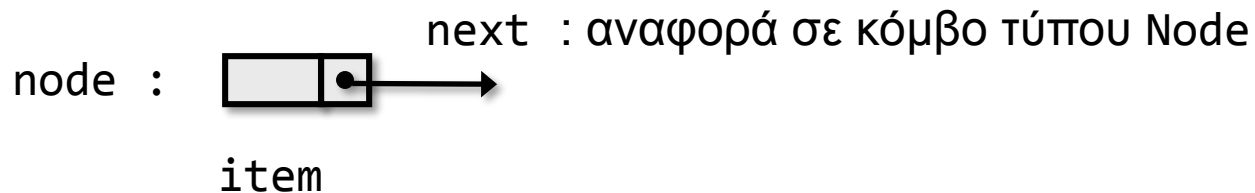


# Συνδεδεμένες Λίστες

Συνδεδεμένη λίστα: Αποθηκεύει ένα σύνολο στοιχείων σε κόμβους.  
Κάθε κόμβος περιλαμβάνει ένα σύνδεσμο προς τον επόμενο κόμβο.

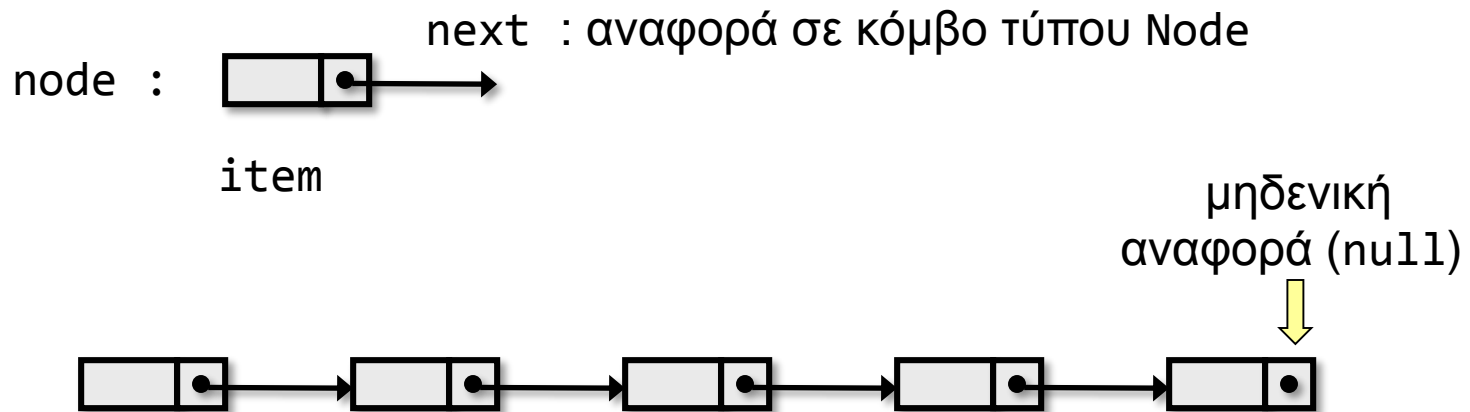
```
private class Node  
{Item item; Node next;}
```



# Συνδεδεμένες Λίστες

Συνδεδεμένη λίστα: Αποθηκεύει ένα σύνολο στοιχείων σε κόμβους.  
Κάθε κόμβος περιλαμβάνει ένα σύνδεσμο προς τον επόμενο κόμβο.

