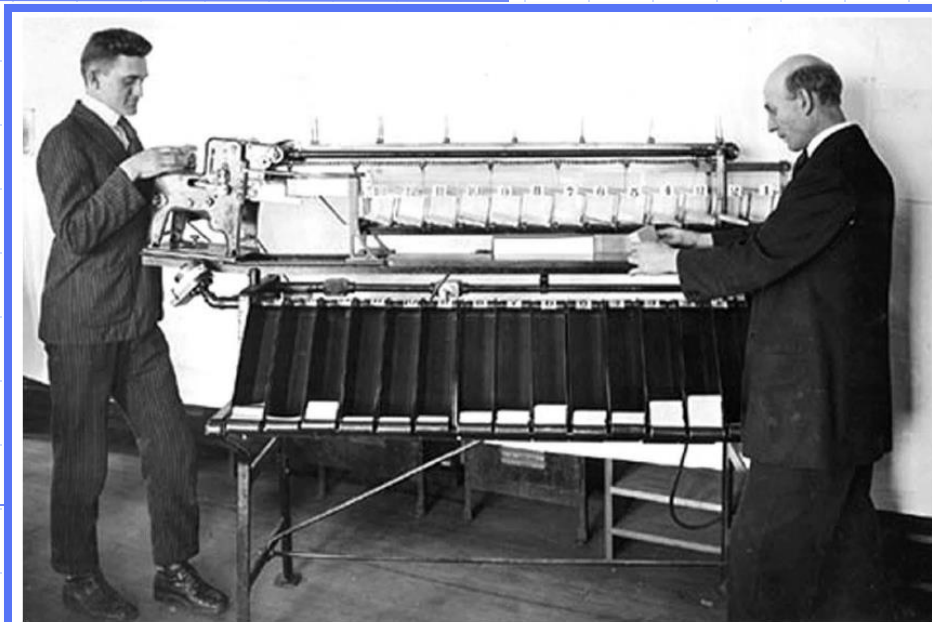


Παρουσίαση για χρήση με το σύγγραμμα, **Αλγόριθμοι Σχεδίαση και Εφαρμογές**, των Μ. Τ. Goodrich and R. Tamassia, Wiley, 2015 (στα ελληνικά από εκδόσεις Μ. Γκιούρδας)

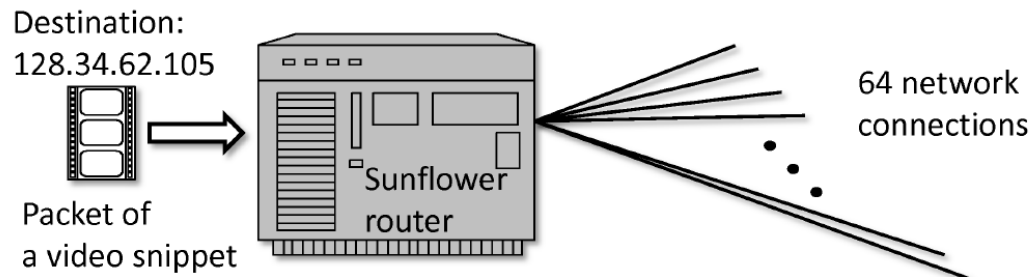
Πίνακες αντιστοίχισης (χάρτες - maps)



Operating a card sorter, 1920. U.S. Census Bureau.

Εφαρμογή: Δρομολογητές δικτύου

- Οι δρομολογητές δικτύου επεξεργάζονται πακέτα πληροφοριών από πολλές συνδέσεις σε υψηλή ταχύτητα.
- Για την επεξεργασία ενός πακέτου, (k,x) , όπου k είναι το κλειδί για τον προορισμό και x τα δεδομένα που περιέχει, ένας δρομολογητής πρέπει πολύ γρήγορα να αποφασίσει σε ποια από τις συνδέσεις του δικτύου να στείλει το πακέτο.
- Ένα τέτοιο σύστημα πρέπει να υποστηρίζει αναζητήσεις βάση κλειδιού, δλδ, λειτουργίες $get(k)$, καθώς και λειτουργίες $put(k,c)$ για την προσθήκη μίας νέας σύνδεσης δικτύου, c , για κλειδί προορισμού, k .
- Ιδανικά θέλουμε να επιτύχουμε $O(1)$ χρόνο τόσο για τη λειτουργία get όσο και για τη λειτουργία put .



Πίνακες αντιστοίχισης

- Ένας πίνακας αντιστοίχισης είναι μία συλλογή τιμών με δυνατότητα αναζήτησης για εγγραφές της μορφής κλειδί-τιμή.
- Οι κύριες λειτουργίες ενός πίνακα αντιστοίχισης είναι η αναζήτηση, η εισαγωγή, και η διαγραφή στοιχείων.
- Πολλαπλές εγγραφές με το ίδιο κλειδί **δεν** επιτρέπονται.
- Άλλες εφαρμογές:
 - Βιβλίο διευθύνσεων.
 - Βάση δεδομένων εγγραφών σπουδαστών.



