Name: Subhapreet Patro

Roll No.: 2211CS010547(Sem-ID:33)

Group: 7A

ADS ASSIGNMENT-1

1) Write a program to build a linked list using the built in linked list collection class.

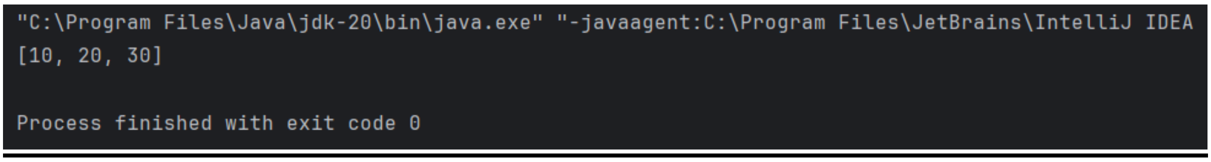
Ans:

Program:

```
import java.util .*;
class SingleLinkedList{
   LinkedList linklist = new LinkedList();
   public void add(int item){
      linklist.add(item);
   }
   public int remove(){
      return (int) linklist.remove();
   }
   public void printList(){
      System.out.println(linklist);
   }
}
public class SingleLinkedListUsingLinkedList {
   public static void main(String[] args) {
      SingleLinkedList s1 = new SingleLinkedList();
      s1.add(10);
      s1.add(20);
```

```
        s1.add(30);
        s1.printList();
    }
}
```

Output:



```
"C:\Program Files\Java\jdk-20\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA
[10, 20, 30]

Process finished with exit code 0
```

2) Write a program to implement Doubly Linked List using the built in Linked List collection class

Ans:

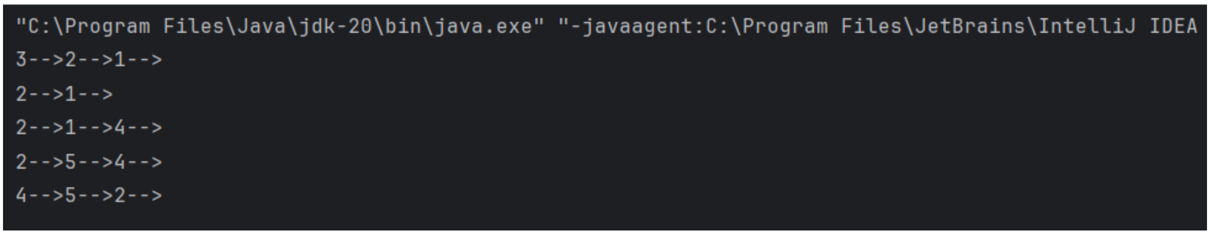
Program:

```
import java.util.LinkedList;
import java.util.ListIterator;

public class DoublyLinkedList {

    private LinkedList<Integer> list;
    public DoublyLinkedList() {
        list = new LinkedList<>();
    }
    public void addFirst(int data) {
        list.addFirst(data);
    }
    public void addLast(int data) {
```

```
    list.addLast(data);
}
public int removeFirst() {
    return list.removeFirst();
}
public int removeLast() {
    return list.removeLast();
}
public int get(int index) {
    return list.get(index);
}
public void set(int index, int data) {
    list.set(index, data);
}
public int size() {
    return list.size();
}
public boolean isEmpty() {
    return list.isEmpty();
}
public void print() {
    for (int i = 0; i < list.size(); i++) {
        System.out.print(list.get(i)+"-->");
    }
    System.out.println();
}
// Method to traverse the linked list in reverse order
```

```
    public void printReverse() {
        ListIterator<Integer> iterator = list.listIterator(list.size());
        while (iterator.hasPrevious()) {
            System.out.print(iterator.previous() + "-->");
        }
        System.out.println();
    }
    public static void main(String[] args) {
        DoublyLinkedList list = new DoublyLinkedList();
        list.addFirst(1);
        list.addFirst(2);
        list.addFirst(3);
        list.print();
        list.removeFirst();
        list.print();
        list.addLast(4);
        list.print();
        list.set(1, 5);
        list.print();
        list.printReverse();
    }
}
```

Output:



```
"C:\Program Files\Java\jdk-20\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA
3-->2-->1-->
2-->1-->
2-->1-->4-->
2-->5-->4-->
4-->5-->2-->
```

3) Stack using Linked List (Built in)

Ans:

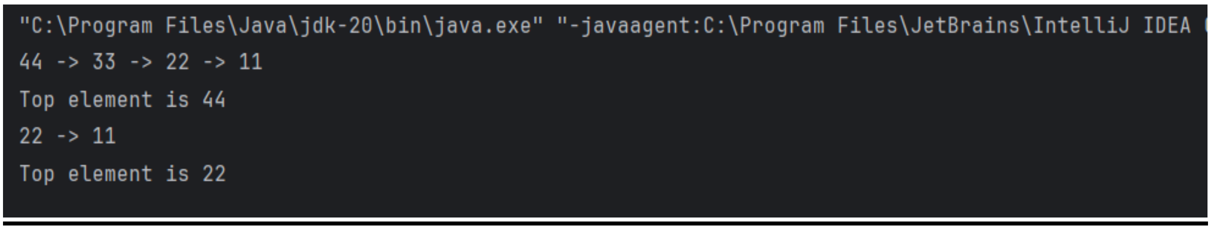
Program:

```
import static java.lang.System.exit;
class StackUsingLL {
    public static void main(String[] args)
    {
        StackUsingLinkedlist obj = new StackUsingLinkedlist();
        obj.push(11);
        obj.push(22);
        obj.push(33);
        obj.push(44);
        obj.display();
        System.out.printf("\nTop element is %d\n", obj.peek());
        obj.pop();
        obj.pop();
        obj.display();
        System.out.printf("\nTop element is %d\n", obj.peek());
    }
}
class StackUsingLinkedlist {
    private class Node {
```

```
    int data;
    Node link;
}
Node top;
StackUsingLinkedlist() { this.top = null; }
public void push(int x)
{
    Node temp = new Node();
    if (temp == null) {
        System.out.print("\nHeap Overflow");
        return;
    }
    temp.data = x;
    temp.link = top;
    top = temp;
}
public boolean isEmpty() { return top == null; }
public int peek()
{
    if (!isEmpty()) {
        return top.data;
    }
    else {
        System.out.println("Stack is empty");
        return -1;
    }
}
```

```
    public void pop()
    {
        if (top == null) {
            System.out.print("\nStack Underflow");
            return;
        }
        top = (top).link;
    }
    public void display()
    {
        if (top == null) {
            System.out.printf("\nStack Underflow");
            exit(1);
        }
        else {
            Node temp = top;
            while (temp != null) {
                System.out.print(temp.data);
                temp = temp.link;
                if(temp != null)
                    System.out.print(" -> ");
            }
        }
    }
}
```

Output:



```
"C:\Program Files\Java\jdk-20\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA
44 -> 33 -> 22 -> 11
Top element is 44
22 -> 11
Top element is 22
```

4) Stacks using Array and ArrayList using generics

Ans:

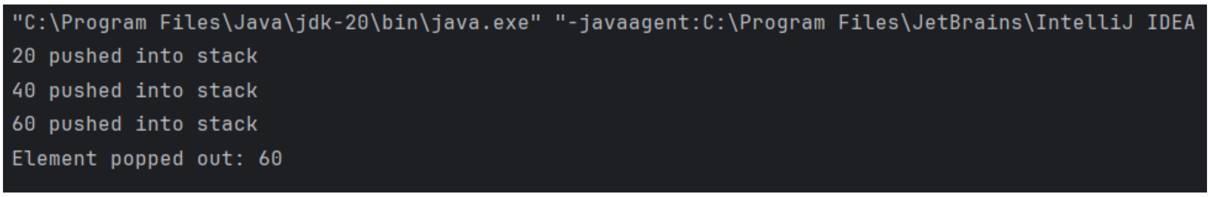
(i) Stacks using Arrays:

Program:

```
class Stack {
   static int MAX=100;
   int top;
   int a[]=new int[MAX];
   boolean isEmpty(){
      return (top<0);
   }
   Stack(){
      top=-1;
   }
   boolean push(int x){
      if (top>=MAX-1){
         System.out.println("Overflow");
         return false;
      }
      else{
         a[++top]=x;
         System.out.println(x+" pushed into stack");
```

```
            return true;
        }
    }
    int pop(){
        if (top<0){
            System.out.println("Underflow");
            return 0;
        }
        else{
            int x=a[top--];
            return x;
        }
    }
    int peek(){
        if (top<0){
            System.out.println("Underflow");
            return 0;
        }
        else{
            int x=a[top];
            return x;
        }
    }
}
class StackUsingArray{
    public static void main(String[] args) {
        Stack s=new Stack();
        s.push(20);
        s.push(40);
        s.push(60);
        System.out.println("Element popped out: "+ s.pop());
    }
}
```