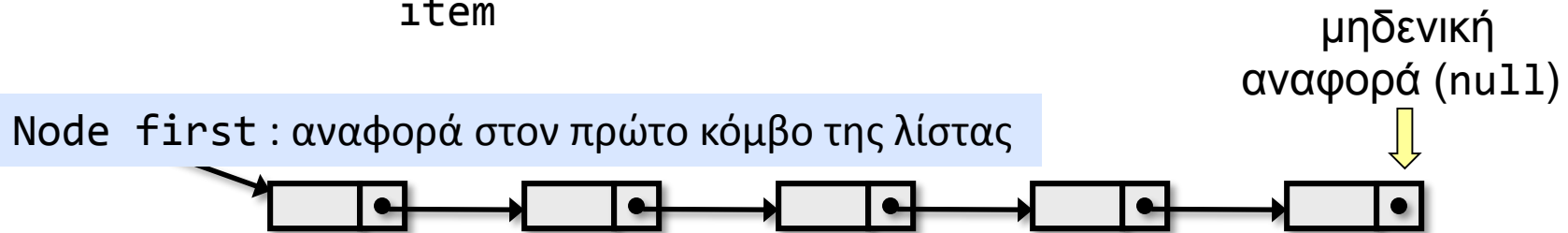
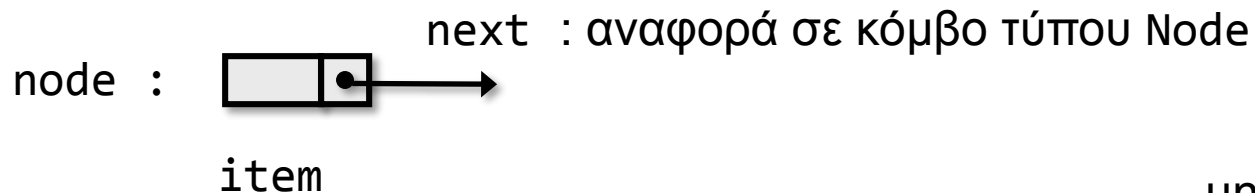
```
private class Node  
{Item item; Node next;}
```



# Συνδεδεμένες Λίστες

Συνδεδεμένη λίστα: Αποθηκεύει ένα σύνολο στοιχείων σε κόμβους.  
Κάθε κόμβος περιλαμβάνει ένα σύνδεσμο προς τον επόμενο κόμβο.

```
private class Node  
{Item item; Node next;}
```

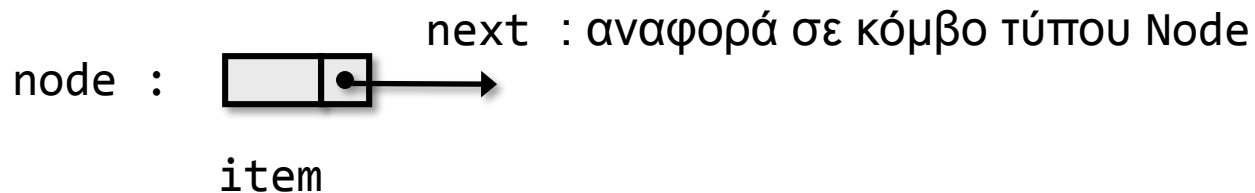


Για να προσπελάσουμε τα στοιχεία της λίστας χρειαζόμαστε μια αναφορά στον πρώτο κόμβο της λίστας. Η αναφορά αποθηκεύεται στη μεταβλητή `first` (τύπου `Node`).

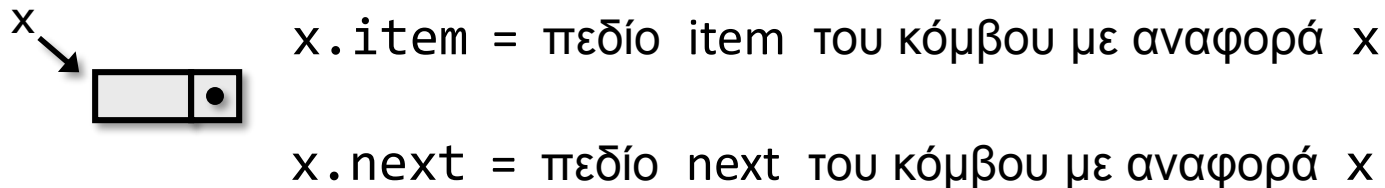
# Συνδεδεμένες Λίστες

Συνδεδεμένη λίστα: Αποθηκεύει ένα σύνολο στοιχείων σε κόμβους.  
Κάθε κόμβος περιλαμβάνει ένα σύνδεσμο προς τον επόμενο κόμβο.

```
private class Node  
{Item item; Node next;}
```



