Κατασκευαστής αντιγραφής και αντιγραφή ανάθεσης

#12

Τμήμα Πληροφορικής και Τηλεπικοινωνιών

Πανεπιστήμιο Ιωαννίνων (Άρτα)

Γκόγκος Χρήστος

V 1.0

Συναρτήσεις που δημιουργούνται αυτόματα

- Οι ακόλουθες δύο συναρτήσεις μέλη που έχουμε ήδη συναντήσει, κάτω από ορισμένες περιστάσεις δημιουργούνται αυτόματα από τον μεταγλωττιστή
  - **Κατασκευαστής**: Ένας κενός προκαθορισμένος κατασκευαστής (δηλαδή ένας κατασκευαστής χωρίς παραμέτρους και με κενό σώμα) δημιουργείται αυτόματα αν δεν οριστεί κάποιος άλλος κατασκευαστής.
  - **Καταστροφέας**: Ένας κενός καταστροφέας δημιουργείται αν δεν οριστεί καταστροφέας (επίσης με κενό σώμα).
- Δύο άλλες συναρτήσεις μέλη που επίσης, κάτω από ορισμένες περιστάσεις δημιουργούνται αυτόματα είναι ο κατασκευαστής αντιγραφής (**copy constructor**) και ο τελεστής ανάθεσης (**assignment operator**)

Κατασκευαστής αντιγραφής (copy constructor)

- Ο κατασκευαστής αντιγραφής είναι κατασκευαστής, και συνεπώς:
  - Έχει το ίδιο όνομα με την κλάση.
  - Δεν έχει τύπο επιστροφής (αν και φαίνεται σαν να επιστρέφει ένα αντικείμενο κλάσης όταν καλείται ρητά).
- Όπως και με τον κατασκευαστή μετατροπής υπάρχουν περιπτώσεις που ο copy constructor καλείται υπονοούμενα. Αυτό συμβαίνει:
  - Όταν ένα αντικείμενο δηλώνεται να έχει την ίδια τιμή με ένα άλλο αντικείμενο
    - Παράδειγμα:

```
Fraction f1(1,2);
Fraction f2 = f1; // το νέο αντικείμενο αρχικοποιείται ως ένα αντίγραφο του f1
```

  - Όταν ένα αντικείμενο **περνά με τιμή (by value)** σε μια συνάρτηση
  - Όταν ένα αντικείμενο **επιστρέφεται με τιμή (by value)** από μια συνάρτηση

Δήλωση κατασκευαστή αντιγραφής

- Καθώς ο σκοπός ενός κατασκευαστή αντιγραφής είναι να αρχικοποιηθεί ένα νέο αντικείμενο ως αντίγραφο ενός άλλου αντικειμένου, δέχεται ένα αντικείμενο ως παράμετρο.
  - `classname(const classname&)`
- Η παράμετρος είναι `const` (αν και δεν απαιτείται) διότι ο κατασκευαστής αντιγραφής **δεν θα πρέπει** να τροποποιεί την παράμετρο.
- Η παράμετρος θα πρέπει να περνά με αναφορά. Αν περνούσε με τιμή τότε λόγω αναδρομικής κλήσης της ίδιας συνάρτησης, δηλαδή του copy constructor θα οδηγούμασταν σε stack overflow (ωστόσο, ο μεταγλωττιστής δεν επιτρέπει να δηλώσουμε σε κατασκευαστή αντιγραφής την παράμετρο να περνά με τιμή).
- Παραδείγματα:
  - `Fraction(const Fraction &f);`
  - `Mixed(const Mixed& m);`

Κατασκευαστής αντιγραφής

- Τι θα πρέπει να κάνει ένας copy constructor;
  - Να αντιγράφει τα μέλη δεδομένων από την παράμετρο στο νέο αντικείμενο.
  - Παράδειγμα:

```
Fraction(const Fraction &f)
{
numer = f.numer;
denom = f.denom;
}
```

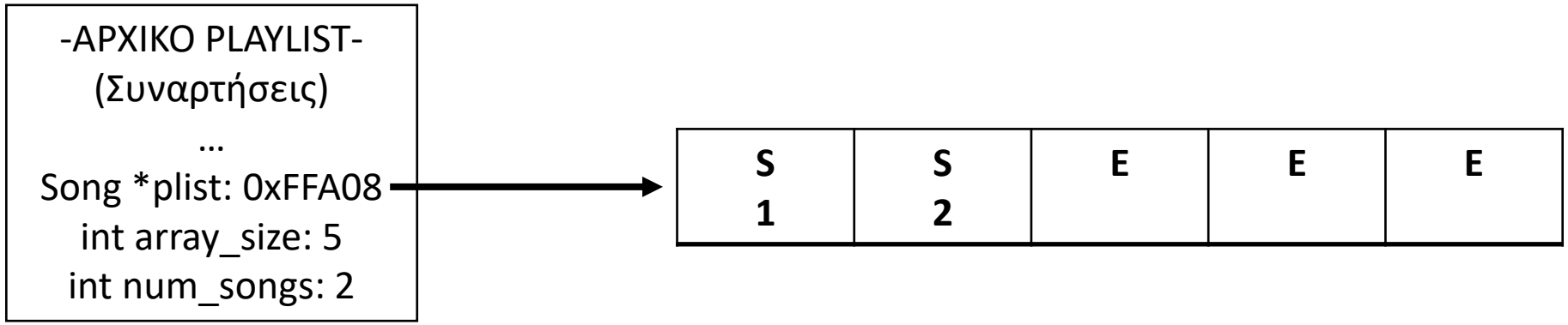
ή

```
Fraction(const Fraction &f)
{
this->numer = f.numer;
this->denom = f.denom;
}
```

