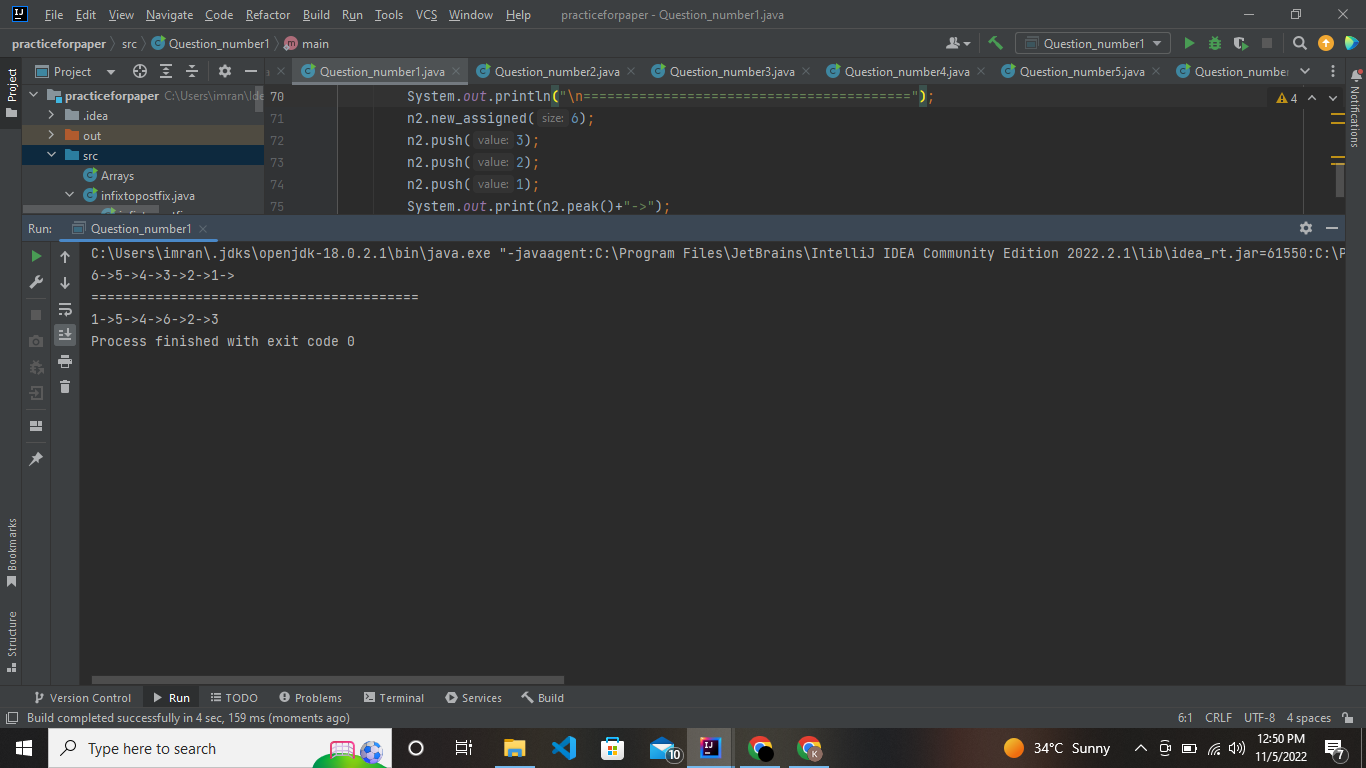
**LAB-8 Assignment:**

**Question number 1**

**Code:**

public class Question\_number1 {  
 int arr[];  
 int size;  
 int top;  
 int capacity;  
  
 void new\_assigned(int size) {  
 arr = new int[size];  
 capacity = size;  
 this.size = size;  
 top = -1;  
 }  
  
 public void push(int value) {  
 if (isFull()) {  
 System.*out*.println("Error!");  
 return;  
 }  
 top++;  
 arr[top] = value;  
  
