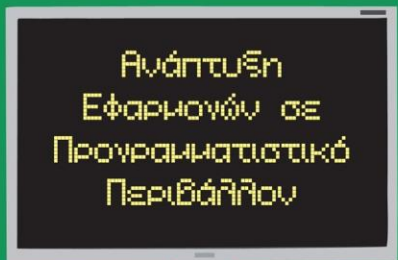


# Ανάπτυξη Εφαρμογών σε Προγραμματιστικό Περιβάλλον:

Η θεωρία του ΑΕΠΠ για τις Πανελλήνιες:  
Προηγμένες Δομές Δεδομένων



ΒΙΒΛΙΟ ΜΑΘΗΤΗ

Γ' ΓΕΝΙΚΟΥ ΛΥΚΕΙΟΥ

Ομάδας Προσανατολισμού Θετικών Σπουδών και Σπουδών Οικονομίας & Πληροφορικής



ΙΝΣΤΙΤΟΥΤΟ ΤΕΧΝΟΛΟΓΙΑΣ ΥΠΟΛΟΓΙΣΤΩΝ ΚΑΙ ΕΚΔΟΣΕΩΝ «ΔΙΟΦΑΝΤΟΣ»

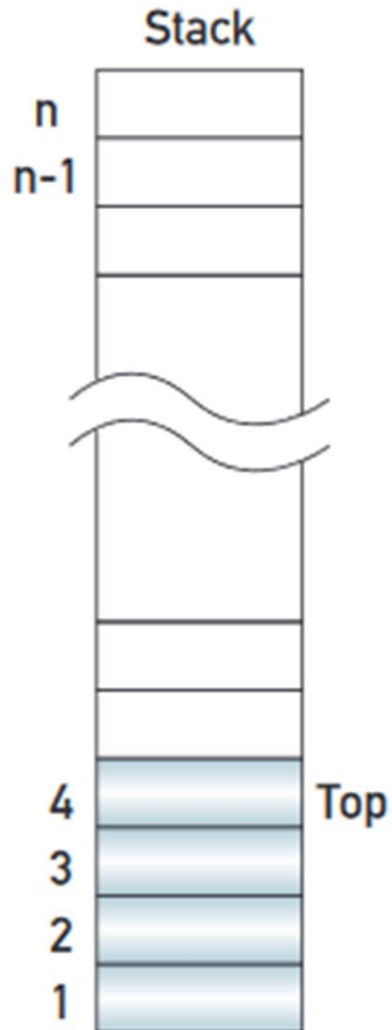
Δημήτρης Βαγιακάκος  
DimitrisV SVISJP

[www.youtube.com/LinuxOSblog](http://www.youtube.com/LinuxOSblog)  
[dimitrislinuxos@protonmail.ch](mailto:dimitrislinuxos@protonmail.ch)



ΔΩΡΕΑΝ Hardcore λυμένες Ασκήσεις και βοηθητικές σημειώσεις για Πανελλήνιες

# ΣΤΟΪΒΑ ( STACK )



LIFO (Last In, First Out) Δομή Δεδομένων , **δηλαδή:**

**Πώς να σκεφτώ μια στοίβα;**

Μια στοίβα με ταχυδρομικές επιστολές  
Εξέταση των επιστολών από πάνω προς τα κάτω.



Μια στοίβα από πιάτα  
Πλύσιμο των πιάτων από πάνω προς τα κάτω.



**Άρα ο,τι εισέρχεται τελευταίο (Last in) ,  
εξέρχεται πρώτο (First out)**

**Βασικές ενέργειες μιας στοίβας :**

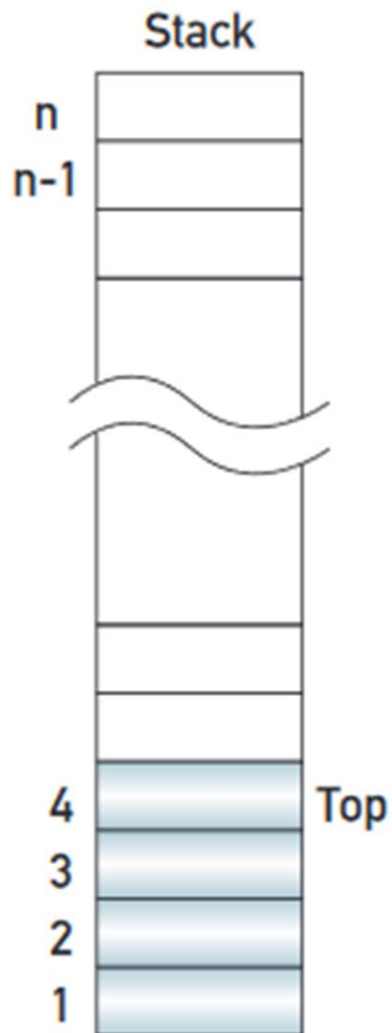
Τοποθέτηση στοιχείου στην κορυφή μιας στοίβας.

