίσκονται τα παιδιά):



Ειδικά Δυαδικά Δένδρα

- **Δυαδικά Δένδρα Αναζήτησης**

- Το αριστερό υποδένδρο κάθε κόμβου έχει τιμές μικρότερες από την τιμή του κόμβου.
- Το δεξί υποδένδρο κάθε κόμβου έχει τιμές μεγαλύτερες από την τιμή του κόμβου.
- **Χρησιμοποιούνται στην αναζήτηση στοιχείων**, όπου βελτιώνουν σημαντικά το χρόνο (αντίστοιχα με τη δυαδική αναζήτηση σε ταξινομημένο πίνακα).
- Με **ενδοδιατεταγμένη διαπέραση** παίρνουμε τα στοιχεία **ταξινομημένα κατά αύξουσα σειρά**.

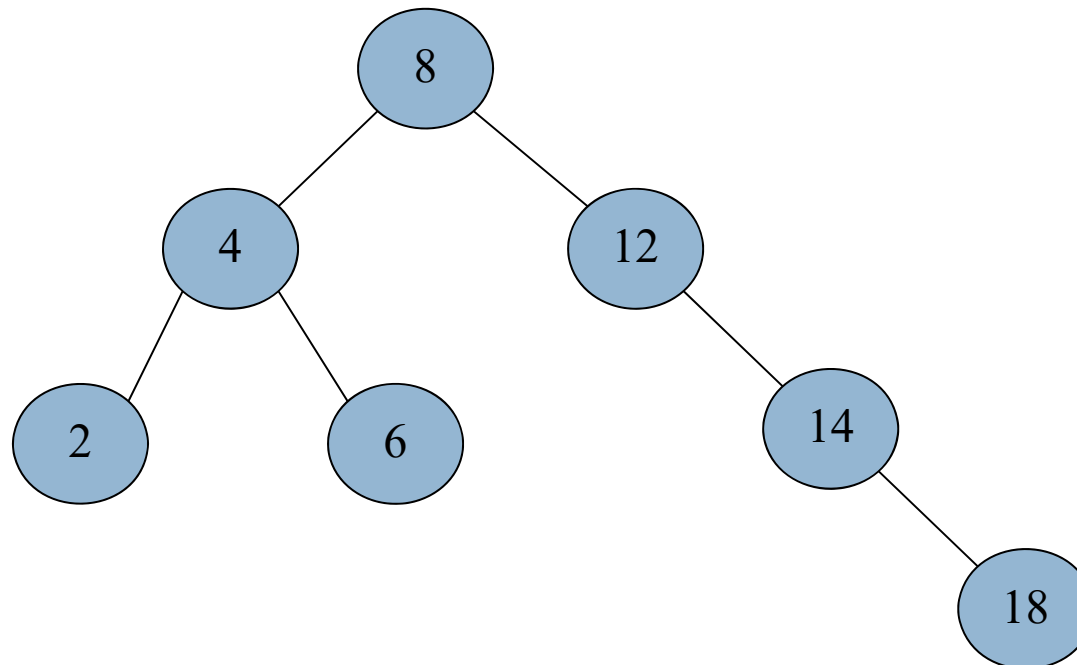
- **Δένδρα - Σωροί**

- Πλήρη δυαδικά δένδρα.
- Στο σωρό ελαχίστων (μεγίστων) η τιμή κάθε κόμβου είναι μικρότερη (μεγαλύτερη) από τις τιμές των παιδιών του.

Δυαδικά Δένδρα Αναζήτησης

Εισαγωγή στοιχείων:

8, 4, 12, 2, 6, 14, 18



Δυαδικά Δένδρα Αναζήτησης

- **Διαγραφές (περιπτώσεις)**
 - Ο κόμβος που διαγράφεται δεν έχει υποδένδρα.
 - Ο κόμβος που διαγράφεται έχει μόνο ένα υποδένδρο.
 - Ο κόμβος που διαγράφεται έχει δύο υποδένδρα.

