**Java Collection Assignment**

**Java Collection: ArrayList Exercises**

1. Write a Java program to create a new array list, add some Movie names (string) and print out the

collection.

-Write a Java program to insert an element into the array list at the first -position.

-Write a Java program to retrieve an element (at a specified index) from a given array list.

-Write a Java program to update specific array elements by given element.

-Write a Java program to remove the third element from an array list.

-Write a Java program to search for an element in an array list.

-Write a Java program to sort a given array list.

-Write a Java program to reverse elements in an array list.

-Write a Java program to empty an array list.

Ans:-

import java.util.\*;

public class Moviearr

{

public static void main(String[] args)

{

ArrayList <String> movie = new ArrayList <String>();

movie.add("PK");

movie.add("Sita Ramam");

movie.add("Bahubali");

movie.add("Wake up sid");

movie.add("Golmal");

System.out.println(movie);

//insert an element into the array list at the first-position

System.out.println("1.Insert an element into the array list at the first -position.");

movie.add(0, "Dilwale");

System.out.println(movie);

//retrieve an element (at a specified index) from a given array list.

System.out.println("2.Retrieve an element (at a specified index) from a given array list.");

System.out.println(movie.get(0));

//update specific array elements by given element.

System.out.println("3.Update specific array elements by given element.");

movie.set(0,"KGF");

System.out.println(movie);

//remove the third element from an array list.

System.out.println("4.remove the third element from an array list.");

movie.remove(2);

System.out.println(movie);

//search for an element in an array list.

System.out.println("5.search for an element in an array list.");

System.out.println(movie.contains("Bahubali"));

//sort a given array list.

System.out.println("6.sort a given array list.");

Collections.sort(movie);

System.out.println(movie);

//to reverse elements in an array list.

System.out.println("7.reverse elements in an array list.");

Collections.reverse(movie);

System.out.println(movie);

//empty an array list.

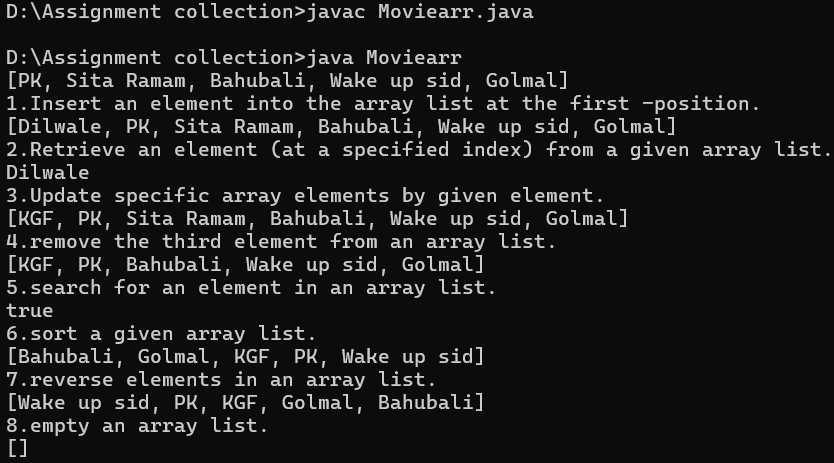
System.out.println("8.empty an array list.");

movie.clear();

System.out.println(movie);

}

}



**Java Collection: LinkedList**

**1.** Write a Java program to append the specified element to the end of a linked list of names.

-Write a Java program to iterate through all elements in a linked list starting at the specified position.

-Write a Java program to iterate a linked list in reverse order.

-Write a Java program to insert the specified element at the specified position in the linked list.

-Write a Java program to insert elements into the linked list at the first and last position.

-Write a Java program to insert the specified element at the front of a linked list.

-Write a Java program to insert some elements at the specified position into a linked list.

-Write a Java program to get the first and last occurrence of the specified elements in a linked list.

-Write a Java program to remove the first and last element from a linked list.

-Write a Java program to swap two elements in a linked list.

-Write a Java program to join two linked lists.

-Write a Java program to check if a particular element exists in a linked list.

-Write a Java program to convert a linked list to an array list. ‘’’’’’’’

-Write a Java program to compare two linked lists.

-Write a Java program to test whether a linked list is empty or not.