δημιουργία νέου κόμβου: `Node x = new Node();`

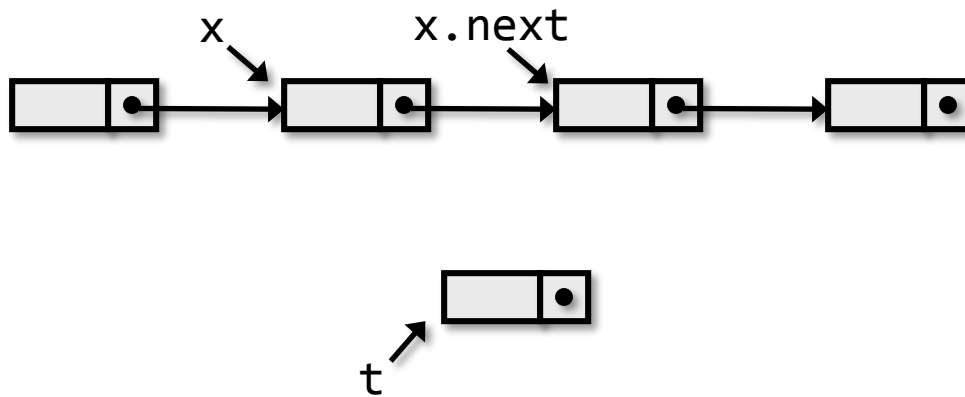


# Συνδεδεμένες Λίστες

Συνδεδεμένη λίστα: Αποθηκεύει ένα σύνολο στοιχείων σε κόμβους.  
Κάθε κόμβος περιλαμβάνει ένα σύνδεσμο προς τον επόμενο κόμβο.

```
private class Node  
{Item item; Node next;}
```

Εισαγωγή του κόμβου t μετά τον x

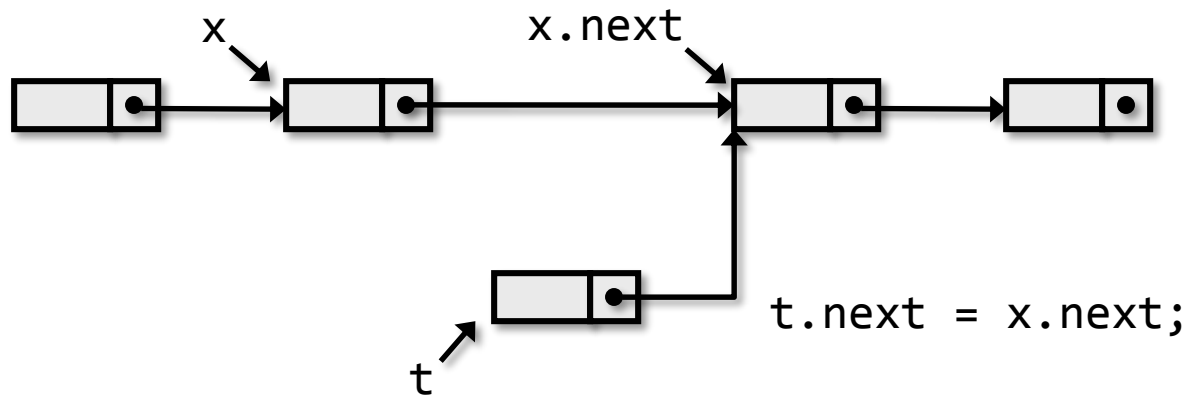


# Συνδεδεμένες Λίστες

Συνδεδεμένη λίστα: Αποθηκεύει ένα σύνολο στοιχείων σε κόμβους.  
Κάθε κόμβος περιλαμβάνει ένα σύνδεσμο προς τον επόμενο κόμβο.

```
private class Node  
{Item item; Node next;}
```