Output:



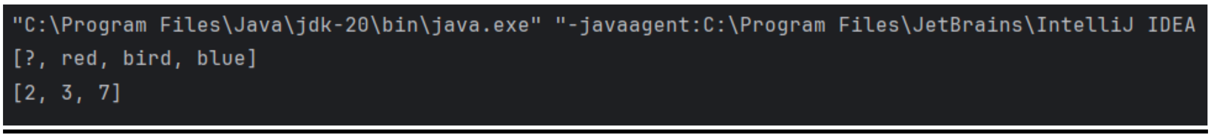
```
"C:\Program Files\Java\jdk-20\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA
20 pushed into stack
40 pushed into stack
60 pushed into stack
Element popped out: 60
```

(ii) ArrayList using Generic Arrays:

Program:

```
import java.util.ArrayList;
public class ArrayListUsingGenericArrays {
    public static void main(String[] args) {
        ArrayList<String> strings = new ArrayList<>();
        strings.add("?");
        strings.add("red");
        strings.add("bird");
        strings.add("blue");
        System.out.println(strings);
        ArrayList<Integer> ints = new ArrayList<>();
        ints.add(2);
        ints.add(3);
        ints.add(7);
        System.out.println(ints);
    }
}
```

Output:



```
"C:\Program Files\Java\jdk-20\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA
[?, red, bird, blue]
[2, 3, 7]
```

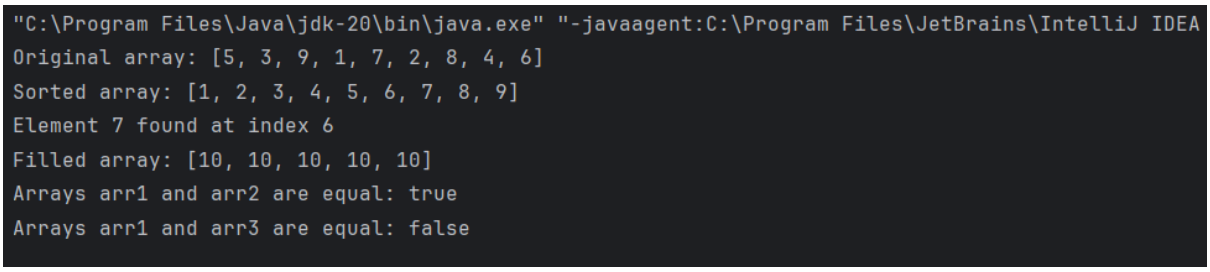
5) Develop a program to demonstrate the built in arrays class and perform various operations using utility methods for arrays.

Ans:

Program:

```
import java.util.Arrays;

public class ArraysDemo {
    public static void main(String[] args) {
        // Initialize an array
        int[] numbers = {5, 3, 9, 1, 7, 2, 8, 4, 6};

        // Print original array
        System.out.println("Original array: " + Arrays.toString(numbers));

        // Sorting array in ascending order
        Arrays.sort(numbers);
        System.out.println("Sorted array: " + Arrays.toString(numbers));

        // Binary search for an element
        int elementToSearch = 7;
        int index = Arrays.binarySearch(numbers, elementToSearch);
        if (index >= 0) {
            System.out.println("Element " + elementToSearch + " found at index " + index);
        } else {
            System.out.println("Element " + elementToSearch + " not found");
        }

        // Filling array with a specific value
        int[] filledArray = new int[5];
        Arrays.fill(filledArray, 10);
        System.out.println("Filled array: " + Arrays.toString(filledArray));

        // Comparing arrays for equality
        int[] arr1 = {1, 2, 3};
        int[] arr2 = {1, 2, 3};
        int[] arr3 = {3, 2, 1};
        boolean isEqual1And2 = Arrays.equals(arr1, arr2);
        boolean isEqual1And3 = Arrays.equals(arr1, arr3);
        System.out.println("Arrays arr1 and arr2 are equal: " + isEqual1And2);
        System.out.println("Arrays arr1 and arr3 are equal: " + isEqual1And3);
    }
}
```

Output:



```
"C:\Program Files\Java\jdk-20\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA
Original array: [5, 3, 9, 1, 7, 2, 8, 4, 6]
Sorted array: [1, 2, 3, 4, 5, 6, 7, 8, 9]
Element 7 found at index 6
Filled array: [10, 10, 10, 10, 10]
Arrays arr1 and arr2 are equal: true
Arrays arr1 and arr3 are equal: false
```