 }  
 public void pop(){  
 if (isEmpty()){  
 System.*out*.println("Error");  
 return;  
 }  
  
 top--;  
 }  
 // peak  
 public int peak(){  
 if (isEmpty()){  
 System.*out*.println("Error");  
 }  
 return arr[top];  
 }  
  
 boolean isFull(){  
 if (capacity==top){  
 return true;  
 }  
 return false;  
 }  
 boolean isEmpty(){  
 if (top==-1){  
 return true;  
 }  
 return false;  
  
 }  
  
  
 public static void main(String[] args) {  
 Question\_number1 n1 = new Question\_number1();  
 n1.new\_assigned(6);  
 n1.push(1);  
 n1.push(2);  
 n1.push(3);  
 n1.push(4);  
 n1.push(5);  
 n1.push(6);  
 int n=0;  
 while (n<6){  
 System.*out*.print(n1.peak()+"->");  
 n1.pop();  
 n++;  
 }  
 Question\_number1 n2 = new Question\_number1() ;  
 System.*out*.println("\n=========================================");  
 n2.new\_assigned(6);  
 n2.push(3);  
 n2.push(2);  
 n2.push(1);  
 System.*out*.print(n2.peak()+"->");  
 n2.pop();  
 n2.push(4);  
 n2.push(5);  
 System.*out*.print(n2.peak()+"->");  
 n2.pop();  
 System.*out*.print(n2.peak()+"->");  
 n2.pop();  
 n2.push(6);  
 System.*out*.print(n2.peak()+"->");  
 n2.pop();  
 System.*out*.print(n2.peak()+"->");  
 n2.pop();  
 System.*out*.print(n2.peak());  
 n2.pop();  
  