  - Ο copy constructor που δημιουργείται αυτόματα (default copy constructor) κάνει ακριβώς αυτό.
  - Ωστόσο, δεν είναι αυτή η συμπεριφορά που θέλουμε πάντα.

«Προβληματικός» προκαθορισμένος κατασκευαστής αντιγραφής: ρηχή αντιγραφή

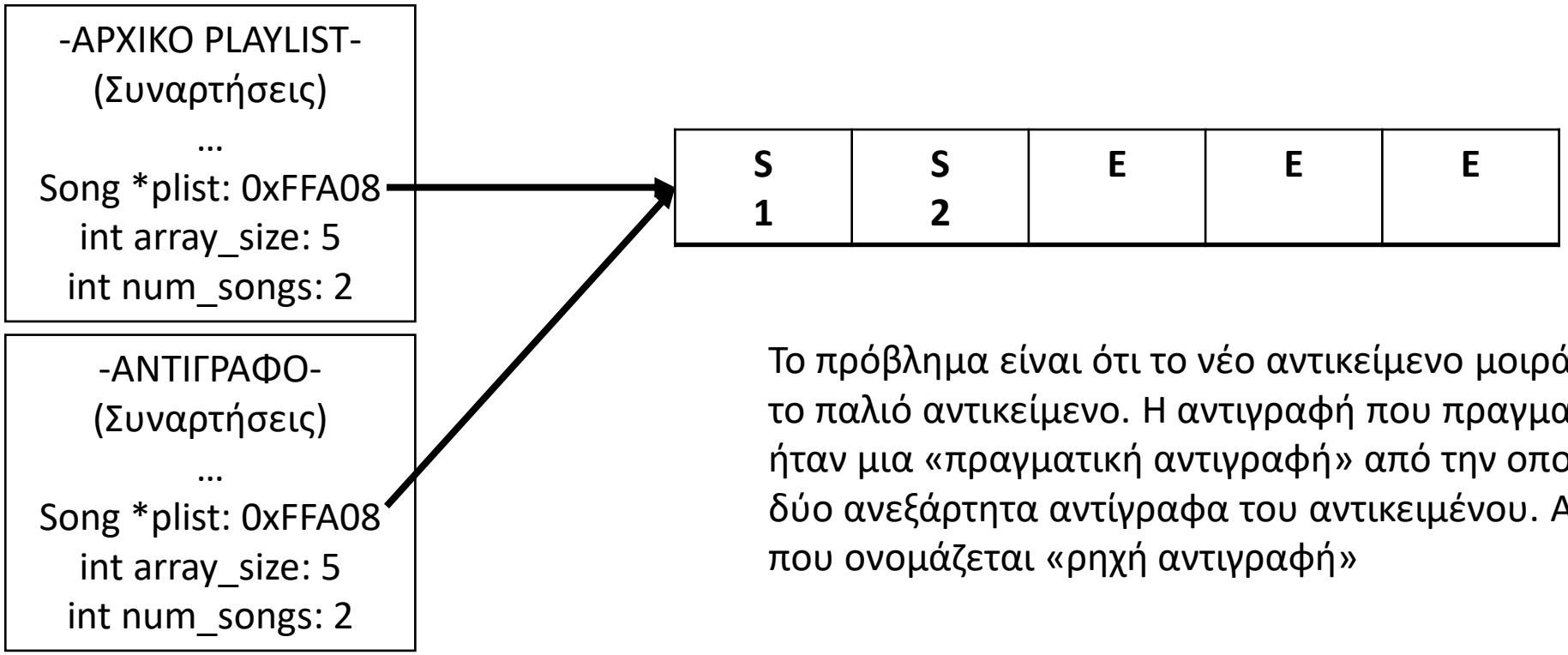
- Έστω ότι θέλουμε να αντιγράψουμε ένα αντικείμενο playlist



- Η ρηχή αντιγραφή (shallow copy) κάνει **ακριβής** αντιγραφή όλων των μελών δεδομένων από το παλιό αντικείμενο στο νέο

