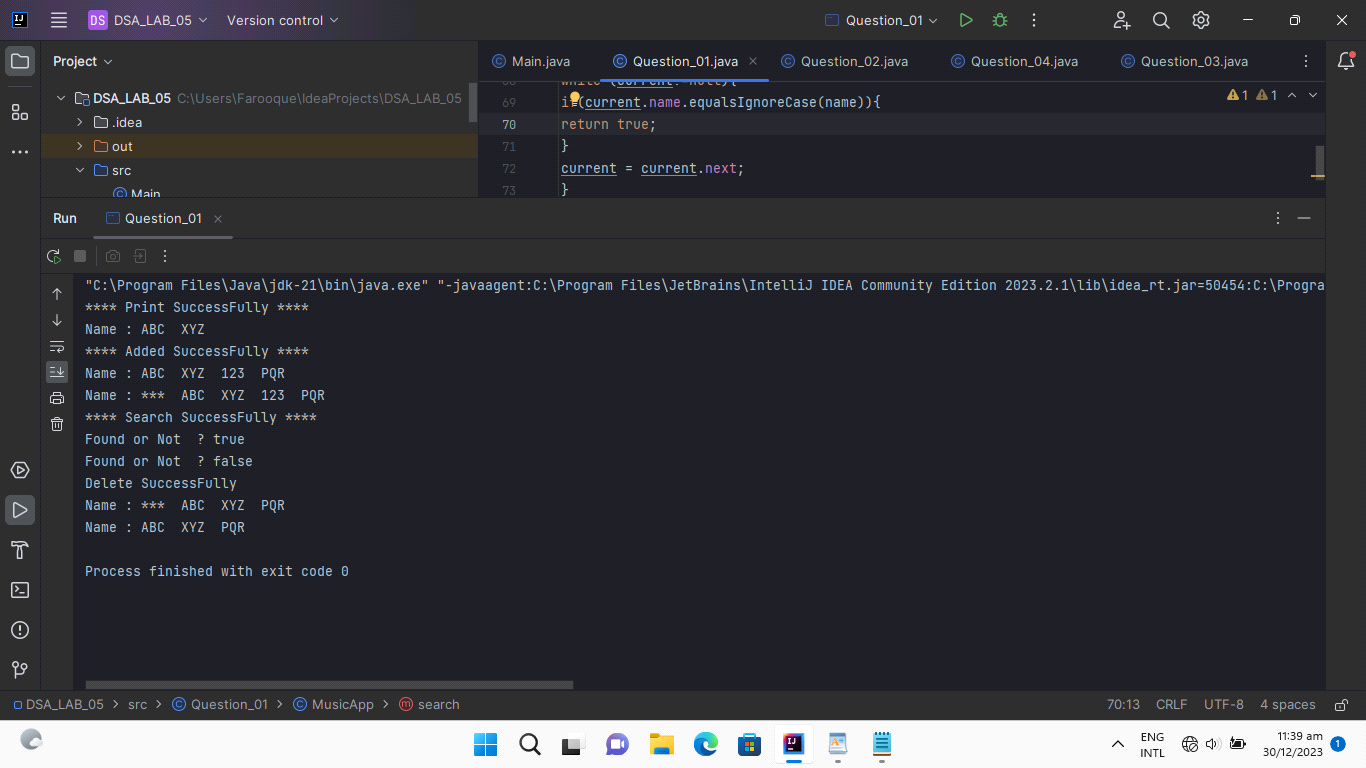
|  |  |
| --- | --- |
| Student Name | Farooque Sajjad |
| Roll Number  Section # | 22SW040  01 |
| Lab # | 05 |