  
 }  
}

**Output:**

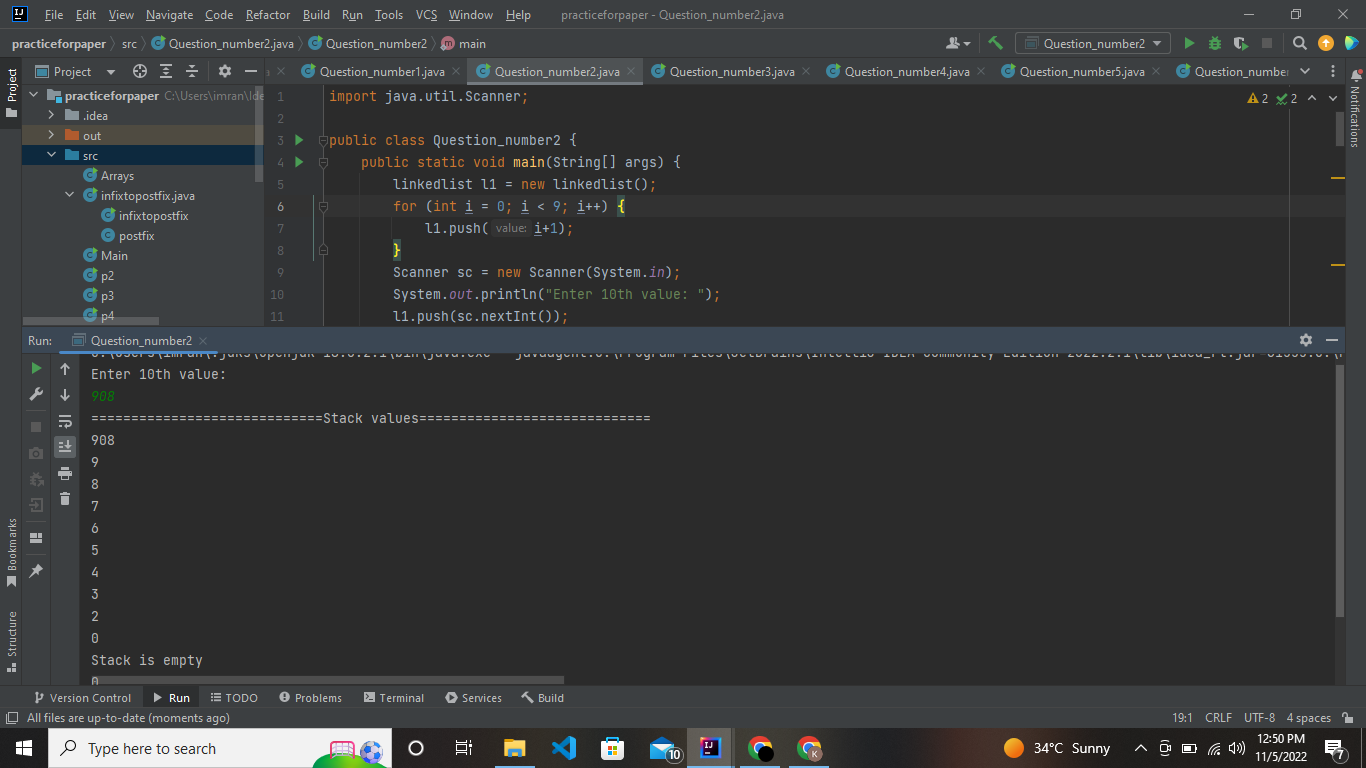
****

**Question number 2**

**Code:**

import java.util.Scanner;  
  
public class Question\_number2 {  
 public static void main(String[] args) {  
 linkedlist l1 = new linkedlist();  
 for (int i = 0; i < 9; i++) {  
 l1.push(i+1);  
 }  
 Scanner sc = new Scanner(System.*in*);  
 System.*out*.println("Enter 10th value: ");  
 l1.push(sc.nextInt());  
 System.*out*.println("=============================Stack values=============================");  
 for (int i = 0; i <= 10; i++) {  
 System.*out*.println(l1.peak());  
 l1.pop();  
  
 }  
 }  
  
}  
class linkedlist{  
 class node{  
 int data;  
 node next;  
  
 }  
 node head = null;  
 public void push(int value){  
 node newnode = new node();  
 if (head==null){  
 head = newnode;  
 return;  
 }  
 newnode.data = value;  
 newnode.next = head;  
 head = newnode;  
 }  
 public void pop(){  
 if (isEmpty()){  
// System.out.println("Stack is empty");  
 return;  
 }  
 head = head.next;  
 }  
 public boolean isEmpty(){  
 if (head==null){  
 return true;  
 }  
 return false;  
 }  
  
 public int peak(){  
 if (isEmpty()){  
 System.*out*.println("Stack is empty");  
 return 0;  
 }  
 return head.data;  
 }  
  
