DAY 8

1) **Search an Element**

```
import java.util.*;
public class SearchElementArrayList {
public static void main(String[] args) {
ArrayList<Integer> list = new ArrayList(Arrays.asList(15, 25, 35, 45, 55));
Scanner sc = new Scanner(System.in);
System.out.print("Enter a number to search: ");
int num = sc.nextInt();
if (list.contains(num)) {
System.out.println(num + " was found in the list.");
} else {
System.out.println(num + " was not found in the list.");
}
}
}
```

Output:

Enter a number to search: 35 35 was found in the list.

2) **Remove a Specific Element**

```
import java.util.*;
public class RemoveFruit {
public static void main(String[] args) {
ArrayList<String> fruits = new ArrayList(Arrays.asList("Strawberry", "Pineapple", "Kiwi", "Mango"));
fruits.remove("Kiwi");
System.out.println(fruits);
}
}
```

Output: [Strawberry, Pineapple, Mango]

3) **Sort Elements**

```
import java.util.*;
public class SortArrayList {
public static void main(String[] args) {
ArrayList<Integer> list = new ArrayList(Arrays.asList(12, 5, 80, 42, 3));
Collections.sort(list);
System.out.println("Sorted list: " + list);
}
}
```

Output: Sorted list: [3, 5, 12, 42, 80]

4) **Reverse the ArrayList**

```
import java.util.*;
public class ReverseArrayList {
public static void main(String args) {
ArrayList<Character> chars = new ArrayList(Arrays.asList('P', 'Q', 'R', 'S', 'T'));
Collections.reverse(chars);
System.out.println("Reversed list: " + chars);
}
}
```

Output: Reversed list: [T, S, R, Q, P]

5) **Update an Element**

```
import java.util.*;
public class UpdateElement {
public static void main(String[ args) {
ArrayList<String> subjects = new ArrayList(Arrays.asList("History", "Geography", "English"));
System.out.println("Before update: " + subjects);
subjects.set(subjects.indexOf("Geography"), "Social Science");
System.out.println("After update: " + subjects);
}
}
```

Output: Before update: [History, Geography, English] and After update: [History, Social Science, English]

6) **Remove All Elements**

```
import java.util.*;
public class ClearArrayList {
public static void main(String args) {
ArrayList<Integer> list = new ArrayList(Arrays.asList(50, 60, 70));
list.clear();
System.out.println("New size of the list: " + list.size());
}
}
```

Output: New size of the list: 0

7) **Iterate using Iterator**

```
import java.util.*;
public class IterateCities {
public static void main(String args) {
ArrayList<String> cities = new ArrayList(Arrays.asList("London", "Paris", "Tokyo"));
Iterator<String> it = cities.iterator();
while (it.hasNext()) {
System.out.println("City: " + it.next());
}
}
}
```

Output: City: London, City: Paris, City: Tokyo

8) **Store Custom Objects**

```
import java.util.*;
class Student {
int id;
String name;
double marks;
Student(int id, String name, double marks) {
this.id = id; this.name = name; this.marks = marks;
}
}
public class StudentArrayList {
public static void main(String[] args) {
ArrayList<Student> students = new ArrayList();
students.add(new Student(10, "Varun", 95.0));
students.add(new Student(20, "Sonia", 88.5));
for (Student s : students) {
System.out.println("ID: " + s.id + ", Name: " + s.name + ", Marks: " + s.marks);
}
}
}
```
Output: ID: 10, Name: Varun, Marks: 95.0 and ID: 20, Name: Sonia, Marks: 88.5

9) **Copy One ArrayList to Another**
```
import java.util.*;
public class CopyArrayList {
public static void main(String[] args) {
ArrayList<String> list1 = new ArrayList(Arrays.asList("Hi", "Hello", "Helo"));
ArrayList<String> list2 = new ArrayList<>();
list2.addAll(list1);
System.out.println("Original List: " + list1);
System.out.println("Copied List: " + list2);
}
}
```
Output: Original List: [Hi,Hello,Helo] and Copied List: [Hi,Hello,Helo]

