# Day 4

**ArrayList:**

**Exercise 1.**

With a file name as input, read all the words in this file as a list.

a) Print all the contens of file.
b) Print in reverse order.
c) Print all words (ending in "s").
d) Remove all duplicate words and print the rest.
e) Find all the duplicated words in this list
f) Print the words and the frequency
g) Replaces every word ending with an "es" with its lower version.
h) Removes every word in the list ending with an "as", case-insensitively
i) Insert a * after each element

**Exercise 2:**

With a file name as input, reads the content of this file with full of numbers then:

a) Prints the max, min, average of the numbers.
b) Sort all this number using Insertion Sort, Bubble Sort and Heap Sort.
c) Find all the prime number
d) Find all the Fibonacci value in this list
e) Remove all the duplcate values and print the rest (keep the order)

**Exercise 3:**

With two array lists of integers (not sorted) as input, returns a new list that contains only the duplicate elements both lists.

Example:

With two list [1, 4, 8, 9, 11, 15, 17, 28, 41, 59] [4, 7, 11, 17, 19, 20, 23, 28, 37, 59, 81]

The returned list: [4, 11, 17, 28, 59]

**LinkedList:**

Use this declaration of the Node class:

```
final class Node
{
    char info;
    Node next;
    public Node(char letter, Node node)
    {
       info = letter;
       next = node;
    }
}
```

```
n1 = n4.next; n1.next = null; // Assignment
n3.next.next = new Node('B',null); //Node instantiation
n1.info = n4.next.info; n1.next.info = 'C'; // Assignment

if (n1==null) {...} else {...} // 4.If statement

while ( (n1!=null) && (n1.info=='A') ) {...} // While loop
```

**Exercise 4:**

Implement all the necessary methods for LinkedList.

Insert at the beginning of the list

Inserts to the end of this list.

Return the first element in the list

Removes the first element in the list.

Returns the last element in the list

Removes the last element in the list.

Removes all nodes from the list

Returns true if this list contains the specified element

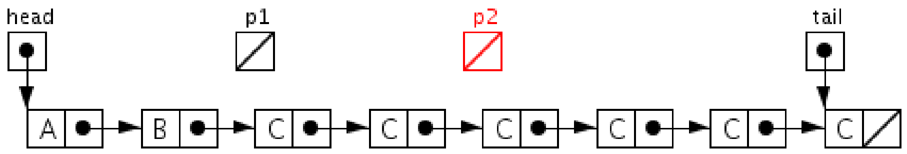
Returns the data at the givem position in the list

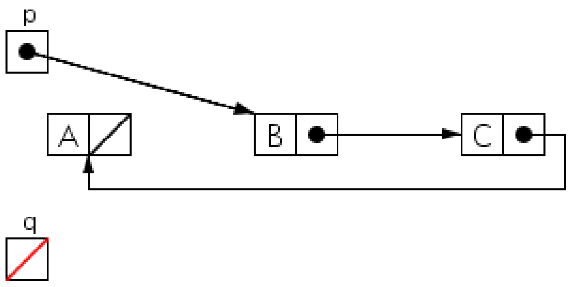
Inserts a new node after a node containing the key

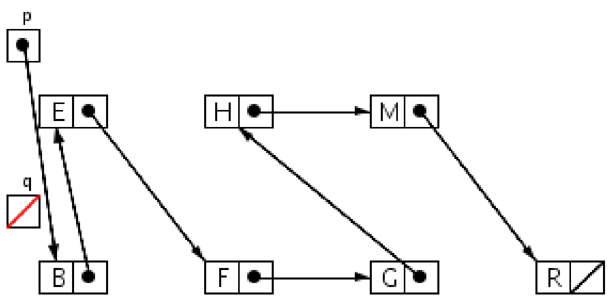
Inserts a new node before a node containing the key
Removes the first occurrence of the specified element in this list

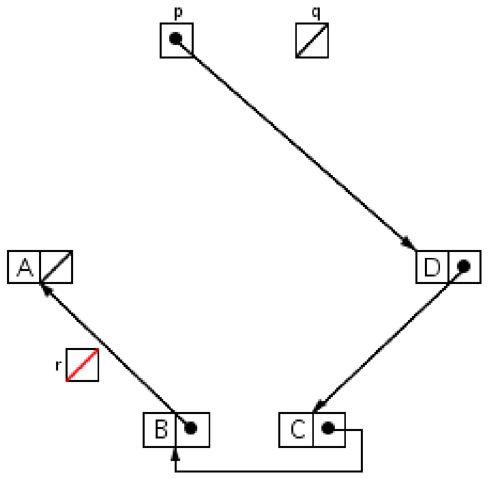
**Exercise 5:**

Using the Exercise 1, implement the following LinkedList





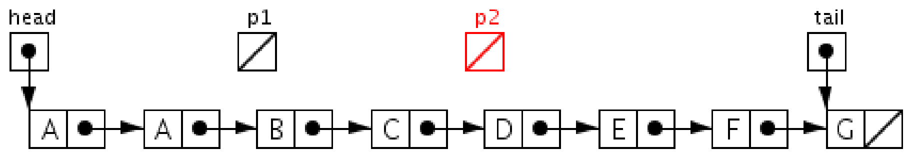




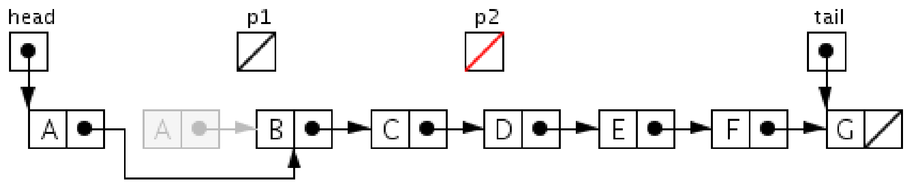
**Exercise 6:**

With a linked list of characters as user input, remove all consecutive duplite values (non-consecutive duplicates need not be deleted). The original order must be kept.

Original:



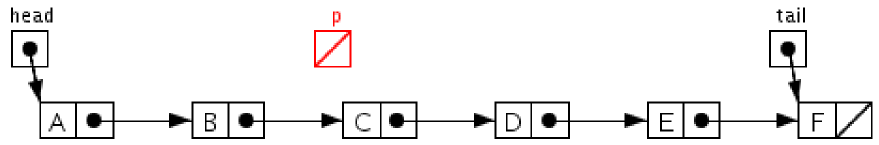
After remove:



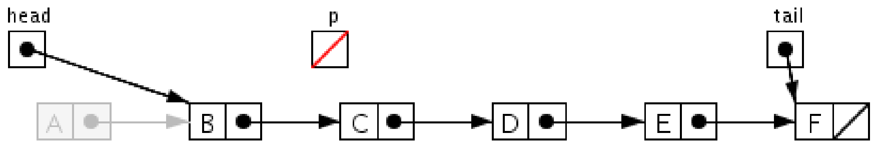
**Exercise 7:**

With a linked list of characters as user input, removes all Nodes with info 'A' from the list. The original order must be kept.

Original:



After:


head
p
tail
A
B
C
D
E
F