}

**Output:**

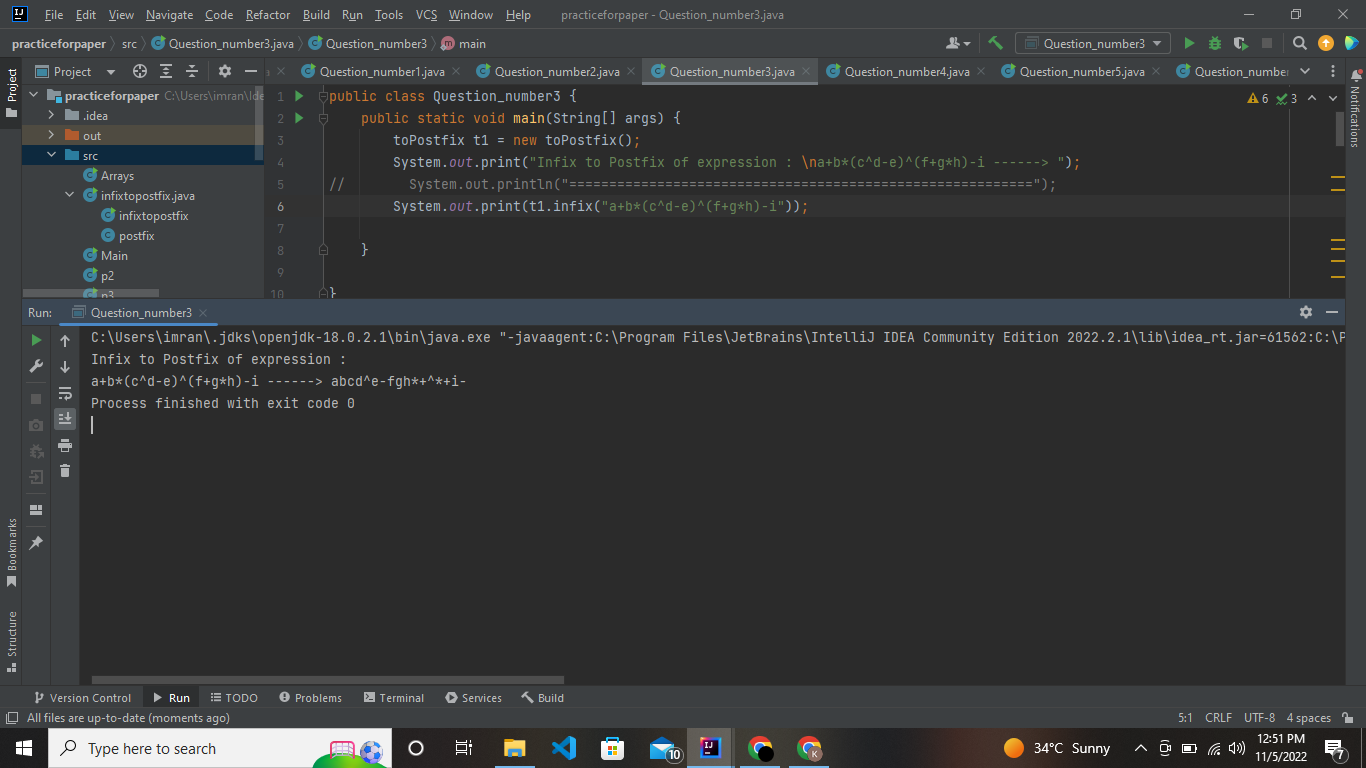
****

**Question number 3**

**Code:**

public class Question\_number3 {  
 public static void main(String[] args) {  
 toPostfix t1 = new toPostfix();  
 System.*out*.print("Infix to Postfix of expression : \na+b\*(c^d-e)^(f+g\*h)-i ------> ");  
// System.out.println("==========================================================");  
 System.*out*.print(t1.infix("a+b\*(c^d-e)^(f+g\*h)-i"));  
  
 }  
  
}  
class toPostfix{  
 int top = -1;  
 char[] arr = new char[100];  
 int size = 100;  
  
 public void push(char data){  
 if (isfull()){  
 System.*out*.println("Full");  
 return;  
 }  
 arr[++top] = data;  
 }  
 public void pop(){  
 if (isempty()){  
 System.*out*.println("empty");  
 return ;  
 }  
 top--;  
 }  
 boolean isfull(){  
 if (top==size){  
 return true;  
 }  
 return false;  
 }  
 boolean isempty(){  
 if (top==-1){  
 return true;  
 }  
 return false;  
 }  
 char peak(){  
  
 return arr[top];  
 }  
 public int prec(char val){  
 if (val=='+'||val=='-'){  
 return 1;  
 } else if (val=='/'||val=='\*') {  
 return 2;  
 } else if (val=='^') {  
 return 3;  
 }else  
 return -1;  
 }  
  
 public String infix(String infix){  
 String postfix ="";  
 for (int i = 0; i < infix.length(); i++) {  
 char character = infix.charAt(i);  
 if (Character.*isLetter*(character)){  
 postfix = postfix+ character;  
 } else if (character == '(' ) {  
 push(character);  
 } else if (character==')') {  
 while (!isempty() && peak()!='(' ){  
 postfix = postfix + peak();  
 pop();  
 }  
 pop();  
 }else {  
  
 while (!isempty() && !(peak()=='(') && prec(character) <= prec(peak())){  
 postfix = postfix+peak();  
 pop();  
 }  
 push(character);  
 }  
  
  
 }  
 while (!isempty()){  
 postfix = postfix+peak();  
 pop();  
 }  
 return postfix;  
  
