CS162 ASSIGNMENT 4

NAME:

ARCHIT AGRAWAL

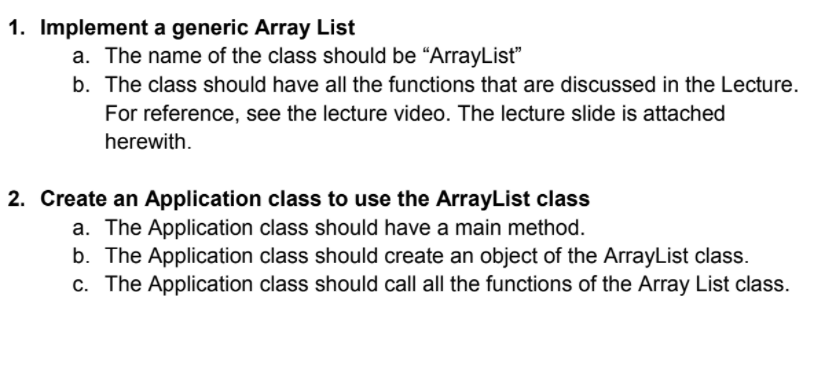
ROLL NO. :

202052307

SECTION:

A

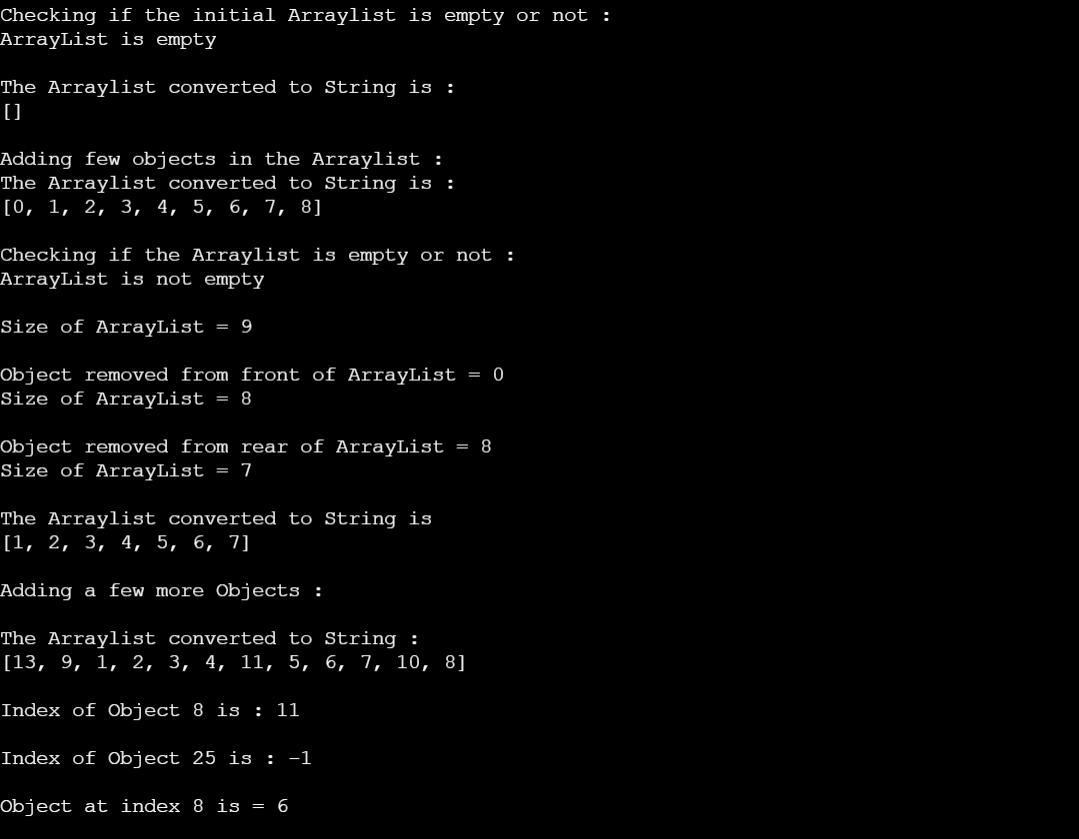
**Question**

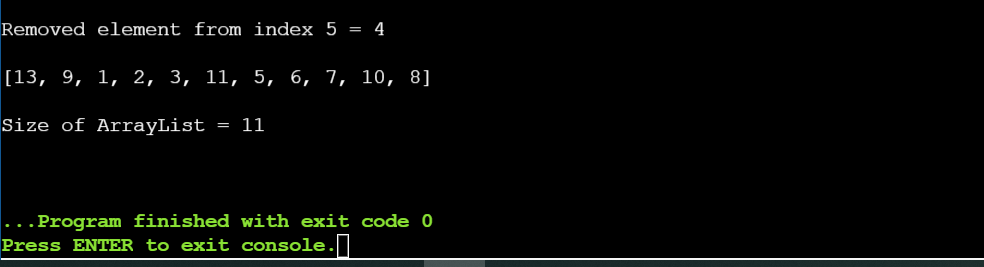


***CODE***

//package com.company;  
public class Application{  
 public static void main(String []args){  
 ArrayList <Integer> list = new ArrayList<Integer>(10); // creating an arraylist with initial capacity of 10  
  