-Write a Java program to replace an element in a linked list.

import java.util.\*;

public class Movielin

{

public static void main(String[] args)

{

LinkedList <String> movie = new LinkedList <String>();

movie.add("PK");

movie.add("Sita Ramam");

movie.add("Bahubali");

movie.add("Wake up sid");

movie.add("Golmal");

System.out.println(movie);

//1.append the specified element to the end of a linked list of names.

System.out.println("1.append the specified element to the end of a linked list of names.");

movie.addLast("ABCD");

System.out.println(movie);

//2.iterate through all elements in a linked list starting at the specified position.

System.out.println("2.iterate through all elements in a linked list starting at the specified position.");

ListIterator<String> it = movie.listIterator(2);

while(it.hasNext())

{

System.out.println(it.next());

}

//3.iterate a linked list in reverse order.

System.out.println("iterate a linked list in reverse order.");

Iterator<String> descIter = movie.descendingIterator();

while (descIter.hasNext())

{

System.out.println(descIter.next());

}

//4.7insert the specified element at the specified position in the linked list.

System.out.println("3.insert the specified element at the specified position in the linked list.");

movie.add(2, "matrix");

System.out.println(movie);

//5.insert elements into the linked list at the first and last position.

System.out.println("4.insert elements into the linked list at the first and last position.");

movie.addFirst("Bagi");

movie.addLast("Ruhi");

System.out.println(movie);

//6 .insert the specified element at the front of a linked list.

movie.addFirst("Jogi");

System.out.println(movie);

// remove the first and last element from a linked list.

System.out.println("remmove first and last");

movie.removeFirst();

movie.removeLast();

System.out.println(movie);

//swap two elements in a linked list.

System.out.println("swap two elements in a linked list.");

Collections.swap(movie, 4, 2);

System.out.println(movie);

//check if a particular element exists in a linked list.

System.out.println("check if a particular element exists in a linked list. ");

System.out.println(movie.contains("Jogi"));

//test whether a linked list is empty or not.

System.out.println("test whether a linked list is empty or not.");

if (movie.isEmpty())

{

System.out.println("Linked list is empty");

}

else

{

System.out.println("Linked list is not empty");

}

//replace an element in a linked list.

System.out.println("replace an element in a linked list. ");

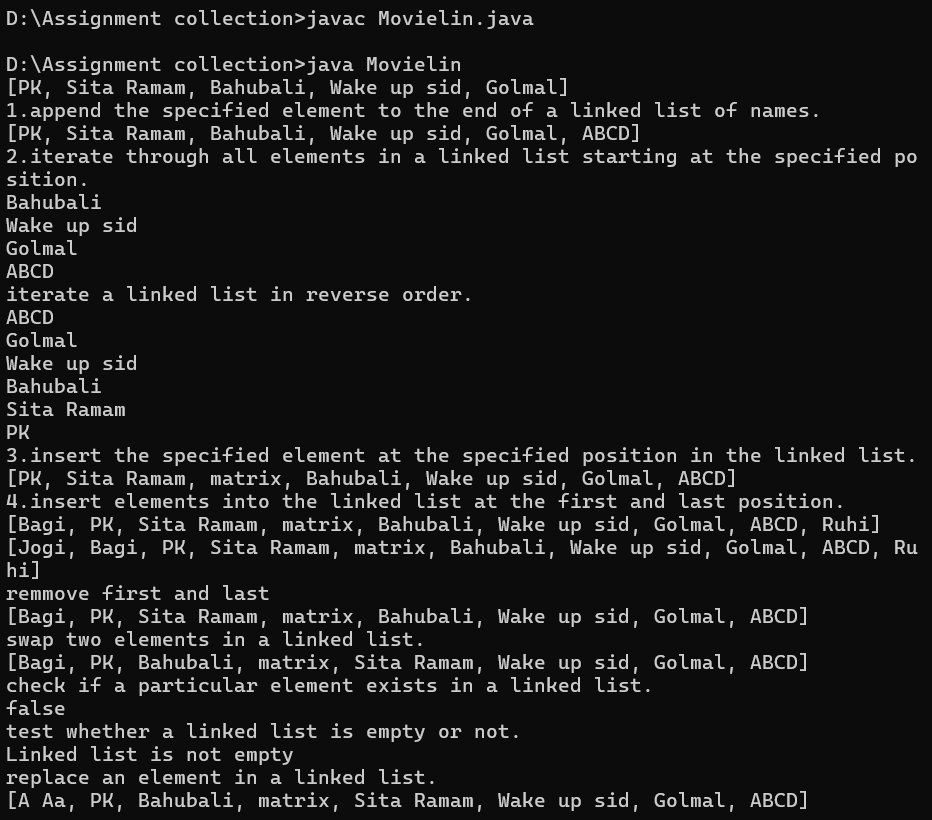
movie.set(0,"A Aa");

System.out.println(movie);

//compare two linked lists.