 }  
}

**Output:**

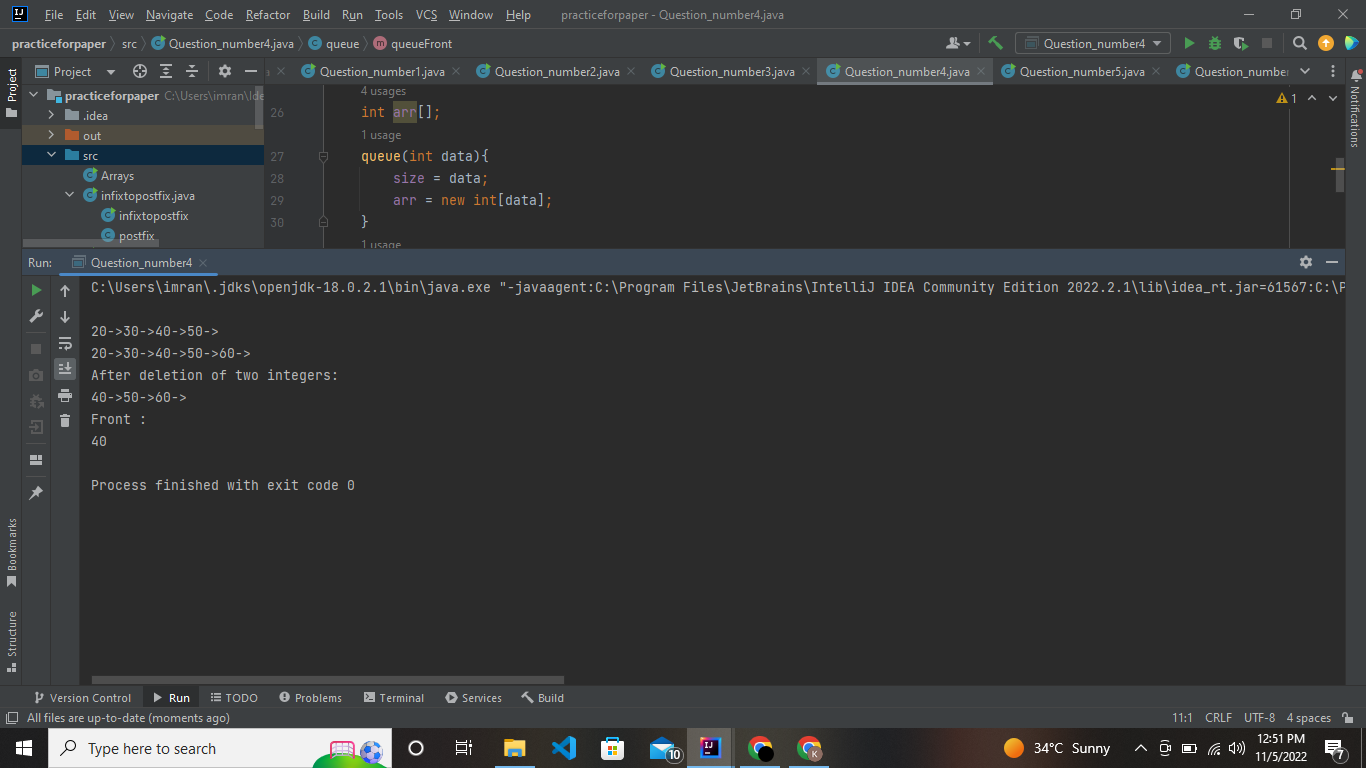
****

**Question number 4**

**Code:**

public class Question\_number4 {  
 public static void main(String[] args) {  
 queue q1 = new queue(10);  
 q1.display();  
 System.*out*.println();  
 q1.Enqueue(20);  
 q1.Enqueue(30);  
 q1.Enqueue(40);  
 q1.Enqueue(50);  
 q1.display();  
 System.*out*.println();  
 q1.Enqueue(60);  
 q1.display();  
 System.*out*.println();  
 q1.Dequeue();  
 q1.Dequeue();  
 System.*out*.println("After deletion of two integers: ");  
 q1.display();  
 System.*out*.println("\nFront : ");  
 q1.queueFront();  
 }  
}  
class queue{  
 int front =0;int rear = 0;  
 int size;  
 int arr[];  
 queue(int data){  
 size = data;  
 arr = new int[data];  
 }  
 public void queueFront(){  
 System.*out*.println(arr[front]);  
 }  
 public void display(){  
 int i = front;  
 while (i!=rear){  
 System.*out*.print(arr[i++]+"->");  
 }  
 }  
 public void Enqueue(int data){  
 if (rear==size){  
 System.*out*.println("Full");  
 return;  
 }  
  