Εισαγωγή του κόμβου t μετά τον x

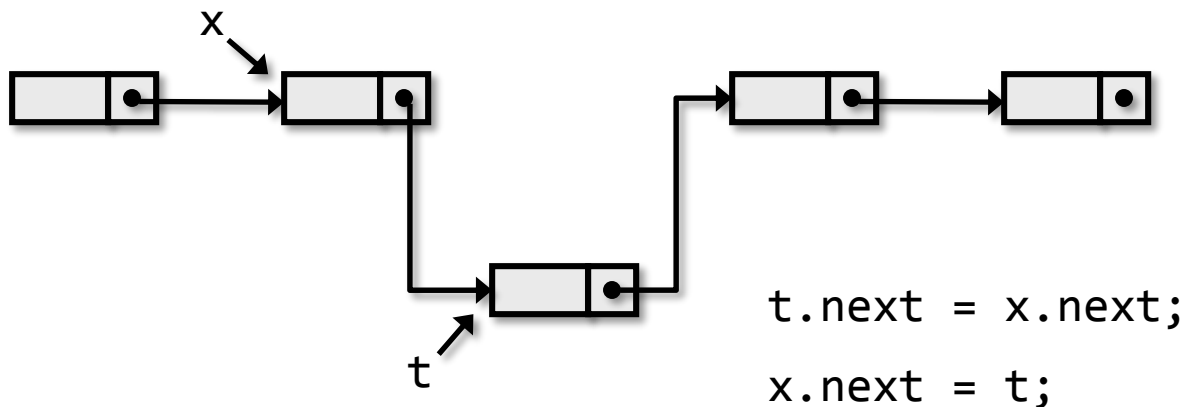


# Συνδεδεμένες Λίστες

Συνδεδεμένη λίστα: Αποθηκεύει ένα σύνολο στοιχείων σε κόμβους.  
Κάθε κόμβος περιλαμβάνει ένα σύνδεσμο προς τον επόμενο κόμβο.

```
private class Node  
{Item item; Node next;}
```

Εισαγωγή του κόμβου t μετά τον x

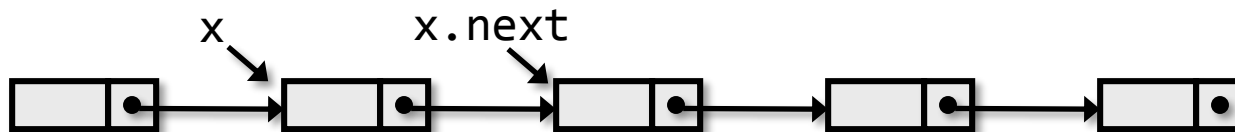


# Συνδεδεμένες Λίστες

Συνδεδεμένη λίστα: Αποθηκεύει ένα σύνολο στοιχείων σε κόμβους.  
Κάθε κόμβος περιλαμβάνει ένα σύνδεσμο προς τον επόμενο κόμβο.

```
private class Node  
{Item item; Node next;}
```

Διαγραφή του κόμβου μετά τον x



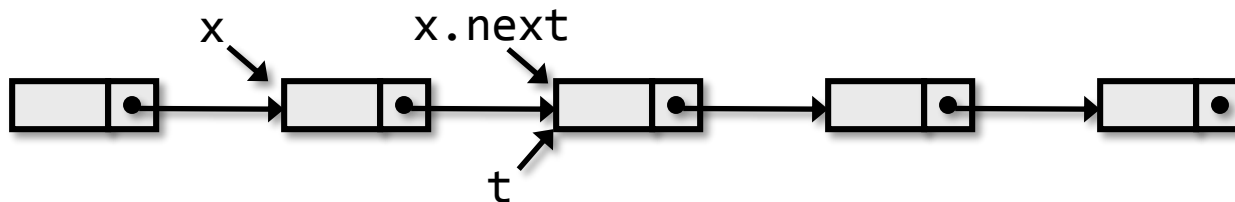


# Συνδεδεμένες Λίστες

Συνδεδεμένη λίστα: Αποθηκεύει ένα σύνολο στοιχείων σε κόμβους.  
Κάθε κόμβος περιλαμβάνει ένα σύνδεσμο προς τον επόμενο κόμβο.

```
private class Node  
{Item item; Node next;}
```