}

}



import java.util.LinkedList;

import java.util.ArrayList;

public class twolink {

public static void main(String[] args) {

LinkedList<Integer> list1 = new LinkedList<>();

LinkedList<Integer> list2 = new LinkedList<>();

list1.add(10);

list1.add(20);

list1.add(30);

list2.add(10);

list2.add(20);

list2.add(30);

//compare two linked lists.

if (list1.equals(list2)) {

System.out.println("Both linked lists are equal");

} else {

System.out.println("Linked lists are not equal");

}

//join two linked lists.

list1.addAll(list2);

System.out.println("Joined linked list"+list1);

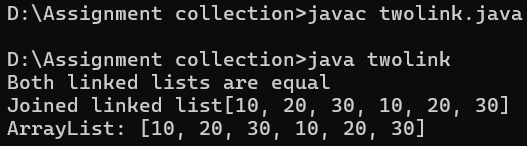
//convert a linked list to an array list.

ArrayList<Integer> arrayList = new ArrayList<>(list1);

System.out.println("ArrayList: " + arrayList);

}

}



**Java Collection: HashSet Exercises**

1. Write a Java program to append the specified element to the end of a hash set for Employee Id and

Employee name.

-Write a Java program to get the number of elements in a hash set.

-Write a Java program to convert a hash set to an array.

-Write a Java program to convert a hash set to a tree set.

-Write a Java program to convert a hash set to a List/ArrayList.

-Write a Java program to remove all of the elements from a hash set.

Ans:-

import java.util.\*;

public class HashSetOperations {

public static void main(String[] args) {

// 1️Create a HashSet and append Employee entries

Set<String> employeeSet = new HashSet<>();

employeeSet.add("E101 - Alice");

employeeSet.add("E102 - Bob");

employeeSet.add("E103 - Charlie");

System.out.println("1HashSet after adding employees: " + employeeSet);

// 2️Get the number of elements in the HashSet

System.out.println("2Number of employees: " + employeeSet.size());

// 3️Convert HashSet to an Array

String[] employeeArray = employeeSet.toArray(new String[0]);

System.out.println("3Employees in array:");

for (String emp : employeeArray) {

System.out.println(emp);

}

// 4️Convert HashSet to TreeSet (sorted set)

Set<String> employeeTreeSet = new TreeSet<>(employeeSet);

System.out.println("4Employees in TreeSet (sorted): " + employeeTreeSet);

// 5️Convert HashSet to ArrayList

List<String> employeeList = new ArrayList<>(employeeSet);

System.out.println("5Employees in ArrayList: " + employeeList);

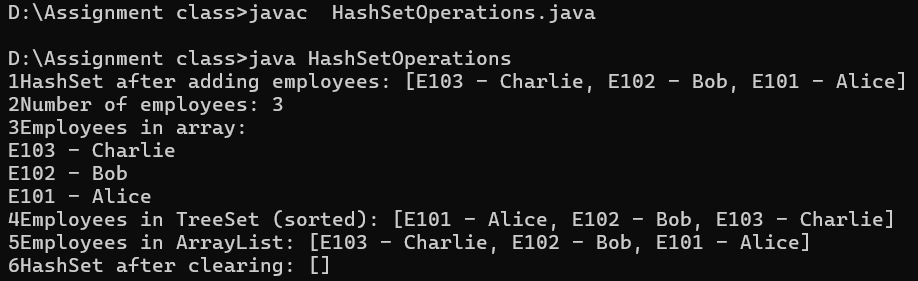
// 6️Remove all elements from the HashSet

employeeSet.clear();

System.out.println("6HashSet after clearing: " + employeeSet);

}

}



**Java Collection: TreeSet**

1. Write a Java program to create a new tree set, add some fruits (string) and print out the tree set.

-Write a Java program to iterate through all elements in a tree set.