***Question No : 01***

***source code***

**public class Question\_01 {  
static class MusicApp{  
String name;  
MusicApp next;  
public MusicApp(String name) {  
this.name = name;  
}  
public void print(MusicApp head ){  
MusicApp temp = head;  
System.*out*.print("Name : ");  
while (temp!=null){  
System.*out*.print( temp.name+ " ");  
temp=temp.next;  
}  
System.*out*.println();  
}  
public MusicApp add(MusicApp head , int index , String newName) {  
MusicApp newNode = new MusicApp(newName);  
if(index<0){  
System.*out*.println("Invalid Index ");  
return head;  
}  
if(index==0){  
newNode.next=head;  
head=newNode;  
}  
else {  
MusicApp current = head;  
int i=0;  
while (current!=null&&i<index){  
current=current.next;  
i++;  
}  
if(current!=null){  
newNode.next=current.next;  
current.next=newNode;  
}  
else{  
System.*out*.println("Index Out of Bound");  
}  
}  
return head;  
}  
public MusicApp deleteAtIndex(MusicApp head , int index) {  
if (head == null) {  
System.*out*.println("List is empty.");  
return head;  
}  
if (index == 0) {  
head = head.next;  
} else {  
MusicApp current = head;  
int currentIndex = 0;  
while (current != null && currentIndex < index - 1) {  
current = current.next;  
currentIndex++;  
}  
if (current != null && current.next != null) {  
current.next = current.next.next;  
} else {  
System.*out*.println("Index out of bounds.");  
}  
}  
return head;  
}  
public boolean search(MusicApp head , String name){  
MusicApp current = head;  
while (current!=null){  
if(current.name.equalsIgnoreCase(name)){  
return true;  
}  
current = current.next;  
}  
return false;  
}  
}  
public static void main(String[] args) {  
MusicApp s1 = new MusicApp("ABC");  
MusicApp s2 = new MusicApp("XYZ");  
s1.next = s2;  
System.*out*.println("\*\*\*\* Print SuccessFully \*\*\*\*");  
s1.print(s1);  
s1=s1.add(s1,1,"123");  
s1=s1.add(s1,2,"PQR");  
System.*out*.println("\*\*\*\* Added SuccessFully \*\*\*\* ");  
s1.print(s1);  
s1=s1.add(s1,0,"\*\*\*");  
s1.print(s1);  
System.*out*.println("\*\*\*\* Search SuccessFully \*\*\*\*");  
boolean result = s1.search(s1,"123");  
System.*out*.println("Found or Not ? " + result );  
result = s1.search(s1,"ABD");  
System.*out*.println("Found or Not ? " + result );  
System.*out*.println("Delete SuccessFully");  
s1 = s1.deleteAtIndex(s1,3);  
s1.print(s1);  
s1 = s1.deleteAtIndex(s1,0);  
s1.print(s1);  
}  
}**