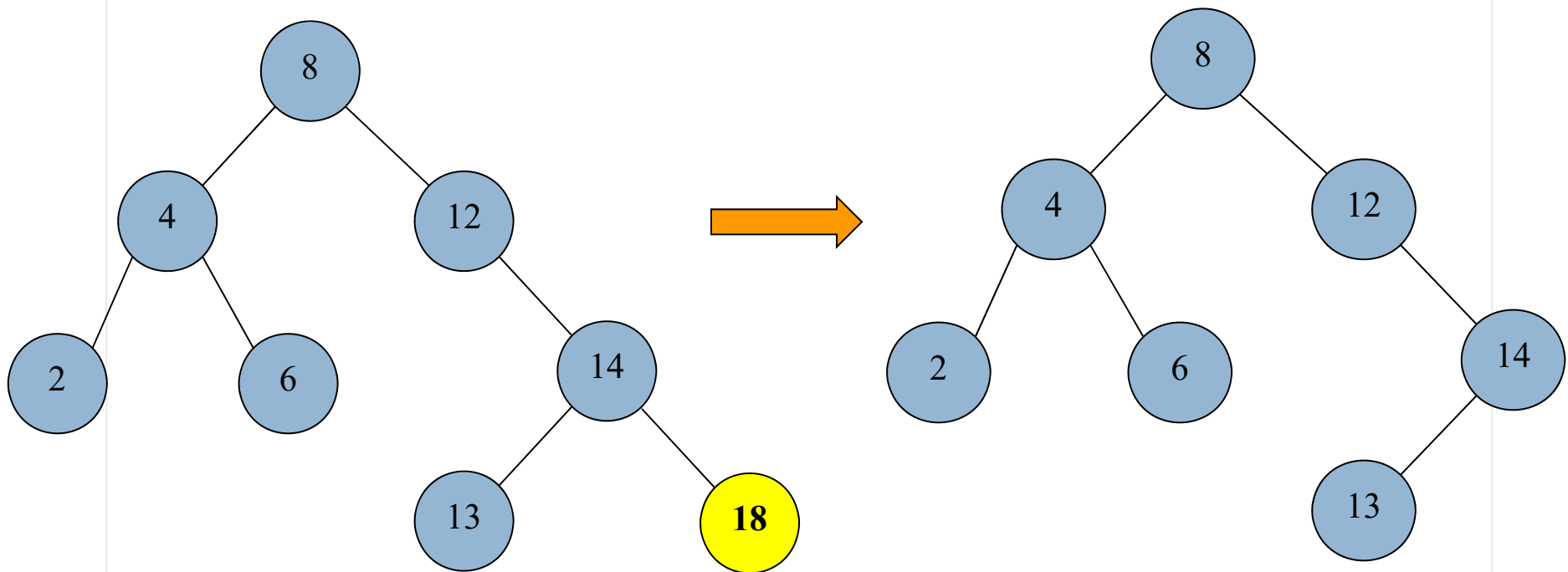
Δυαδικά Δένδρα Αναζήτησης

- **Διαγραφές**

- Οι πρώτες δύο περιπτώσεις είναι πιο εύκολες.
- Έστω ότι διαγράφουμε έναν κόμβο t που **δεν έχει** υποδένδρα (παιδιά). Τότε απλά διαγράφουμε τον κόμβο και θέτουμε το δείκτη του γονιού σε NULL.
- Αν ο κόμβος που διαγράφεται έχει **μόνο ένα** υποδένδρο (παιδί), τότε μετά τη διαγραφή του κόμβου t τη θέση του παίρνει η ρίζα του υποδένδρου (παιδιού) του t και ενημερώνεται κατάλληλα ο δείκτης του γονιού του t .