Ονομάζεται εισαγωγή (**push**).

Απομάκρυνση στοιχείου από την κορυφή.

Ονομάζεται εξαγωγή (**pop**).

# ΣΤΟΪΒΑ ( STACK )



Η διαδικασία της ώθησης πρέπει οπωσδήποτε να ελέγχει, αν η στοίβα είναι γεμάτη, οπότε λέγεται ότι συμβαίνει υπερχείλιση (overflow) της στοίβας.

Αντίστοιχα, η διαδικασία απώθησης ελέγχει, αν υπάρχει ένα τουλάχιστον στοιχείο στη στοίβα, δηλαδή ελέγχει αν γίνεται υποχείλιση (underflow) της στοίβας

Η δομή στοίβα μπορεί να υλοποιηθεί με :  
έναν πίνακα και  
έναν δείκτη στο στοιχείο της κορυφής της στοίβας. (μια βοηθητική μεταβλητή)

Μια βοηθητική μεταβλητή (με όνομα συνήθως top) χρησιμοποιείται για να δείχνει το στοιχείο που τοποθετήθηκε τελευταίο στην κορυφή της στοίβας. Για την εισαγωγή ενός νέου στοιχείου στη στοίβα (**ώθηση**) αρκεί να αυξηθεί η μεταβλητή top κατά ένα και στη θέση αυτή να εισέλθει το στοιχείο. Αντίθετα για την εξαγωγή ενός στοιχείου από τη στοίβα (**απώθηση**) εξέρχεται πρώτα το στοιχείο που δείχνει η μεταβλητή top και στη συνέχεια η top μειώνεται κατά ένα για να δείχνει τη νέα κορυφή.



# ΠΑΡΑΔΕΙΓΜΑ ΣΤΟΙΒΑΣ

push(10)	push(5)	pop()	push(15)	push(7)	pop()
				7	
	5		15	15	15
10	10	10	10	10	10

Εισαγωγή  
του 10

Εισαγωγή  
του 5

Αφαίρεση  
του πιο  
πάνω  
στοιχείου

Εισαγωγή  
του 15

Εισαγωγή  
του 7

Αφαίρεση  
του πιο  
πάνω  
στοιχείου



# ΟΥΡΑ ( QUEUE )



**FIFO (First In, First Out)** Δομή Δεδομένων , **δηλαδή:**  
**Πώς να σκεφτώ μια ουρά;**

Μια ουρά αναμονής για επιβίβαση στον Προαστιακό Σιδηρόδρομο.

Μία ουρά αναμονής στο Super Market.

Ουρά εκτυπωτή με εργασίες εκτύπωσης που αναμένουν επεξεργασία.

Ουρά πακέτων δεδομένων που περιμένουν την σειρά τους για να μεταδοθούν στο Διαδίκτυο.

**Άρα ο,τι εισέρχεται πρώτο (First in) ,  
εξέρχεται πρώτο (First out)**

**Βασικές ενέργειες μιας ουράς :**

Εισαγωγή (**enqueue**) στοιχείου στο πίσω άκρο της ουράς.

Εξαγωγή (**dequeue**) στοιχείου από το εμπρός άκρο της ουράς.

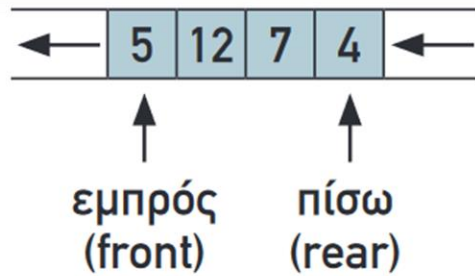
# ΟΥΡΑ ( QUEUE )



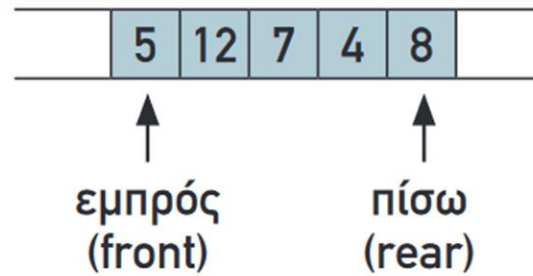
Άρα, σε αντίθεση με τη δομή της στοίβας, στην περίπτωση της ουράς απαιτούνται δύο δείκτες: ο εμπρός (**front**) και ο πίσω (**rear**) δείκτης, που μας δίνουν τη θέση του στοιχείου που σε πρώτη ευκαιρία θα εξαχθεί και τη θέση του στοιχείου που μόλις εισήλθε.

