LinkedList from scratch

package FirstJavaProgram.com.company;  
  
public class Main {  
  
 Node head;  
 private int size;  
  
 Main()  
 {  
 this.size=0;  
 }  
  
 class Node {  
 String data;  
 Node next;  
  
 Node(String data) {  
 this.data = data;  
 this.next = null;  
 }  
 }  
  
 *//add a node - first* public void addFirst(String data) {  
 Node newNode = new Node(data);  
 if (head == null) {  
 head = newNode;  
 size++;  
 return;  
 }  
 size++;  
 newNode.next = head;  
 head = newNode;  
 }  
  
 *//add last* public void addLast(String data) {  
 Node newNode = new Node(data);  
 if (head == null) {  
 head = newNode;  
 size++;  
 return;  
 }  
 size++;  
 Node currNode = head;  
 while (currNode.next != null) {  
 currNode = currNode.next;  
 }  
  
 currNode.next = newNode;  
 }  
  
 *//print* public void printll() {  
 if (head == null) {  
 System.*out*.print("list is empty");  
 return;  
 }  
 Node currNode = head;  
 while (currNode != null) {  
 System.*out*.print(currNode.data + " ->");  
 currNode = currNode.next;  
 }  
  
 System.*out*.print("Null");  
 }  
  
 *//delete first* public void deleteFirst()  
 {  
 if(head==null)  
 {  
 System.*out*.println("list is empty");  
 return;  
 }  
 size--;  
  
 head=head.next;  
 }  
  
 *//delete last* public void deleteLast()  
 {  
 if(head==null)  
 {  
 System.*out*.println("list is empty");  
 return;  
 }  
 size--;  
 Node prev=head;  
 Node curr=head.next;  
 if(head.next==null)  
 {  
 head=null;  
 return;  
 }  
  
 while(curr.next!=null)  
 {  
 prev=prev.next;  
 curr=curr.next;  
 }  
 prev.next=null;  
 }  
  
 *//return the size* public int getSize()  
 {  
 return size;  
 }  
  
 *//reverse the linked list* public void reverseIterate()  
 {  
 if(head==null || head.next==null)  
 {  
 return;  
 }  
 Node prev=head;  
 Node curr=head.next;  
 while(curr!=null)  
 {  
 Node nextn=curr.next;  
 curr.next=prev;  
  
 *//update* prev=curr;  
 curr=nextn;  
 }  
 head.next=null;  
 head=prev;  
 }  
  
 public static void main(String[] args) {  
 *// write your code here* Main list = new Main();  
  
 list.addFirst("this");  
 list.addFirst("is");  
 list.addLast("diksha");  
 list.printll();  
 System.*out*.println();  
 list.addLast("pathak");  
 list.printll();  
 System.*out*.println();  
 list.deleteFirst();  
 list.printll();  
 System.*out*.println();  
 list.deleteLast();  
 list.printll();  
 int x= list.getSize();  
 System.*out*.println();  
 System.*out*.println(x);  
  
 list.reverseIterate();  
 System.*out*.println();  
 list.printll();  
  
  
  
 }  
}

Output

is ->this ->diksha ->Null

is ->this ->diksha ->pathak ->Null

this ->diksha ->pathak ->Null

this ->diksha ->Null

2

diksha ->this ->Null

LinkedList class from Collection framework

package FirstJavaProgram.com.company;  
  
import java.util.LinkedList;  
  
public class Main {  
  
 public static void main(String[] args) {  
 *// write your code here* LinkedList<String> list = new LinkedList<String>();  
  
 list.addFirst("diksha");  
 list.addFirst("pathak");  
 System.*out*.println(list);  
 list.addLast("this");  
 System.*out*.println(list);  
  
 System.*out*.println(list.size());  
 for(int i=0; i<list.size(); i++)  
 {  
 System.*out*.print(list.get(i)+ " ->");  
 }  
  
  
 System.*out*.println("NULL");  
  
 list.removeFirst();  
 System.*out*.println(list);  
  
 System.*out*.println();  
  
 list.removeLast();  
 System.*out*.println(list);  
  
 list.remove(0);  
 System.*out*.println(list);  
  
  
  
 }  
}

Stack implementation using LinkedList

package FirstJavaProgram.com.company;  
  
public class Main {  
  
 static class Node{  
 int data;  
 Node next;  
 public Node(int data)  
 {  
 this.data=data;  
 next=null;  
 }  
 }  
  
 static class Stack{  
 public static Node *head*;  
  
 public static boolean isEmpty()  
 {  
 if(*head*==null)  
 {  
 return true;  
 }  
 return false;  
 }  
 public static void push(int data)  
 {  
 Node newnode=new Node(data);  
  
 if(*isEmpty*())  
 {  
 *head*=newnode;  
  