Λειτουργία της προκαθορισμένης ρηχής αντιγραφής

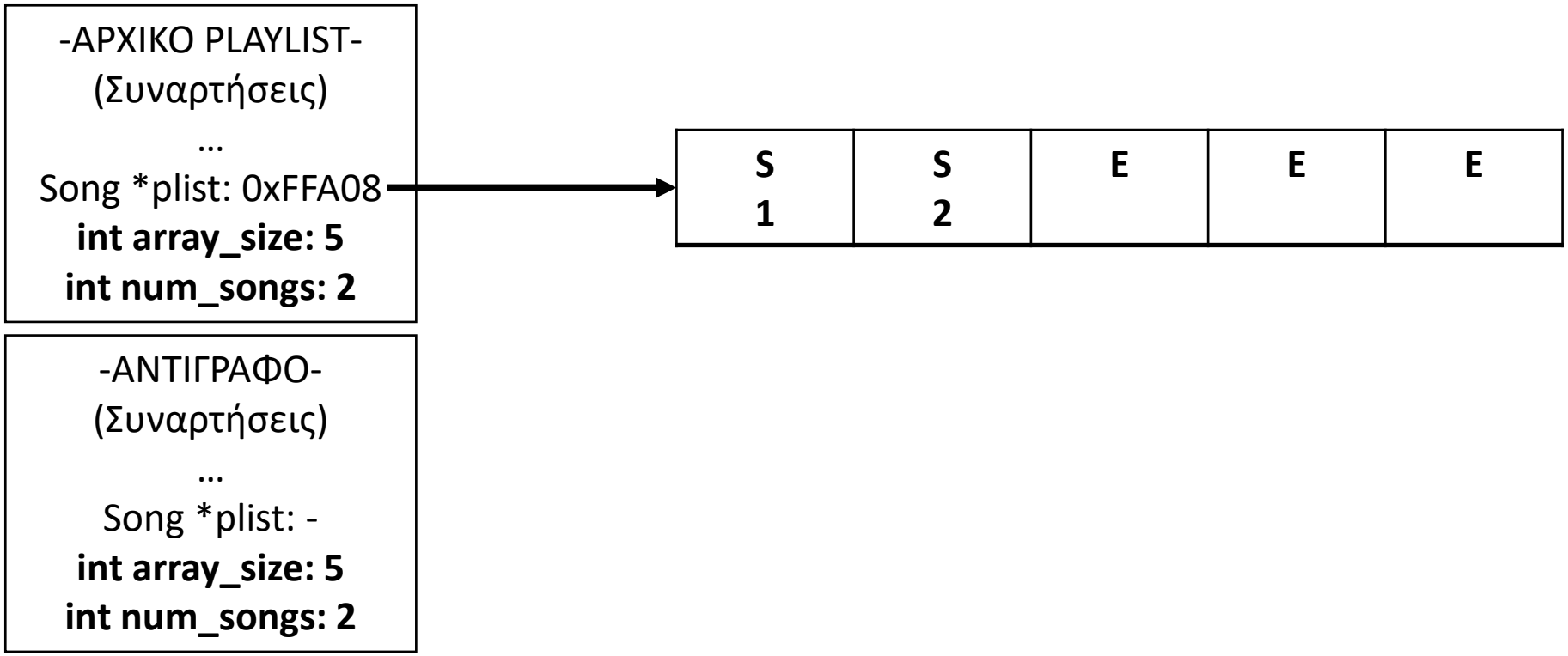
- Στον κατασκευαστή αντιγραφής το αρχικό αντικείμενο είναι η παράμετρος.
- Στο αντίγραφο ορίζονται τα μέλη δεδομένων να είναι ίδια με αυτά του αρχικού αντικειμένου.



Το πρόβλημα είναι ότι το νέο αντικείμενο μοιράζεται την plist με το παλιό αντικείμενο. Η αντιγραφή που πραγματοποιήθηκε δεν ήταν μια «πραγματική αντιγραφή» από την οποία θα προκύπτανε δύο ανεξάρτητα αντίγραφα του αντικειμένου. Αυτός είναι ο λόγος που ονομάζεται «ρηχή αντιγραφή»