Μία ουρά μπορεί να υλοποιηθεί με τη βοήθεια ενός μονοδιάστατου πίνακα. Για την εισαγωγή ενός νέου στοιχείου στην ουρά **αυξάνεται ο δείκτης rear κατά ένα** και στη θέση αυτή αποθηκεύεται το στοιχείο. Αντίστοιχα για τη λειτουργία της εξαγωγής, εξέρχεται το στοιχείο που δείχνει ο δείκτης front, ο οποίος στη συνέχεια **αυξάνεται κατά ένα, για να δείχνει το επόμενο στοιχείο που πρόκειται να εξαχθεί**. Σε κάθε περίπτωση όμως, πρέπει να ελέγχεται πριν από οποιαδήποτε ενέργεια, αν υπάρχει ελεύθερος χώρος στον πίνακα για την εισαγωγή και αν υπάρχει ένα τουλάχιστον στοιχείο για την εξαγωγή.

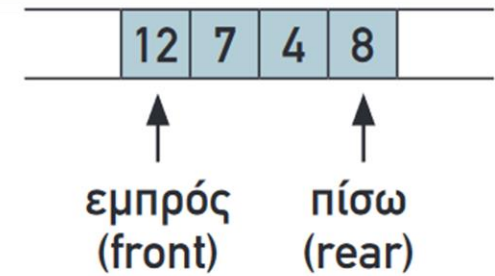
# ΠΑΡΑΔΕΙΓΜΑ ΟΥΡΑΣ



α) Μια ουρά  
με 4 στοιχεία

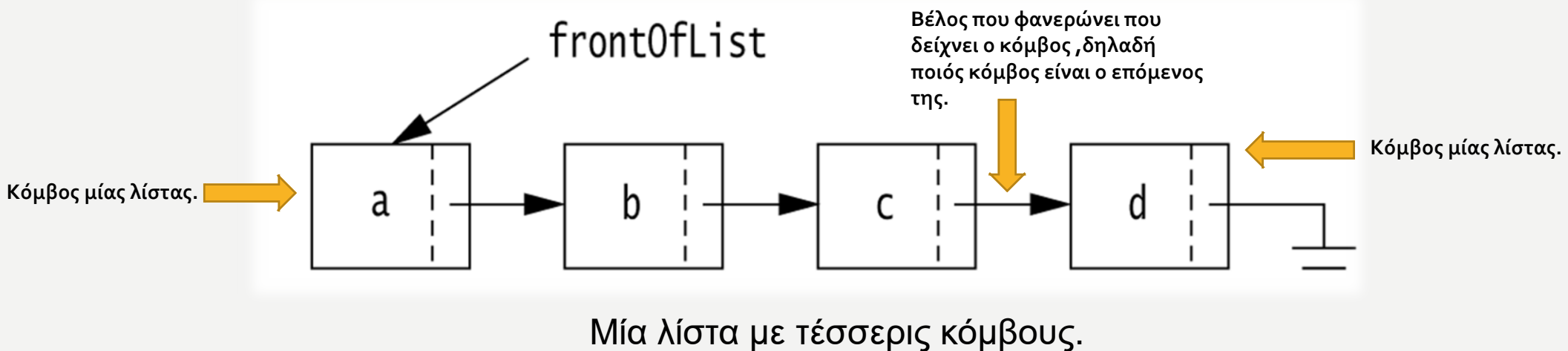


β) Η ουρά μετά  
την εισαγωγή του  
στοιχείου 8



γ) Η ουρά μετά  
την εξαγωγή του  
στοιχείου 5

# ΛΙΣΤΑ ( LIST )



Στις λίστες το κύριο χαρακτηριστικό είναι ότι οι κόμβοι τους συνήθως βρίσκονται σε απομακρυσμένες θέσεις μνήμης και η σύνδεσή τους γίνεται με δείκτες (pointers).

Ο δείκτης **δεν λαμβάνει αριθμητικές τιμές** όπως ακέραιες, πραγματικές κ.ά., αλλά **οι τιμές του είναι διευθύνσεις στην κύρια μνήμη** και χρησιμοποιείται ακριβώς για τη σύνδεση των διαφόρων στοιχείων μιας δομής, που είναι αποθηκευμένα σε μη συνεχόμενες θέσεις μνήμης. Συνήθως ο δείκτης είναι ένα πεδίο κάθε κόμβου της δομής, όπως φαίνεται στο διπλανό σχήμα.