 System.*out*.println();  
 System.*out*.println("Checking if the initial Arraylist is empty or not : ");  
 if(list.isEmpty()){  
 System.*out*.println("ArrayList is empty");  
 } else {  
 System.*out*.println("ArrayList is not empty");  
 }  
 System.*out*.println();  
  
 System.*out*.println("The Arraylist converted to String is :");  
 System.*out*.println(list.toString());  
 System.*out*.println();  
  
 System.*out*.println("Adding few objects in the Arraylist :");  
 for(int i = 0; i < 9; i++){  
 list.add(i , i);  
 }  
  
 System.*out*.println("The Arraylist converted to String is :");  
 System.*out*.println(list.toString());  
 System.*out*.println();  
  
 System.*out*.println("Checking if the Arraylist is empty or not : ");  
  
 if(list.isEmpty()){  
 System.*out*.println("ArrayList is empty");  
 } else {  
 System.*out*.println("ArrayList is not empty");  
 }  
 System.*out*.println();  
  
 System.*out*.println("Size of ArrayList = "+list.size());  
 System.*out*.println();  
  
 System.*out*.println("Object removed from front of ArrayList = "+list.removeFront());  
 System.*out*.println("Size of ArrayList = "+list.size());  
 System.*out*.println();  
  
 System.*out*.println("Object removed from rear of ArrayList = "+list.removeRear());  
 System.*out*.println("Size of ArrayList = "+list.size());  
 System.*out*.println();  
  
 System.*out*.println("The Arraylist converted to String is");  
 System.*out*.println(list.toString());  
 System.*out*.println();  
  
 System.*out*.println("Adding a few more Objects :");  
 System.*out*.println();  
 list.addToFront(9);  
 list.addToRear(10);  
 list.add(11, 5);  
 list.addToRear(8);  
 list.addToFront(13);  
  
 System.*out*.println("The Arraylist converted to String :");  
 System.*out*.println(list.toString());  
 System.*out*.println();  
  
 System.*out*.println("Index of Object 8 is : "+list.indexOf(8));  
 System.*out*.println();  
  
 System.*out*.println("Index of Object 25 is : "+list.indexOf(25));  
 System.*out*.println();  
  
 System.*out*.println("Object at index 8 is = "+list.get(8));  
 System.*out*.println();  
  
 System.*out*.println("Removed element from index 5 = " +list.remove(5));  
 System.*out*.println();  
  
 System.*out*.println(list.toString());  
 System.*out*.println();  
  
 System.*out*.println("Size of ArrayList = "+list.size());  
 System.*out*.println();  
  
 }  
}  
  
class ArrayList <T>{  
  
 private T[] arr;  
 private int size;  
  
 public ArrayList(int initialCapacity){  
 if(initialCapacity < 1){  
 throw new IllegalArgumentException("Initial Capacity must be >= 1");  
 }  
 arr = (T[])new Object[initialCapacity];  
 size = 0;  
 }  
  
 public ArrayList(){  
 this(10);  
 }  
  
 public void addToFront(T obj){  
 add(obj, 0);  
 }  
  
 public void addToRear(T obj){  
 add(obj, size);  
 }  
  
 public boolean isEmpty(){  
 return size == 0;  
 }  
  
 public int size(){  
 return size;  
 }  
 /\*  
  
 in the methods 'removeFront' and 'removeRear' the user will not enter any index  
 the user will just call the method and it will remove either the front (0) index or the rear(size - 1) index  
 but if the list is empty, the remove(int index) method will be called  
 where checkIndex will be called and it will throw IndexOutOfBoundsException  
 as the user has not entered any index, I do not want to throw IndexOutOfBoundsException  
 so I made a custom exception, 'EmptyListException'  
  
 \*/  
  
 public T removeFront(){  
 checkListNotEmpty(); //EmptyListException is thrown here, if needed  
 T obj = remove(0);  
 return obj;  
 }  
  
 public T removeRear(){  
 checkListNotEmpty(); //EmptyListException is thrown here, if needed  
 T obj = remove(size - 1);  
 return obj;  
 }  
  
 public void checkIndex(int index){  
 if(index < 0 || index > size - 1){  
 throw new IndexOutOfBoundsException("Index = "+ index + " Size = "+size);  
 }  
 }  
  
 public T get(int index){  
 checkIndex(index);  
 return arr[index];  
 }  
  
 public int indexOf(T obj){  
 for(int i = 0; i < size; i++){  
 //the 'if' condition used is to prevent any 'NullPointerException'  
 if((arr[i] == null && obj == null) || (arr[i] != null && arr[i].equals(obj))){  
 return i;  
 }  
 }  
 return -1;  
 }  
  
 public T remove(int index){  
 checkIndex(index);  
 T temp = arr[index];  
 for(int i = index + 1; i < size; i++){  
 arr[i - 1] = arr[i];  
 }  
 size--;  
 arr[size] = null;  
 return temp;  
 }  
  