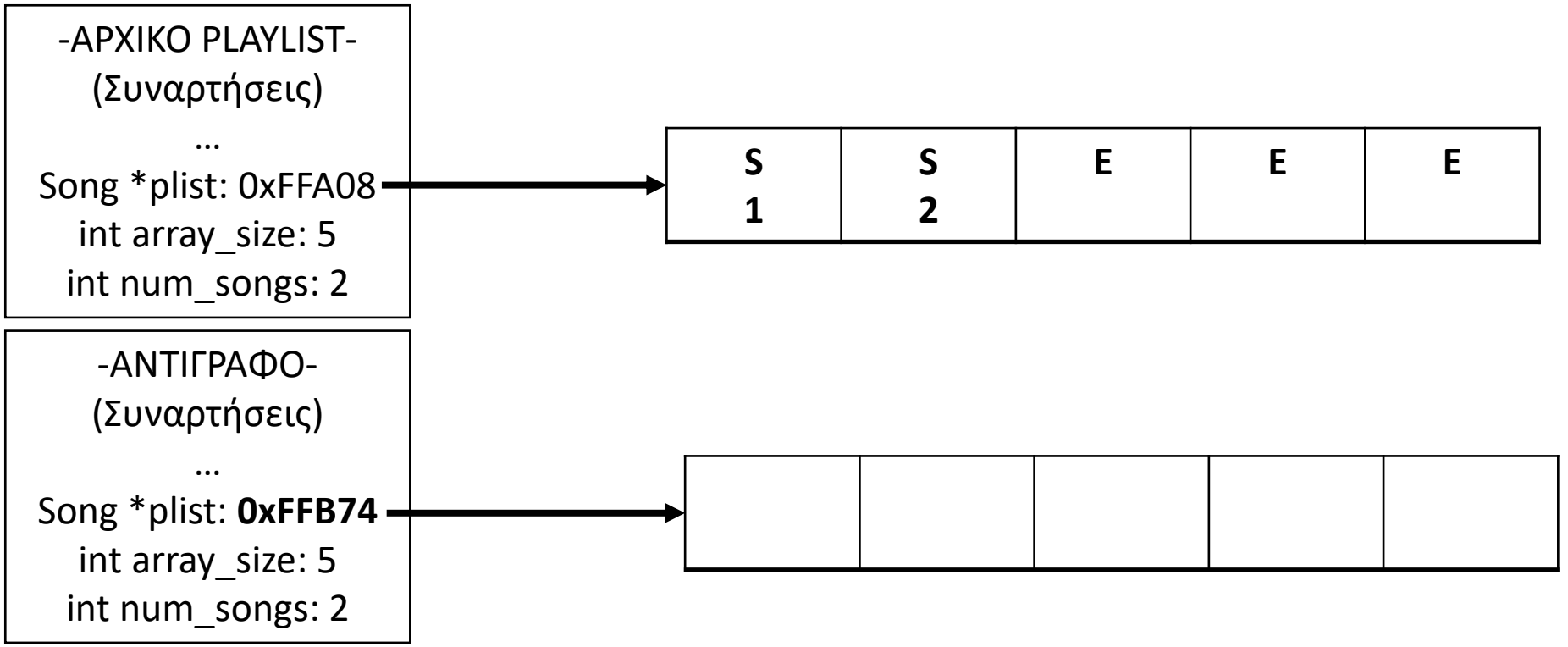
Βαθιά αντιγραφή (πρέπει να υλοποιηθεί ως κατασκευαστής αντιγραφής από το χρήστη)

- Ο κατασκευαστής αντιγραφής έχει το αρχικό αντικείμενο ως παράμετρο.
- Τα μέλη δεδομένα που δεν είναι δείκτες ορίζονται να έχουν ίσες τιμές με τα μέλη δεδομένων του αρχικού αντικειμένου



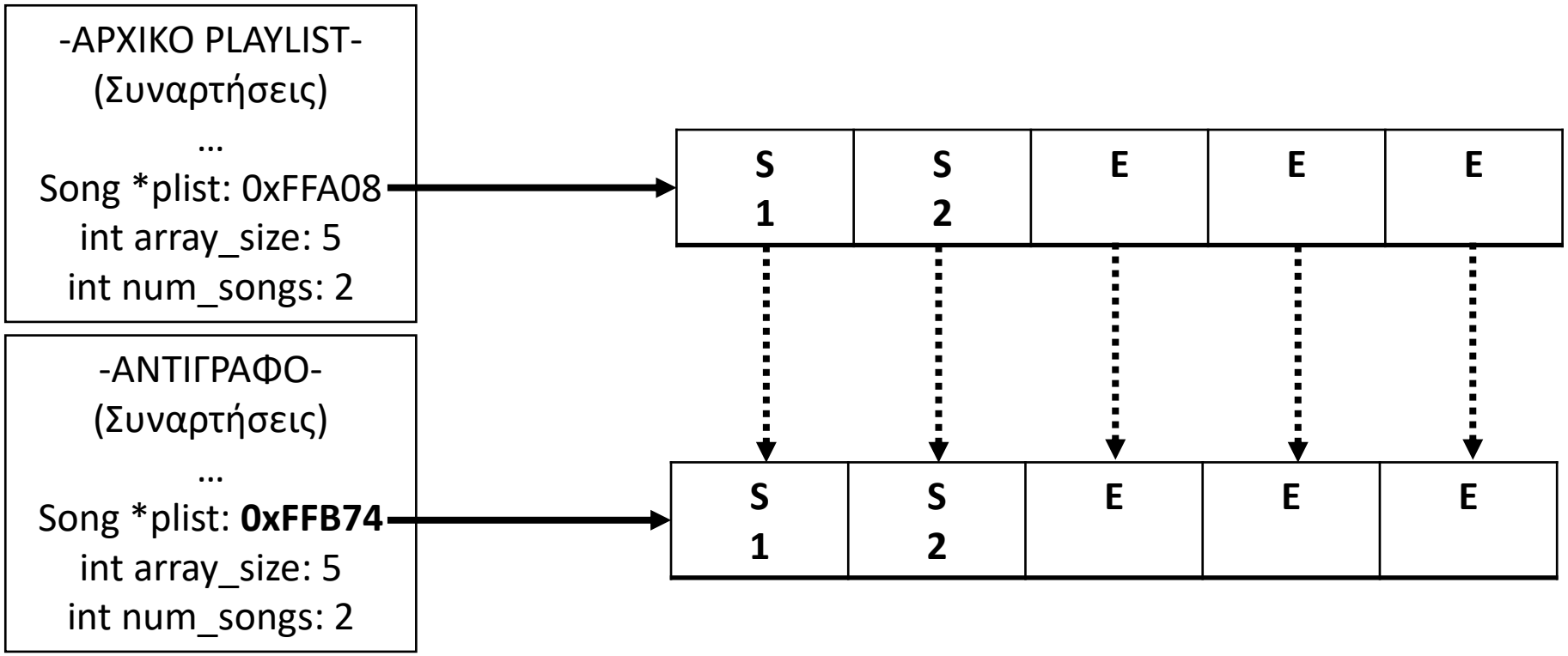
Βαθιά αντιγραφή (deep copy)