**output**

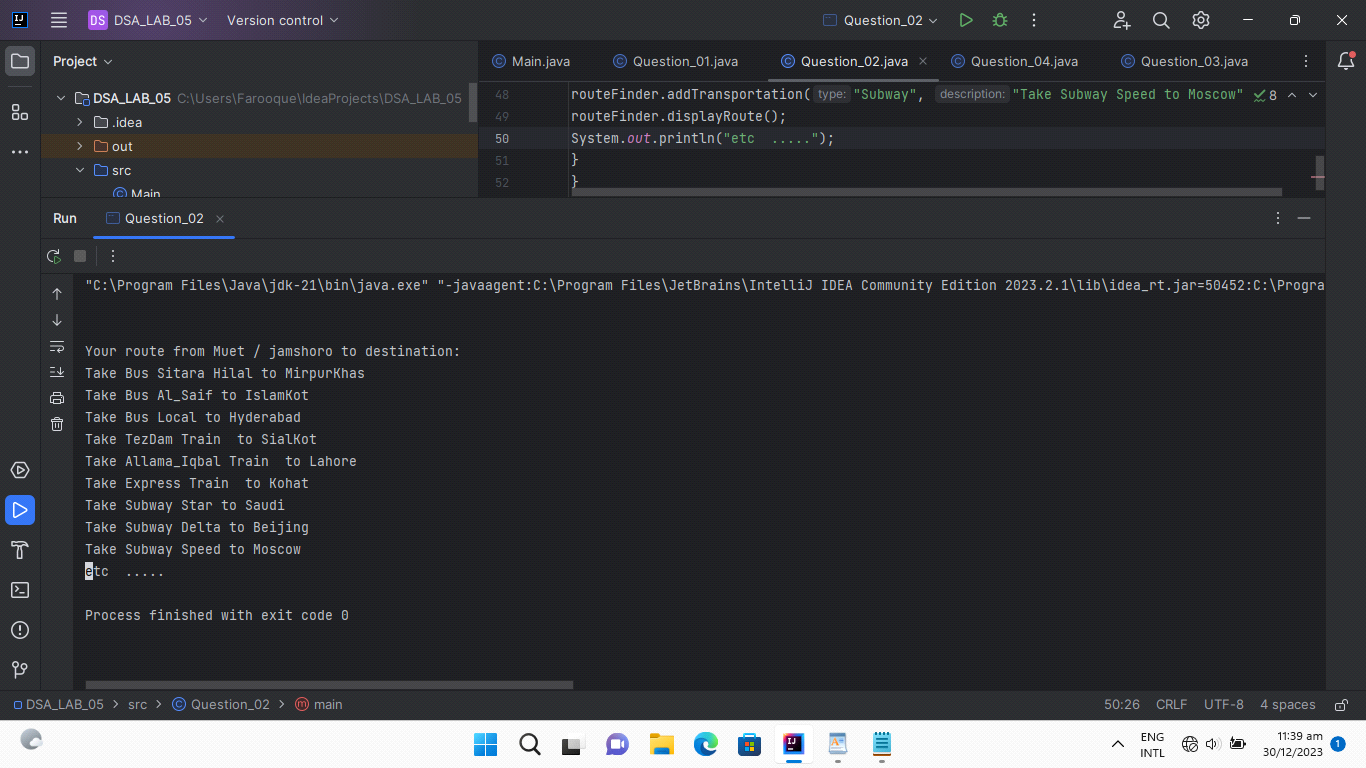


***Question No : 02***

***source code***

**public class Question\_02 {  
static class TransportationNode {  
String type;  
String description;  
TransportationNode next;  
public TransportationNode(String type, String description) {  
this.type = type;  
this.description = description;  
this.next = null;  
}  
}  
  
static class RouteFinder {  
TransportationNode origin;  
public void addTransportation(String type, String description) {  
TransportationNode newNode = new TransportationNode(type, description);  
if (origin == null) {  
origin = newNode;  
} else {  
TransportationNode current = origin;  
while (current.next != null) {  
current = current.next;  
}  
current.next = newNode;  
}  
}  
public void displayRoute() {  
System.*out*.println("Your route from Muet / jamshoro to destination:");  
TransportationNode current = origin;  
while (current != null) {  
System.*out*.println(current.description );  
current = current.next;  
}  
}  
}  
public static void main(String[] args) {  
RouteFinder routeFinder = new RouteFinder();  
System.*out*.println();  
routeFinder.addTransportation("Bus", "Take Bus Sitara Hilal to MirpurKhas");  
System.*out*.println();  
routeFinder.addTransportation("Bus", "Take Bus Al\_Saif to IslamKot ");  
routeFinder.addTransportation("Bus", "Take Bus Local to Hyderabad");  
routeFinder.addTransportation("Train", "Take TezDam Train to SialKot");  
routeFinder.addTransportation("Train", "Take Allama\_Iqbal Train to Lahore");  
routeFinder.addTransportation("Train", "Take Express Train to Kohat");  
routeFinder.addTransportation("Subway", "Take Subway Star to Saudi");  
routeFinder.addTransportation("Subway", "Take Subway Delta to Beijing");  
routeFinder.addTransportation("Subway", "Take Subway Speed to Moscow");  
routeFinder.displayRoute();  
System.*out*.println("etc .....");  
}  
}**

