ΠΙΝΑΚΑΣ – ΣΤΟΙΒΑ – ΟΥΡΑ

ΔΟΜΕΣ ΔΕΔΟΜΕΝΩΝ & ΑΛΓΟΡΙΘΜΟΙ
(ΚΕΦ. 3, ΠΑΡ. 3.3 - 3.5)

ΣΤΑΤΙΚΗ ΔΟΜΗ

- Το απαιτούμενο μέγεθος της κύριας μνήμης που χρειάζεται να δεσμευτεί για την αποθήκευση των δεδομένων της δομής, καθορίζεται κατά τη διάρκεια της **δημιουργίας** του προγράμματος κι όχι κατά τη διάρκεια της εκτέλεσής του.
- Τα στοιχεία των στατικών δομών αποθηκεύονται σε **συνεχόμενες** θέσεις μνήμης.

ΠΙΝΑΚΑΣ

- Η δομή του «πίνακα» υποστηρίζεται σχεδόν απ' όλες τις γλώσσες προγραμματισμού.
- Περιέχει στοιχεία ίδιου τύπου (π.χ. ακέραιους, πραγματικούς, κλπ).
- Η αναφορά στα στοιχεία ενός πίνακα, γίνεται γενικά με τη χρήση του συμβολικού ονόματός του, ακολουθούμενο από την τιμή ενός ή περισσοτέρων δεικτών, μέσα σε παρένθεση ή αγγύλη (π.χ. A[9], ή A[1,2], B[3,1], κλπ)

ΠΙΝΑΚΑΣ

- Ο πίνακας μπορεί να είναι μονοδιάστατος, δισδιάστατος και γενικά πολυδιάστατος.
- Όταν στους δισδιάστατους πίνακες το μέγεθος των δύο διαστάσεων είναι ίδιο, τότε ο πίνακας λέγεται τετραγωνικός (square) και συμβολίζεται ως πίνακας “n x n”.

ΠΙΝΑΚΑΣ Α:4 Χ 3		
A[1,1]	A[1,2]	A[1,3]
A[2,1]	A[2,2]	A[3,2]
A[3,1]	A[3,2]	A[3,3]
A[4,1]	A[4,2]	A[4,3]

ΠΑΡΑΔΕΙΓΜΑ 1

Δίνεται ένας μονοδιάστατος πίνακας (table) 100 στοιχείων. Να σχεδιασθεί αλγόριθμος που να βρίσκει το μεγαλύτερο στοιχείο του.

Αλγόριθμος Max_Table
Δεδομένα // table //
max ← table[1]
Για i **από** 2 **μέχρι** 100
    Αν table[i] > max **τότε**
    max ← table[i]
Τέλος_επανάληψης
Αποτελέσματα // max //
Τέλος Max_Table