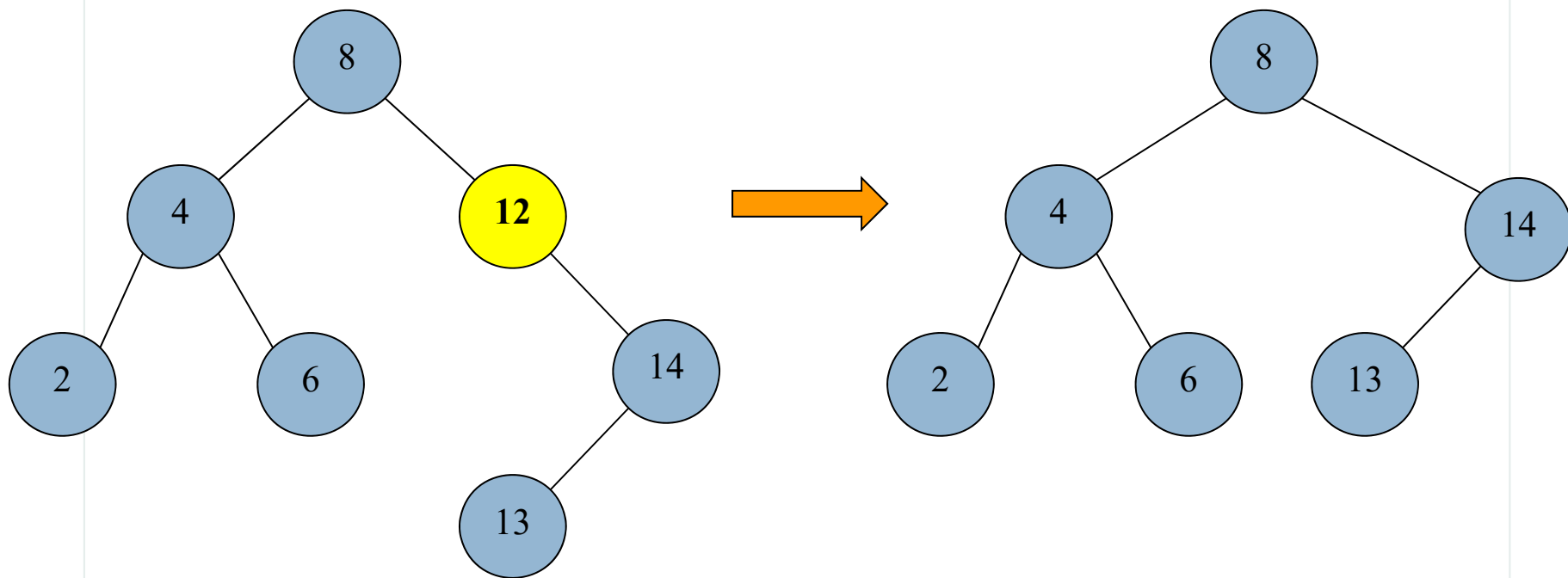
Δυαδικά Δένδρα Αναζήτησης

- **Διαγραφή του 18:** ο κόμβος που περιέχει το 18 **δεν έχει** υποδένδρα



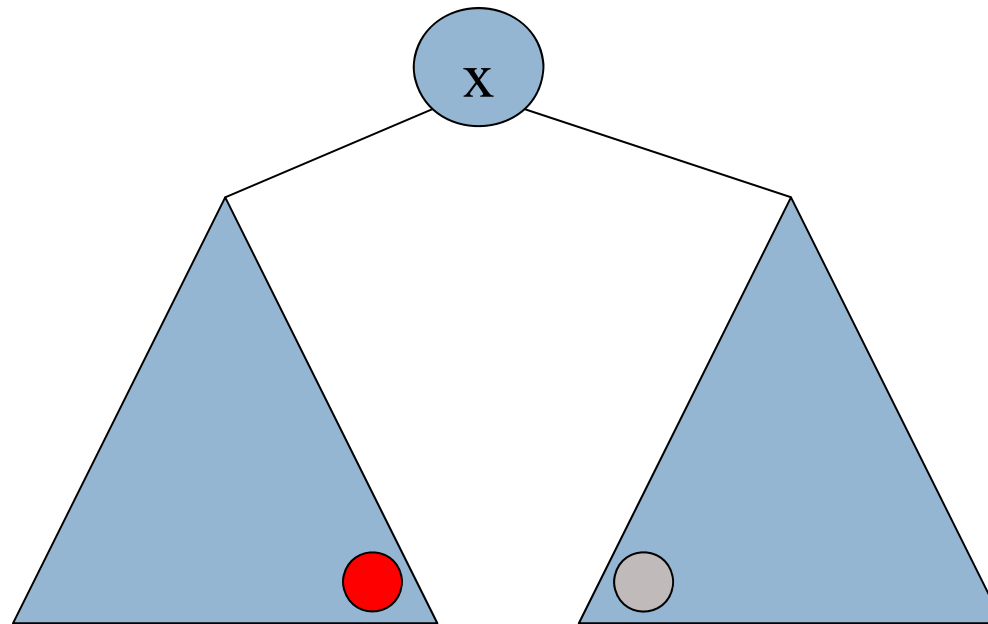
Δυαδικά Δένδρα Αναζήτησης

- **Διαγραφή του 12:** ο κόμβος που περιέχει το 12 έχει μόνο δεξί υποδένδρο



Δυαδικά Δένδρα Αναζήτησης

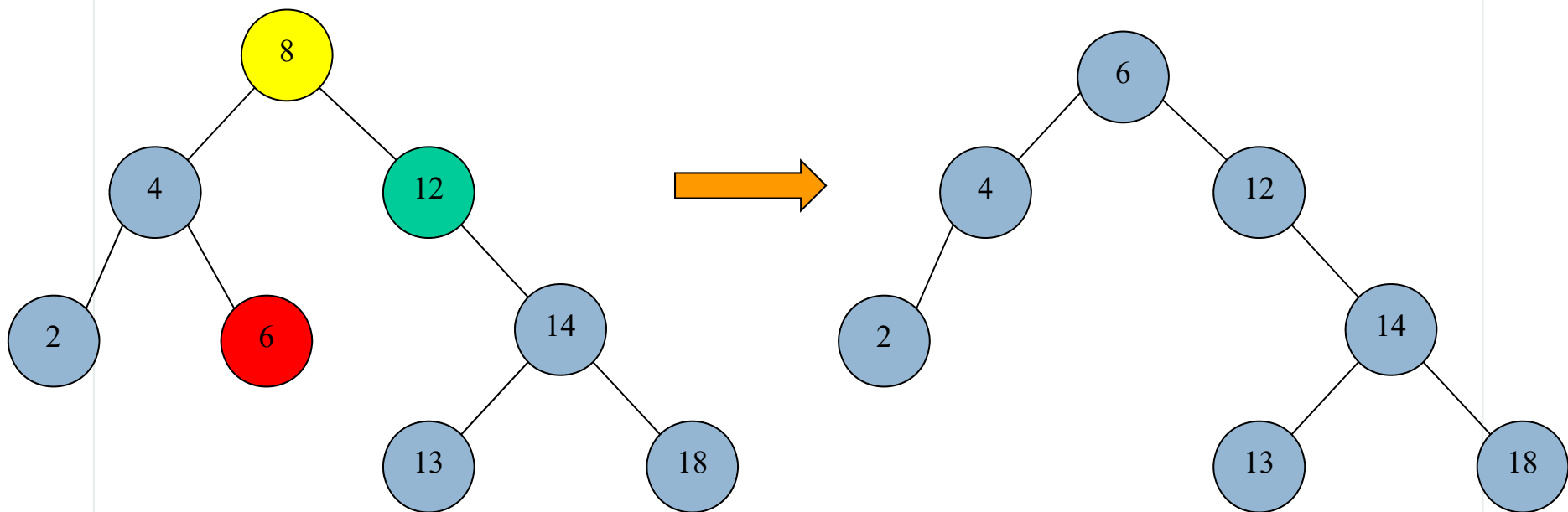
- **Διαγραφή κόμβου που έχει δύο υποδένδρα.** Τη θέση του στοιχείου που διαγράφεται θα πάρει είτε το **μεγαλύτερο στοιχείο του αριστερού υποδένδρου** είτε το **μικρότερο στοιχείο του δεξιού υποδένδρου** (αυτή η επιλογή περιγράφεται στον τόμο μελέτης).



Πώς βρίσκουμε τα δύο αυτά στοιχεία;