Διαγραφή του κόμβου μετά τον x



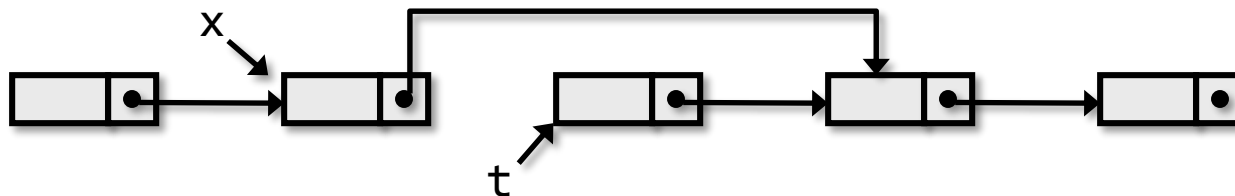
```
t = x.next;
```

# Συνδεδεμένες Λίστες

Συνδεδεμένη λίστα: Αποθηκεύει ένα σύνολο στοιχείων σε κόμβους.  
Κάθε κόμβος περιλαμβάνει ένα σύνδεσμο προς τον επόμενο κόμβο.

```
private class Node  
{Item item; Node next;}
```

Διαγραφή του κόμβου μετά τον x



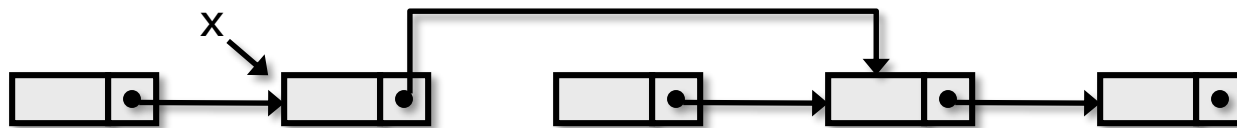
```
t = x.next;  x.next = t.next;
```

# Συνδεδεμένες Λίστες

Συνδεδεμένη λίστα: Αποθηκεύει ένα σύνολο στοιχείων σε κόμβους.  
Κάθε κόμβος περιλαμβάνει ένα σύνδεσμο προς τον επόμενο κόμβο.

```
private class Node  
{Item item; Node next;}
```

