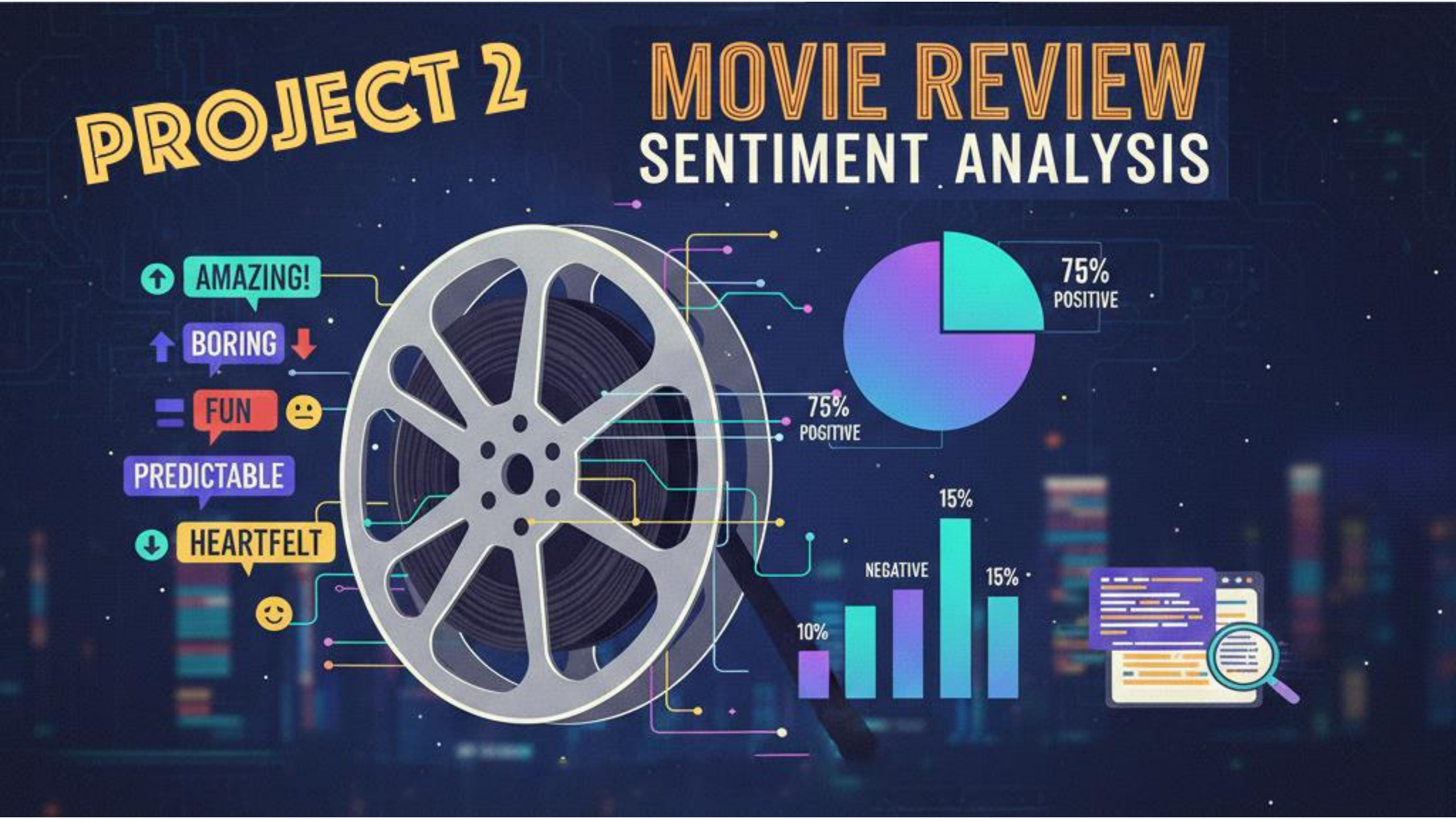
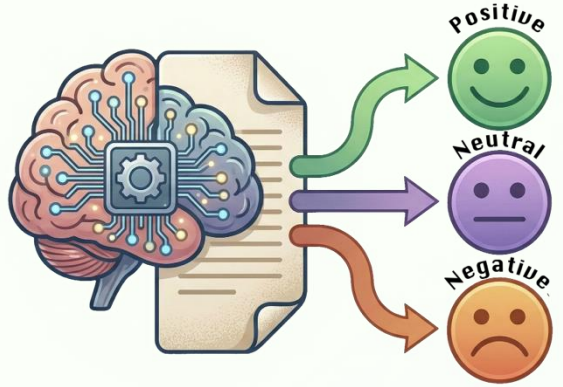
PROJECT 2

MOVIE REVIEW SENTIMENT ANALYSIS



Αντικείμενο της εργασίας

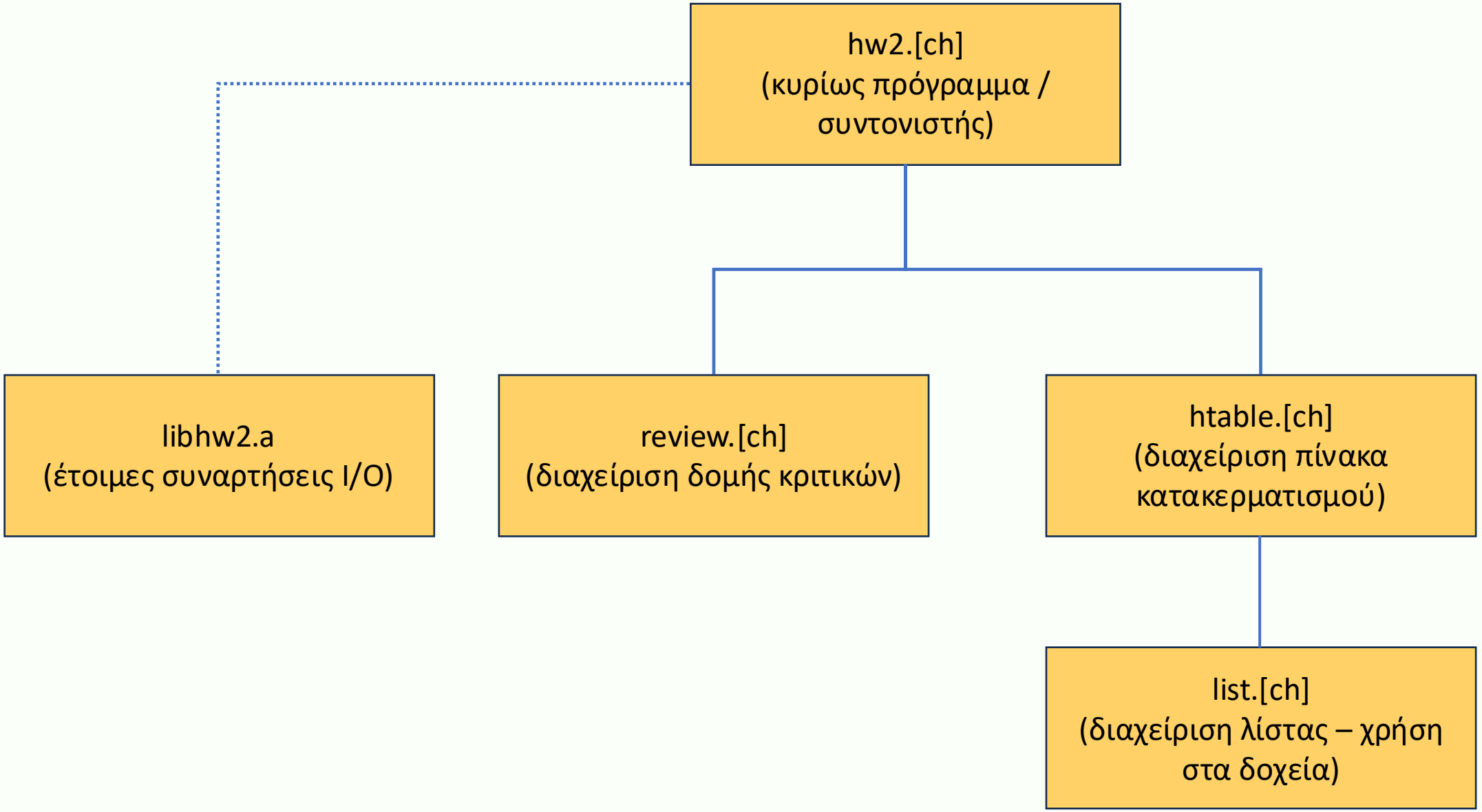
- Δημιουργία και διαχείριση μιας βάσης δεδομένων για κριτικές ταινιών
  - Κριτική -> Τίτλος ταινίας, όνομα κριτικού, κείμενο, βαθμολογία
  - Ενέργειες -> Εισαγωγή, διαγραφή, ανάκτηση, εκτύπωση
- Αυτόματη εκτίμηση νέας κριτικής που δεν έχει βαθμολογία ως θετική, αρνητική ή ουδέτερη.
  - Χρησιμοποιώντας τις υπάρχουσες βαθμολογίες με (κάπως) έξυπνο τρόπο



Στοιχεία υλοποίησης

- Δυναμική δέσμευση μνήμης
  - Στοιχεία κριτικών
  - Δομή αποθήκευσης κριτικών
  - Λίστα
  - Πίνακας κατακερματισμού
- Δομημένη ανάπτυξη με διαχωρισμό σε ενότητες
- Ξεχωριστή μεταγλώττιση και διασύνδεση με χρήση Makefile

Στοιχεία υλοποίησης



Κριτική

- Τίτλος ταινίας (δυναμικά δεσμευμένη συμβολοσειρά)
- Κείμενο κριτικής (δυναμικά δεσμευμένη συμβολοσειρά)
  - Όλη σε μία γραμμή
  - Οι λέξεις διαχωρίζονται με κενά
  - Τα σημεία στίξης έχουν αφαιρεθεί.
- Όνομα κριτικού (δυναμικά δεσμευμένη συμβολοσειρά)
- Βαθμολογία ταινίας (δεκαδικός αριθμός από 0 έως και 4)
- → Ορίστε κατάλληλο struct!

