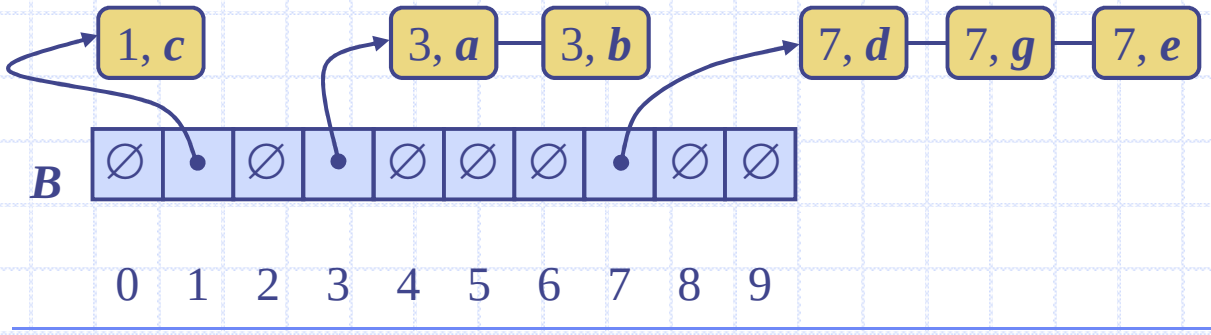
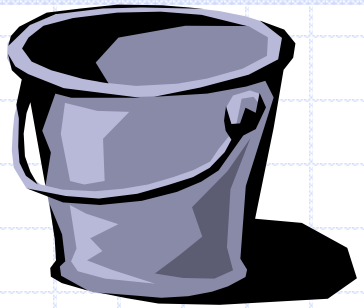
Ταξινόμηση Κάδου και Ταξινόμηση Βάσης



Ταξινόμηση Κάδου



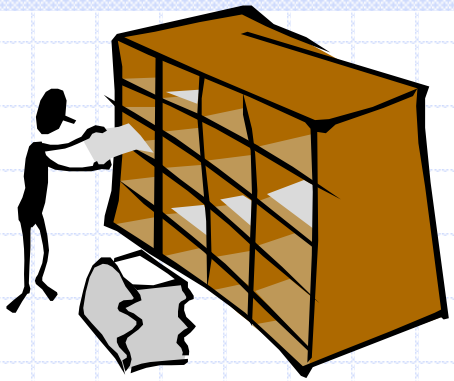
- Έστω S μια ακολουθία από n καταχωρήσεις (κλειδί, στοιχείο) με κλειδιά στο διάστημα $[0, N-1]$
- Η ταξινόμηση κάδου χρησιμοποιεί τα κλειδιά σαν δείκτες σε ένα βοηθητικό πίνακα B ακολουθιών (κάδων)
  - Φάση 1: Αδειάζει η ακολουθία S με μετακίνηση της καταχώρησης (k, o) στον κάδο $B[k]$
  - Φάση 2: Για $i = 0, \ldots, N-1$, μετακίνηση των καταχωρήσεων του κάδου $B[i]$ στο τέλος της ακολουθίας S
- Ανάλυση:
  - Η φάση 1 απαιτεί $O(n)$ χρόνο
  - Η φάση 2 απαιτεί $O(n + N)$ χρόνο