-Write a Java program to add all the elements of a specified tree set to another tree set.

-Write a Java program to create a reverse order view of the elements contained in a given tree set.

-Write a Java program to find the numbers less than 7 in a tree set.

Ans:-

import java.util.\*;

public class TreeSetSruits

{

public static void main(String[] args)

{

TreeSet<String> fruits = new TreeSet<>();

TreeSet<Integer> fruits1 = new TreeSet<>();

TreeSet<String> fruits3 = new TreeSet<>();

//Add1

fruits.add("Apple");

fruits.add("Banana");

fruits.add("Mango");

fruits.add("Orenge");

fruits.add("Kiwi");

//Add2

fruits1.add(1);

fruits1.add(2);

fruits1.add(3);

fruits1.add(4);

fruits1.add(5);

//Add3

fruits3.add("Grapes");

fruits3.add("pear");

System.out.println("Treesetof Fruits: "+fruits);

System.out.println("Treesetof numbers: "+fruits1);

System.out.println("Treesetof Fruits: "+fruits3);

//Iterate

System.out.println("Iterate: ");

for (String fruit : fruits)

{

System.out.println(fruit);

}

//reverse

System.out.println("reverse the String");

Iterator<String> reverseIterator = fruits.descendingIterator();

System.out.println("Fruits in reverse order:");

while (reverseIterator.hasNext())

{

System.out.println(reverseIterator.next());

System.out.println(fruits);

}

// find the numbers less than 7 in a tree set.

System.out.println(" find the numbers less than 7 in a tree set. ");

for (Integer num : fruits1)

{

if (num < 7)

{

System.out.println(num + " is less than 7");

}

else

{

System.out.println(num + " is not less than 7");

}

// add all the elements of a specified tree set to another tree set.

System.out.println("fruits before addAll: " + fruits);

System.out.println("fruits3: " +fruits3 );

fruits.addAll(fruits3);

System.out.println("fruits after addAll: " + fruits);

}

}

}



Java Collection: HashMap

1. Write a Java program to associate the specified value with the specified key in a HashMap.

-Write a Java program to count the number of key-value (size) mappings in a map.

-Write a Java program to copy all of the mappings from the specified map to another map.

-Write a Java program to remove all of the mappings from a map.

-Write a Java program to test if a map contains a mapping for the specified key.

-Write a Java program to test if a map contains a mapping for the specified value.

Ans:-

import java.util.HashMap;

import java.util.Map;

public class HashMapDemo {

public static void main(String[] args) {

// Create a HashMap

Map<String, String> map = new HashMap<>();

// 1. Associate a value with a key

map.put("Red", "Apple");

map.put("Yellow", "Banana");

map.put("Green", "Grapes");

System.out.println("Map after adding values: " + map);

// 2. Count the number of key-value mappings

System.out.println("Size of map: " + map.size());

// 3. Copy all mappings to another map

Map<String, String> copyMap = new HashMap<>();

copyMap.putAll(map);

System.out.println("Copied map: " + copyMap);

// 4. Remove all mappings from the copied map

copyMap.clear();

System.out.println("Copied map after clearing: " + copyMap);

// 5. Check if map contains a specific key

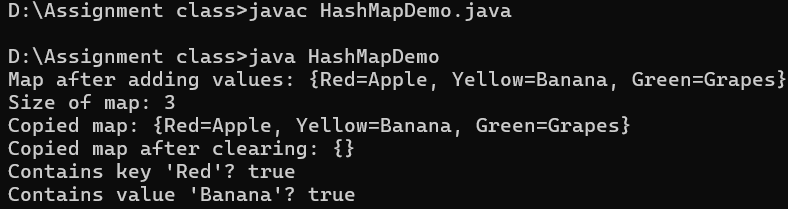
System.out.println("Contains key 'Red'? " + map.containsKey("Red"));

// 6. Check if map contains a specific value

System.out.println("Contains value 'Banana'? " + map.containsValue("Banana"));

}

}



**Practice Problem: Ex:**1

Implement different operations on an ArrayList A.