Ανάγνωση

- Συνάρτηση read_line (δίνεται έτοιμη)
  - 1η παράμετρος: Διεύθυνση στην οποία θα αποθηκευτεί μία συμβολοσειρά
  - 2η παράμετρος:
    - NULL για να διαβάσει από τη συμβατική είσοδο, ή
    - Το όνομα ενός αρχείου για να διαβάσει την επόμενη γραμμή του αρχείου

```
char *text;
read_line(&text, NULL);
...
free(text);
```

Διαβάζει ολόκληρη γραμμή

Αφαιρεί κενά από την αρχή και το τέλος της γραμμής

Δεσμεύει δυναμικά μνήμη κι αποθηκεύει τη διεύθυνση στο text…

… επομένως πρέπει να αποδεσμεύσετε τη μνήμη όταν δεν τη χρειάζεστε πια.

Δομή κριτικών

- Οργάνωση:
  - Δυναμικά δεσμευμένη
  - Μέγεθος ανάλογο του πλήθους των αποθηκευμένων κριτικών
  - Λίστα? Δυναμικός πίνακας? Διαλέξτε!
- Λειτουργία: Προσθήκη νέας κριτικής
  - Δε χρειάζεται έλεγχος διπλότυπων

Δομή κριτικών

- Λειτουργία: Εκτύπωση με βάση κάποιο κριτήριο (string)
  - Κριτήριο: "*"
    - Εκτυπώνονται όλες οι κριτικές
  - Κριτήριο: Άλλη συμβολοσειρά
    - Εκτυπώνονται οι κριτικές των ταινιών που έχουν ως τίτλο αυτή τη συμβολοσειρά.
    - Χρησιμοποιήστε strcasecmp (δε γίνεται διάκριση κεφαλαίων/μικρών)
  - Η σειρά εκτύπωσης πολλαπλών αποτελεσμάτων για το ίδιο κριτήριο εξαρτάται από την υλοποίηση σας.

Δομή κριτικών

- Λειτουργία: Αφαίρεση όλων των κριτικών με κριτήριο το όνομα του ατόμου που τις έγραψε.
  - ΠΡΟΣΟΧΗ: Θα δούμε αργότερα πως πριν διαγραφεί μια κριτική πρέπει να εξετάσουμε τις λέξεις από τις οποίες αποτελείται και να ανανεώσουμε μια δομή αποθήκευσης λέξεων που δεν έχει σχέση με τη δομή κριτικών.
  - ΕΠΟΜΕΝΩΣ: Η συνάρτηση αφαίρεσης πρέπει να επιστρέφει κάθε μία κριτική που αφαιρεί.
  - ΤΡΥΚ: Μπορούμε να "θυμόμαστε" πού είχαμε μείνει στην προηγούμενη επανάληψη; ΝΑΙ! Με χρήση static τοπικών μεταβλητών.

Checkpoint #1

- Υλοποιήστε ό,τι έχει σχέση με τη δομή κριτικών
  - review.h
    - Ορισμοί struct
      - Υποχρεωτικά ένα struct που αναπαριστά κριτική
      - Προαιρετικά βοηθητικό struct με πεδία που αφορούν τη δομή κριτικών (π.χ. πλήθος, διεύθυνση δεσμευμένης μνήμης για τη δομή).
    - Επικεφαλίδες συναρτήσεων για τις λειτουργίες πάνω στη δομή κριτικών
  - review.c
    - Υλοποιήσεις συναρτήσεων για τις λειτουργίες πάνω στη δομή κριτικών

Checkpoint #1

- Υλοποιήστε τα σχετικά κομμάτια στη main
  - hw2.h
    - Επικεφαλίδες βοηθητικών συναρτήσεων για τη main
  - hw2.c
    - Υλοποίηση main
      - Επιλογές που αφορούν τη δομή κριτικών.
    - Υλοποίηση βοηθητικών συναρτήσεων
- Γράψτε κατάλληλο Makefile

Αξιολόγηση κριτικών

- Τι γνωρίζουμε:
  - Τους βαθμούς των κριτικών που έχουν αποθηκευτεί στη δομή κριτικών
- Τι θέλουμε:
  - Να αξιολογήσουμε μια νέα κριτική η οποία δεν έχει βαθμό.
- Ιδέα:
  - Λέξεις με "θετική" σημασία θα εμφανίζονται κυρίως σε κριτικές με μεγάλο βαθμό, ενώ λέξεις με "αρνητική" σημασία σε κριτικές με χαμηλό βαθμό.

Αξιολόγηση κριτικών

- Πράξη:
  - Για κάθε λέξη υπολογίζουμε το μέσο όρο των βαθμών των κριτικών στις οποίες εμφανίζεται.
  - Κάθε εμφάνιση λέξης αντιμετωπίζεται ξεχωριστά (δηλαδή μια λέξη που εμφανίζεται δύο φορές στην ίδια κριτική "μετράει" δύο φορές)
  - Για να αξιολογήσουμε μια νέα κριτική που δεν έχει βαθμό υπολογίζουμε το μέσο όρο των βαθμών των λέξεων που περιέχει
    - Αν δεν έχουμε βαθμό για μια λέξη, χρησιμοποιούμε το 2.0 ("ουδέτερο")

Παράδειγμα

- Κείμενα και βαθμοί κριτικών που είναι αποθηκευμένες στη βάση:
  - "An epic sci-fi movie" με βαθμό 4.0
  - "An average sci-fi drama " με βαθμό 2.5
  - "An amazing epic with an average cast" με βαθμό 3.0
- Βαθμοί που ανατίθενται στις επιμέρους λέξεις

an: (4.0 + 2.5 + 3.0 + 3.0) / 4 = 3.125
epic: (4.0 + 3.0) / 2 = 3.5
…

- Βαθμός νέας κριτικής:

"An overlong epic" : (3.125 + 2.0 + 3.5) / 3 = 2.875

Δομή αποθήκευσης λέξεων & βαθμών

- Πίνακας κατακερματισμού:
  - Δυναμικός πίνακας από δοχεία
  - Κάθε δοχείο υλοποιείται ως μία λίστα
    - Διπλά διασυνδεδεμένη, κυκλική, με τερματικό κόμβο
    - Ταξινομημένη λεξικογραφικά (ως προς τις λέξεις) σε αύξουσα σειρά
- Σημαντικό: Η υλοποίηση της λίστας πρέπει να γίνει ξεχωριστά, ανεξάρτητα από τον πίνακα κατακερματισμού.

Λίστα

- Κάθε κόμβος περιέχει (εκτός από τους δείκτες σύνδεσης):
  - Λέξη (δυναμικά δεσμευμένη συμβολοσειρά)
  - Πλήθος εμφανίσεων της λέξης
  - Άθροισμα βαθμών των κριτικών στις οποίες εμφανίζεται η λέξη
    - Γιατί όχι απευθείας το μέσο όρο;
- Λειτουργία: Δημιουργία άδειας λίστας
  - Τι σημαίνει "άδεια" για το συγκεκριμένο είδος λίστας;

Λίστα

- Λειτουργία: Προσθήκη λέξης (με βαθμό)
  - Αν η λέξη υπάρχει ήδη, απλά θα ανανεωθούν το άθροισμα + οι εμφανίσεις
  - Αν δεν υπάρχει, πρέπει να προστεθεί νέος κόμβος
    - Στη σωστή σειρά!
  - Τι πρέπει να επιστρέφει η συνάρτηση προσθήκης?
    - Ένδειξη για το αν προστέθηκε νέος κόμβος ή ανανεώθηκε ένας ήδη υπάρχων! Γιατί???