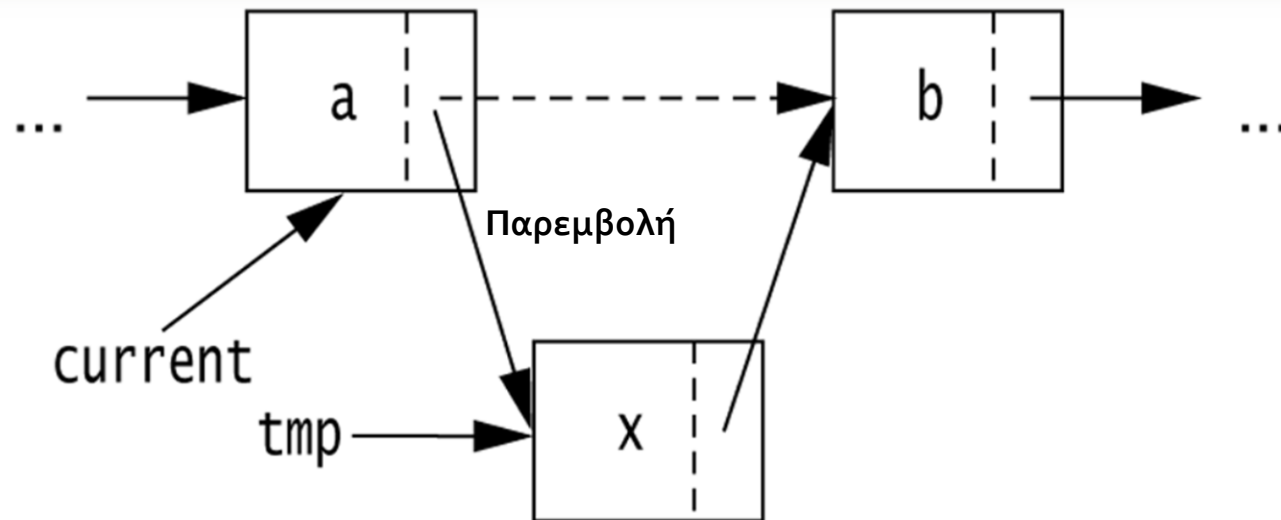
Δεδομένα

Δείκτης

! Οι όροι index και pointer αποδίδονται στα ελληνικά ως δείκτης. Και οι δύο παραπέμπουν σε θέσεις, πίνακα ο πρώτος και μνήμης ο δεύτερος.

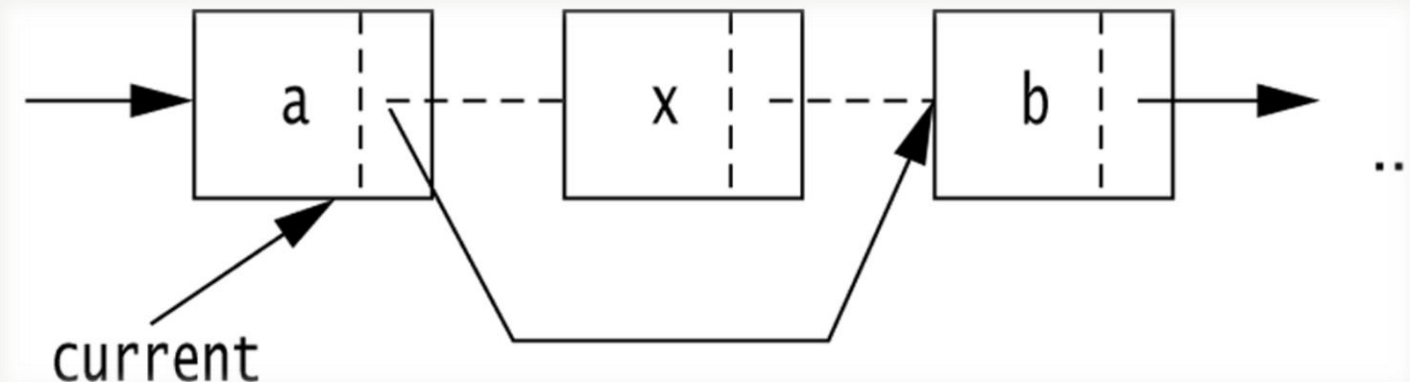


# ΕΙΣΑΓΩΓΗ ΣΕ ΛΙΣΤΑ



Όπως φαίνεται και στο σχήμα, οι απαιτούμενες ενέργειες για την εισαγωγή (**παρεμβολή**) του νέου κόμβου είναι ο δείκτης του δεύτερου κόμβου να δείχνει το νέο κόμβο και ο δείκτης του νέου κόμβου να δείχνει τον τρίτο κόμβο (δηλαδή να πάρει την τιμή που είχε πριν την εισαγωγή ο δείκτης του δεύτερου κόμβου). Έτσι οι κόμβοι της λίστας διατηρούν τη λογική τους σειρά, αλλά οι φυσικές θέσεις στη μνήμη μπορεί να είναι τελείως διαφορετικές