**Input**:

The first line of input contains an integer **T** denoting the no of test cases. Then T test cases follow. The first line of input contains an integer **Q** denoting the no of queries. Then in the next line

are **Q** space separated queries .

A query can be of five types

1. a x (Adds an element x to the ArrayList A at the end)

2. b (Sorts the ArrayList A in ascending order)

3. c (Reverses the ArrayList A)

4. d (prints the size of the ArrayList)

5. e (prints space separated values of the ArrayList)

5. f (Sorts the ArrayList A in descending order)

**Output:**

The output for each test case will be space separated integers denoting the results of each query

**Constraints:**

**1<=T<=**100

**1<**=Q<**=**100

**Example**:

**Input**

2

6

a4 a6 a7bce

**4**

a 55 a 11 de

**Output**

764

2 55 11

**Explanation:**

**For the first test case**

There are six queries. Queries are performed in this order

1. a 4 { ArrayList has 4 }

2. a 7 {ArrayList has 7 }

3. a 6 {ArrayList has 6}

4. b {sorts the ArrayList in ascending order, ArrayList now is 5 6 7}

5. c {reverse the ArrayList}

6. e {prints the element of the ArrayList 7 6 4}

**For the sec test case**

There are four queries. Queries are performed in this order

1. a 55 (ArrayList A has 55}

(prints the size of the ArrayList A ie. 2)

2. a 11

(ArrayList A has 55,11**}**

3. d

**4.** e

(prints the elements of the ArrayList A ie 55 11)

**Ans:-**

**import java.util.\*;**

**public class ArrayListOperations {**

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

**Scanner sc = new Scanner(System.in);**

**int T = sc.nextInt(); // Number of test cases**

**sc.nextLine(); // Consume leftover newline**

**for (int t = 0; t < T; t++) {**

**int Q = sc.nextInt(); // Number of queries**

**sc.nextLine(); // Consume leftover newline**

**String[] queries = sc.nextLine().split(" ");**

**List<Integer> A = new ArrayList<>();**

**List<String> output = new ArrayList<>();**

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

**String query = queries[i];**

**if (query.startsWith("a")) {**

**if (query.length() > 1) {**

**A.add(Integer.parseInt(query.substring(1)));**

**} else {**

**i++;**

**A.add(Integer.parseInt(queries[i]));**

**}**

**} else if (query.equals("b")) {**

**Collections.sort(A);**

**} else if (query.equals("c")) {**

**Collections.reverse(A);**

**} else if (query.equals("d")) {**

**output.add(String.valueOf(A.size()));**

**} else if (query.equals("e")) {**

**for (int num : A) {**

**output.add(String.valueOf(num));**

**}**

**} else if (query.equals("f")) {**

**A.sort(Collections.reverseOrder());**

**}**

**}**

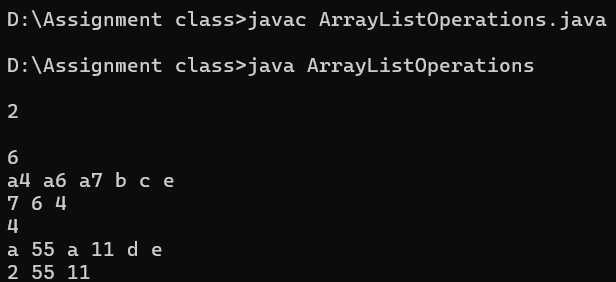
**System.out.println(String.join(" ", output));**

**}**

**sc.close();**

**}**

**}**

****

**Practice Problem: Ex:2**

ArrayLis

t are dynamic size arrays. Try this problem using ArrayList.

Given an ArrayList of **N** elements and an integer **Q** defining the type of query(which will be either 1 or 2): **Q = 1** includes two integers **p** and **r**. Which means insert the value r at index p in the ArrayList and print the whole updated ArrayList.

**Q = 2** includes one integer **p**. In this query print the index at which the value p is last found in the ArrayList. If the value p is not found in the ArrayList then print **"-1"**.