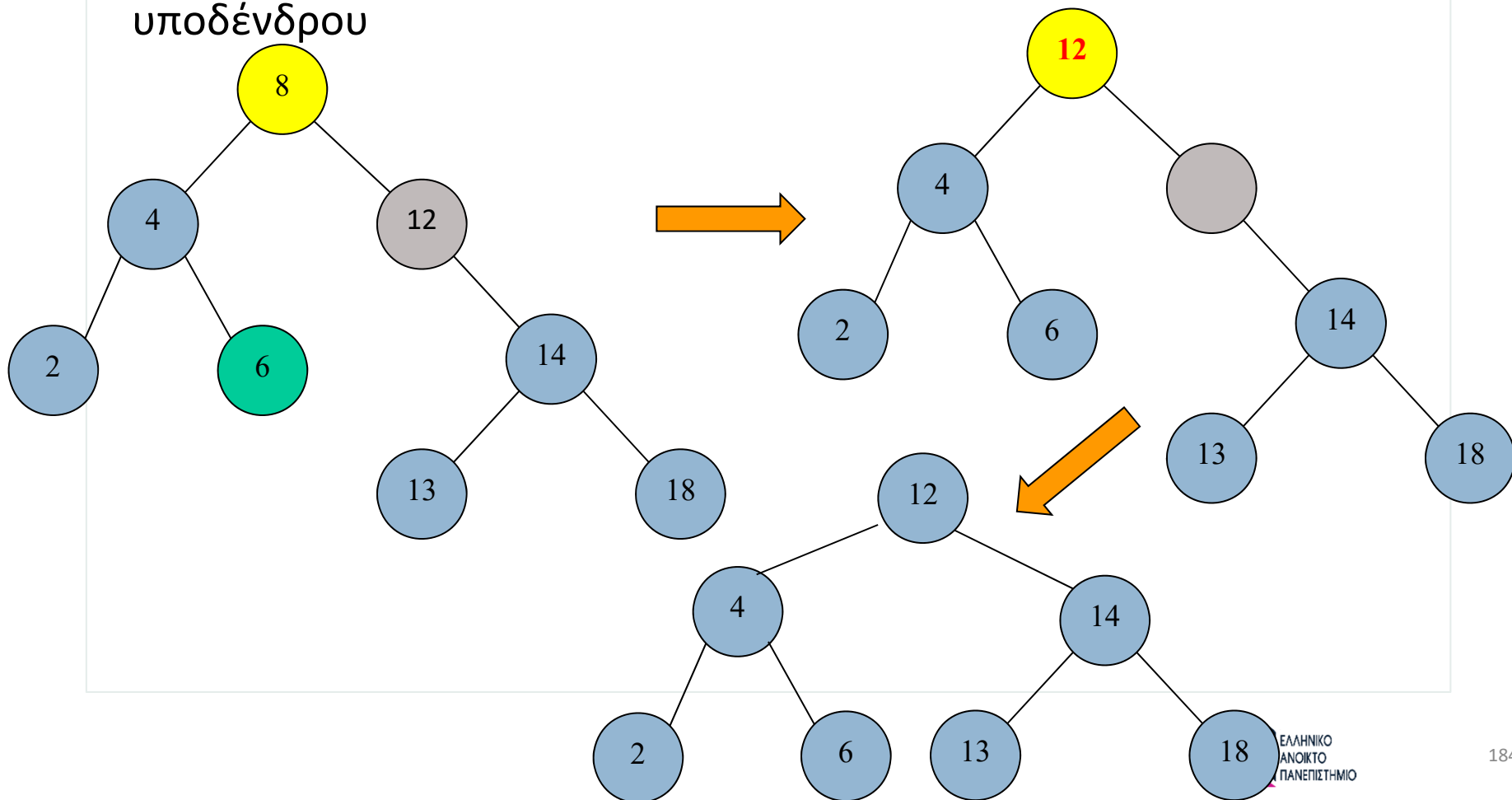
Δυαδικά Δένδρα Αναζήτησης

Διαγραφή του 8: Αντικατάσταση με το μεγαλύτερο του αριστερού υποδένδρου



Δυαδικά Δένδρα Αναζήτησης

- **Διαγραφή του 8:** Αντικατάσταση με το μικρότερο του δεξιού υποδένδρου



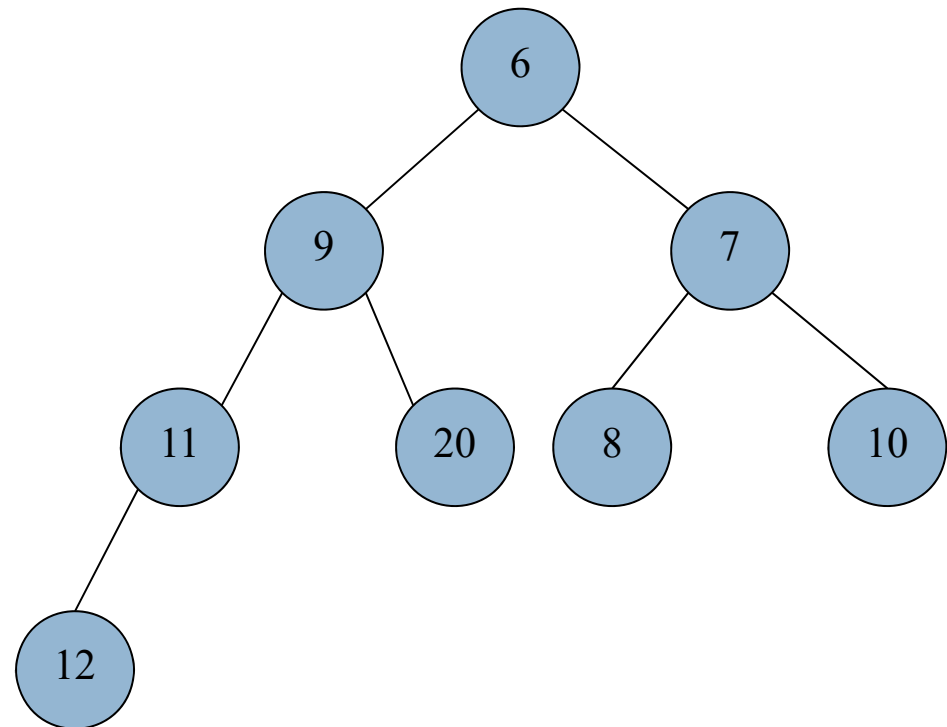
Δένδρο - Σωρός

- Είναι ένα πλήρες δυαδικό δένδρο που:
 - η τιμή κάθε κόμβου είναι μεγαλύτερη από τις τιμές των παιδιών του (**Σωρός μεγίστων** (maxHeap))
ή
 - η τιμή κάθε κόμβου είναι μικρότερη από τις τιμές των παιδιών του (**Σωρός ελαχίστων** (minHeap))
- Υποστηρίζει τις ακόλουθες βασικές λειτουργίες:
 - Εύρεση ελαχίστου (ή μεγίστου) σε σταθερό χρόνο.
 - Εισαγωγή στοιχείου σε χρόνο $O(\log n)$ όπου n το πλήθος των στοιχείων του σωρού.
 - Διαγραφή ελαχίστου (ή μεγίστου) σε χρόνο $O(\log n)$.

Δένδρο - Σωρός

Σωρός ελαχίστων (minHeap)

- Το δένδρο-σωρός μεγαλώνει από πάνω προς τα κάτω και από αριστερά προς τα δεξιά. Μόνο το τελευταίο επίπεδο μπορεί να μην είναι πλήρως συμπληρωμένο.
- Το στοιχείο ενός κόμβου είναι μικρότερο από τα στοιχεία των παιδιών του.
- Το ελάχιστο στοιχείο βρίσκεται πάντα στη ρίζα του δένδρου.