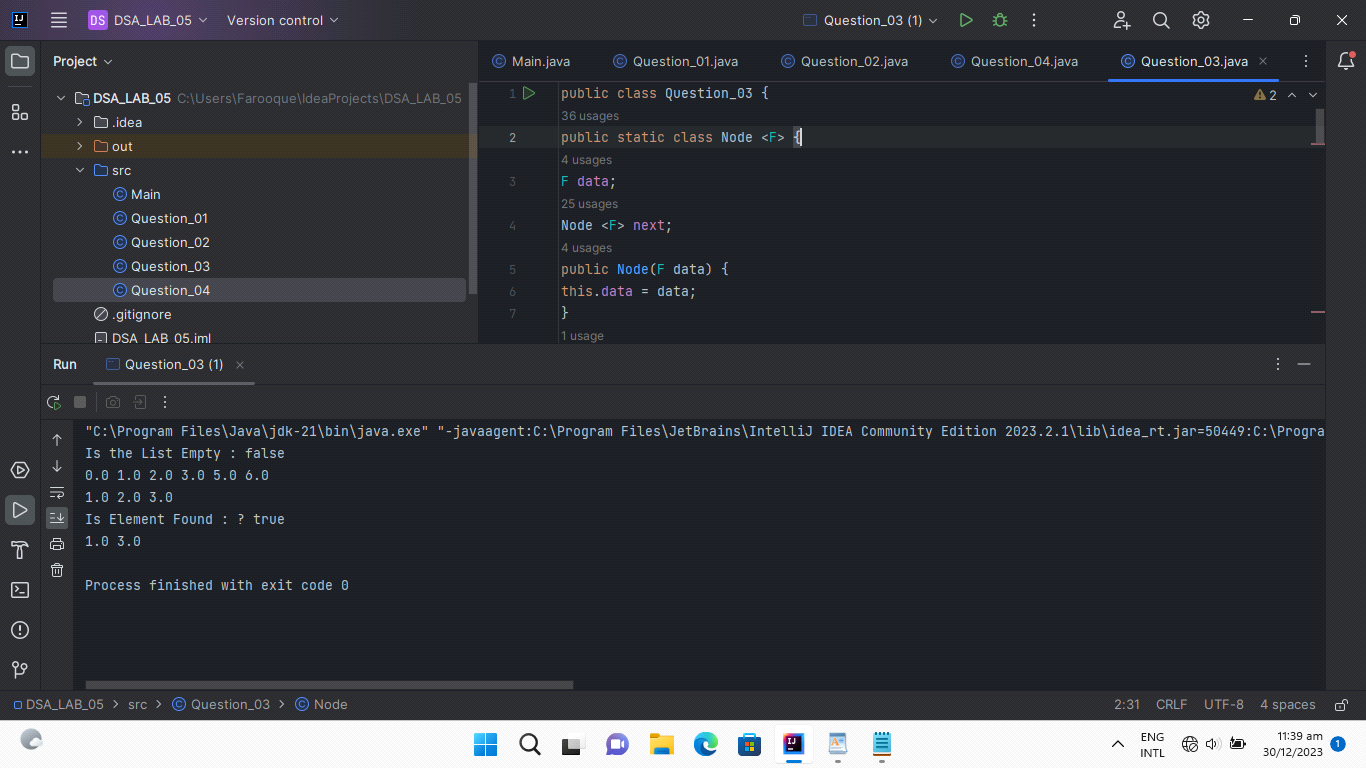
**output**



***Question No : 03***

***source code***

**public class Question\_03 {  
public static class Node <F> {  
F data;  
Node <F> next;  
public Node(F data) {  
this.data = data;  
}  
public boolean isEmpty(Node<F> head){  
return head == null;  
}  
public int calculateSize(Node <F> head){  
Node <F> temp = head;  
int size =0;  
while (temp!=null){  
size++;  
temp=temp.next;  
}  
return size;  
}  
public Node<F> insertAtHead(Node <F> head, F data ){  
Node<F> newNode = new Node<>(data);  
newNode.next=head;  
head=newNode;  
return head;  
}  
public Node<F> insertAtLast(Node <F> head , F data){  
Node<F> newNode = new Node<>(data);  
if(head==null){  
head=newNode;  
}  
else{  
Node <F> temp = head;  
while (temp.next!=null){  
temp=temp.next;  
}  
temp.next=newNode;  
}  
return head;  
}  
public Node<F> insertAtPosition(Node <F> head , F data ,int position){  
if(position<0||position>calculateSize(head)){  
System.*out*.println("Invalid Position");  
return head;  
}  
if(position==0){  
insertAtHead(head,data);  
}  
else if(position==calculateSize(head)){  
insertAtLast(head,data);  
}  
else{  
Node<F> newNode = new Node<>(data);  
Node <F> temp = head;  
for(int i=0;i<position-1;i++){  
temp=temp.next;  
}  
newNode.next = temp.next;  
temp.next=newNode;  
}  
return head;  
}  
public Node<F> deleteAtFirst(Node<F> head){  
if(head!=null){  
return head.next;  
}  
return null;  
}  
public Node <F> deleteAtLast(Node <F> head){  
if(head==null||head.next==null){  
head=null;  
return head;  
}  
Node <F> temp = head;  
while (temp.next.next!=null){  
temp=temp.next;  
}  
temp.next=null;  
return head;  
}  
public Node <F> deleteAtPosition(Node <F> head , int position ){  
if(position<0||position>=calculateSize(head)||head==null){  
System.*out*.println("Invalid Position");  
return head;  
}  
if(position==0){  
deleteAtFirst(head);  
}  
else{  
Node <F> temp = head;  
for(int i=0;i<position-1;i++){  
temp=temp.next;  
}  
temp.next=temp.next.next;  
}  
return head;  
}  
public boolean search(Node<F> head , F data ){  
Node <F> temp = head;  
while (temp!=null){  
if(temp.data.equals(data)){  
return true;  
}  
temp=temp.next;  
}  
return false;  
}  
public Node<F> deleteValue(Node <F> head , F value){  
Node <F> temp = head;  
Node <F> prev = null;  
while (temp!=null&&!temp.data.equals(value)){  