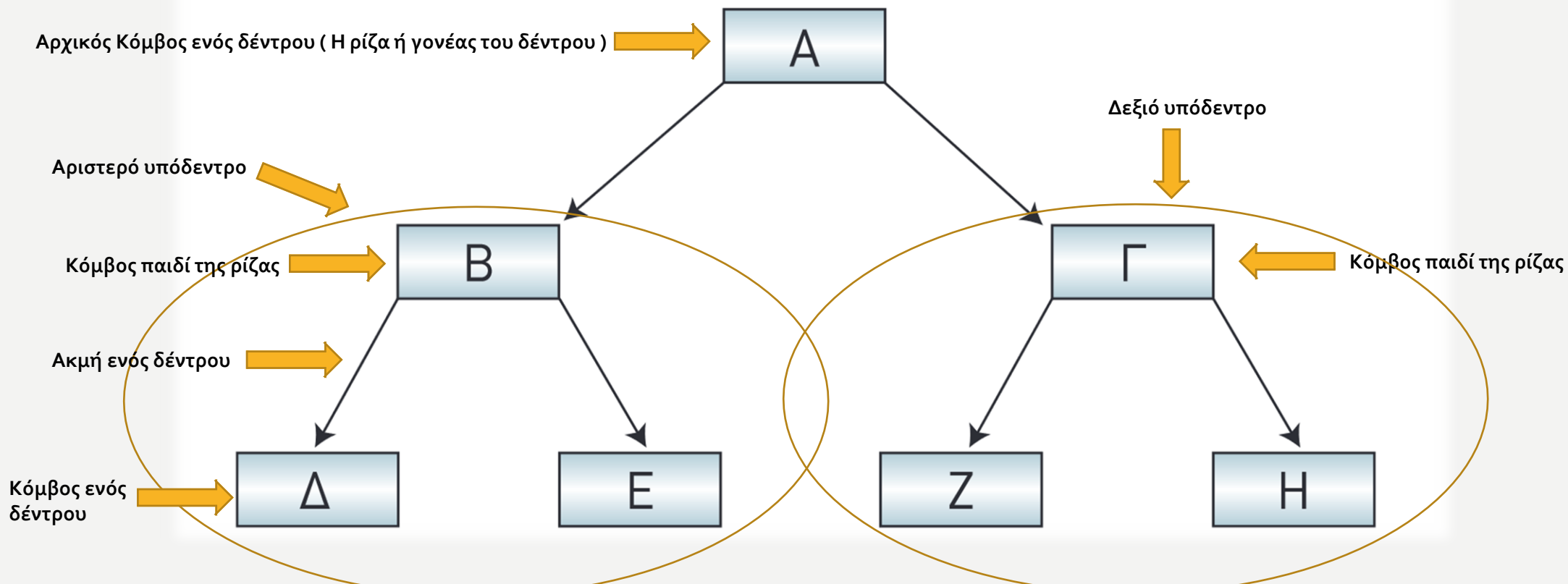
# ΔΙΑΓΡΑΦΗ ΣΕ ΛΙΣΤΑ



Αντίστοιχα για τη διαγραφή ενός κόμβου αρκεί να αλλάξει τιμή ο δείκτης του προηγούμενου κόμβου και να δείχνει πλέον τον επόμενο αυτού που διαγράφεται. Ο κόμβος που διαγράφηκε (ο τρίτος) αποτελεί "άχρηστο δεδομένο" και ο χώρος μνήμης που καταλάμβανε, παραχωρείται για άλλη χρήση.

# ΔΕΝΤΡΑ

Το κύριο χαρακτηριστικό των δένδρων είναι, ότι από έναν κόμβο δεν υπάρχει ένας μόνο επόμενος κόμβος, αλλά περισσότεροι. Υπάρχει ένας μόνο κόμβος, που λέγεται ρίζα, από τον οποίο ξεκινούν όλοι οι άλλοι κόμβοι.



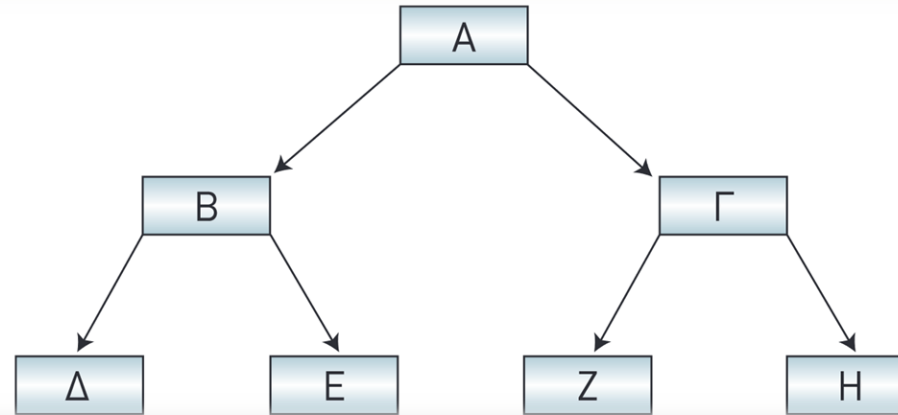
Τα δέντρα είναι μια από τις βασικές δομές δεδομένων που χρησιμοποιούνται στον προγραμματισμό.