Διαγραφή του κόμβου μετά τον x

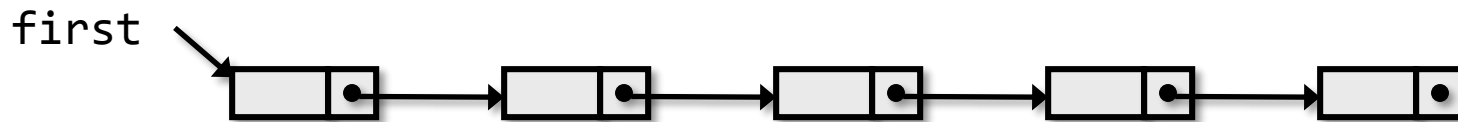


ή πιο απλά `x.next = x.next.next;`

# Συνδεδεμένες Λίστες

## Αναζήτηση στοιχείου σε λίστα

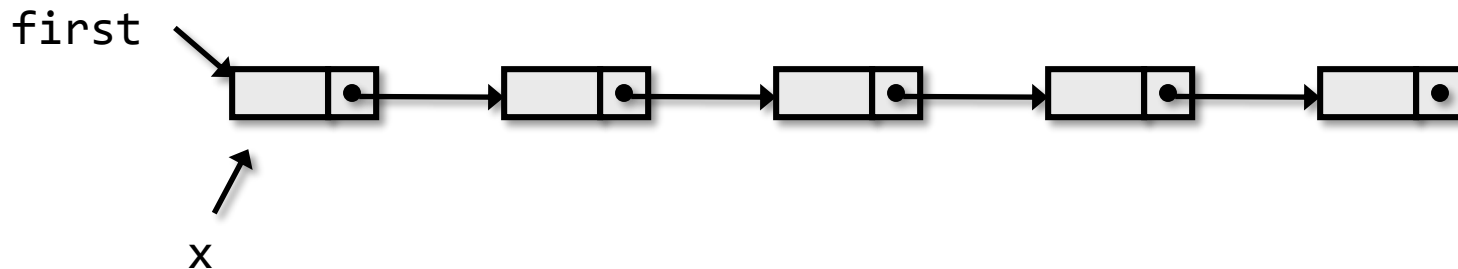
```
public boolean contains(Item item) {  
    Node x = first;  
    while (x != null) {  
        if (x.item.equals(item)) return true;  
        x = x.next;  
    }  
    return false;    // δεν βρέθηκε  
}
```



# Συνδεδεμένες Λίστες

## Αναζήτηση στοιχείου σε λίστα

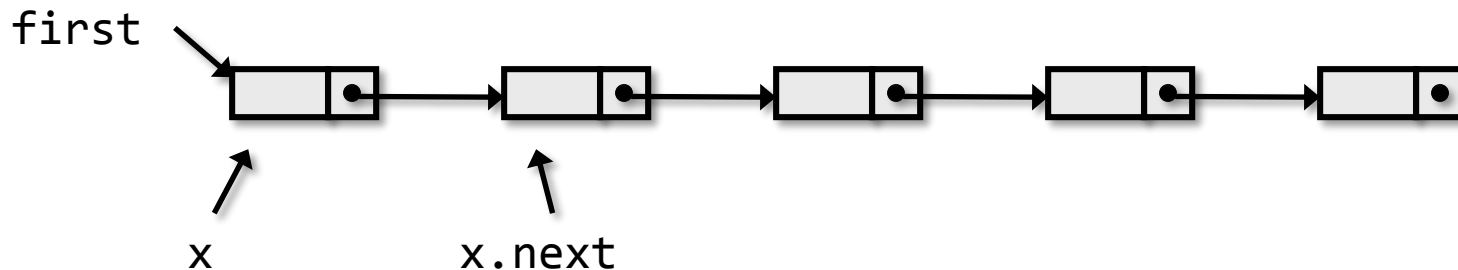
```
public boolean contains(Item item) {  
    Node x = first;  
    while (x != null) {  
        if (x.item.equals(item)) return true;  
        x = x.next;  
    }  
    return false;    // δεν βρέθηκε  
}
```



# Συνδεδεμένες Λίστες

## Αναζήτηση στοιχείου σε λίστα

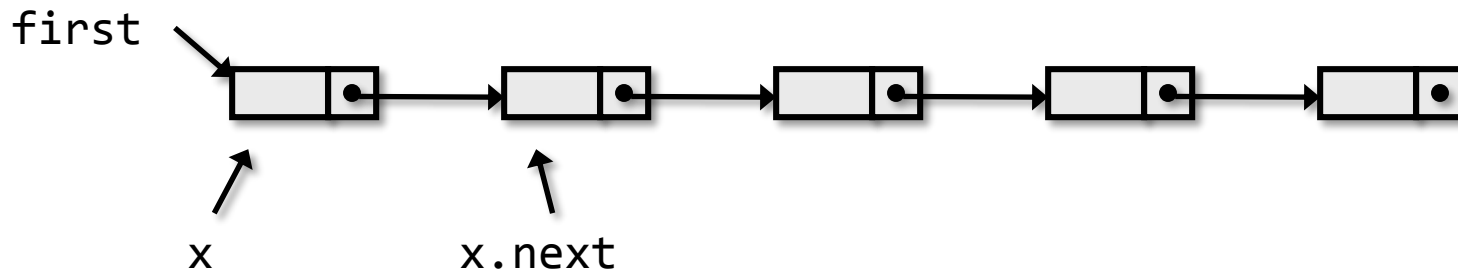
```
public boolean contains(Item item) {  
    Node x = first;  
    while (x != null) {  
        if (x.item.equals(item)) return true;  
        x = x.next;  
    }  
    return false;    // δεν βρέθηκε  
}
```