  Η ταξινόμηση κάδου απαιτεί $O(n + N)$ χρόνο

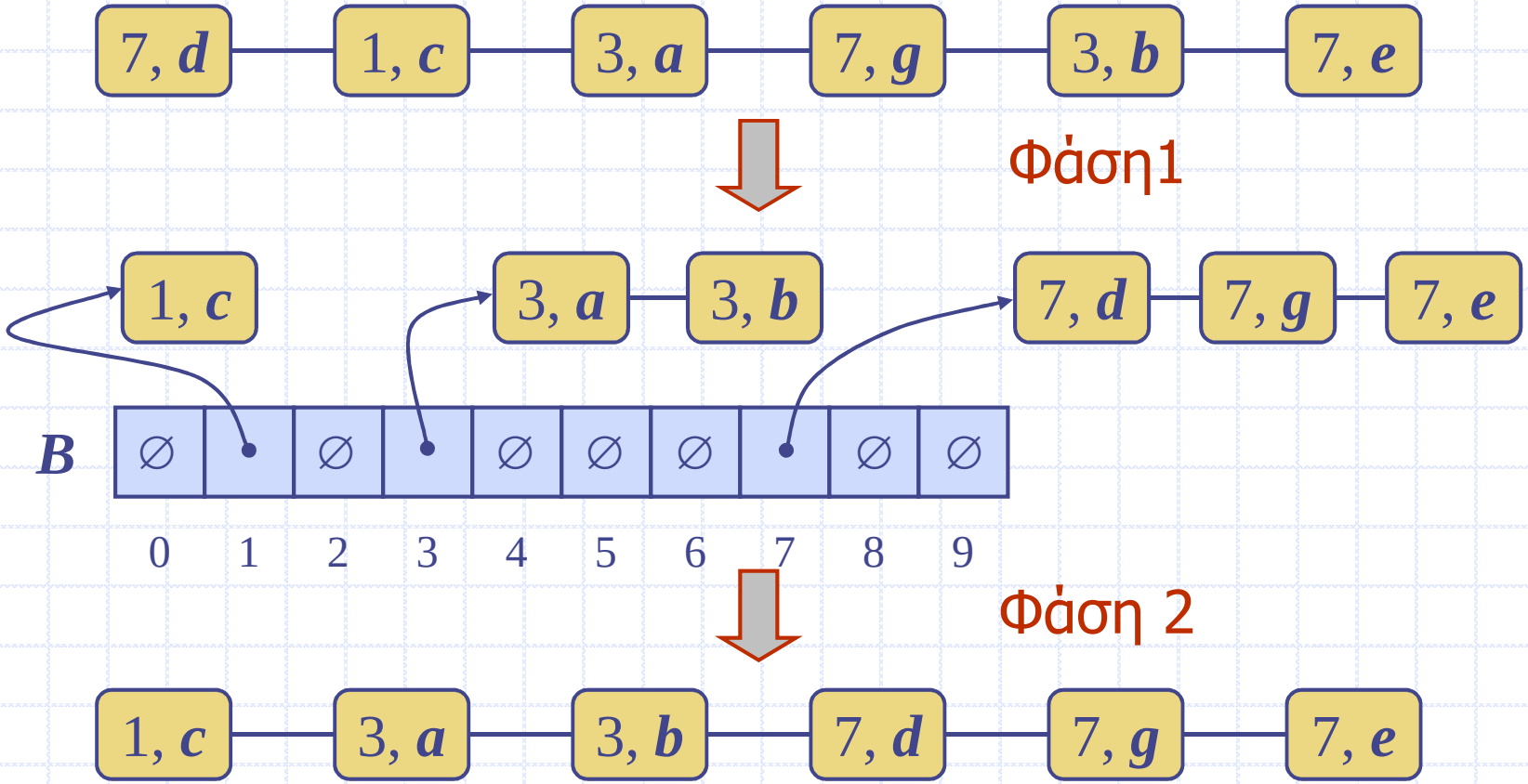
Algorithm *bucketSort*(S, N)

```
Input sequence S of (key, element) items with keys in the range [0, N − 1]
Output sequence S sorted by increasing keys
B ← array of N empty sequences
while ¬S.isEmpty()
    f ← S.first()
    (k, o) ← S.remove(f)
    B[k].addLast((k, o))
for i ← 0 to N − 1
    while ¬B[i].isEmpty()
        f ← B[i].first()
        (k, o) ← B[i].remove(f)
        S.addLast((k, o))
```