 }  
 newnode.next=*head*;  
 *head*=newnode;  
 }  
  
 public static int pop()  
 {  
 if(*isEmpty*())  
 {  
 return -1;  
 }  
  
 int top=*head*.data;  
 *head*=*head*.next;  
 return top;  
 }  
  
 public static int peek()  
 {  
 if(*isEmpty*())  
 {  
 return -1;  
 }  
  
   
 return *head*.data;  
 }  
 }  
  
 public static void main(String[] args) {  
 *// write your code here* Stack s =new Stack();  
  
 s.*push*(1);  
 s.*push*(2);  
 s.*push*(3);  
 s.*push*(4);  
 s.*push*(5);  
  
 while(!s.*isEmpty*()){  
 System.*out*.println(s.*peek*());  
 s.*pop*();  
 }  
  
  
  
 }  
}

stack implementation using array lists

package FirstJavaProgram.com.company;  
  
import java.util.ArrayList;  
  
public class Main {  
  
 static class Stack{  
 static ArrayList<Integer> *list* = new ArrayList<>();  
  
 public static boolean isEmpty()  
 {  
 return *list*.size()==0;  
 }  
  
 public static void push(int data)  
 {  
 *list*.add(data);  
 }  
  
 public static int pop()  
 {  
  
 if(*isEmpty*())  
 {  
 return -1;  
  
 }  
 int top=*list*.get(*list*.size()-1);  
 *list*.remove(*list*.size()-1);  
  
 return top;  
 }  
  
 public static int peek()  
 {  
 if(*isEmpty*())  
 return -1;  
 return *list*.get(*list*.size()-1);  
 }  
 }  
  
 public static void main(String[] args) {  
 *// write your code here* Stack s =new Stack();  
  
 s.*push*(1);  
 s.*push*(2);  
 s.*push*(3);  
 s.*push*(4);  
 s.*push*(5);  
  
 while(!s.*isEmpty*()){  
 System.*out*.println(s.*peek*());  
 s.*pop*();  
 }  
  
  
  
 }  
}

implement stack using collections framework

package FirstJavaProgram.com.company;  
  
import java.util.ArrayList;  
import java.util.Stack;  
  
public class Main {  
  
 public static void main(String[] args) {  
 *// write your code here* Stack<Integer> s=new Stack<>();  
   
  
 s.push(1);  
 s.push(2);  
 s.push(3);  
 s.push(4);  
 s.push(5);  
  
 while(!s.isEmpty()){  
 System.*out*.println(s.peek());  
 s.pop();  
 }  
  
  
  
 }  
}

bubble sort

package FirstJavaProgram.com.company;  
  
public class Main {  
 public static void printarr(int[] arr)  
 {  
 for(int k=0; k<arr.length; k++)  
 {  
 System.*out*.println(arr[k]);  
 }  
 }  
 public static void main(String[] args) {  
  
 int[] arr={7,8,3,1,2};  
 int temp;  
  
 *//time complexity - O(n^2)  
 //bubble sort* for(int i=0; i<arr.length-1; i++)  
 {  
 for(int j=0; j<arr.length-i-1; j++)  
 {  
 if(arr[j]>arr[j+1])  
 {  
 temp=arr[j];  
 arr[j]=arr[j+1];  
 arr[j+1]=temp;  
 }  
 }  
  
 }  
 *printarr*(arr);  
 }  
}

selection sort

package FirstJavaProgram.com.company;  
  
public class Main {  
 public static void printarr(int[] arr)  
 {  
 for(int k=0; k<arr.length; k++)  
 {  
 System.*out*.println(arr[k]);  
 }  
 }  
 public static void main(String[] args) {  
  
 int[] arr={7,8,3,1,2};  
 int min,temp;  
  
 *//time complexity - O(n^2)  
 //selection sort* for(int i=0; i<arr.length-1; i++)  
 {  
 min=i;  
 for(int j=i+1; j<arr.length; j++)  
 {  
 if(arr[j]<arr[min])  
 {  
 min=j;  
 }  
 }  
 temp=arr[i];  
 arr[i]=arr[min];  
 arr[min]=temp;  
 }  
 *printarr*(arr);  
 }  
}

insertion sort

package FirstJavaProgram.com.company;  
  
public class Main {  
 public static void printarr(int[] arr)  
 {  
 for(int k=0; k<arr.length; k++)  
 {  
 System.*out*.println(arr[k]);  
 }  
 }  
 public static void main(String[] args) {  
 int[] arr={7,8,3,1,2};  
  
  
 *//time complexity - O(n^2)  
 //insertion sort* for (int i = 1; i < arr.length; ++i) {  
 int key = arr[i];  
 int j = i - 1;  
  
 */\* Move elements of arr[0..i-1], that are  
 greater than key, to one position ahead  
 of their current position \*/* while (j >= 0 && arr[j] > key) {  
 arr[j + 1] = arr[j];  
 j = j - 1;  
 }  
 arr[j + 1] = key;  
 }  
 *printarr*(arr);  
 }  
}