# Συνδεδεμένες Λίστες

Αναζήτηση σε συνδεδεμένη λίστα λέξεων μέχρι να βρούμε τη λέξη που βρίσκεται αμέσως μετά τη word στη λεξικογραφική διάταξη

```
public String successor(String word) {  
    Node x;  
    for (x = first; x != null; x = x.next)  
        if (x.str.compareTo(word) > 0) break;  
    if (x!=null) return x.str;  
    else return null;  
}
```

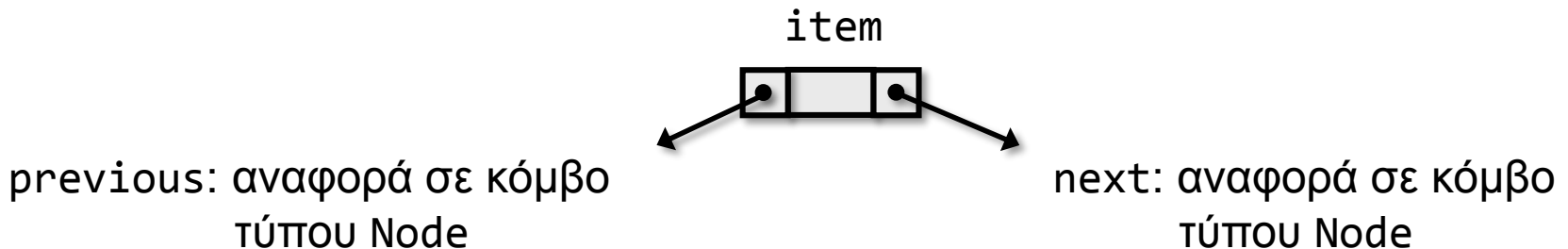


# Διπλά Συνδεδεμένες Λίστες

Κάθε κόμβος περιλαμβάνει ένα σύνδεσμο προς τον επόμενο και προς τον προηγούμενο κόμβο.

```
private class Node  
{Item item; Node next; Node previous}
```

node :



Θα πρέπει να προσαρμόσουμε κατάλληλα τις προηγούμενες μεθόδους ώστε να διατηρούν σωστά τη διπλά συνδεδεμένη λίστα.

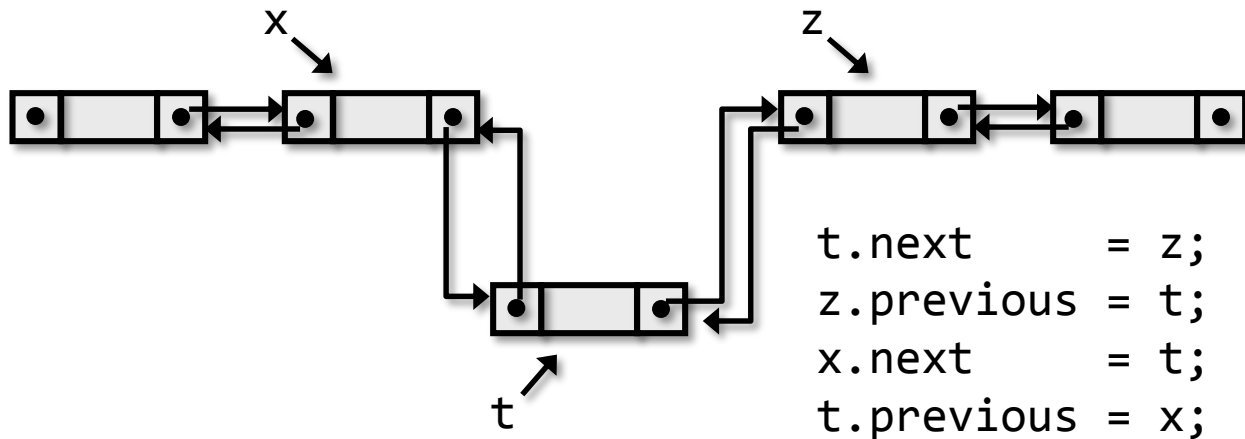


# Συνδεδεμένες Λίστες

Κάθε κόμβος περιλαμβάνει ένα σύνδεσμο προς τον επόμενο και προς τον προηγούμενο κόμβο.

```
private class Node  
{Item item; Node next; Node previous}
```