**NOTE: Assume 0 based indexing**

**Example 1:**

**Input:**

N

=

5, Q

= 1

A[] = {1, 4, 5, 9, 3}

Query []

**Output:**

= {2,6}

**1** 4 6 5 9 3

**Explanation:**

p=Query [0]=2

r=Query [1]=6

After inserting the element r=6 at index p=2, the updated arraylist ={1,4,6,5,9,3}

**Example 2**:

**Input:**

N = 4, Q

= 2

A[]= {1, 9, 2, 4}

Query[] = {4}

**Output:**

**Explanation:**

3

р

= 4

The element 4 is last found

in A at index = 3

**Your Task:**

You don't need to read input or print anything. Your task is to complete the function **solve()** which takes the **N** (number of elements in Array A),ArrayList **A, Q**(Type of the of query) and the ArrayList **Query**. If the Q = 1 then return the updated ArrayList of integers. else return the ArrayList which contains the index at which the value p is last found in the ArrayList A (where p = Query[0]),If the value of p is not found then return the ArrayList which contains -1.

**Expected Time Complexity:** O(N)

**Expected Auxiliary Space:** O(N) **Constraints:**

1 **<=** N **<=** 104

1 **<=**Q **<= 2**

If Q = 1 then size of Query is 2,

where Query[0] represents the value of p and Query[0] represents the value of r.

If Q = 2 then size of Query is 1,

where Query[0] represents the value of p.

1 **<**= A[i] **<**= 103

Ans:-

import java.util.\*;

public class ArrayListQuerySolver {

public static ArrayList<Integer> solve(int N, ArrayList<Integer> A, int Q, ArrayList<Integer> Query) {

ArrayList<Integer> result = new ArrayList<>();

if (Q == 1) {

int p = Query.get(0);

int r = Query.get(1);

A.add(p, r);

result.addAll(A);

} else if (Q == 2) {

int p = Query.get(0);

int lastIndex = -1;

for (int i = A.size() - 1; i >= 0; i--) {

if (A.get(i) == p) {

lastIndex = i;

break;

}

}

result.add(lastIndex);

}

return result;

}

public static void main(String[] args) {

ArrayList<Integer> A1 = new ArrayList<>(Arrays.asList(1, 4, 5, 9, 3));

ArrayList<Integer> Query1 = new ArrayList<>(Arrays.asList(2, 6));

ArrayList<Integer> result1 = solve(5, A1, 1, Query1);

System.out.println("After insertion: " + result1);

ArrayList<Integer> A2 = new ArrayList<>(Arrays.asList(1, 9, 2, 4));

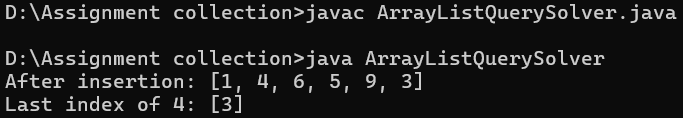
ArrayList<Integer> Query2 = new ArrayList<>(Arrays.asList(4));

ArrayList<Integer> result2 = solve(4, A2, 2, Query2);

System.out.println("Last index of 4: " + result2);

}

}



**Practice Problem: Ex:3**

Java provides an inbuilt object type called **Stack**. It is a collection that is based on the last in first out (LIFO) principle. Try this problem using Stack.

Given **n** elements of a stack **st** where the first value is the bottom-most value of the stack and the last one is the element at top of the stack, delete the middle element of the stack without using any additional data structure.

**Example 1**:

**Input:** n = 5

st

=

{1**,** 2, 3, 4, 5}

**Output:** 5 4 2 1

**Explanation:** The middle element is 3. If

**it** is deleted and then the values are seen from top, this will be the order.

**Example 2**:

**Input:** n = 6

st **=**

{1**,** 4, 9, 2, 6, 5}

**Output:** 5 6 2 4 1

**Explanation**: The middle element is 9 and **if**

**it** is deleted this will be the stack traversal.

**Your Task:**

You do not need to read input or print anything. Your task is to complete the function **deleteMid()** which takes n and st as input parameters and returns a stack where the middle element is deleted.

**Expected Time Complexity:** O(n)

**Expected Auxiliary Space:** O(n)

**Constraints:**

2 ≤ n ≤ 103

1 ≤ st[i] ≤ 104

**Practice Problem: Ex:4**

Implement different operations on a set s.

**Input**:

The first line of input contains an integer **T** denoting the no of test cases. Then T test cases follow. The first line of input contains an integer **Q** denoting the no of queries. Then in the next line are **Q** space separated queries .