 arr[rear++] = data;  
 }  
 public void Dequeue(){  
 if (front==size){  
 System.*out*.println("Empty");  
 return;  
 }  
 front++;  
 }  
}

**Output:**

****

**Question number 5**

**Code:**

**public class Qno5Lab8 {**

**public static void main(String[] args) {**

**Circular\_Queue cq = new Circular\_Queue(3);**

**cq.Queue("A");**

**cq.Queue("B");**

**cq.Queue("C");**

**cq.Print();**

**cq.Dequeue();**

**cq.Queue("D");**

**cq.Print();**

**System.out.println(cq.Peak\_Dequeue());**

**cq.Dequeue();**

**cq.Queue("E");**

**cq.Print();**

**System.out.println(cq.Peak\_Dequeue());**

**cq.Dequeue();**

**}**

**}**

**class Circular\_Queue**

**{**

**int front=-1;**

**int rear=-1;**

**int size;**

**String arr[];**

**public Circular\_Queue(int size) {**

**this.size=size;**

**arr=new String[size];**

**}**

**public void Queue(String data)**

**{**

**if(front==(rear+1)%size)**

**{**

**System.out.println("Queue is Full");**

**}**

**else if(front==-1&&rear==-1)**

**{**

**front=0;**

**rear=0;**

**arr[rear]=data;**

**}**

**else**

**{**

**rear=(rear+1)%size;**

**arr[rear]=data;**

**}**

**}**

**public void Dequeue()**

**{**

**if(front==-1&&rear==-1)**

**{**

**System.out.println("Queue is Empty");**

**}**

**else if(front==rear)**

**{**

**System.out.println("Queue is Empty");**

**front=-1;**

**rear=-1;**

**}**

**else**

**{**

**front=(front+1)%size;**

**}**

**}**

**public String Peak\_Dequeue()**

**{**

**return arr[front];**

**}**

**public void Print()**

**{**

**for(int i=0;i<arr.length;i++)**

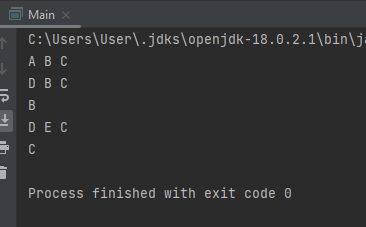