Λίστα

- Λειτουργία: Διαγραφή λέξης
  - Αν η λέξη υπάρχει θα ανανεωθούν το άθροισμα + οι εμφανίσεις
    - Αν το πλήθος εμφανίσεων μηδενιστεί, πρέπει να αφαιρεθεί ο κόμβος!
  - Πρέπει να επιστραφεί ένδειξη για το αν αφαιρέθηκε κόμβος ή όχι ή δεν υπήρχε καν η λέξη.
- Λειτουργία: Εύρεση κι επιστροφή βαθμού λέξης
  - Προκύπτει από το άθροισμα και το πλήθος εμφανίσεων

Λίστα

- Όλες οι προηγούμενες λειτουργίες απαιτούν αναζήτηση λέξης
- Το τι κάνουμε αν βρεθεί (ή όχι) η λέξη διαφέρει ανά λειτουργία
- ΑΡΑ: Γράφουμε μια συνάρτηση αναζήτησης που μπορεί να χρησιμοποιηθεί από όλες (αντί να αντιγράφουμε τον ίδιο κώδικα σε πολλά σημεία!)
- Τι πρέπει να επιστρέφει η συνάρτηση αναζήτησης;
  - Ένδειξη επιτυχίας
  - Δείκτη προς κόμβο που θα διευκολύνει την εκάστοτε λειτουργία
    - Αν βρέθηκε η λέξη → τον κόμβο που την περιέχει
    - Αν όχι → κάποιον "διπλανό". Προηγούμενο; Επόμενο; Τι βολεύει και γιατί;

Checkpoint #2

- Υλοποιήστε τις λειτουργίες της λίστας
- Φτιάξτε ένα προσωρινό αρχείο που περιέχει μια συνάρτηση main η οποία καλεί τις συναρτήσεις της λίστας.
  - Κάντε εισαγωγές, διαγραφές, εκτυπώσεις, αναζητήσεις.
  - Βεβαιωθείτε ότι λειτουργούν σωστά οι οριακές περιπτώσεις
    - Λειτουργίες σε άδεια λίστα
    - Λειτουργίες που "αποτυγχάνουν" (π.χ. προσπάθεια διαγραφής λέξης που δεν υπάρχει)
    - Λειτουργίες στην αρχή ή στο τέλος της λίστας (π.χ. ταξινομημένη εισαγωγή πριν την $1^η$ υπάρχουσα λέξη)

Για το στάδιο 1

- Υλοποιήστε τη δομή κριτικών
- Υλοποιήστε τη λίστα (παρότι δεν ελέγχεται από το autolab)

Πίνακας κατακερματισμού (hash table)

- Η βασική δομή στην οποία αποθηκεύονται οι λέξεις (με άθροισμα βαθμών κι εμφανίσεις) για γρήγορη αναζήτηση!
- Δυναμικός πίνακας από δοχεία
  - Κάθε δοχείο μια λίστα
  - Πόσα δοχεία έχει
  - Πόσες αποθηκευμένες λέξεις έχει
  - Ορίστε κατάλληλο/α struct!

} Από αυτά υπολογίζεται ο βαθμός πληρότητας (load factor)

Πίνακας κατακερματισμού

- INIT_HSIZE: Αρχικό μέγεθος
- HIGH_LOAD: Μέγιστος επιτρεπτός βαθμός πληρότητας
- LOW_LOAD: Ελάχιστος επιτρεπτός βαθμός πληρότητας
- hash(): συνάρτηση κατακερματισμού
  - Δίνεται έτοιμη
  - ΠΡΟΣΟΧΗ: Η συνάρτηση hash επιστρέφει ένα "μεγάλο" αριθμό. Μην ξεχάσετε να τον φέρετε στα "μέτρα" του πίνακα με χρήση %

Πίνακας κατακερματισμού

- Λειτουργία: Δημιουργία άδειου πίνακα
  - Δυναμικός πίνακας από δοχεία
    - Αποφασίστε τι είναι ένα δοχείο. Δείκτης προς κεφαλή λίστας? Κάποιο struct που περιλαμβάνει δείκτη προς κεφαλή λίστας κι επιπλέον πληροφορίες?
  - Αρχικό μέγεθος πίνακα: HSIZE
  - Τα δοχεία αρχικά είναι άδεια
    - Code reuse: Σε αυτό το σημείο θα έχετε ήδη κάποια συνάρτηση δημιουργίας άδειας λίστας.

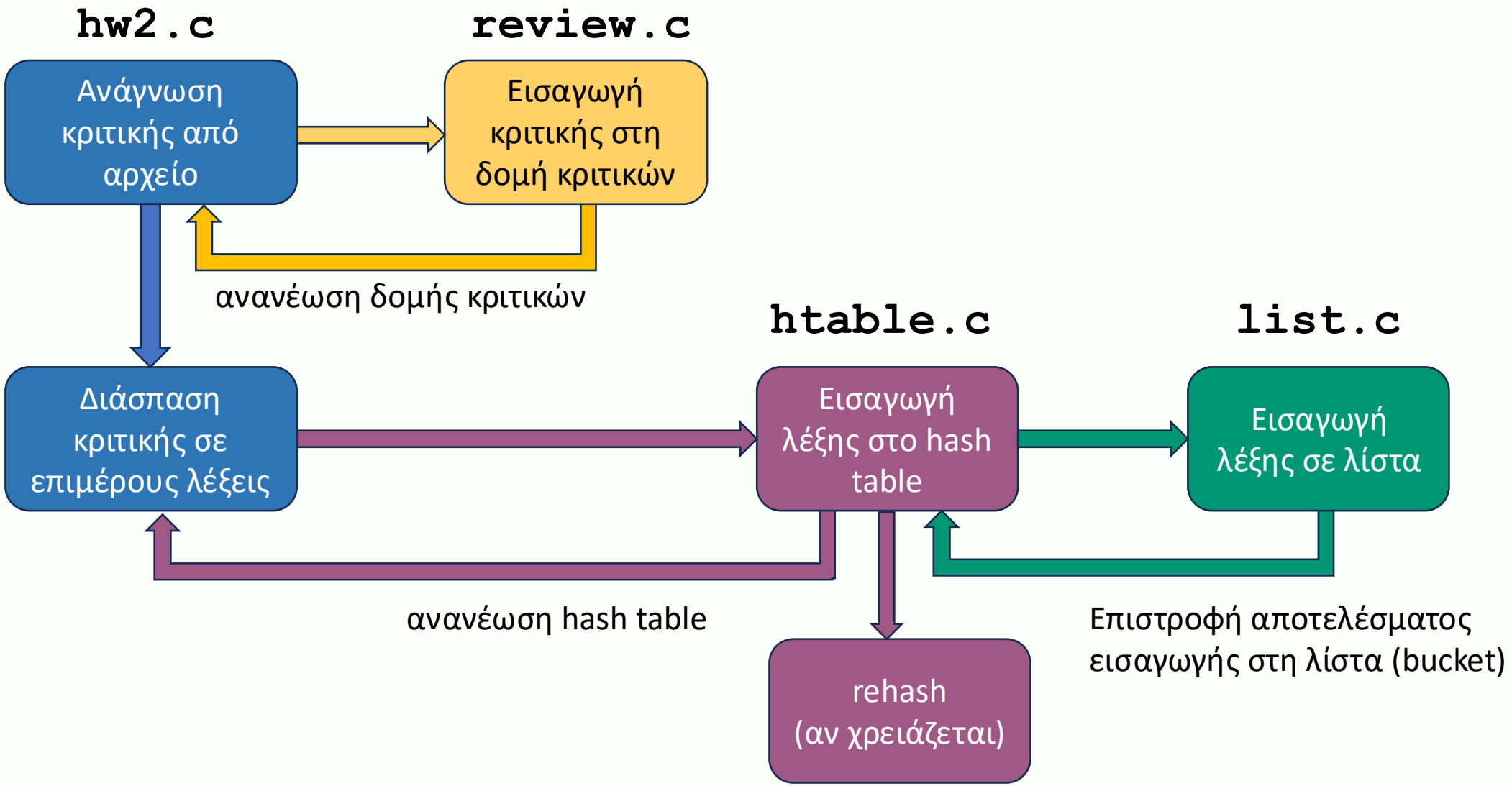
Πίνακας κατακερματισμού

- Λειτουργία: Προσθήκη/Αφαίρεση λέξης
  - Χρήση hash() για εύρεση δοχείου
  - Code reuse: Έχετε ήδη κάποια συνάρτηση προσθήκης/αφαίρεσης λέξης στη λίστα του δοχείου!
- Λειτουργία: Εκτύπωση
  - Χρήση hash() για εύρεση δοχείου
  - Code reuse: Έχετε ήδη κάποια συνάρτηση εκτύπωσης λίστας!
- Λειτουργία: Εύρεση βαθμού μιας λέξης
  - You get the idea…

Πίνακας κατακερματισμού

- Λειτουργία: rehash
  - Αν **κατά την προσθήκη** λέξης ο νέος βαθμός πληρότητας γίνει >= HIGH_LOAD ο πίνακας διπλασιάζεται και γίνεται ανακατανομή των στοιχείων
  - Αν **κατά την αφαίρεση** λέξης ο νέος βαθμός πληρότητας γίνει <= LOW_LOAD ο πίνακας υποδιπλασιάζεται (εκτός αν πέφτει κάτω από INIT_HSIZE) και γίνεται ανακατανομή των στοιχείων
    - Για ευκολία φτιάξτε νέο πίνακα διπλάσιου/μισού μεγέθους και μεταφέρετε τα στοιχεία του παλιού στον καινούργιο
    - Code reuse: Έχετε ήδη συναρτήσεις αφαίρεσης κόμβου από λίστα και προσθήκης κόμβου σε λίστα!
  - Αφήστε το για το τέλος!