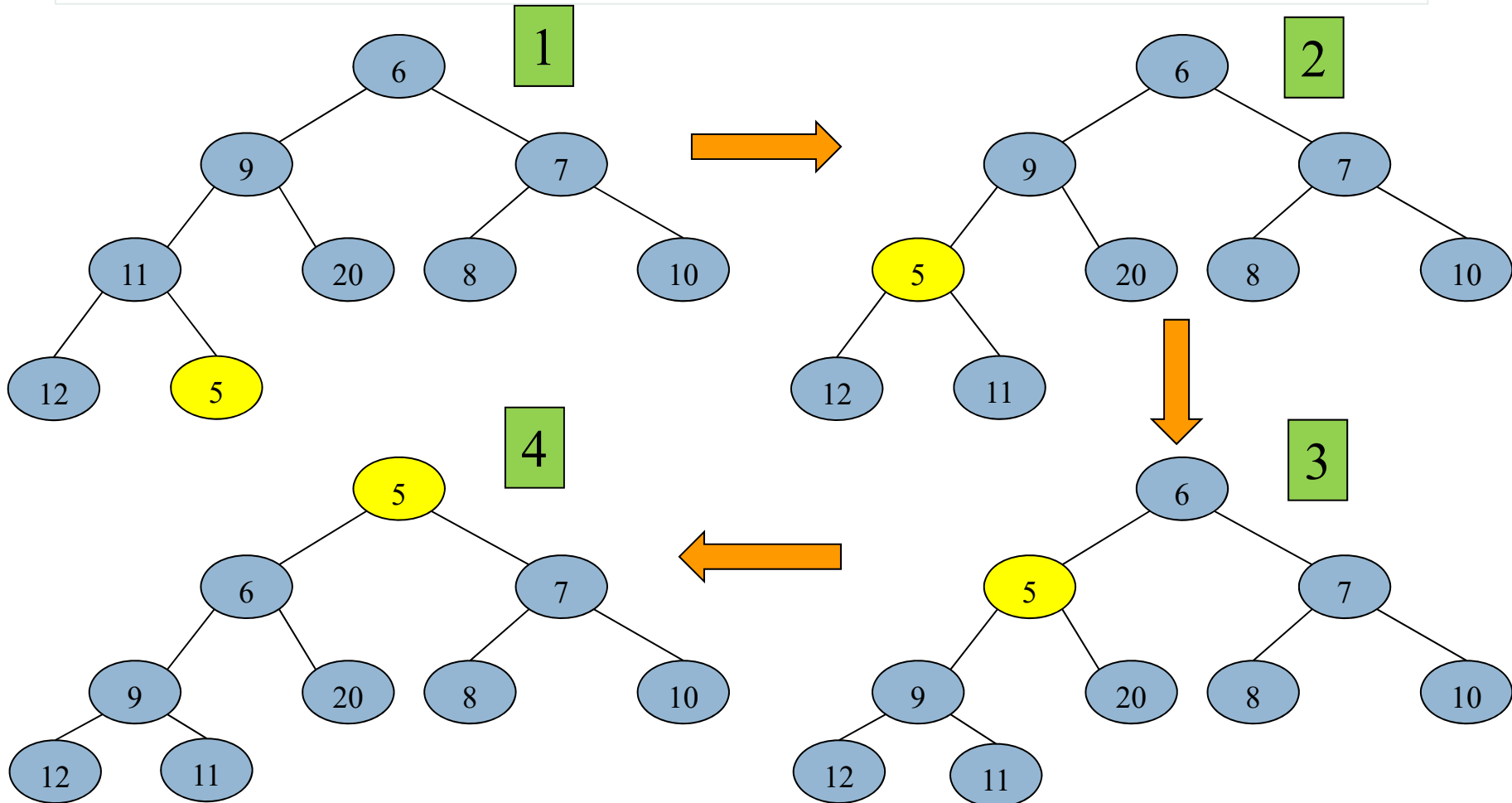
Δένδρο - Σωρός

- **Διαδικασία εισαγωγής**

- Το νέο στοιχείο τοποθετείται σε έναν κόμβο στο τέλος του σωρού.
- Στη συνέχεια, λαμβάνουν χώρα αντιμεταθέσεις ώστε το νέο στοιχείο να τοποθετηθεί τελικά στο σωστό επίπεδο.
- Το νέο στοιχείο μπορεί να φτάσει μέχρι τη ρίζα του σωρού.

Δένδρο - Σωρός

- Παράδειγμα εισαγωγή του 5

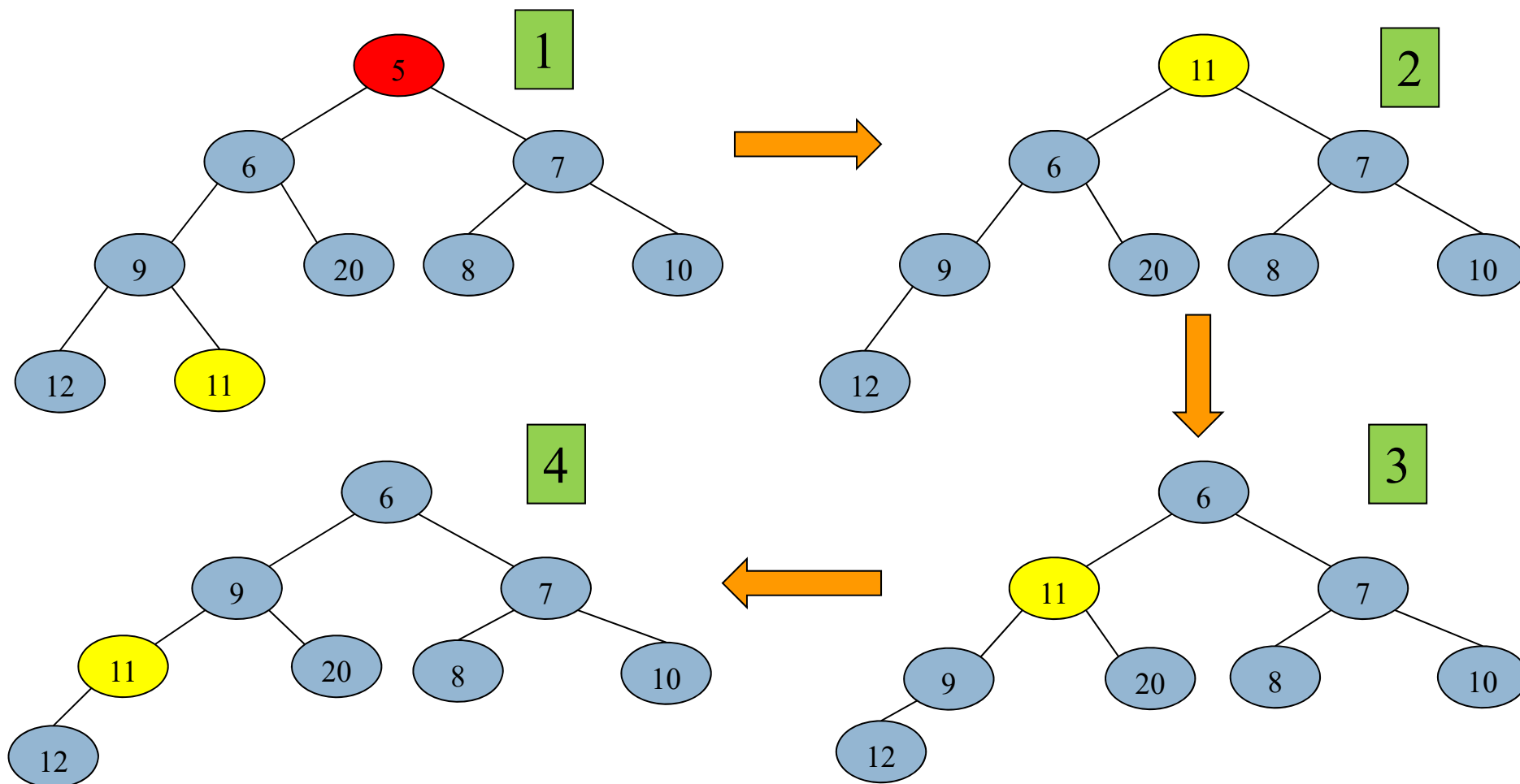


Δένδρο - Σωρός

- **Διαδικασία διαγραφής ελαχίστου**
 - Παίρνουμε το τελευταίο στοιχείο του σωρού και το τοποθετούμε στη ρίζα.
 - Στη συνέχεια, λαμβάνουν χώρα αντιμεταθέσεις ώστε το στοιχείο που μπήκε στη ρίζα να τοποθετηθεί τελικά στο σωστό επίπεδο.
 - Το στοιχείο που τοποθετήθηκε στη ρίζα μπορεί να φτάσει μέχρι το τελευταίο επίπεδο του σωρού.
 - Το μέγεθος του σωρού (=πλήθος στοιχείων) μειώνεται κατά 1.