**{**

**System.out.print(arr[i]+" ");**

**}**

**System.out.println();**

**}}**

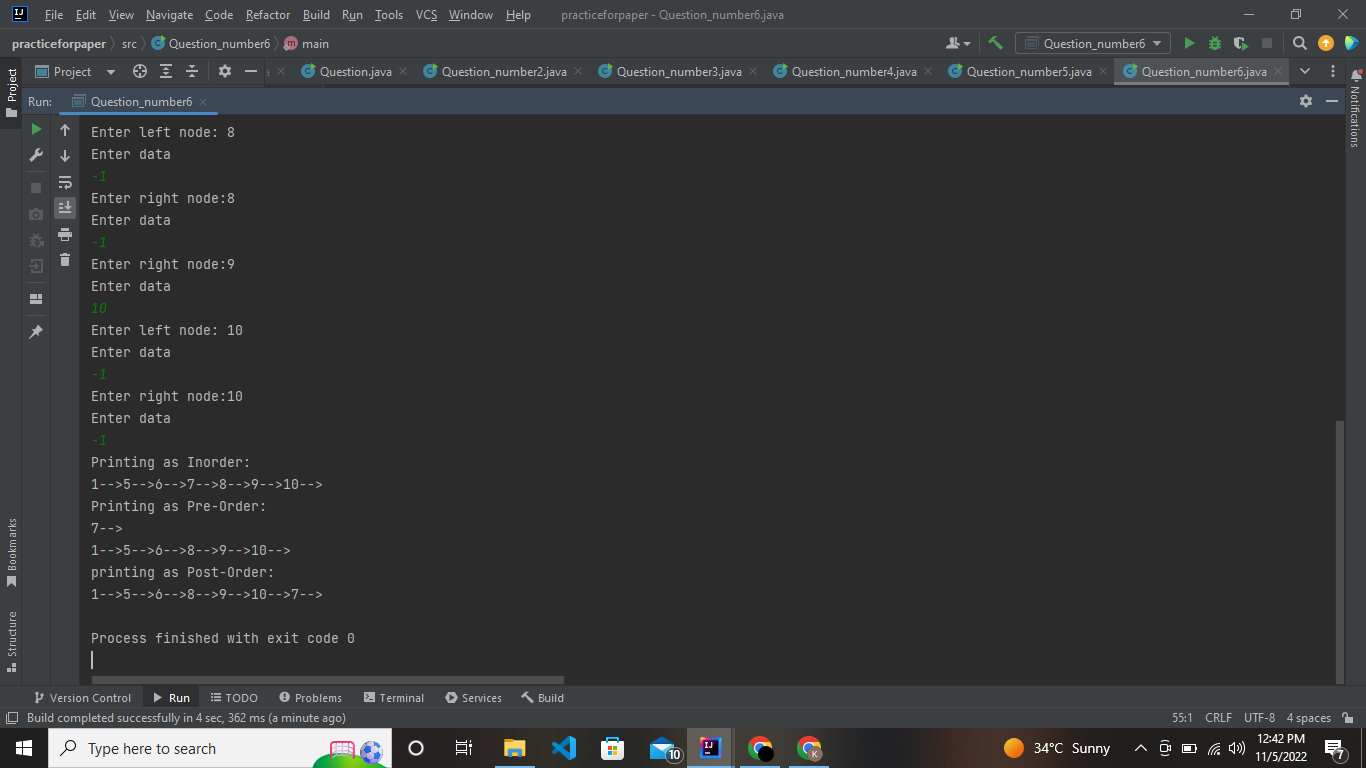
****

**Question number 6**

**Code:**

import java.util.Scanner;  
public class Question\_number6 {  
 public static void main(String[] args) {  
 Question\_number6 q1 = new Question\_number6();  
 node rootnode =null;  
 rootnode = q1.inserting();  
 System.*out*.println("Printing as Inorder: ");  
 q1.inorder(rootnode);  
 System.*out*.println("\nPrinting as Pre-Order: ");  
 q1.preorder(rootnode);  
 System.*out*.println("\nprinting as Post-Order: ");  
 q1.postorder(rootnode);  
 }  
  
class node{  
 node left;  
 node right;  
 int data;  
}  
 public node inserting(){  
 node root = new node();  
 System.*out*.println("Enter data");  
 Scanner sc = new Scanner(System.*in*);  
 int data=sc.nextInt();  
 root.data = data;  
 if (data == -1){  
 return null;  
 }  
 System.*out*.println("Enter left node: "+data);  
 root.left = inserting();  
 System.*out*.println("Enter right node:" +data);  
 root.right = inserting();  
 return root;  
 }  
 public void inorder(node root){  
 if (root==null){  
 return;  
 }  
 inorder(root.left);  
 System.*out*.print(root.data+"-->");  
 inorder(root.right);  
 }  
 public void preorder(node root){  
 if (root==null){  
 return;  
 }  
 System.*out*.println(root.data+"-->");  
 inorder(root.left);  
 inorder(root.right);  
 }  
 public void postorder(node root){  
 if (root==null){  
 return;  
 }  
 inorder(root.left);  
 inorder(root.right);  
 System.*out*.println(root.data+"-->");  
 }  
}

**Output:**

****

**=====================End======================**