Παράδειγμα ροής δεδομένων



Παράδειγμα λειτουργιών hash table

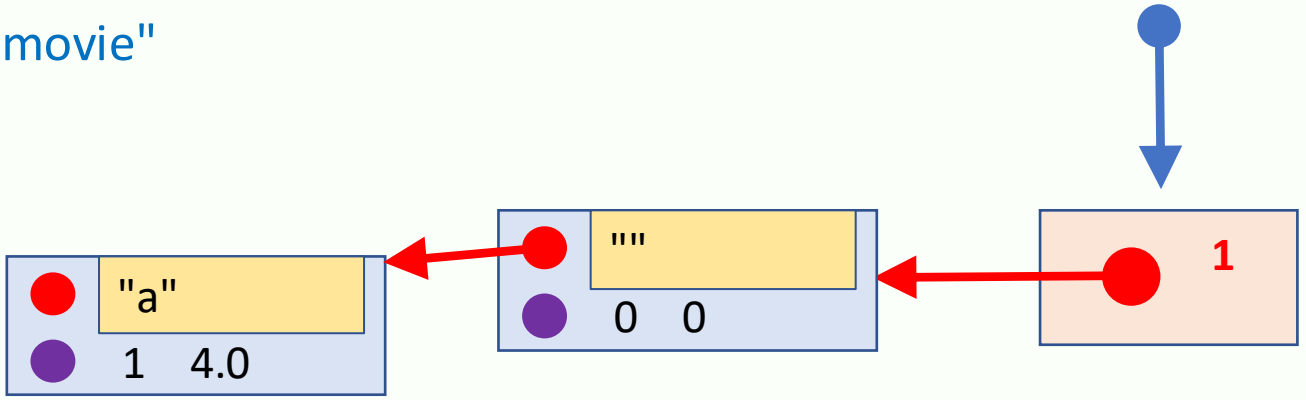
- Στο παράδειγμα που ακολουθεί:
  - Για λόγους σαφήνειας, στα σχήματα της λίστας εμφανίζονται μόνο οι δείκτες προς τον επόμενο κόμβο, εκτός από αυτόν που δείχνει από τον τελευταίο κόμβο προς τον sentinel.
  - Τα δοχεία μπορείτε να τα υλοποιήσετε με διαφορετικό τρόπο από αυτόν που φαίνεται στο παράδειγμα
    - Στο παράδειγμα ένα δοχείο είναι struct που περιέχει δείκτη προς την κεφαλή λίστας και το μέγεθος της λίστας.

Εισαγωγή λέξεων 1ης κριτικής

HIGH_LOAD: 4
LOW_LOAD: 1

Δόθηκε η κριτική "A stunning epic movie"
με σκορ 4.0

πίνακας κατακερματισμού



Εισαγωγή της λέξης "a"

```
hash("a") % 1 : 0
```

μέγεθος: 1
στοιχείων: **1**
load factor: **1**

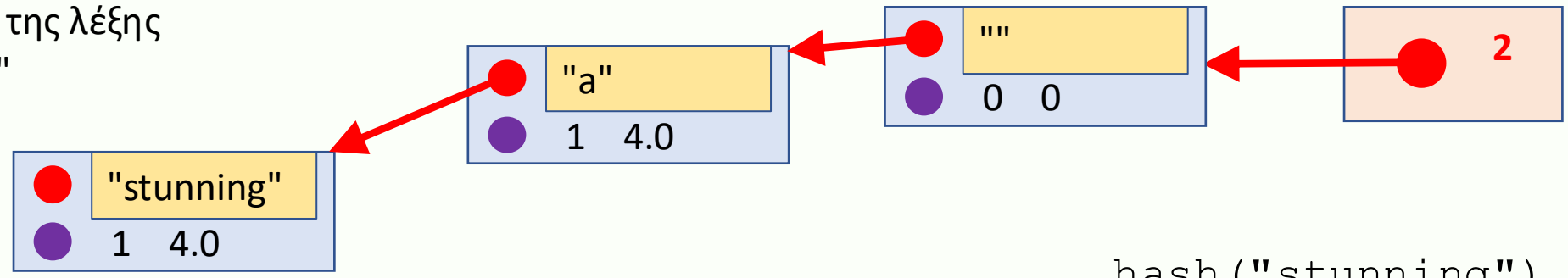
Εισαγωγή λέξεων 1ης κριτικής

HIGH_LOAD: 4
LOW_LOAD: 1

Δόθηκε η κριτική "A stunning epic movie"
με σκορ 4.0

πίνακας κατακερματισμού

Εισαγωγή της λέξης
"stunning"




```
hash("stunning") % 1 : 0
```

μέγεθος: 1
στοιχείων: **2**
load factor: **2**

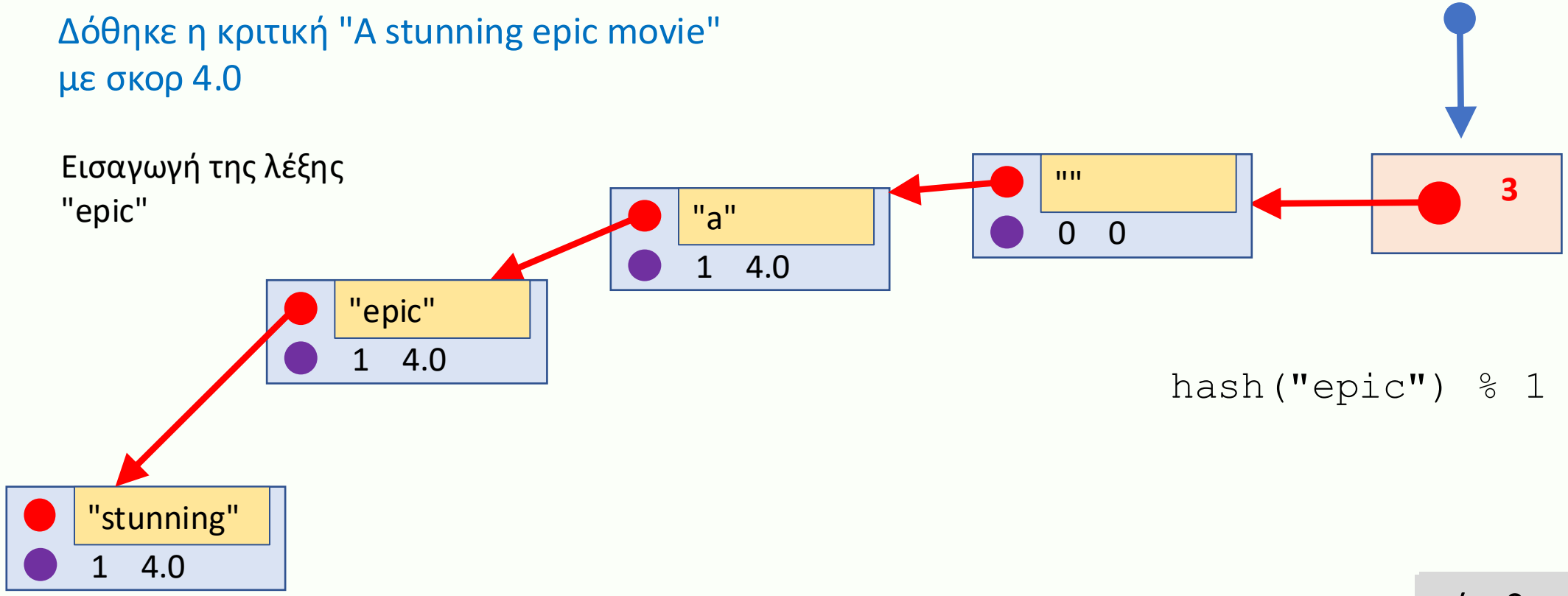
Εισαγωγή λέξεων 1ης κριτικής

HIGH_LOAD: 4
LOW_LOAD: 1

πίνακας κατακερματισμού

Δόθηκε η κριτική "A stunning epic movie"
με σκορ 4.0

Εισαγωγή της λέξης
"epic"




```
hash("epic") % 1 : 0
```

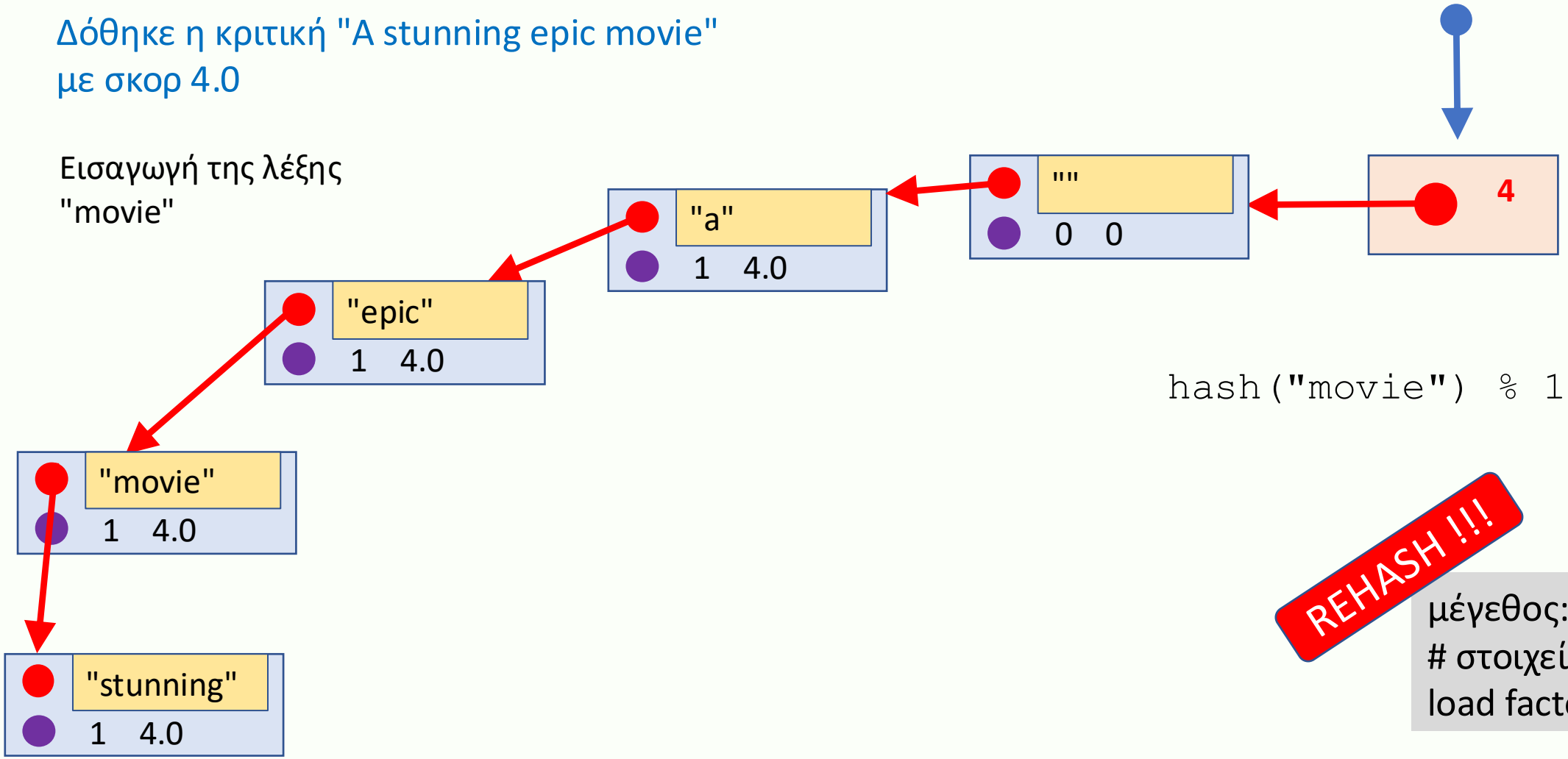
μέγεθος: 1
στοιχείων: **3**
load factor: **3**