Λειτουργίες πίνακα αντιστοίχισης

- **get(k)**: Αν ο πίνακας αντιστοίχισης M περιέχει ένα στοιχείο με κλειδί ίσο με k επιστρέφει την τιμή του, αλλιώς επιστρέφει `null`.
- **put(k, v)**: εισαγωγή στοιχείου (k, v) στον πίνακα αντιστοίχισης M , εάν το κλειδί k δεν υπάρχει στον M επιστρέφει `null`, αλλιώς επιστρέφει την προηγούμενη τιμή στην οποία αντιστοιχούσε το k .
- **remove(k)**: εάν ο πίνακας αντιστοίχισης M έχει ένα στοιχείο με κλειδί k , το αφαιρεί από τον M και επιστρέφει την τιμή του, αλλιώς επιστρέφει `null`.
- **size(), isEmpty()**.

Παράδειγμα

Λειτουργία

isEmpty()
put(5,A)
put(7,B)
put(2,C)
put(8,D)
put(2,E)
get(7)
get(4)
get(2)
size()
remove(5)
remove(2)
get(2)
isEmpty()

Έξοδος

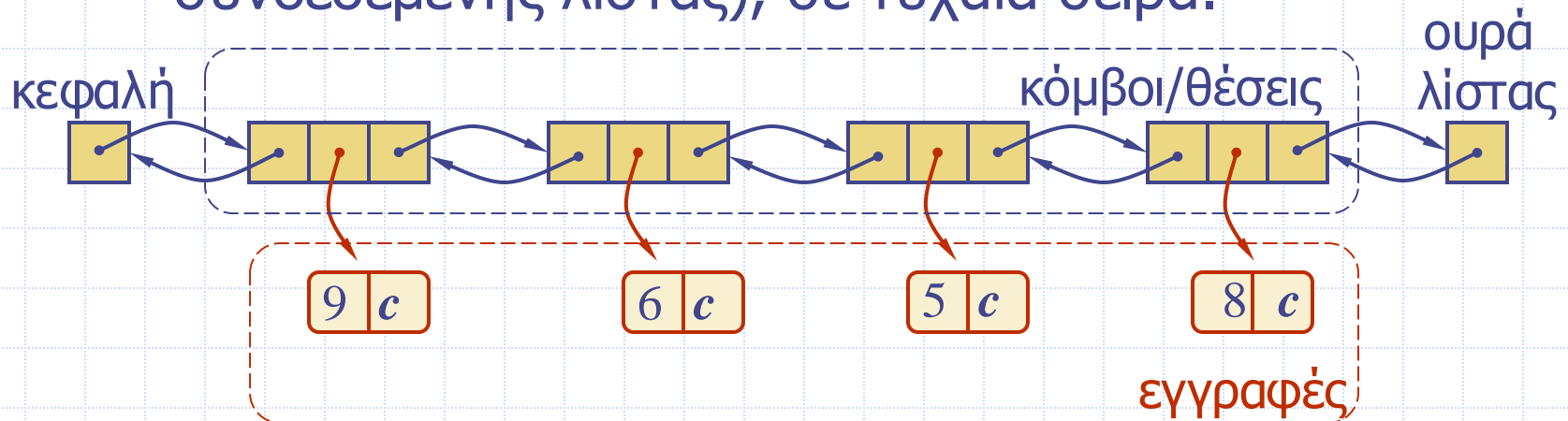
true
null
null
null
C
B
null
E
4
A
E
null
false

Πίνακας αντιστοίχισης (map)

∅
(5,A)
(5,A),(7,B)
(5,A),(7,B),(2,C)
(5,A),(7,B),(2,C),(8,D)
(5,A),(7,B),(2,E),(8,D)
(5,A),(7,B),(2,E),(8,D)
(5,A),(7,B),(2,E),(8,D)
(5,A),(7,B),(2,E),(8,D)
(5,A),(7,B),(2,E),(8,D)
(7,B),(2,E),(8,D)
(7,B),(8,D)
(7,B),(8,D)
(7,B),(8,D)

Ένας απλός πίνακας αντιστοίχισης βασισμένος σε λίστα

- Μπορούμε να υλοποιήσουμε έναν πίνακα αντιστοίχισης με μία αταξινόμητη λίστα.
 - Αποθηκεύουμε τα στοιχεία του πίνακα αντιστοίχισης σε μία λίστα S (βάσει μίας διπλά συνδεδεμένης λίστας), σε τυχαία σειρά.



Απόδοση πίνακα αντιστοίχισης που είναι βασισμένος σε λίστα

□ Απόδοση:

- Η **put** απαιτεί χρόνο $O(1)$ αφού μπορούμε να προσθέσουμε ένα νέο στοιχείο στην αρχή ή στο τέλος της ακολουθίας.
 - Η **get** και η **remove** απαιτούν χρόνο $O(n)$ αφού στην χειρότερη περίπτωση (όπου το στοιχείο δεν υπάρχει) θα διασχίσουν ολόκληρη την ακολουθία ψάχνοντας ένα στοιχείο με το δεδομένο κλειδί.
- Η υλοποίηση βάσει μη ταξινομημένης λίστας είναι κατάλληλη μόνο για πίνακες αντιστοίχισης μικρού μεγέθους ή για πίνακες αντιστοίχισης όπου η συνήθης λειτουργία είναι η **put** ενώ οι αναζητήσεις (**get**) και οι διαγραφές (**remove**) είναι σπάνιες (π.χ. ιστορικό καταγραφής συνδέσεων χρηστών σε έναν Η/Υ).