prev=temp;  
temp=temp.next;  
}  
if(temp==null){  
System.*out*.println("Value Not Found");  
return head ;  
}  
if(prev==null){  
head=temp.next;  
}  
else{  
prev.next=temp.next;  
}  
return head;  
}  
public void display(Node<F> head){  
Node<F> temp = head;  
while(temp!=null){  
System.*out*.print(temp.data+" " );  
temp = temp.next;  
}  
System.*out*.println();  
}  
}  
  
public static void main(String[] args) {  
Node<Float> floatList = new Node<>(1.0f);  
System.*out*.println("Is the List Empty : " + floatList.isEmpty(floatList));  
floatList=floatList.insertAtLast(floatList,5.0f);  
floatList=floatList.insertAtLast(floatList,6.0f);  
floatList=floatList.insertAtHead(floatList,0.0f);  
floatList=floatList.insertAtPosition(floatList,2.0f,2);  
floatList=floatList.insertAtPosition(floatList,3.0f,3);  
floatList.display(floatList);  
floatList=floatList.deleteAtFirst(floatList);  
floatList=floatList.deleteAtLast(floatList);  
floatList=floatList.deleteAtPosition(floatList,3);  
floatList.display(floatList);  
System.*out*.println("Is Element Found : ? " + floatList.search(floatList,2.0f));  
floatList=floatList.deleteValue(floatList,2.0f);  
floatList.display(floatList);  
}  
}  
  
  
output**



***Question No : 04***

***source code***

**public class Question\_04 {  
public static class Node {  
int data;  
Node next;  
public Node(int data) {  
this.data = data;  
}  
public int calculateSize(Node head){  
Node temp = head;  
int size =0;  
while (temp!=null){  
size++;  
temp=temp.next;  
}  
return size;  
}  
  
public int [] LinkedListToArray(Node head){  
int size = calculateSize(head);  
int [] arr = new int[size];  
if(head==null){  
System.*out*.println("List Is Null");  
}  
else{  
Node temp = head;  
for(int i=0;i<size;i++){  
arr[i]=temp.data;  
temp=temp.next;  
}  
System.*out*.println("Linked List Converted into Array SuccessFully");  
return arr;  
}  
return arr;  
}  
}  
public static void main(String[] args) {  
Node n1 = new Node(10);  
Node n2 = new Node(20);  
Node n3 = new Node(30);  
Node n4 = new Node(40);  
Node n5 = new Node(50);  
Node n6 = new Node(60);  
n1.next=n2;  
n2.next=n3;  
n3.next=n4;  
n4.next=n5;  
n5.next=n6;  
  
int [] array = n1.LinkedListToArray(n1);  
for(int i=0;i<array.length;i++){  
System.*out*.print(array[i]+ " ");  
}  
System.*out*.println();  
}  
}**

**output**