Ένα δέντρο συνδυάζει τα πλεονεκτήματα δύο άλλων δομών:

- Ενός ταξινομημένου πίνακα
- Μιας συνδεδεμένης λίστας

# ΔΕΝΤΡΑ

Από τη ρίζα ξεκινούν δύο κόμβοι. Οι κόμβοι αυτοί λέγονται παιδιά της ρίζας. Με την ίδια λογική, από κάθε παιδί της ρίζας ξεκινούν άλλα παιδιά.



Μετακίνηση στους κόμβους του δέντρου μέσω των ακμών:

- Σε μία διαδρομή.
- Από τη ρίζα προς τα κάτω (από πάνω προς τα κάτω).

! Οι δομές δεδομένων που χρησιμοποιούν δείκτες αποκαλούνται **δυναμικές (dynamic)**, γιατί η υλοποίησή τους γίνεται έτσι, ώστε να μην απαιτείται εκ των προτέρων καθορισμός του μέγιστου αριθμού κόμβων. Είναι φανερό ότι οι δομές αυτές είναι πιο ευέλικτες από τη στατική δομή του πίνακα, επειδή επεκτείνονται και συρρικνώνονται κατά τη διάρκεια εκτέλεσης του προγράμματος.

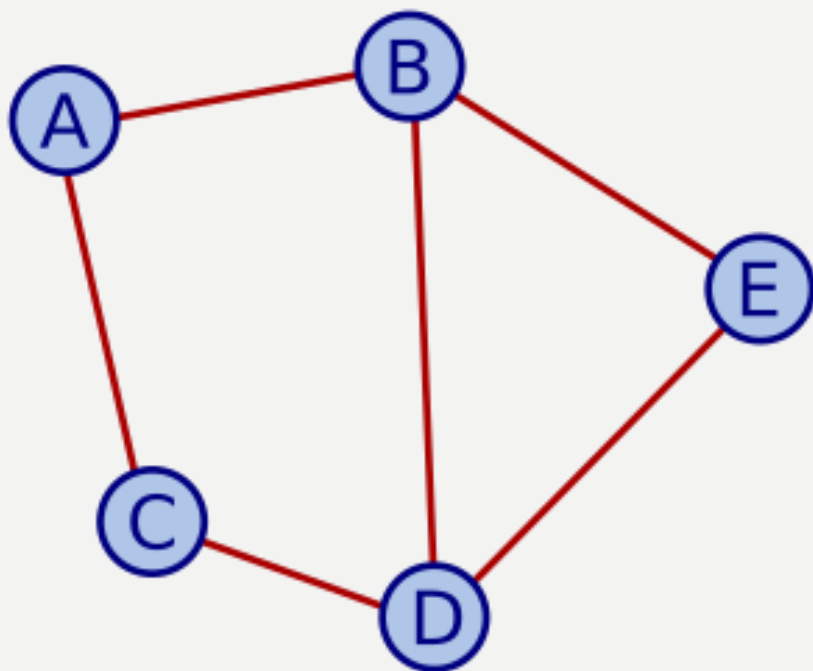


# ΓΡΑΦΟΙ

Ένας γράφος (**graph**) αποτελείται από **ένα σύνολο κόμβων** (ή σημείων ή κορυφών) και ένα σύνολο γραμμών (ή ακμών ή τόξων) που ενώνουν μερικούς ή όλους τους κόμβους.

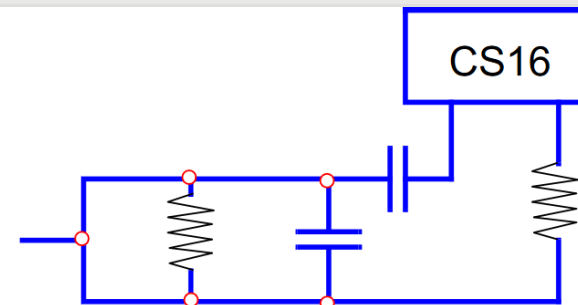
Ο γράφος αποτελεί την πιο γενική δομή δεδομένων, με την έννοια ότι όλες οι προηγούμενες δομές που παρουσιάστηκαν μπορούν να θεωρηθούν περιπτώσεις γράφων.

Πολλά προβλήματα και καταστάσεις της καθημερινής μας ζωής μπορούν να περιγραφούν με τη βοήθεια γράφων.