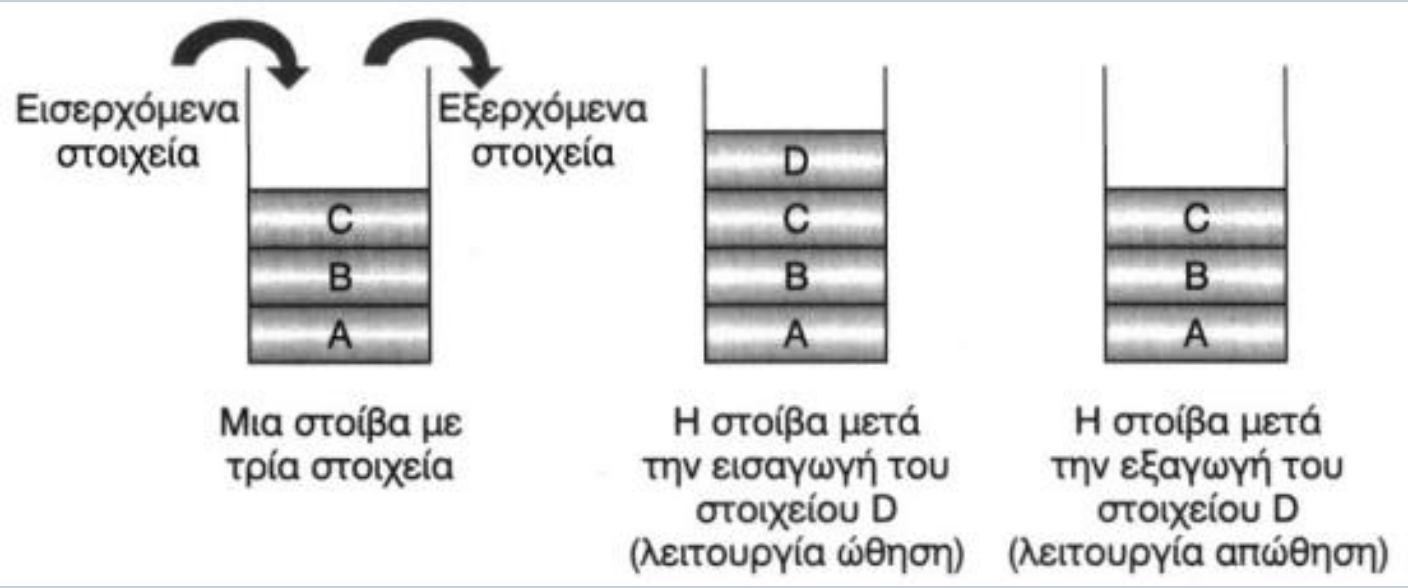
ΠΑΡΑΔΕΙΓΜΑ 2

Να γραφεί ένας αλγόριθμος που να διαβάζει και να αθροίζει τα στοιχεία δύο δυσδιάστατων πινάκων 5 Χ 5, καταχωρώντας το αποτέλεσμα σ' ένα νέο πίνακα.

Αλγόριθμος Sum_Table
Δεδομένα // A, B //
Για i **από** 1 **μέχρι** 5
    Για j **από** 1 **μέχρι** 5
    C[i,j] ← A[i,j] + B[I,j]
    Τέλος_επανάληψης
Τέλος_επανάληψης
Αποτελέσματα // C //
Τέλος Sum_Table

ΣΤΟΙΒΑ (Stack)

- Η **στοίβα(stack)** είναι μια δομή δεδομένων που χρησιμοποιεί πίνακα (μονοδιάστατο) για την υλοποίησή της.
- Για την επεξεργασία της χρησιμοποιήται η τεχνική **LIFO: Last In – First Out**.



ΣΤΟΙΒΑ (Stack)

- Η εισαγωγή στοιχείων στη στοίβα ονομάζεται ***ώθηση*** *(push)* και η εξαγωγή ***απώθηση*** *(pull ή pop)*.
- Κατά την εισαγωγή στοιχείων θα πρέπει να ελέγχεται αν η στοίβα είναι γεμάτη (*υπερχείλιση - overflow*) και κατά την εξαγωγή αν είναι άδεια (*υποχείλιση – underflow*)
- Η μεταβλητή "*top*" δείχνει το τελευταίο στοιχείο.

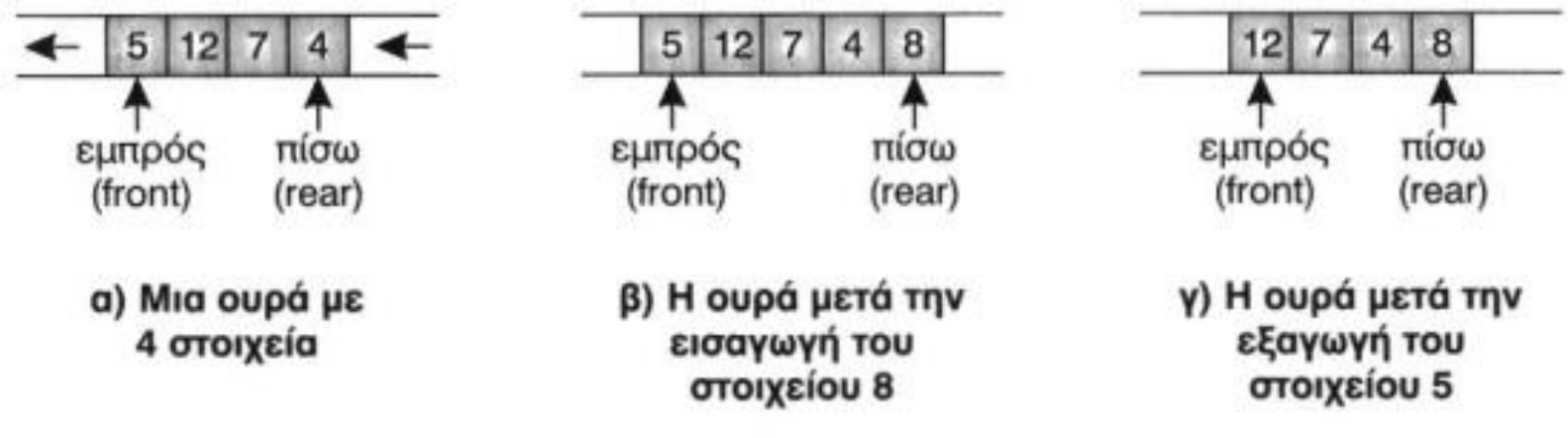
Αλγόριθμος stack_push
Δεδομένα // item, top, size //
Αν top < size **τότε**
    top ← top + 1
    stack[top] ← item
    done ← Αληθής
αλλιώς
    done ← Ψευδής
Τέλος_αν
Αποτελέσματα // top, done //
Τέλος stack_push