Παράδειγμα



- Διάστημα Κλειδιού $[0, 9]$



Ιδιότητες και Επεκτάσεις

- Ιδιότητα τύπου του κλειδιού
  - Τα κλειδιά χρησιμοποιούνται σαν δείκτες σε ένα πίνακα και δεν μπορούν να είναι οποιαδήποτε αντικείμενα
  - Δεν υπάρχει εξωτερικός συγκριτής
- Σταθερή μέθοδος ταξινόμησης
  - Η σχετική σειρά οποιονδήποτε δυο στοιχείων με ίδιο κλειδί διατηρείται μετά την εκτέλεση του αλγόριθμου

Επεκτάσεις

- Ακέραια κλειδιά στο διάστημα $[a, b]$
  - Θέσε την καταχώρηση (k, o) στον κάδο $B[k - a]$
- Κλειδιά συμβολοσειρές από ένα σύνολο D πιθανών συμβολοσειρών, όπου το D σταθερό μέγεθος
  - Ταξινόμηση του D και υπολογισμός της θέσης $r(k)$ κάθε συμβολοσειράς k του D στην ταξινομημένη ακολουθία
  - Θέσε την καταχώρηση (k, o) στον κάδο $B[r(k)]$

Λεξικογραφική Διάταξη

- Μια d-πλειάδα είναι μια ακολουθία από d κλειδιά $(k_1, k_2, \ldots, k_d)$, όπου λέμε ότι το κλειδί k_i είναι η i διάσταση της πλειάδας
- Παράδειγμα:
  - Οι Καρτεσιανές συντεταγμένες ενός σημείου στο χώρο είναι μια 3-πλειάδα
- Η λεξικογραφική διάταξη δυο d-πλειάδων ορίζεται αναδρομικά σαν

$$(x_1, x_2, \ldots, x_d) < (y_1, y_2, \ldots, y_d)$$

$$\Leftrightarrow$$

$$x_1 < y_1 \vee x_1 = y_1 \wedge (x_2, \ldots, x_d) < (y_2, \ldots, y_d)$$

Δηλ., οι πλειάδες συγκρίνονται στην πρώτη διάσταση, στη συνέχεια στη δεύτερη, κοκ.

Λεξικογραφική Ταξινόμηση

- Έστω C_i ο συγκριτής που συγκρίνει δύο πλειάδες στην i διάστασή τους
- Έστω ***stableSort***(S, C) ένας σταθερός αλγόριθμος ταξινόμησης που χρησιμοποιεί τον συγκριτή C
- Η λεξικογραφική ταξινόμηση ταξινομεί μια ακολουθία από d-πλειάδες σε λεξικογραφική σειρά εκτελώντας d φορές τον αλγόριθμο ***stableSort***, μια για κάθε διάσταση
- Η λεξικογραφική ταξινόμηση τρέχει σε χρόνο $O(dT(n))$, όπου $T(n)$ είναι ο χρόνος τρεξίματος της ***stableSort***

Algorithm ***lexicographicSort***(S)
  Input sequence S of d-tuples
  Output sequence S sorted in lexicographic order

  for $i \leftarrow d$ **downto** 1
    stableSort(S, C_i)

Παράδειγμα:

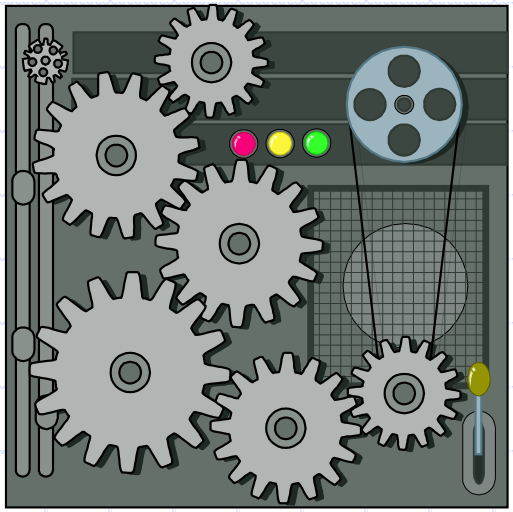
(7,4,6) (5,1,5) (2,4,6) (2, 1, 4) (3, 2, 4)