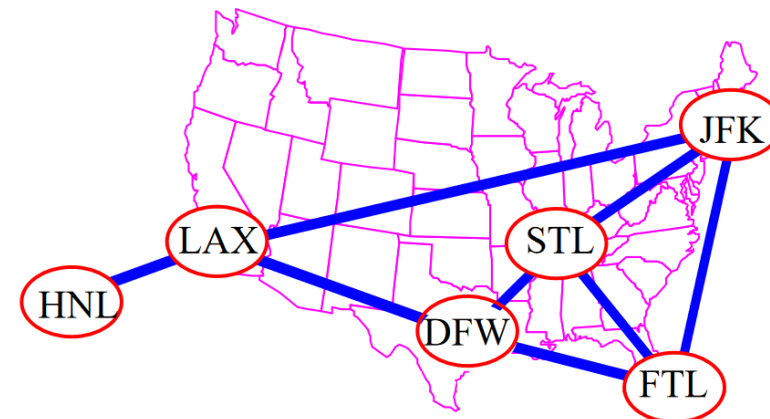


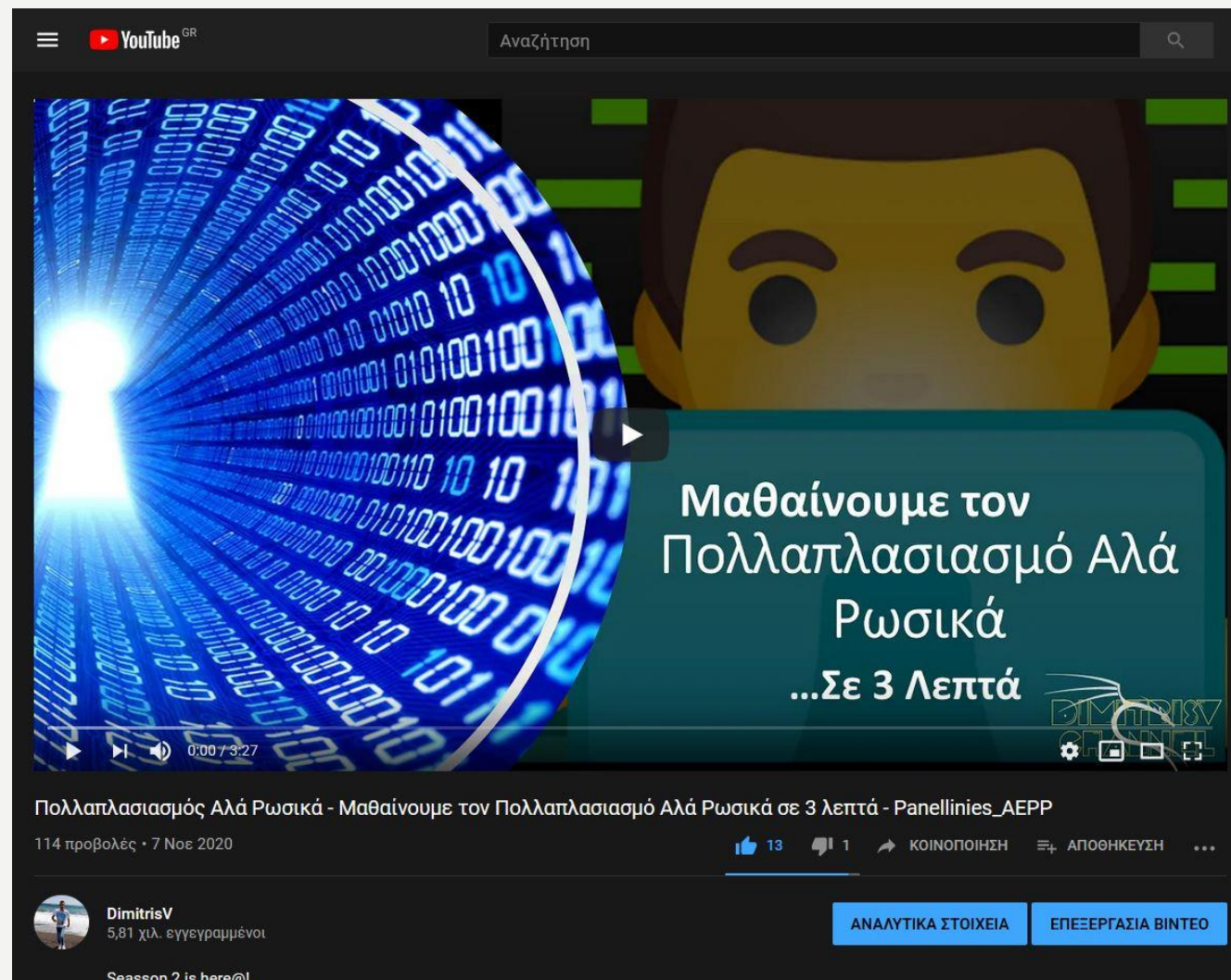
## Παραδείγματα:

Ηλεκτρονικά  
κυκλώματα



Δίκτυα (δρόμων,  
πτήσεων,  
επικοινωνιών)





YouTube GR Αναζήτηση

Μαθαίνουμε τον Πολλαπλασιασμό Αλά Ρωσικά ...Σε 3 Λεπτά

Πολλαπλασιασμός Αλά Ρωσικά - Μαθαίνουμε τον Πολλαπλασιασμό Αλά Ρωσικά σε 3 λεπτά - Panellinies\_AEPP

114 προβολές • 7 Νοε 2020

DimitrisV  
5,81 χιλ. εγγραφμένοι

Season 2 is here@!