Δένδρο - Σωρός

- Παράδειγμα διαγραφής ελαχίστου



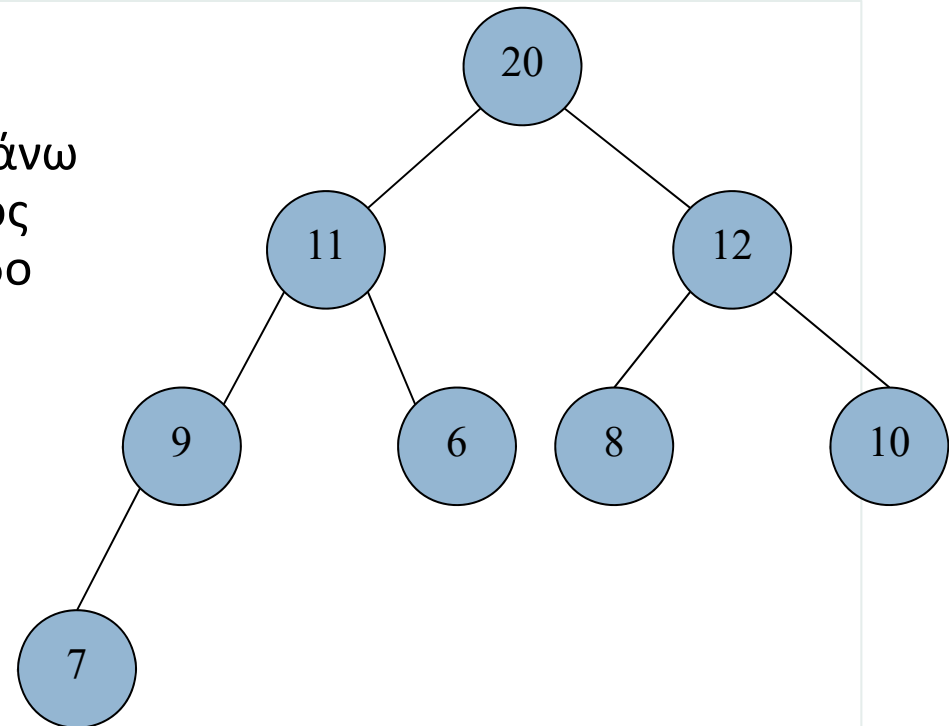
Δένδρο - Σωρός

Σωρός μεγίστων (maxHeap)

Το δένδρο-σωρός μεγαλώνει από πάνω προς τα κάτω και από αριστερά προς τα δεξιά. Μόνο το τελευταίο επίπεδο μπορεί να μην είναι πλήρως συμπληρωμένο.

Το στοιχείο ενός κόμβου είναι μεγαλύτερο από τα στοιχεία των παιδιών του.

Το μέγιστο στοιχείο βρίσκεται πάντα στη ρίζα του δένδρου.



Διάταξη ή Ταξινόμηση

- Αποτελεί βασική λειτουργία στα συστήματα επεξεργασίας δεδομένων.
- **Είσοδος:** σύνολο στοιχείων που βρίσκονται σε έναν πίνακα.
- **Έξοδος:** ταξινομημένος πίνακας σε αύξουσα ή φθίνουσα διάταξη.
- Επειδή η ταξινόμηση χρησιμοποιείται συχνά για την επίλυση δυσκολότερων προβλημάτων θα πρέπει να εκτελείται **γρήγορα**.

Διάταξη

- Γνωστότεροι αλγόριθμοι διάταξης:
 - με επιλογή (selection sort)
 - με εισαγωγή (insertion sort)
 - με ανταλλαγή ή φουσαλίδας (bubble sort)
 - με συγχώνευση (mergesort)
 - με χρήση σωρού (heapsort)
 - γρήγορη ταξινόμηση (quicksort)

Διάταξη Φυσαλίδας

- **Αλγόριθμος (η βασική ιδέα)**

- Έστω A ο πίνακας που περιέχει n στοιχεία. Σε κάθε πέρασμα, συγκρίνονται δύο γειτονικά στοιχεία, έστω στις θέσεις i και $i+1$.
- Εάν $A[i] > A[i+1]$ τότε αντιμεταθέτουμε τα δύο στοιχεία.
- Σε κάθε πέρασμα, το μεγαλύτερο στοιχείο τοποθετείται στο τέλος του πίνακα.

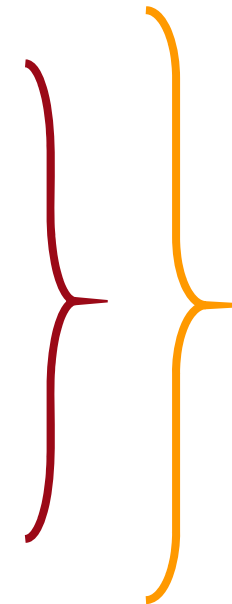
Σημείωση: Εναλλακτικά η μέθοδος μπορεί να τοποθετήσει το μικρότερο στοιχείο στην αρχή του πίνακα, στη συνέχεια παρατίθενται σε ψευδοκώδικα και οι δύο παραλλαγές.

Διάταξη Φυσαλίδας ()

Ψευδοκώδικας (1).

Το μικρότερο στοιχείο Τοποθετείται στην **αρχή**.

```
ΓΙΑ I:=2 ΕΩΣ N ΕΠΑΝΑΛΑΒΕ
  ΓΙΑ K:=N ΕΩΣ I ΜΕ ΒΗΜΑ -1 ΕΠΑΝΑΛΑΒΕ
    ΕΑΝ (P[K] < P[K-1]) ΤΟΤΕ
      TEMP:= P[K-1]
      P[K-1]:= P[K]
      P[K]:= TEMP
    ΕΑΝ-ΤΕΛΟΣ
  ΓΙΑ-ΤΕΛΟΣ
ΓΙΑ-ΤΕΛΟΣ
```



Διάταξη Φυσαλίδας

Το μικρότερο στοιχείο τοποθετείται στην αρχή.