Εισαγωγή του κόμβου t μετά τον x

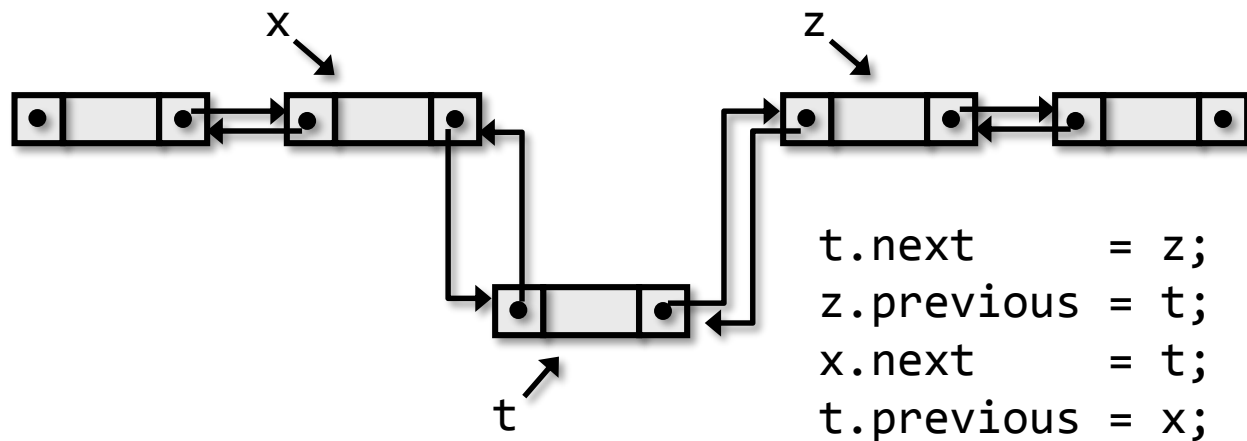


# Συνδεδεμένες Λίστες

Κάθε κόμβος περιλαμβάνει ένα σύνδεσμο προς τον επόμενο και προς τον προηγούμενο κόμβο.

```
private class Node  
{Item item; Node next; Node previous}
```

Εισαγωγή του κόμβου t μετά τον x



Προσοχή στις ειδικές περιπτώσεις! Θα πρέπει να ελέγξουμε αν ο x ή ο z είναι null

# Αποτελέσματα εκτέλεσης

```
Test WordList
linked list construction time = 882
number of linked list nodes = 8111

contains 'and' 3000 times
contains 'astonished' 3 times
contains 'boat' 4 times
contains 'path' 8 times
contains 'the' 3681 times
contains 'train' 0 times
contains 'tom' 673 times
contains 'wondered' 12 times

the 10 most frequent words are:
    the(3681)
    and(3000)
    a(1795)
    to(1705)
    of(1446)
    he(1181)
    was(1166)
    it(1116)
    in(937)
    that(890)
deleting 'in'
deleting 'it'
deleting 'was'

the remaining 10 most frequent words are:
    the(3681)
    and(3000)
    a(1795)
    to(1705)
    of(1446)
    he(1181)
    that(890)
    his(814)
    you(759)
    i(745)

sorting words in lexicographical order
first 10 words in lexicographical order are:
    a(1795)
    aadead(1)
    abandoned(4)
    abash(1)
    abashed(1)
    abide(1)
    ablaze(1)
    able(5)
    aboard(1)
    abounding(1)

total running time = 1149
```