Εισαγωγή λέξεων 1ης κριτικής

HIGH_LOAD: 4
LOW_LOAD: 1

Δόθηκε η κριτική "A stunning epic movie"
με σκορ 4.0

πίνακας κατακερματισμού




```
hash("movie") % 1 : 0
```

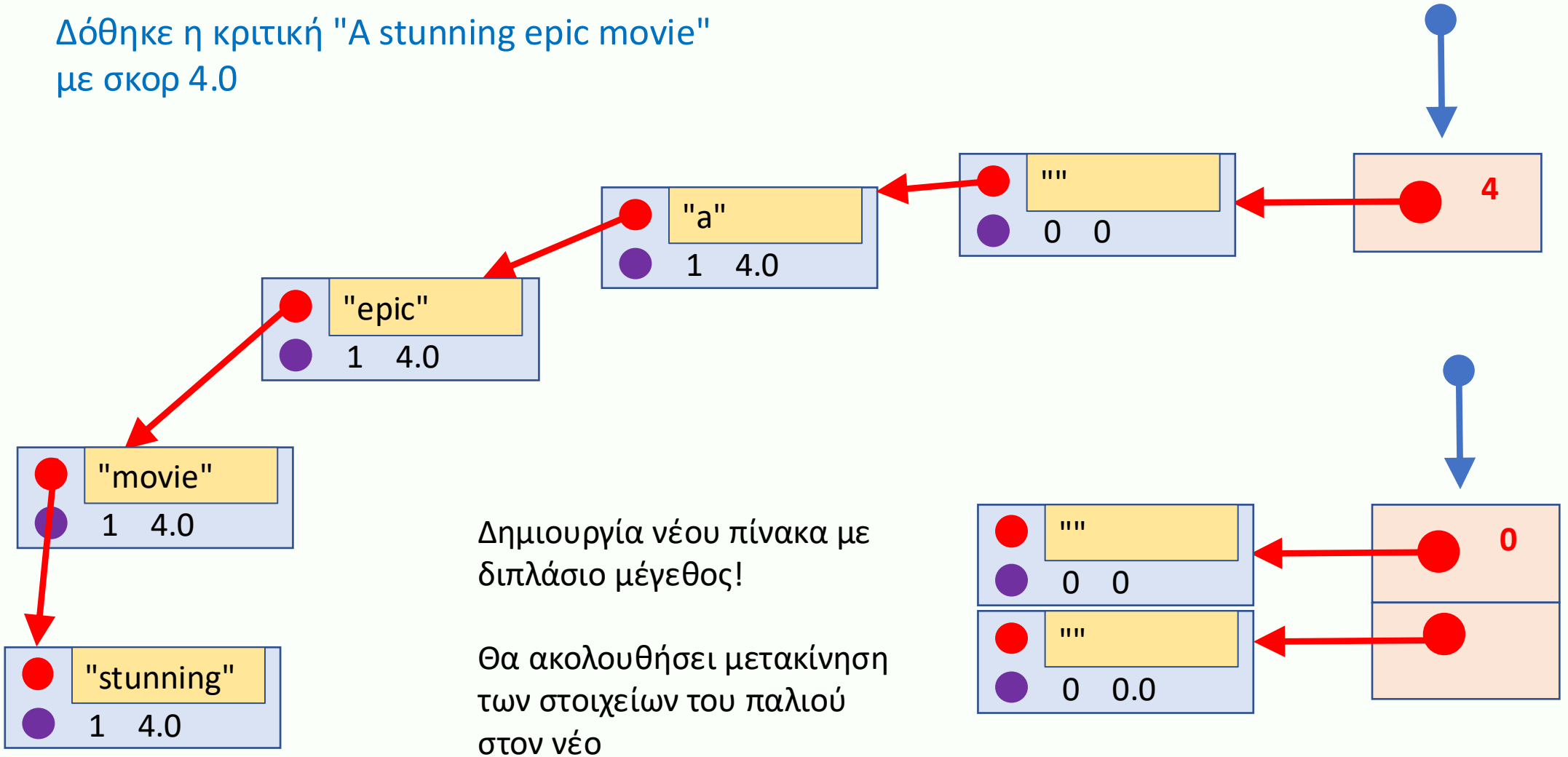
REHASH !!!

μέγεθος: 1
στοιχείων: **4**
load factor: **4**

Εισαγωγή λέξεων 1ης κριτικής

HIGH_LOAD: 4
LOW_LOAD: 1

Δόθηκε η κριτική "A stunning epic movie"
με σκορ 4.0



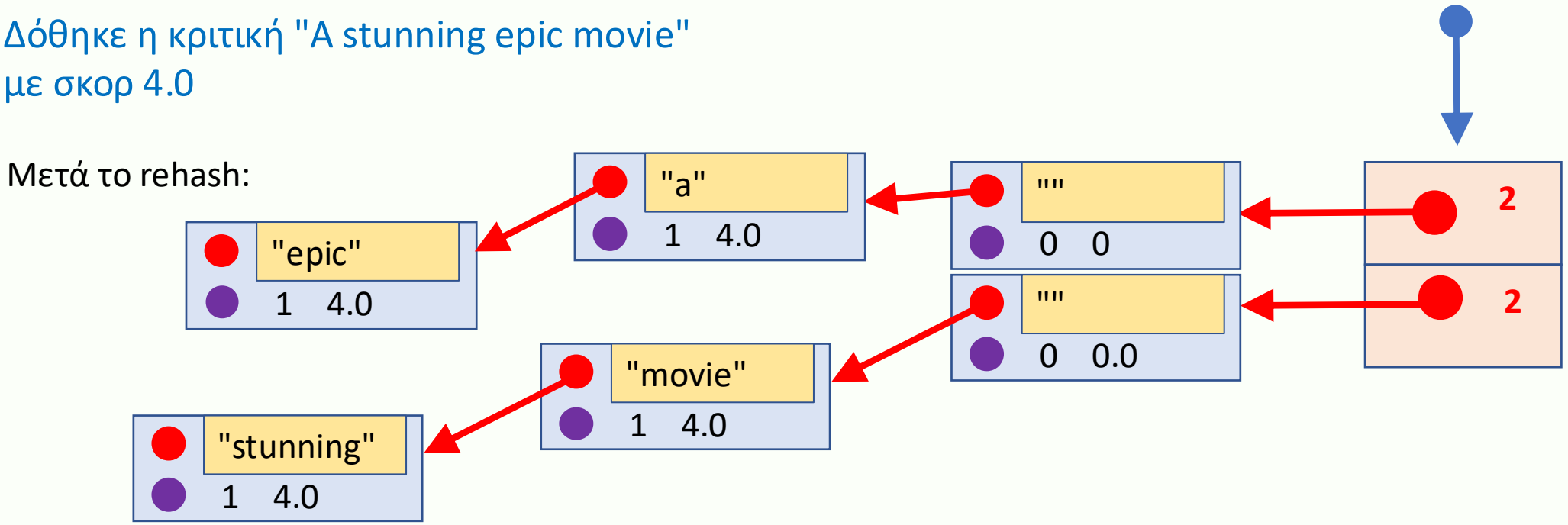
Δημιουργία νέου πίνακα με διπλάσιο μέγεθος!

