**King Saud University**

**College of Computer and Information Sciences Computer Science Department**

**CSC 212 First Semester 1437-1438**

**Tutorial # 7**

**Important**: This tutorial has an online part, which you should complete on LMS (tutorial section). The deadline for online task is Monday 8 November at 8:00 A.M

# Problem 1

1. ~~Write the recursive method~~ **~~search~~** ~~member of the class Linkedlist. That searches for an element e and return true if found. False otherwise.~~ **~~Don’t use auxiliary data structures and don’t call any of the LinkedList methods.~~**

public boolean search (T e){  
 Node<T> temp= head;  
 return search\_rec(e, temp);  
}  
private boolean search\_rec(T e, Node<T> temp){  
 if(temp==null)  
 return false;  
 else{  
 if(temp.data.equals(e))  
 return true;  
 return search\_rec(e,temp.next);  
 }  
 }

1. ~~Write the static recursive method~~ **~~SearchList~~**~~. That search for an element e in a List l and return true if found. False otherwise.~~ **~~Don’t use auxiliary data structures.~~**

public boolean search (list<T> temp ,T e){  
 if(temp.empty())  
 return false;  
 temp.findFirst();  
 return search\_rec(temp,e);  
}  
private boolean search\_rec(list<T> temp,T e){  
 if(temp.last()){  
 if(!temp.retrieve().equals(e))  
 return false;  
 return true;  
 }  
 else if(temp.retrieve().equals(e))  
 return true;  
 else{  
 temp.findNext();  
 return search\_rec(temp,e);  
 }  
}

# ~~Problem 2~~

1. ~~Write the recursive static method~~ **~~CopyStack~~**~~, that takes two Stacks s1 and s2 and copies all the elements in s1 into s2 in the same order.~~ **~~Don’t use auxiliary data structures. s1 should not change at the end of the method.~~**

public static<T> void CopyStack(Stack<T> s1, Stack<T> s2){

if(s1.empty())  
 return;

T e = s1.pop();

CopyStack(s1,s2);

s1.push(e);

s2.push(e);

}

1. ~~Write the recursive static method~~ **~~SearchStack~~**~~, that takes a Stack s and an element e and search for the element e and return true if found. False otherwise.~~ **~~Don’t use auxiliary data structures. s should not change at the end of the method.~~**

public boolean searchStack(Stack<T> s, T e){  
 boolean flag;  
 if(s.empty())  
 return false;  
 else{  
 T temp=s.pop();  
 if(temp.equals(e))  
 flag=true;   
 else{  
 flag= searchStack(s,e);  
 }  
 s.push(temp);  
 return flag;  
 }  
}