SEARCHING IN A LINKED LIST

Given a singly linked list, write a function which will search for a value in the list.

Signature of the function will be as shown below:

```
1    bool search(Node*head, int x);
```

head points to the first Node in the list. The function should return true is x is present in the list as false.

In a linked list only linear search can be made (No-Binary search even if the list is sorted). Hence the solution is simple,

Check each Node of the list,

if value in Node == x

    return TRUE

When the list is over return FALSE

Non-Recursive Solution:

```
1    bool search(Node* head, int x)
2    {
3        for(; head != NULL; head = head->link)
4        {
5            if(head->data == x)
6                return true;
7        }
8        return false;
9    }
```

Recursive Solution:

Before getting fascinated with recursion, you should know that recursion comes with its side effects

```
1     bool search(Node* head, int x)
2     {
3         if(head == NULL)
4             return false;
5
6         if(head->data == x)
7             return true;
```

```
8
9           search(head->link, x);
10      }
```

Time Complexity: O(n) for both recursive & Non-Recursive implementation

Extra Space Used: For Non-Recursive O(1). For Recursive Solution, O(n) (Space used in the stack to store activation records of function getting called recursively.

Source: http://www.ritambhara.in/searching-in-a-linked-list/