Θα ακολουθήσει μετακίνηση των στοιχείων του παλιού στον νέο

Εισαγωγή λέξεων 1ης κριτικής

HIGH_LOAD: 4
LOW_LOAD: 1

Δόθηκε η κριτική "A stunning epic movie"
με σκορ 4.0




```
hash("a") % 2 : 0
hash("epic") % 2 : 0
hash("movie") % 2 : 1
hash("stunning") % 2 : 1
```

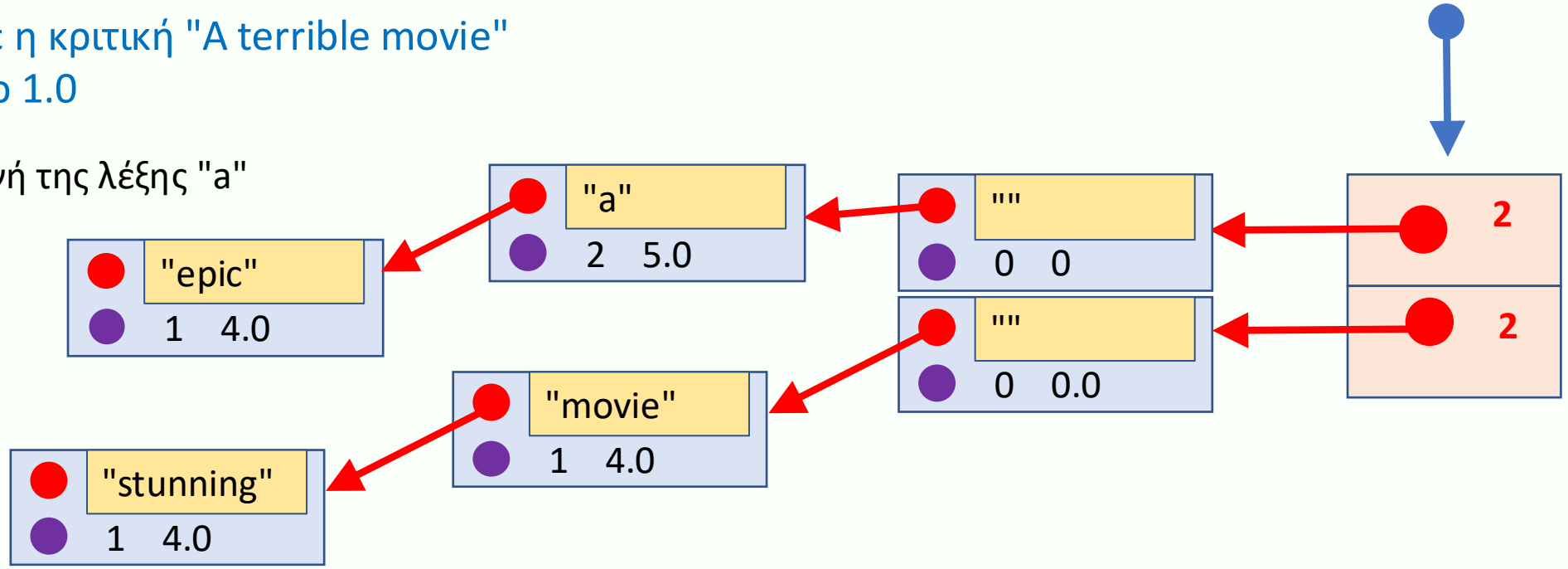
μέγεθος: 2
στοιχείων: **4**
load factor: **2**

Εισαγωγή λέξεων 2ης κριτικής

HIGH_LOAD: 4
LOW_LOAD: 1

Δόθηκε η κριτική "A terrible movie"
με σκορ 1.0

Εισαγωγή της λέξης "a"




```
hash("a") % 2 : 0
```

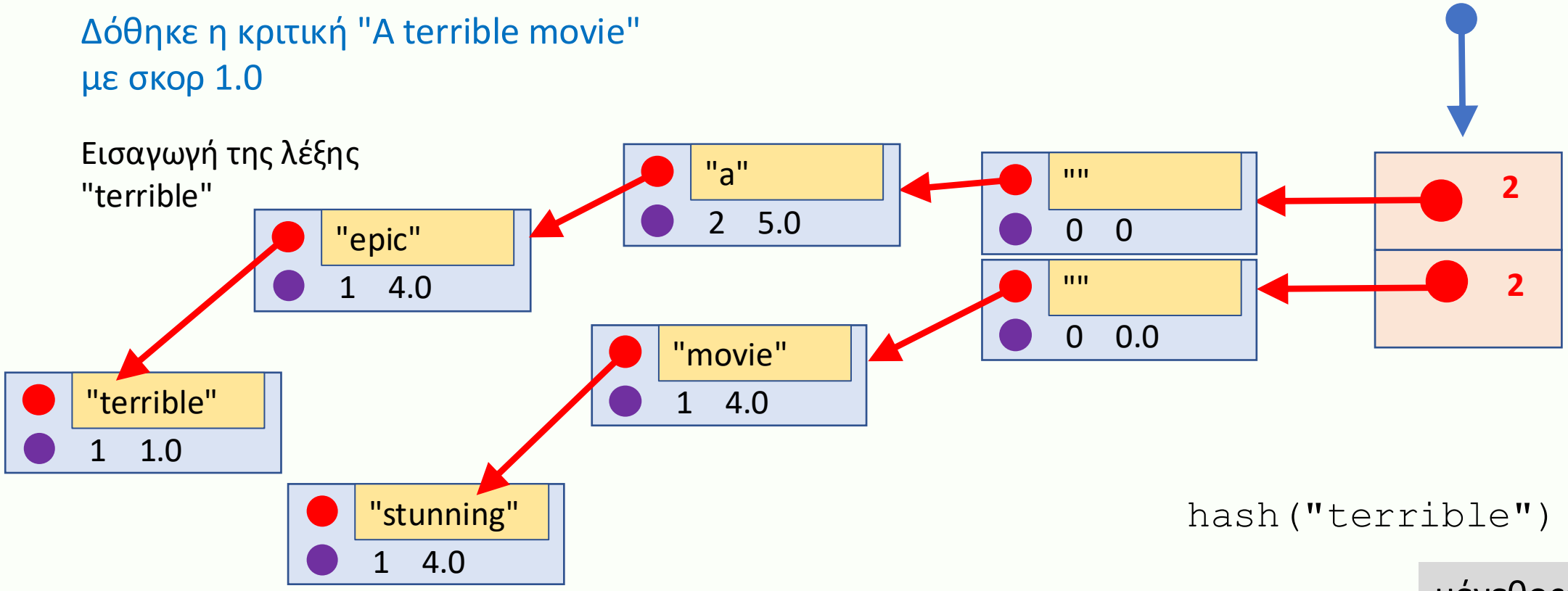
μέγεθος: 2
στοιχείων: **4**
load factor: **2**

Εισαγωγή λέξεων 2ης κριτικής

HIGH_LOAD: 4
LOW_LOAD: 1

Δόθηκε η κριτική "A terrible movie"
με σκορ 1.0

Εισαγωγή της λέξης
"terrible"




```
hash("terrible") % 2 : 0
```

μέγεθος: 2
στοιχείων: **5**
load factor: **2.5**

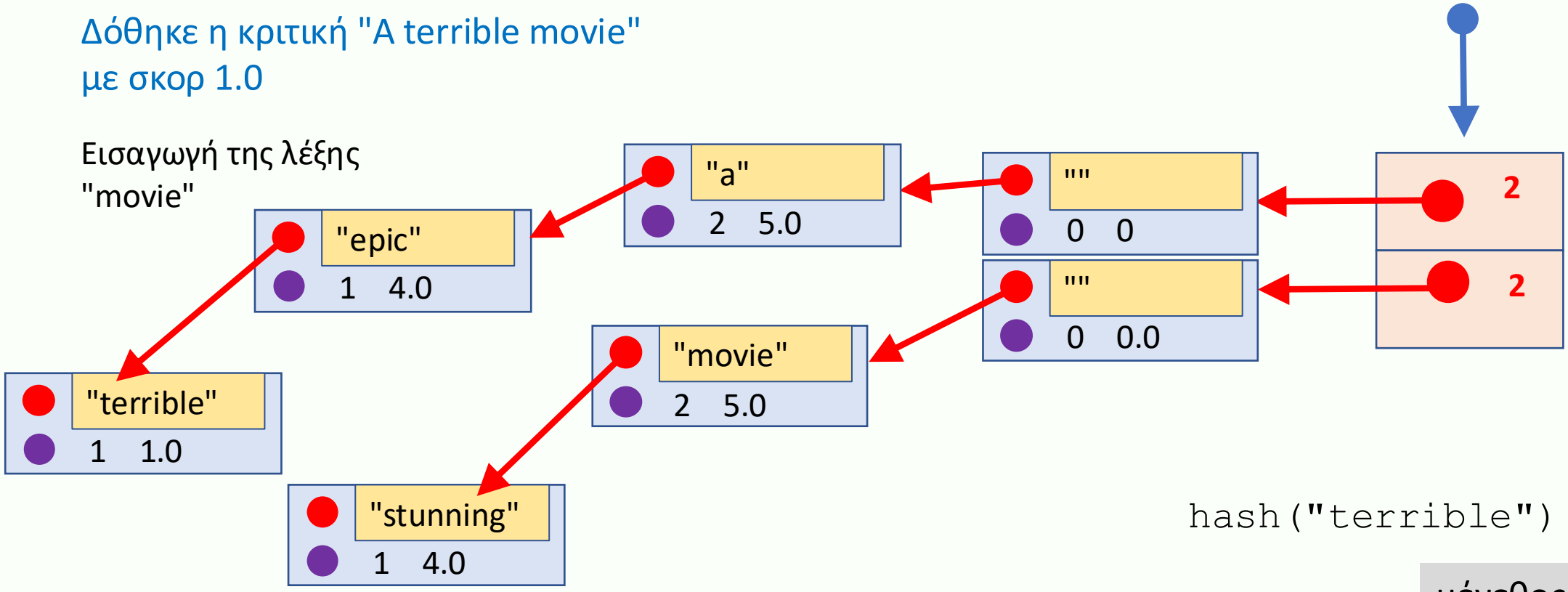
Εισαγωγή λέξεων 2ης κριτικής

HIGH_LOAD: 4
LOW_LOAD: 1

πίνακας κατακερματισμού

Δόθηκε η κριτική "A terrible movie"
με σκορ 1.0

Εισαγωγή της λέξης
"movie"



