**Enser 1.**

**public class** taher{  
 **static int** findRepeating(**int** arr[], **int** n)  
 {  
  
 **int** ta = 0;  
 **for** (**int** i = 0; i < n - 1; i++)  
 ta = ta ^ (i + 1) ^ arr[i];  
 ta = ta ^ arr[n - 1];  
  
 **return** ta;  
 }  
 **public static void** main(String[] args)  
 {  
 **int** arr[] = {1,2,3,4,5,6,6,7,8,5,4};  
 **int** t = arr.length;  
 System.out.println(findRepeating(arr,t));  
 }  
}

**Enser 2.**

**public class** taher<t,d > {  
 **public** tarray1[];  
 **public** d array2[];  
 **public** ()  
 {}  
 **public** taher (t[] array1, d[] array2) {  
 **this**.array1 = array1;  
 **this**.array2 = array2;}  
  
 **public** <t> **void** Merging( t []array1,d[] array2) {  
 Object[] array3 = **new** Object[array1.length+array2.length];  
  
 **for** (**int** i = 0; i <array1.length ; i++) {  
 array3[i] = (Object) array1[i];}  
 **for** (**int** i = 0; i < array2.length; i++) {  
 array3[array1.length + i] = (Object) array2[i];  
 }  
 **for** (**int** i = 0; i < array3.length; i++) {  
 System.out.println(array3[i]);}}  
  
 **public static void** main(String[] args) {  
 Int arr1[]={10,20,30,40,50,60,70,80};  
 Int arr2 []={90,100 ,110 };

**import** java.util.Arrays;  
**public taherdafallh**<T> {  
**public static** <T> **void** reverse(T[] array){  
 **int** q=array.length-1;  
 **for** (**int** i = 0; i < array.length/2; i++) {  
 T tem=(T) array[i];  
 array [i]=array[s];  
 array[q]=tem;  
 q--;  
 }System.out.println(Arrays.toString(array));  
 **for** (**int** i = 0; i <array.length ; i++)  
 {}}  
  
 **public static void** main(String[] args) {  
 String arr1[]={**"taher"**,**"dafallah"**};  
 Integer arr2[]={,1,5,77,3,8,5};  
 reverse(arr1);  
 reverse(arr2);}  
  
  
}

}  
}

**Enser 3.**

**public class** tLinkedList<E> {  
 **private static class** Node<E> {  
 **private** E data;  
 **private** Node<E> next;  
 **public** Node(E data) {  
 **this**(data,**null**);  
 }  
 **public** Node(E data, Node<E> next) {  
 **this**.data = data;  
 **this**.next = next;  
 }  
 **public** E getData() { **return this**.data; }  
 **public** Node<E> getNext() { **return this**.next; }  
 **public void** setNext(Node<E> node) { **this**.next = node; }  
 **public** String toString() { **return this**.data.toString(); }  
 }  
 **private** Node<E> tail = **null**;  
 **private int** size = 0;  
  
 **public** tLinkedList() { }  
 **public int** size() { **return** size; }  
 **public boolean** isEmpty() { **return** size == 0; }  
  
 **public** String toString() {  
 StringBuffer sb = **new** StringBuffer();  
 sb.append(**"[ "**);  
 **if**(isEmpty()) {  
 sb.append(**"]"**);  
 **return** sb.toString();  
 } **else** {  
 Node<E> runner = tail.getNext();  
 sb.append(runner);  
 **for**(**int** i=1; i<size; i++) {  
 runner = runner.getNext();  
 sb.append(**", "**);  
 sb.append(runner);  
 }  
 sb.append(**" ]"**);  
 **return** sb.toString();  
 }}  
 **public void** rotate() {  
 **if**(tail != **null**) {  
 tail = tail.getNext();  
 }  
 }  
  
 **public void** addFirst(E e) {  
 **if**( size==0 ) {  
 tail = **new** Node<E>(e);  
 tail.setNext(tail);  
 } **else** {  
 Node<E> newest = **new** Node<E>(e, tail.getNext());  
 tail.setNext(newest);  
 }  
 size++;  
 }  
  