- Πρώτο πέρασμα: το στοιχείο 1 τοποθετείται στην αρχή του πίνακα.

Ερώτηση:

- Πόσες συγκρίσεις / αντιμεταθέσεις πραγματοποιούνται στη χειρότερη περίπτωση στο πρώτο πέρασμα;
- Πόσες στο δεύτερο;
- Πόσες συνολικά συγκρίσεις θα εκτελεστούν σε σχέση με το μέγεθος (n) του πίνακα;

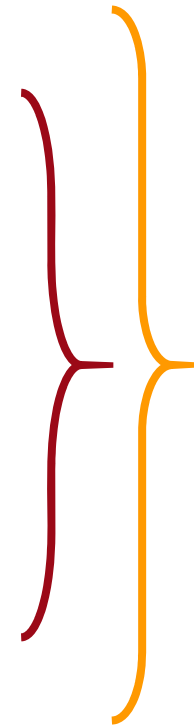
9	2	1	5	7	6	3	4
9	2	1	5	7	3	6	4
9	2	1	5	7	3	6	4
9	2	1	5	3	7	6	4
9	2	1	5	3	7	6	4
9	2	1	3	5	6	3	4
9	2	1	3	5	6	3	4
9	1	2	3	5	6	3	4
9	1	2	3	5	6	3	4
1	9	2	3	5	6	3	4

Διάταξη Φυσαλίδας

Ψευδοκώδικας (2).

Το μεγαλύτερο στοιχείο τοποθετείται στο **τέλος**

```
ΓΙΑ I:=2 ΕΩΣ N ΕΠΑΝΑΛΑΒΕ
  ΓΙΑ K:=1 ΕΩΣ N-I+1 ΕΠΑΝΑΛΑΒΕ
    ΕΑΝ (P[K] > P[K+1]) ΤΟΤΕ
      TEMP:= P[K+1]
      P[K+1]:= P[K]
      P[K]:= TEMP
    ΕΑΝ-ΤΕΛΟΣ
  ΓΙΑ-ΤΕΛΟΣ
ΓΙΑ-ΤΕΛΟΣ
```



Διάταξη Φυσαλίδας

Το μεγαλύτερο στοιχείο τοποθετείται στο τέλος

- Πρώτο πέρασμα: το στοιχείο 9 τοποθετείται στο τέλος του πίνακα.

Ερώτηση:

- Πόσες συγκρίσεις/αντιμεταθέσεις πραγματοποιούνται στη χειρότερη περίπτωση στο πρώτο πέρασμα;
- Πόσες στο δεύτερο;
- Πόσες συνολικά συγκρίσεις θα εκτελεστούν σε σχέση με το μέγεθος (n) του πίνακα;

$$\sum_{k=1}^n k = \frac{n(n+1)}{2}.$$

9	2	1	5	7	6	3	4
2	9	1	5	7	6	3	4
2	9	1	5	7	6	3	4
2	1	9	5	7	6	3	4
2	1	9	5	7	6	3	4
2	1	5	9	7	6	3	4
2	1	5	9	7	6	3	4
2	1	5	7	9	6	3	4
2	1	5	7	9	6	3	4
2	1	5	7	6	9	3	4
2	1	5	7	6	9	3	4
2	1	5	7	6	3	9	4
2	1	5	7	6	3	9	4
2	1	5	7	6	3	4	9

Διάταξη με Επιλογή

- Από τις πλέον κατανοητές «διαισθητικά».
- Διαίρεσε τον πίνακα σε έναν ταξινομημένο υποπίνακα και σε έναν μη-ταξινομημένο.
- Σε κάθε πέρασμα, το μικρότερο (μεγαλύτερο) κλειδί του μη-ταξινομημένου υποπίνακα τοποθετείται στην αρχή (τέλος) του.

Διάταξη με Επιλογή

- Σε κάθε πέρασμα,
 - Βρες το μικρότερο κλειδί στην μη ταξινομημένη λίστα
 - Ενάλλαξε το επιλεγμένο κλειδί με το τελευταίο κλειδί της μη-ταξινομημένης λίστας

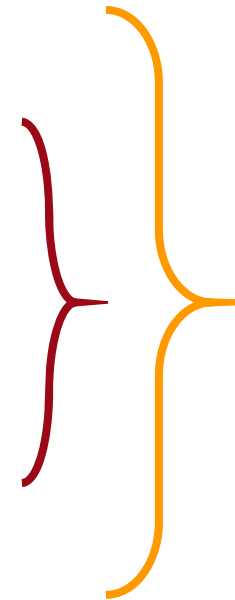
Σημείωση: Εναλλακτικά ο αλγόριθμος βρίσκει κάθε φορά το μεγαλύτερο στοιχείο και το τοποθετεί στο τέλος.

Διάταξη με Επιλογή

Ψευδοκώδικας

Εύρεση μικρότερου στοιχείου.

```
ΓΙΑ Κ:=1 ΕΩΣ Ν-1 ΕΠΑΝΑΛΑΒΕ
  ΜΙΝ:= Ρ[Κ]; ΡΟS:=Κ;
  ΓΙΑ Ι:=Κ+1 ΕΩΣ Ν ΕΠΑΝΑΛΑΒΕ
    ΕΑΝ (Ρ[Ι] < ΜΙΝ) ΤΟΤΕ
      ΜΙΝ:= Ρ[Ι];
      ΡΟS:= Ι
    ΕΑΝ-ΤΕΛΟΣ
  ΓΙΑ-ΤΕΛΟΣ
  ΤΕΜΡ:=Ρ[Κ]; Ρ[Κ]:=ΜΙΝ; Ρ[ΡΟS]:=ΤΕΜΡ
ΓΙΑ-ΤΕΛΟΣ
```