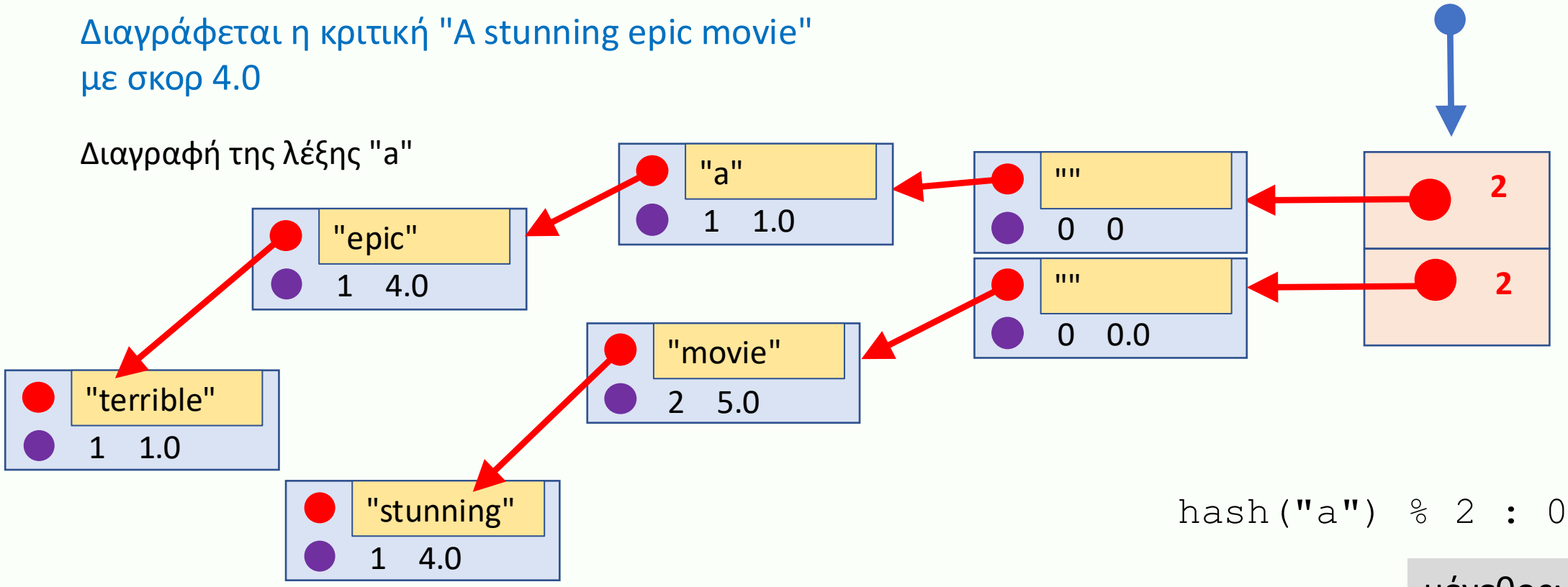

```
hash("terrible") % 2 : 0
```

μέγεθος: 2
στοιχείων: **5**
load factor: **2.5**

Διαγραφή 1ης κριτικής

HIGH_LOAD: 4
LOW_LOAD: 1

Διαγράφεται η κριτική "A stunning epic movie"
με σκορ 4.0



μέγεθος: 2
στοιχείων: **5**
load factor: **2.5**

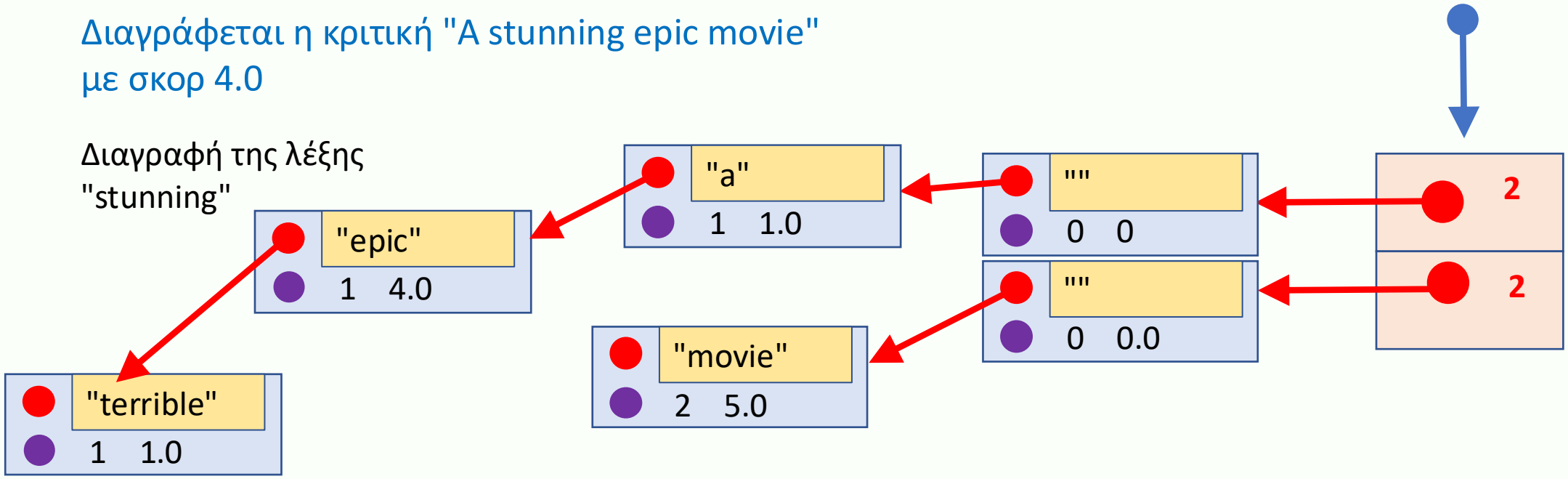
α) Μειώνεται ο αριθμός εμφανίσεων της λέξης κατά 1
β) Μειώνεται το άθροισμα βαθμών της λέξης κατά όσο είναι το σκορ της κριτικής (4.0)

Διαγραφή 1ης κριτικής

HIGH_LOAD: 4
LOW_LOAD: 1

πίνακας κατακερματισμού

Διαγράφεται η κριτική "A stunning epic movie"
με σκορ 4.0




```
hash("stunning") % 2 : 1
```

μέγεθος: 2
στοιχείων: **4**
load factor: **2**

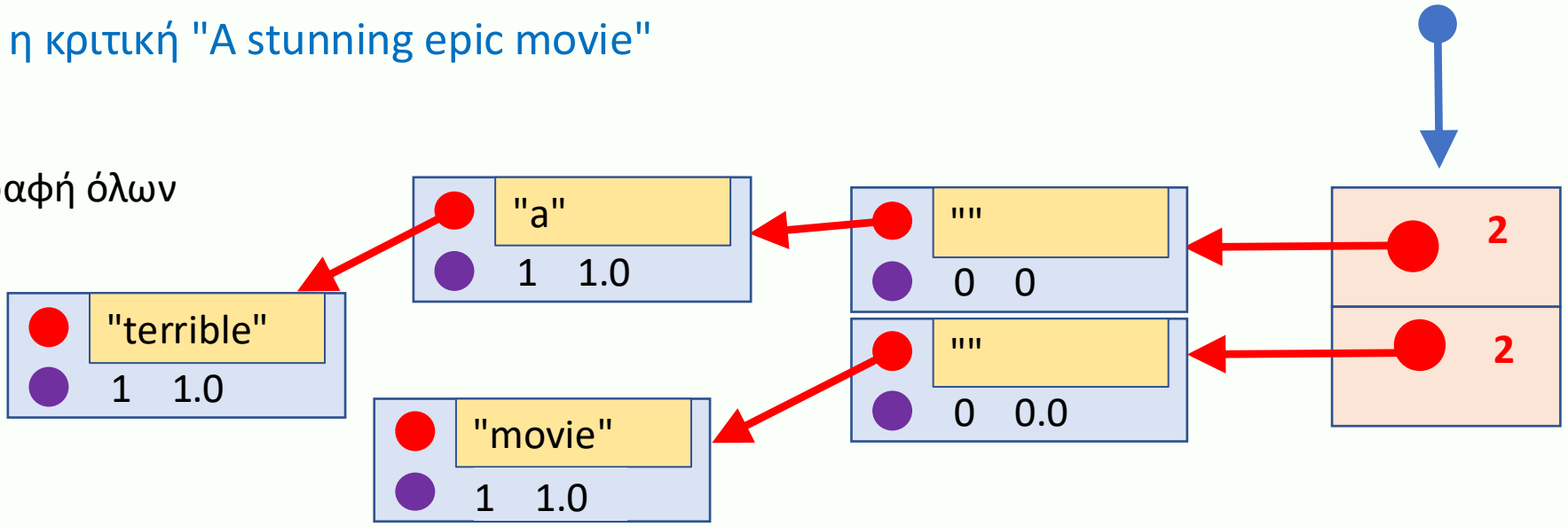
α) Μειώνεται ο αριθμός εμφανίσεων της λέξης κατά 1
β) Επειδή ο αριθμός εμφανίσεων έγινε 0, διαγράφεται ο κόμβος.