ΑΝΑΛΥΤΙΚΑ ΣΤΟΙΧΕΙΑ ΕΠΕΞΕΡΓΑΣΙΑ ΒΙΝΤΕΟ

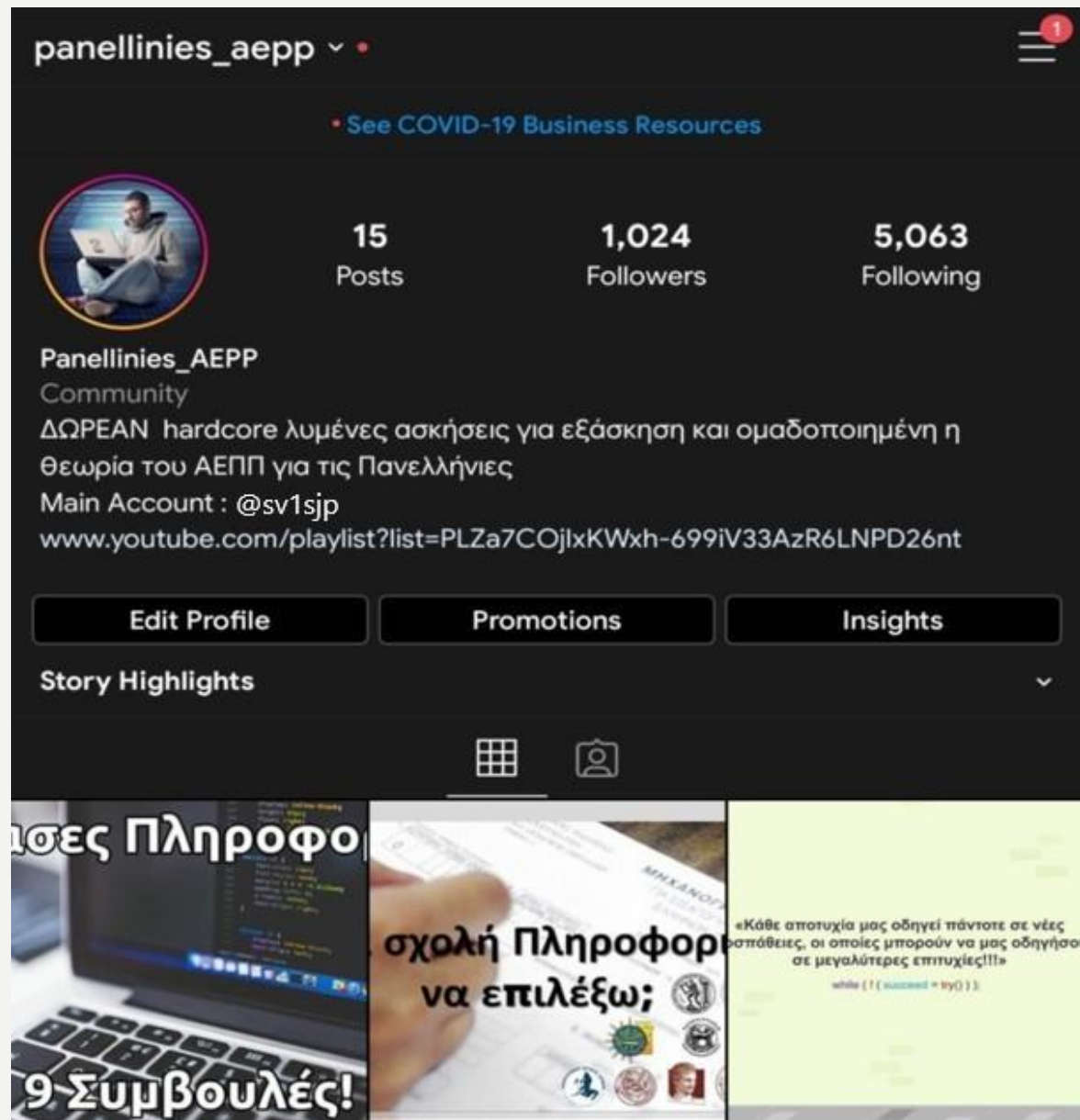


Δεν ξεχνάμε να διαβάσουμε και τον Πολλαπλασιασμό Αλά Ρωσικά !!!!!

<https://www.youtube.com/watch?v=y57gXCblpMA>

# ΑΠΟΡΙΕΣ;

Για οποιαδήποτε απορία ή διευκρίνηση ,στείλτε μήνυμα στην σελίδα μας στο Facebook και στο Instagram panellinies\_aepp!



Και μία εγγραφή στο κανάλι [DimitrisV](#) θα με βοηθήσει να συνεχίσω να παρέχω Hardcore λυμένες Ασκήσεις και βοηθητικές σημειώσεις για Πανελλήνιες & όχι μόνο !