 public void add(T obj, int index){  
 if(index < 0 || index > size){  
 throw new IndexOutOfBoundsException("Index = "+ index +" Size = "+size);  
 }  
 //checking capacity  
 if(size == arr.length){  
 T []newArray = (T[])new Object[2 \* size];  
 int i = 0;  
 for(T o : arr){  
 newArray[i++] = o;  
 }  
 this.arr = newArray;  
 }  
 //shift elements to right   
 for(int i = size; i > index; i--){  
 arr[i] = arr[i - 1];  
 }  
 arr[index] = obj;  
 size++;  
 }  
  
 public String toString(){  
 if(size == 0){  
 return "[]";  
 }  
 StringBuilder sb = new StringBuilder();  
 sb.append("[");  
 for(int i = 0; i < size - 1; i++){  
 sb.append((arr[i] != null ? arr[i].toString() : "null") + ", ");  
 }  
 sb.append((arr[size - 1] != null ? arr[size - 1].toString() : "null") + "]");  
 return sb.toString();  
 }  
  
 public void checkListNotEmpty(){  
 if(size == 0){  
 throw new EmptyListException();  
 }  
 }  
}  
  
class EmptyListException extends RuntimeException{  
 public EmptyListException(){  
 super("List is Empty");  
 }  
}

***OUTPUT***





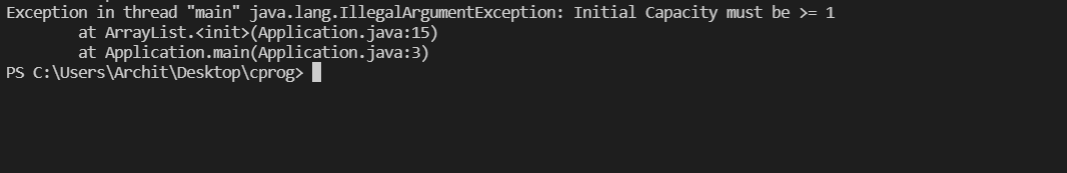
***Code for throwing Exceptions -:***

The changes required in the code above in order to throw exceptions are to be done in Application class only.

Hence, instead of pasting the whole code again and again, only the class Application code is pasted.

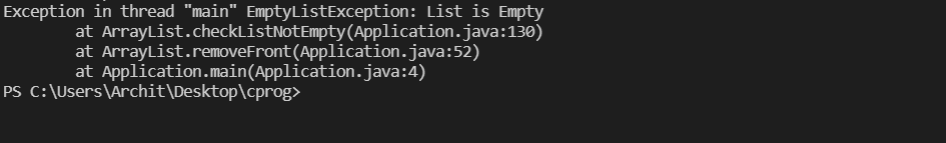
* *For throwing IllegalArgumentException in constructor of ArrayList*

public class Application {  
 public static void main(String []args){  
 ArrayList <Integer> list = new ArrayList<Integer>(0); // creating an arraylist with initial capacity of 10  
  
 }  
}



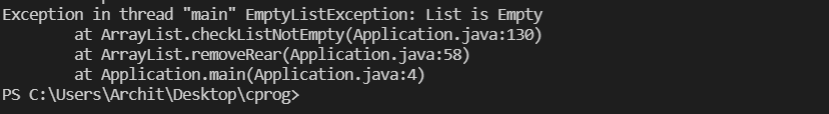
* *For throwing EmptyListException in method removeFront*

public class Application {  
 public static void main(String []args){  
 ArrayList <Integer> list = new ArrayList<Integer>(10);  
 System.*out*.println("Object removed from front : "+list.removeFront());  
 }  
}



* *For throwing EmptyListException in method removeRear*

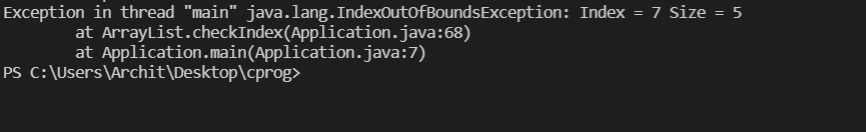
public class Application {  
 public static void main(String []args){  
 ArrayList <Integer> list = new ArrayList<Integer>(10);  
 System.*out*.println("Object removed from rear : "+list.removeRear());  
 }  
}



* *For throwing IndexOutOfBoundsException in method checkIndex, get, remove*

Since the method ‘get’ and ‘remove’ call the ‘checkIndex’ method , hence *IndexOutOfBoundsException* will be thrown if any inconsistent index is entered

public class Application {  
 public static void main(String []args){  
 ArrayList <Integer> list = new ArrayList<Integer>(10);  
 for(int i = 0; i < 5; i++){  
 list.add(i, i);  
 }  
 list.checkIndex(7); //list.get(7) or list.remove(7)  
 }  
}



* *For throwing IndexOutOfBoundsException in method add*

public class Application {  
 public static void main(String []args){  
 ArrayList <Integer> list = new ArrayList<Integer>(10);  
 for(int i = 0; i < 5; i++){  
 list.add(i, i);  
 }  
 list.add(6, 6);  
 }  
}