 **public void** addLast(E e) {  
 addFirst(e);  
 tail = tail.getNext();  
 }  
  
 **public** E removeFirst() {  
 **if**(isEmpty()) {  
 **return null**;  
 }  
 Node<E> head = tail.getNext();  
 size--;  
 **if**(head.equals(tail)) {  
 E value = tail.getData();  
 tail = **null**;  
 **return** value;  
  
 } **else** {  
 tail.setNext(head.getNext());  
 **return** head.getData();  
 }  
 }  
  
 **public** E first() {  
 **if**(isEmpty()) {  
 **return null**;  
 }  
 **return** tail.getNext().getData();}  
  
 **public** E last() {  
 **if**(isEmpty()) {  
 **return null**;  
 }  
 **return** tail.getData();  
 }  
 **public static void** main(String[] args) {  
 tLinkedList<Integer> t = **new** tLinkedList<Integer>();  
 **for**(**int** i=5; i<10; i++) {  
 t.addFirst(i);}  
 System.out.println(t);  
   
 **for**(**int** i=150; i<160; i++) {  
 t.addLast(i);}  
 System.out.println(t);  
   
 t.rotate();  
 t.rotate();  
 t.rotate();  
 System.out.println(t);  
   
 t.removeFirst();  
 t.removeFirst();  
 System.out.println(t);  
  
 }  
}

**Enser 4.**

**public class** Node<E> {  
 **int** data;  
 Node next;  
 **public** Node(**int** data, Node next) {  
 **this**.data = data;  
 **this**.next = next;  
 }  
  
 Node(**int** t)  
   
 {  
 **this**.data = t;  
 **this**.next = **null**;}}  
**class** LinkedList<E>  
{  
 Node s;  
 LinkedList()  
 {  
 start = **null**;  
 }  
 **public void** push(**int** data)  
 {  
 **if**(**this**.s == **null**)  
 {  
 Node temp = **new** Node(data);  
 **this**.s = temp;  
 }  
 **else** {  
 Node temp = **new** Node(data);  
 temp.next = **this**.start;  
 **this**.st = temp;}}  
 **public int** findSecondL(Node ptr)  
 {  
 Node temp = ptr;  
 **if**(temp == **null** || temp.next == **null**)  
 **return** -1;  
 **while**(temp.next.next != **null**)  
 {  
 temp = temp.next;  
 }  
 **return** temp.data;}  
 **public static void** main(String[] args){  
 LinkedList a=**new** LinkedList();  
 a.push(1);  
 a.push(2);  
 a.push(3);  
 a.push(4);  
 a.push(5);  
 System.out.println(a.findSecondL(a.st));}}

**Enser 5**.

**import** java.util.Stack;  
**public class** Converts {  
  
 **public static** Stack<Integer> stackPush(Stack<Integer> stack){  
  
 **for**(**int** i = 0; i < 5; i++){  
  
 **int** push = (**int**) stack.push(i);  
  
 System.out.println(push);  
 }  
 **return** stack;  
  
 }  
  
 **public static void** pop(Stack<Integer> stack){  
  
 **for**(**int** i = 0; i < 5; i++){  
  
 **int** pop = (**int**) stack.pop();  
 System.out.println(pop);  
  
  
 }  
  
 }  
  
  
 **public static void** peek(Stack<Integer> stack){  
  
 **int** peel = (**int**) stack.peel();  
  
 System.out.println(**"Top of the element is: "** + peel);  
  
 }  
  
  
 **public static void** search(Stack<Integer> stack, **int** element){  
  
 System.out.println(**"Element is searched : "**);  
  
 Integer search = (Integer) stack.search(element);  
  
 **if**(element == -1){  
  
 System.out.println(**"Stack is emptya."**);  
 }  
 **else**{  
 System.out.println(**"Element is: "** + search);  
 }  
  
  
 }  
  
 **public static** Stack<**int**> Converts(Stack<Integer> stackr, Stack<**int**> stackt{  
  
  
 stackt = stackPush(stackr);  
  
 System.out.println(**"Stack Converts successfully: "**+ stackt);  
  
 **return** stackt;  
  
 }  
  
 **public static void** main(String[] args) {  
  
 Stack<Integer> stackr = **new** Stack<>();  
 Stack<Integer> stackt = **new** Stack<>();  
  
 Stack<**int**> Stackd = Converts(stackr, stackt);}}