- Δεσμεύεται νέα μνήμη στην οποία θα δείχνει ο δείκτης του αντιγράφου.



Βαθιά αντιγραφή (deep copy)

- Αντιγράφονται τα δεδομένα από την παλιά δυναμική μνήμη στη νέα.



Τελεστής ανάθεσης (assignment operator)

- Ο τελεστής ανάθεσης (=) καλείται όταν ένα αντικείμενο εκχωρείται σε ένα άλλο.
- Μοιάζει με τον κατασκευαστή αντιγραφής αλλά υπάρχουν ορισμένες βασικές διαφορές:
  - Ο τελεστής ανάθεσης είναι μια κανονική συνάρτηση μέλος της κλάσης και όχι ένας κατασκευαστής, άρα αφορά δύο αντικείμενα που ήδη υπάρχουν και έχουν αρχικοποιηθεί.
  - Ο τελεστής ανάθεσης επιστρέφει την τιμή που του ανατίθεται (έτσι, επιτρέπονται και διαδοχικές (cascading) κλήσεις του τελεστή ανάθεσης)

```
Fraction f1(1,2), f2, f3;
f3 = f2 = f1; // τα αντικείμενα f2 και f3 λαμβάνουν την ίδια τιμή.
```

Τελεστής ανάθεσης

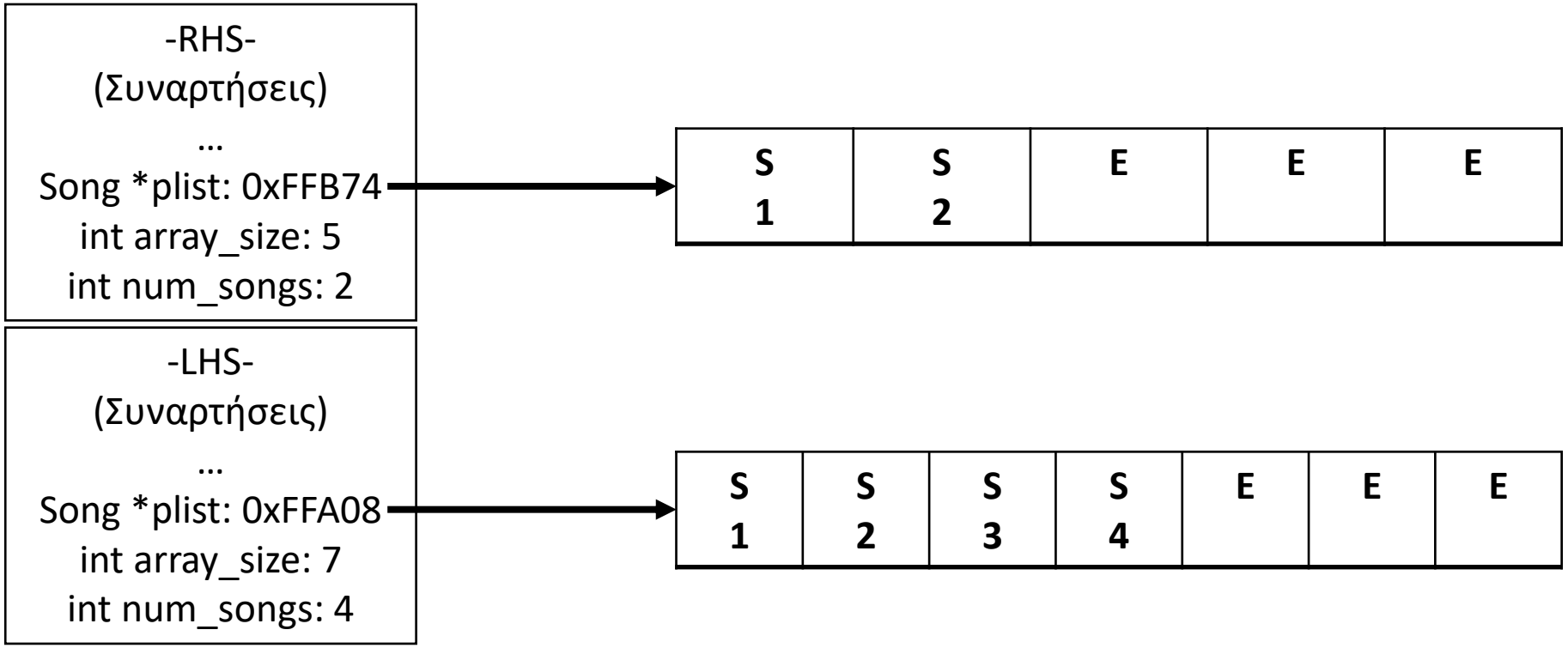
- Μορφή:
  - `classname& operator=(const classname&)`
- Παράδειγμα:
  - `Fraction lhs(1,2), rhs(2,5);`
    `lhs = rhs;`
  - `lhs` είναι το καλών αντικείμενο, ενώ `rhs` είναι η παράμετρος, η συνάρτηση του τελεστή ανάθεσης αλλάζει το `lhs` έτσι ώστε να είναι ένα αντίγραφο του `rhs` και επιστρέφει μια αναφορά στο `lhs` μέσω του δείκτη `this`.