A query can be of four types

**1.** a x (inserts an element x to the set s)

**2.** b (prints the contents of the set s in increasing order)

**3.** cx (erases an element x from the sets)

**4.** d x (prints 1 if the element x is present in the set else print **-1)**

**5.** e (prints the size of the set s)

**Output:**

The output for each test case will be space separated integers denoting the results of each query. **Constraints:**

1 **<=** T **<=** 100

1 **<=** Q **<=** 100

**Example:**

**Input**:

2

6

a 1 a 2 a 3 bc2b

5

a 1a5ed5d2

**Output**:

12313

21-1

**Explanation:**

**Testcase 1:**

There are six queries. Queries are performed in this order

1. a 1

2. a 2 3. a 3 4. b

5. c **2**

6. b

{ insert 1 to set now set has {1}}

{inserts 2 to set now set has {1,2} } {inserts 3 to set now set has {1,2,3}} {prints the set contents ie 1,2,3}

{removes 2 from the set}

{prints the set contents ie 1,3}

**Testcase 2:**

There are five queries. Queries are performed in this order

1. a 1

{inserts 1 to set now set has {1}}

2. a 11

{inserts 11 to set now set has {1,11}}

3. e

{prints the size of the set ie 2}

4. d 5

{since five is present prints 1}

5. d 2

{since 2 is not present in the set prints **-1}**

**Ans:-**

**import java.util.\*;**

**public class CombinedOperations {**

**public static void deleteMidHelper(Stack<Integer> st, int current, int mid) {**

**if (current == mid) {**

**st.pop();**

**return;**

**}**

**int temp = st.pop();**

**deleteMidHelper(st, current + 1, mid);**

**st.push(temp);**

**}**

**public static Stack<Integer> deleteMid(int n, Stack<Integer> st) {**

**int mid = n / 2;**

**deleteMidHelper(st, 0, mid);**

**return st;**

**}**

**public static void processSetQueries(List<String> queries) {**

**TreeSet<Integer> set = new TreeSet<>();**

**StringBuilder output = new StringBuilder();**

**for (String query : queries) {**

**if (query.startsWith("a")) {**

**int x = Integer.parseInt(query.substring(2));**

**set.add(x);**

**} else if (query.equals("b")) {**

**for (int val : set) output.append(val);**

**} else if (query.startsWith("c")) {**

**int x = Integer.parseInt(query.substring(1));**

**set.remove(x);**

**} else if (query.startsWith("d")) {**

**int x = Integer.parseInt(query.substring(2));**

**output.append(set.contains(x) ? "1" : "-1");**

**} else if (query.equals("e")) {**

**output.append(set.size());**

**}**

**}**

**System.out.println("Set Output: " + output.toString());**

**}**

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

**Stack<Integer> st = new Stack<>();**

**st.addAll(Arrays.asList(1, 2, 3, 4, 5));**

**Stack<Integer> updatedStack = deleteMid(st.size(), st);**

**System.out.print("Stack after deleting middle: ");**

**while (!updatedStack.isEmpty()) {**

**System.out.print(updatedStack.pop() + " ");**

**}**

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

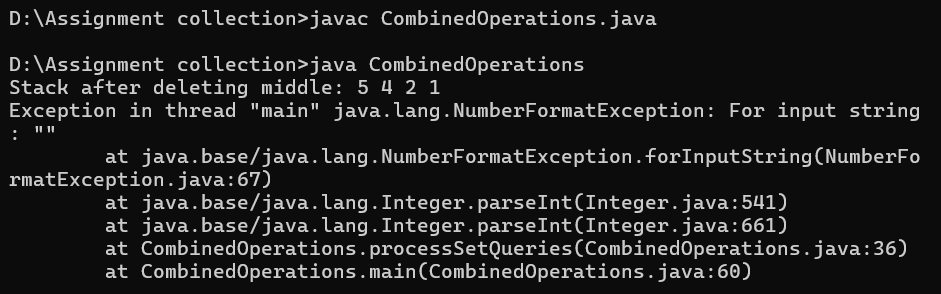
**// Task 2: Set operations**

**List<String> queries = Arrays.asList("a 1", "a 2", "a 3", "b", "c2", "b", "e", "d3", "d5");**

**processSetQueries(queries);**

**}**

**}**

****