Διαγραφή 1ης κριτικής

HIGH_LOAD: 4
LOW_LOAD: 1

Διαγράφεται η κριτική "A stunning epic movie" με σκορ 4.0

Μετά τη διαγραφή όλων των λέξεων



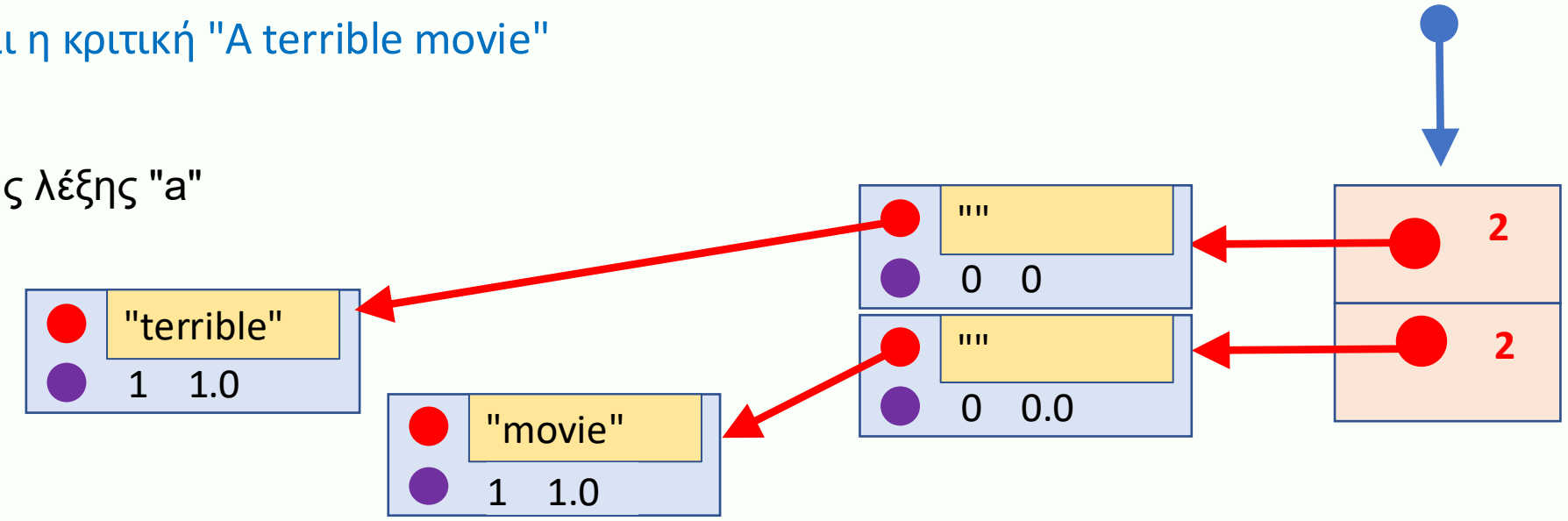
μέγεθος: 2
στοιχείων: **3**
load factor: **1.5**

Διαγραφή 2ης κριτικής

HIGH_LOAD: 4
LOW_LOAD: 1

Διαγράφεται η κριτική "A terrible movie"
με σκορ 4.0

Διαγραφή της λέξης "a"




```
hash("a") % 2 : 0
```

REHASH !!!

μέγεθος: 2
στοιχείων: **2**
load factor: **1**

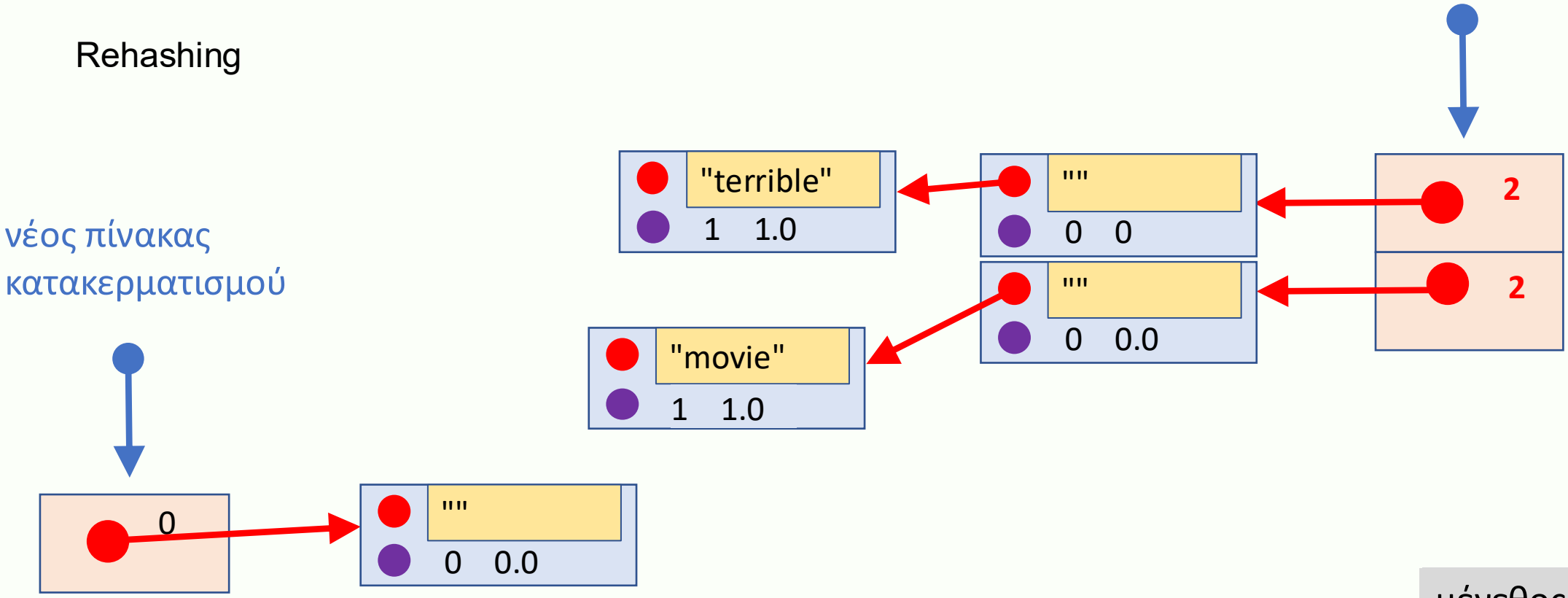
α) Μειώνεται ο αριθμός εμφανίσεων της λέξης κατά 1
β) Επειδή ο αριθμός εμφανίσεων έγινε 0, διαγράφεται ο κόμβος.

Διαγραφή 2ης κριτικής

HIGH_LOAD: 4
LOW_LOAD: 1

πίνακας κατακερματισμού

Rehashing



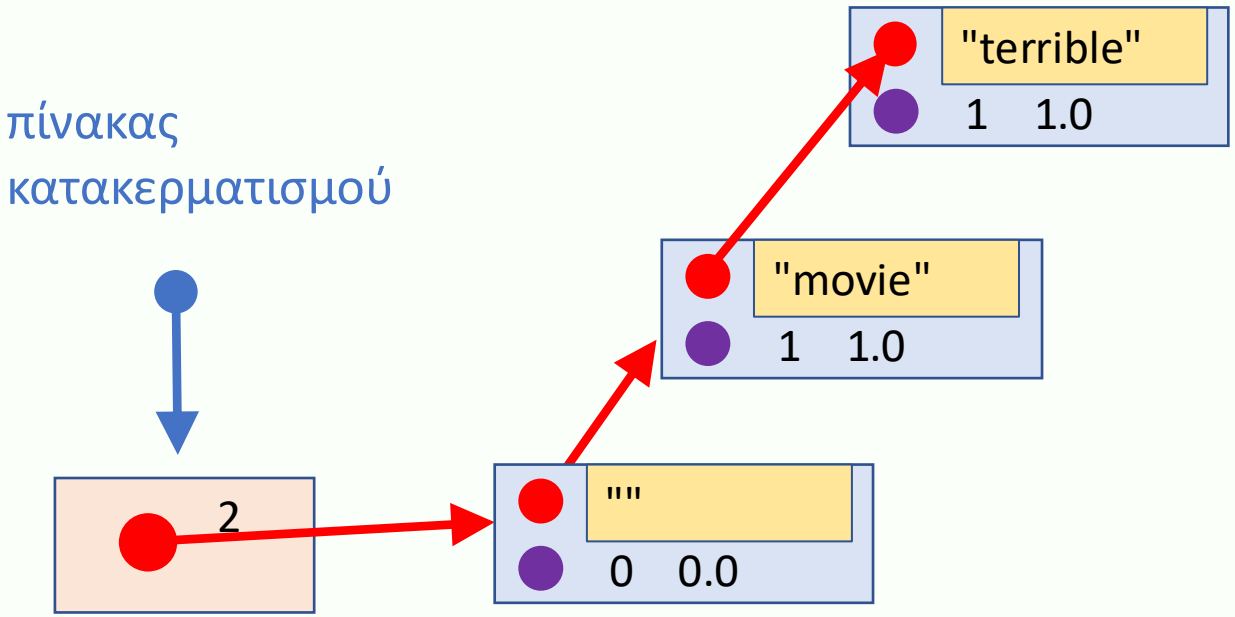
μέγεθος: 2
στοιχείων: **2**
load factor: **1**

μέγεθος: 1
στοιχείων: 0
load factor: 0

Διαγραφή 2ης κριτικής

HIGH_LOAD: 4
LOW_LOAD: 1

Μετά το rehashing



μέγεθος: 1
στοιχείων: **2**
load factor: **2**