Ο δείκτης `this`

- Σε κάθε αντικείμενο υπάρχει ένας δείκτης που ονομάζεται `this`
- Είναι σαν να υπάρχει η ακόλουθη δήλωση στα μέλη δεδομένων του αντικειμένου:
  - `classname *this;`
- Ο δείκτης `this` δείχνει προς το ίδιο το αντικείμενο.
- Μπορούμε να καλούμε άλλες συναρτήσεις μέλη με την εντολή:
  - `this->memberFunction();`
- Χρησιμοποιώντας το δείκτη `this` επιστρέφουμε μια αναφορά προς το ίδιο το αντικείμενο στον τελεστή ανάθεσης.

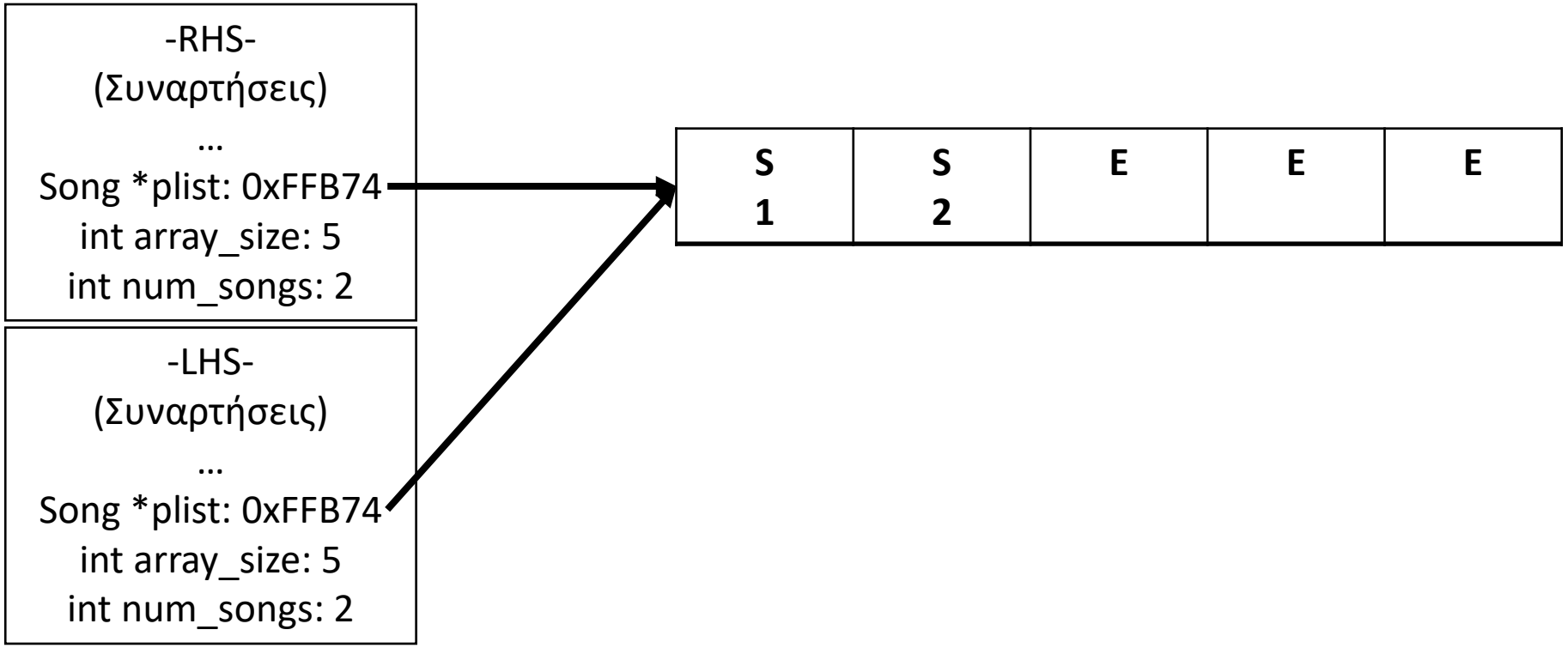
Ρηχή αντιγραφή με ανάθεση

- Έστω ότι αναθέτουμε το αντικείμενο playlist RHS στο LHS.
  - `LHS = RHS;`



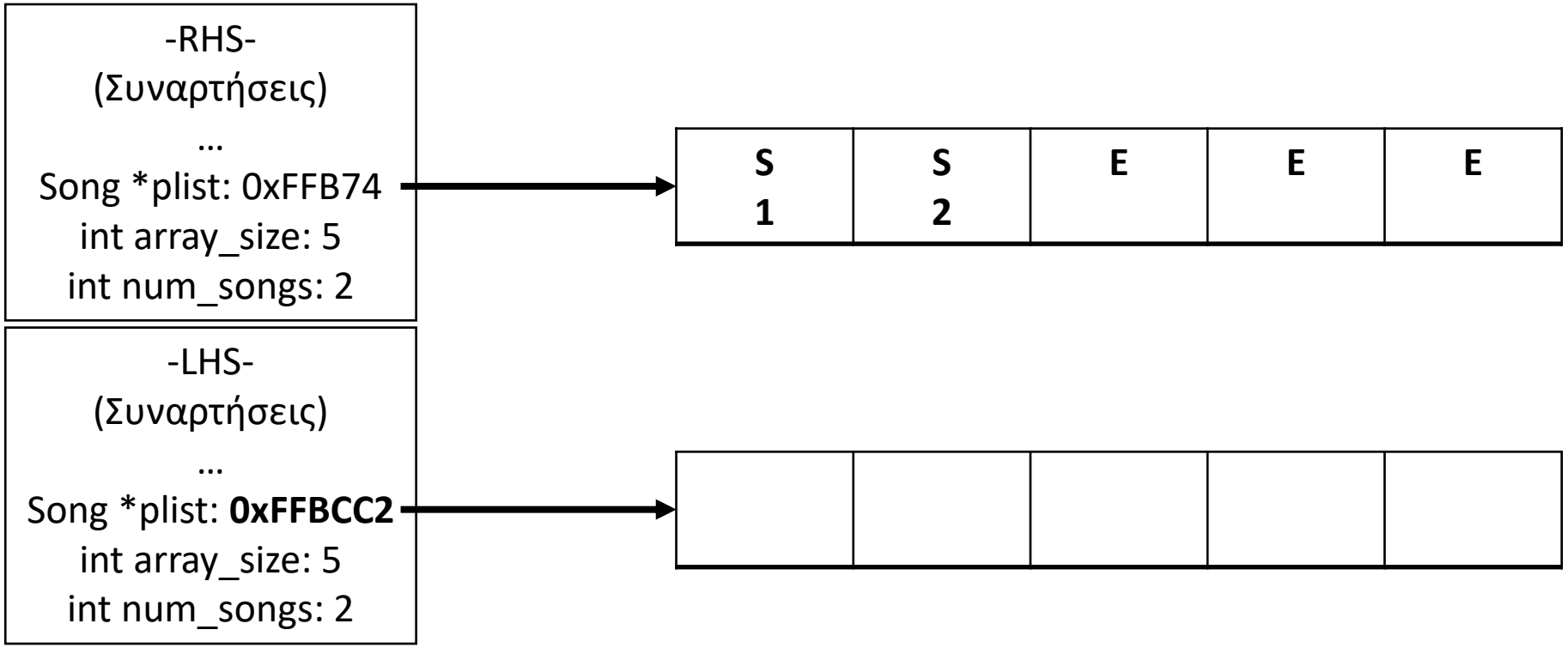
Ρηχή αντιγραφή με ανάθεση

- `LHS = RHS;`
- Ο τελεστής ανάθεσης **που αυτόματα δημιουργείται** από το μεταγλωττιστή πραγματοποιεί ρηχή αντιγραφή.