Διάταξη με Επιλογή

Βρες το ελάχιστο

9	2	1	5	7	6	3	4
---	---	---	---	---	---	---	---

Τοποθέτησε το ελάχιστο στην αρχή

1	2	9	5	7	6	3	4
---	---	---	---	---	---	---	---

Βρες το ελάχιστο

1	2	9	5	7	6	3	4
---	---	---	---	---	---	---	---

Τοποθέτησε το ελάχιστο στην αρχή

1	2	9	5	7	6	3	4
---	---	---	---	---	---	---	---

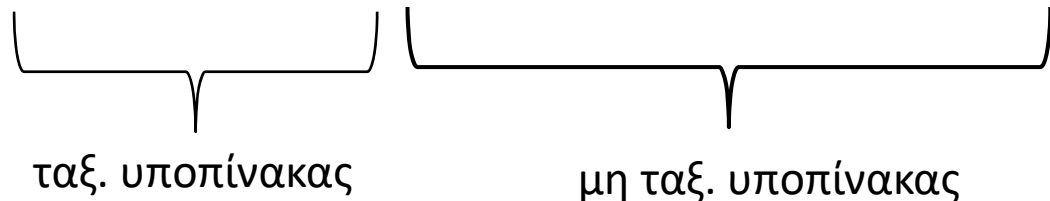
Βρες το ελάχιστο

1	2	9	5	7	6	3	4
---	---	---	---	---	---	---	---

Τοποθέτησε το ελάχιστο στην αρχή

1	2	3	5	7	6	9	4
---	---	---	---	---	---	---	---

**Σχολιάστε για το πλήθος
συγκρίσεων που απαιτούνται**

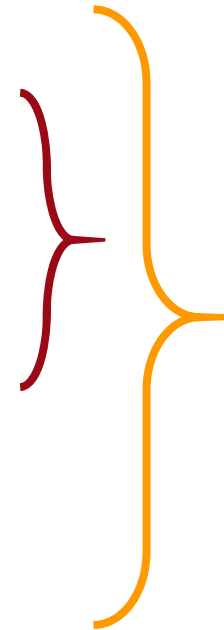


Διάταξη με Επιλογή

Ψευδοκώδικας

(με εύρεση **μεγαλύτερου** στοιχείου)

```
ΓΙΑ Κ:=1 ΕΩΣ Ν-1 ΕΠΑΝΑΛΑΒΕ
  ΜΑΧ:= Ρ[1]; ΡΟΣ:=1;
  ΓΙΑ Ι:=2 ΕΩΣ Ν-Κ+1 ΕΠΑΝΑΛΑΒΕ
    ΕΑΝ (Ρ[Ι] > ΜΑΧ) ΤΟΤΕ
      ΜΑΧ:= Ρ[Ι];
      ΡΟΣ:= Ι
    ΕΑΝ-ΤΕΛΟΣ
  ΓΙΑ-ΤΕΛΟΣ
  ΤΕΜΡ:=Ρ[Ν-Κ+1];
  Ρ[Ν-Κ+1]:=ΜΑΧ;
  Ρ[ΡΟΣ]:=ΤΕΜΡ
ΓΙΑ-ΤΕΛΟΣ
```



Διάταξη με Επιλογή

Βρες το μέγιστο

9	2	1	5	7	6	3	4
---	---	---	---	---	---	---	---

Τοποθέτησε το μέγιστο στο τέλος

4	2	1	5	7	6	3	9
---	---	---	---	---	---	---	---

Βρες το μέγιστο

4	2	1	5	7	6	3	9
---	---	---	---	---	---	---	---

Τοποθέτησε το μέγιστο στο τέλος

4	2	1	5	3	6	7	9
---	---	---	---	---	---	---	---

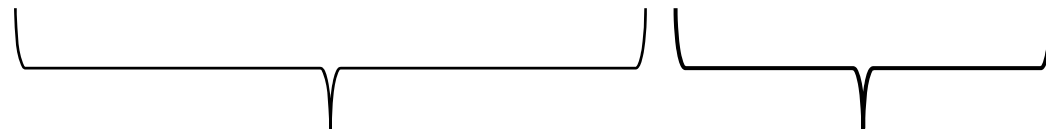
Βρες το μέγιστο

4	2	1	5	3	6	7	9
---	---	---	---	---	---	---	---

Τοποθέτησε το μέγιστο στο τέλος

4	2	1	5	3	6	7	9
---	---	---	---	---	---	---	---

Σχολιάστε για το πλήθος συγκρίσεων που απαιτούνται



μη ταξ. υποπίνακας

ταξ. υποπίνακας

Γρήγορη Διάταξη

- Η «γρήγορη» διάταξη βασίζεται στην αρχή του «**Διαίρει και Βασίλευε**».
- Στον προγραμματισμό της, χρησιμοποιούμε την τεχνική της «**αναδρομής**».
- Είναι αισθητά πιο αποδοτική από τις προηγούμενες δύο.

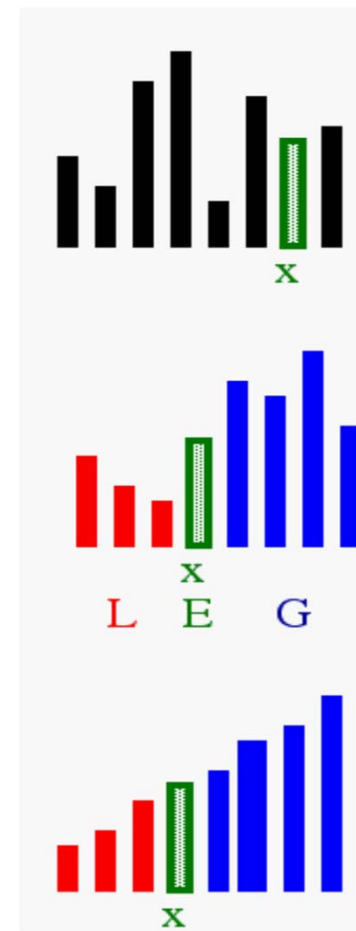
Γρήγορη Διάταξη

- **Διαίρει:** επέλεξε ένα στοιχείο «**άξονα**» του πίνακα. Με βάση αυτό, χώρισε τον πίνακα στα δύο:
 - στον αριστερό υποπίνακα βάλε όλα τα μικρότερα στοιχεία και
 - στο δεξιό υποπίνακα όλα τα μεγαλύτερα από το στοιχείο-άξονα
- **Αναδρομή:** Αναδρομικά κάνε το ίδιο στους δύο υποπίνακες.
- **Βασίλευε:** Βάλε στη σειρά τον αριστερό υποπίνακα, το στοιχείο-άξονα και τέλος το δεξιό υποπίνακα.

Σημείωση: Στην απλή εκδοχή της *quicksort* ως στοιχείο-άξονας επιλέγεται συνήθως το πρώτο στοιχείο του κάθε υποπίνακα.

Γρήγορη Διάταξη

1. **Επιλογή:** Επίλεξε το στοιχείο - άξονα.
2. **Διαίρει:** Βάλε το σε θέση ώστε αριστερά να είναι τα μικρότερα και δεξιά τα μεγαλύτερα.
3. **Βασίλευε:** ταξινόμησε τον πίνακα αναδρομικά.



Καλή επιτυχία στις
εξετάσεις!