Week 2 - Linked Lists

Brenden West - 2023

Contents

Learning Outcomes

- Linked Lists

Reading & Videos

- *LaFore - Ch. 5*
- https://www.geeksforgeeks.org/data-structures/linked-list/ (review)

Linked List

Linked Lists are a common alternative to arrays for structuring a data collection and are independent of any specific programming language.

Each item in a linked list is an object with data and a **pointer** to the next item in the list. Lists are **traversed** by following these pointers from the **head** item.

Linked lists have some key differences from arrays:

- Items are not stored in contiguous memory locations
- The list can be defined without knowing the list size
- List items do not have an **index position** and can only be accessed by traversing the list
- List items can be added or removed more easily
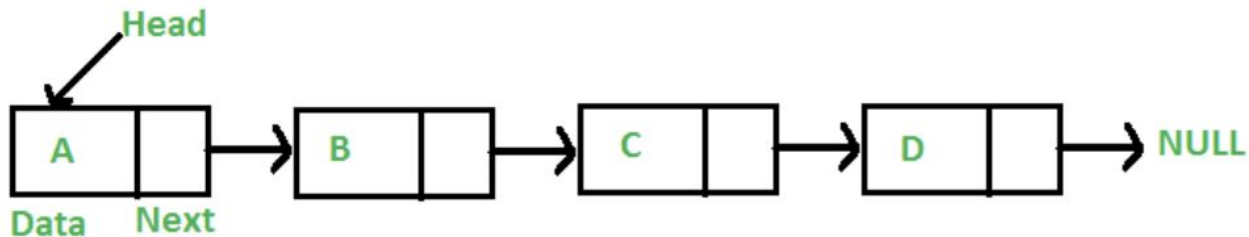
Linked Lists - construction

To build a linked list, we
- Define a Node object
- Create a `Node` for each item,
- Set the item (or data) field to the desired value
- Set the **next** field to the next node in the list

```
public class Main {
  private static class Node {
        private String item;
        private Node next;
  }

  public static void main(String[]
args) {

      // list operations

  }
}
```
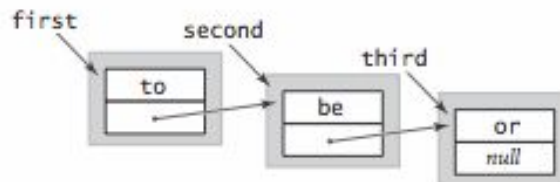
```
Node first  = new Node();
first.item  = "to";
```
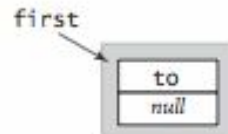


```
Node second = new Node();
second.item = "be";
first.next  = second;
```
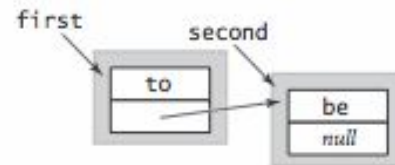


```
Node third  = new Node();
third.item  = "or";
second.next = third;
```

Linked List - traversing

Linked lists are traversed from the head node until reaching the desired node or the end of the list.

Traversing can use a **loop** or a **recursive** method.

```
Node first = new Node();
first.item = "to";

Node second = new Node();
second.item = "be";
first.next = second;

Node current = first;
while (current != null) {

System.out.println(current.item);
        current = current.next;
    }
```

Linked List - inserting items

Inserting a list item involves creating a new item and setting values for this and related list items.

The process differs slightly for these scenarios:

- Insert at start
- Insert at end
- Insert in the middle

```
Node newItem = new Node();

// insert item at start
newItem.next = first;
first = newItem;

// insert item at end
Node current = first;
while (current != null) {
  current = current.next;          }
current.next = newItem
```

Linked List - inserting items

Inserting a list item involves creating a new item and setting values for this and related list items.

The process differs slightly for these scenarios:

- Insert at start
- Insert at end
- Insert in the middle

```
Node newItem = new Node();

// insert after target node
String target = "target value";
Node current = first;
while (current != null) {
  if (current.item == target) {
        Node oldNext = current.next;
        current.next = newItem;
        newItem.next = oldNext;
        break;
  }
  current = current.next;          }
```