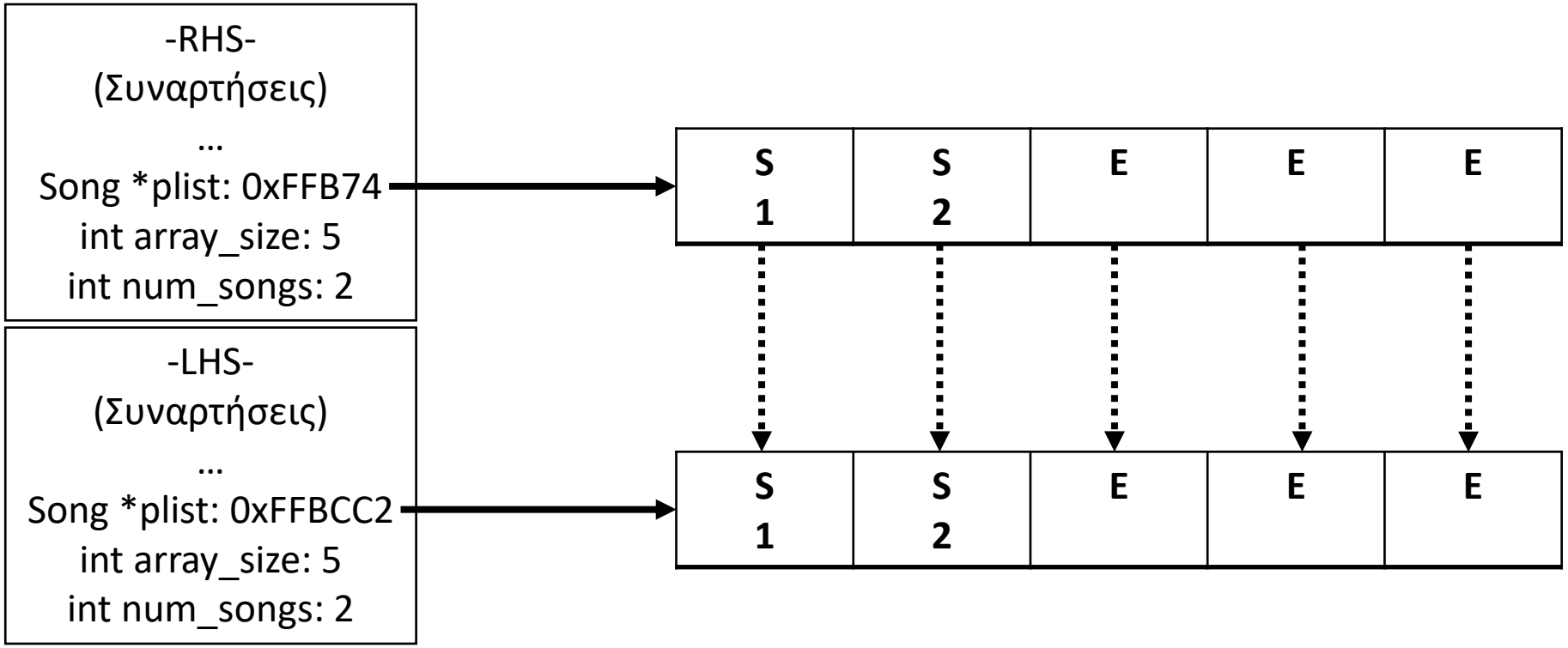
Βαθιά αντιγραφή με ανάθεση

- `LHS = RHS;`
- Καθώς ο πίνακας που έχει δεσμεύσει το LHS για `plist` δεν έχει το σωστό μέγεθος, θα πρέπει να αποδεσμευθεί η μνήμη του και να δεσμευθεί μνήμη κατάλληλου μεγέθους.



Βαθιά αντιγραφή με ανάθεση

- `LHS = RHS;`
- Στη συνέχεια θα πρέπει να αντιγραφούν τα στοιχεία του `plist` του RHS στο LHS, και επιπλέον να αντιγραφούν τα υπόλοιπα μέλη δεδομένων.



Κατασκευαστής αντιγραφής και αντιγραφή ανάθεσης

- Αν οριστεί κατασκευαστής αντιγραφής, και δεν οριστεί κάποιος άλλος κατασκευαστής, ο μεταγλωττιστής **δεν θα δημιουργήσει** προκαθορισμένο κατασκευαστή.
- Ο τελεστής ανάθεσης θα πρέπει να είναι συνάρτηση μέλος και όχι friend συνάρτηση.
- Ο τελεστής ανάθεσης ονομάζεται αλλιώς και τελεστής αντιγραφής και υλοποιεί την αντιγραφή ανάθεσης (copy assignment).
- Ο τελεστής ανάθεσης πάντα τελειώνει με:

```
return *this;
```

Κώδικας song και playlist για shallow copy και deep copy

https://github.com/chgogos/oop/blob/master/various/COP3330/lect12/playlist_shallow_copy.cpp

https://github.com/chgogos/oop/blob/master/various/COP3330/lect12/playlist_deep_copy.cpp

https://github.com/chgogos/oop/blob/master/various/COP3330/lect12/playlist_deleted.cpp

Ερωτήσεις σύνοψης

- Ποιο είναι το πρωτότυπο του κατασκευαστή αντιγραφής (copy constructor);
- Ποια είναι η διαφορά ανάμεσα στη ρηχή αντιγραφή (shallow copy) και στη βαθιά αντιγραφή (deep copy); Ποια είναι η προκαθορισμένη;
- Πότε καλείται ο κατασκευαστής αντιγραφής για ένα αντικείμενο;
- Γιατί η παράμετρος στον κατασκευαστή αντιγραφής περνά με const αναφορά; Τι θα συνέβαινε αν περνούσε με τιμή;
- Ποιο είναι το πρωτότυπο για τον τελεστή ανάθεσης (assignment operator);
- Ποιες είναι οι διαφορές ανάμεσα στον κατασκευαστή αντιγραφής και στην ανάθεση αντιγραφής;
- Τι επιστρέφει ο `operator=`;

Αναφορές

- http://www.cs.fsu.edu/~xyuan/cop3330/