(2, 1, 4) (3, 2, 4) (5,1,5) (7,4,6) (2,4,6)

(2, 1, 4) (5,1,5) (3, 2, 4) (7,4,6) (2,4,6)

(2, 1, 4) (2,4,6) (3, 2, 4) (5,1,5) (7,4,6)

Ταξινόμηση Βάσης



- Η ταξινόμηση βάσης είναι μια εξειδίκευση της λεξικογραφικής ταξινόμησης που χρησιμοποιεί την ταξινόμηση κάδου σαν σταθερό αλγόριθμο ταξινόμησης σε κάθε διάσταση
- Η ταξινόμηση βάσης εφαρμόζεται σε πλειάδες όπου τα κλειδιά σε κάθε διάσταση i είναι ακέραιοι στο διάστημα $[0, N-1]$
- Η ταξινόμηση βάσης τρέχει σε χρόνο $O(d(n+N))$

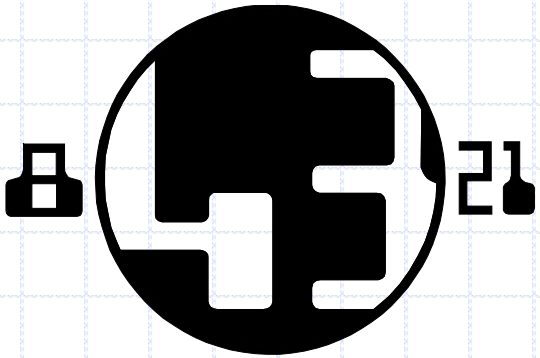
Algorithm *radixSort*(S, N)
  Input sequence S of d-tuples such that $(0, \ldots, 0) \leq (x_1, \ldots, x_d)$ and $(x_1, \ldots, x_d) \leq (N-1, \ldots, N-1)$ for each tuple $(x_1, \ldots, x_d)$ in S
  Output sequence S sorted in lexicographic order
  for $i \leftarrow d$ **downto** 1
    bucketSort(S, N)

Ταξινόμηση Βάσης για Δυαδικούς Αριθμούς

- Έστω μια ακολουθία από ακέραιους n b-bit

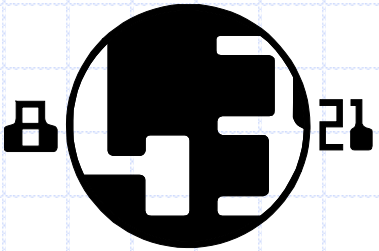
$$x = x_{b-1} \ldots x_1 x_0$$

- Παριστάνουμε κάθε στοιχείο σαν μια b-πλειάδα από ακεραίους στο διάστημα $[0, 1]$ και εφαρμόζουμε ταξινόμηση βάσης με $N = 2$
- Αυτή η εφαρμογή του αλγόριθμου της ταξινόμησης βάσης τρέχει σε χρόνο $O(bn)$
- Για παράδειγμα, μπορούμε να ταξινομήσουμε μια ακολουθία από ακέραιους 32-bit σε γραμμικό χρόνο



Algorithm *binaryRadixSort*(*S*)
  Input sequence *S* of *b*-bit integers
  Output sequence *S* sorted
  replace each element *x* of *S* with the item (0, *x*)
  for *i* ← 0 **to** *b* – 1
    replace the key *k* of each item (*k*, *x*) of *S* with bit x_i of *x*
    bucketSort(*S*, 2)

Παράδειγμα



- Ταξινόμηση μιας ακολουθία από ακέραιους 4-bit