Ουρά (Queue)

- Θεωρία Ουρών (*Queueing Theory*), ειδικός κλάδος των μαθηματικών.
- Υλοποίηση: μονοδιάστατοι πίνακες.
- FIFO (*First-In-First-Out*).
- Οι κύριες λειτουργίες στην ουρά είναι δύο:
  - **Εισαγωγή** (*enqueue*) και
  - **Εξαγωγή** (*dequeue*)

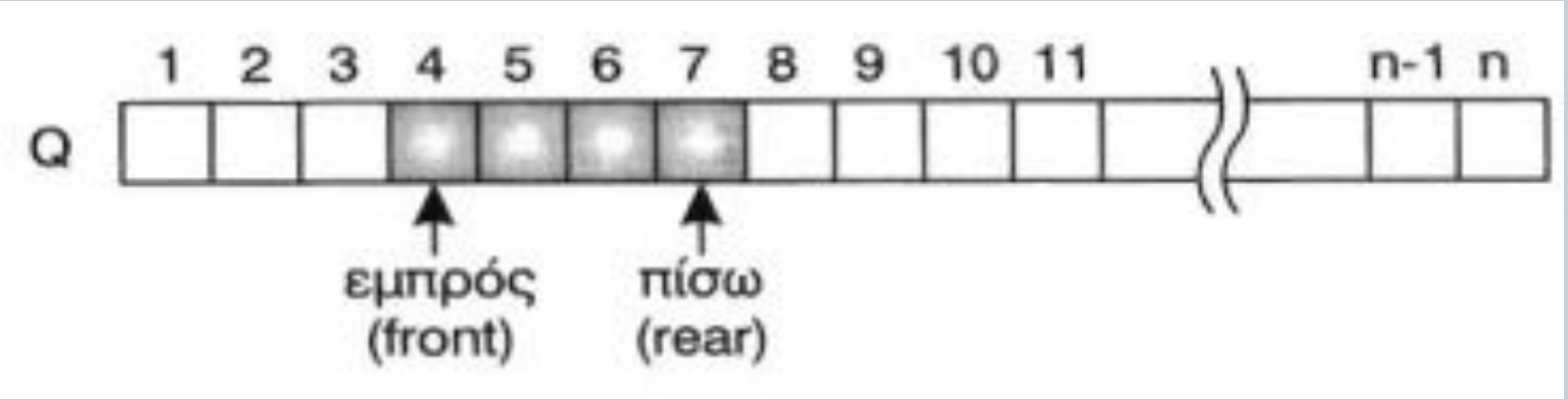
Ουρά (Queue)

- Στην περίπτωση της ουράς χρειάζονται δύο δείκτες για τον προσδιορισμό της:
  - **Εμπρός** (front): δείχνει το στοιχείο που εισήλθε και πρόκειται να εξαχθεί πρώτο.
  - **Πίσω** (rear): δείχνει το στοιχείο που μόλις έχει εισαχθεί στην ουρά (τελευταίο).



Ουρά (Queue)

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.



Ουρά (Queue)

enqueue

```
1.  Αλγόριθμος en_queue
2.  Δεδομένα // rear, size, item //
3.  Αν rear < size τότε
4.          rear ← rear + 1
5.          queue[rear] ← item
6.          done ← Αληθής
7.  αλλιώς
8.          done ← Ψευδής
9.  Τέλος_αν
10. Αποτελέσματα // rear, done //
11. Τέλος en_queue
```

dequeue

```
1.  Αλγόριθμος de_Queue
2.  Δεδομένα // rear, front, queue//
3.  Αν front <= rear τότε
4.          item  ← queue[front]
5.          front ← front + 1
6.          done ← Αληθής
7.  αλλιώς
8.          done ← Ψευδής
9.          item  ← null
10. Τέλος_αν
11. Αποτελέσματα // item, rear, done //
12. Τέλος de_Queue
```

ΑΕΠΠ

ΠΙΝΑΚΑΣ – ΣΤΟΙΒΑ – ΟΥΡΑ

ΚΕΦ. 3, ΠΑΡ. 3.3 – 3.5.