10) **Create and Display a LinkedList**
```
import java.util.*;
public class LinkedListColors {
public static void main(String[ args) {
LinkedList<String> colors = new LinkedList(Arrays.asList("Cyan", "Magenta", "Yellow", "Black"));
for (String color: colors) {
System.out.println("Color: " + color);
}
}
}
```
Output: Color: Cyan, Color: Magenta, Color: Yellow, Color: Black

11) **Add Elements at First and Last Position**
```
import java.util.*;
public class LinkedListAddFirstLast {
public static void main(String[] args) {
LinkedList<Integer> numbers = new LinkedList(Arrays.asList(100, 200, 300));
numbers.addFirst(50);
numbers.addLast(400);
System.out.println(numbers);
}
}
```
Output: [50, 100, 200, 300, 400]

12) **Insert Element at Specific Position**
```
import java.util.*;
public class LinkedListInsert {
public static void main(String args) {
LinkedList<String> names = new LinkedList(Arrays.asList("Lisa", "Sam", "Tom"));
System.out.println("Before insert: " + names);
names.add(1, "David");
System.out.println("After insert: " + names);
}
}
```
Output: Before insert: [Lisa, Sam, Tom] and After insert: [Lisa, David, Sam, Tom]

13) **Remove Elements**

```
import java.util.*;
public class LinkedListRemove {
public static void main(String[] args) {
LinkedList<String> animals = new LinkedList(Arrays.asList("Lion", "Tiger", "Bear", "Wolf"));
animals.removeFirst();
System.out.println("Removed first element: " + animals);
animals.removeLast();
System.out.println("Removed last element: " + animals);
animals.remove("Bear");
System.out.println("Removed 'Bear': " + animals);
}
}
```
Output: Removed first element: [Tiger, Bear, Wolf], Removed last element: [Tiger, Bear], and Removed 'Bear': [Tiger]

14) **Search for an Element**

```
import java.util.*;
public class LinkedListSearch {
public static void main(String[] args) {
LinkedList<String> list = new LinkedList(Arrays.asList("Cherry", "Grape", "Melon"));
Scanner sc = new Scanner(System.in);
System.out.print("Enter fruit name to search: ");
String fruit = sc.nextLine();
if (list.contains(fruit)) {
System.out.println(fruit + " was found in the list.");
} else {
System.out.println(fruit + " was not found in the list.");
}
}
}
```
Input : Grape
Output : Grape was found in the list.

Input : Mango
Output: Mango was not found in the list.

15) **Iterate using ListIterator**

```
import java.util.*;
public class LinkedListListIterator {
public static void main(String args) {
LinkedList<String> cities = new LinkedList(Arrays.asList("Pune", "Goa", "Hyd"));
ListIterator<String> it = cities.listIterator();
System.out.println("Forward iteration:");
while (it.hasNext()) {
System.out.println(it.next());
}
System.out.println("Backward iteration:");
while (it.hasPrevious()) {
```

```
System.out.println(it.previous());
}
}
}
```
Output: Forward iteration:
Pune
Goa
Hyd
Backward iteration:
Hyd
Goa
Pune

16) **Sort a LinkedList**
```
import java.util.*;
public class LinkedListSort {
public static void main(String[] args) {
LinkedList<Integer> list = new LinkedList(Arrays.asList(90, 70, 60, 80, 50));
Collections.sort(list);
System.out.println("Sorted list: " + list);
}
}
```
Output: Sorted list: [50, 60, 70, 80, 90]

17) **Convert LinkedList to ArrayList**
```
import java.util.*;
public class LinkedListToArrayList {
public static void main(String args) {
LinkedList<String> ll = new LinkedList(Arrays.asList("Red", "Blue", "Green"));
ArrayList<String> al = new ArrayList(ll);
System.out.println("Original LinkedList: " + ll);
System.out.println("Converted ArrayList: " + al);
}
}
```
Output:
Original LinkedList: [Red, Blue, Green]
Converted ArrayList: [Red, Blue, Green]

18) **Store Custom Objects**
```
import java.util.*;
class Book {
int id;
String title;
String author;
Book(int id, String title, String author) {
this.id = id; this.title = title; this.author = author;
}
}
public class LinkedListBooks {
public static void main(String[] args) {
LinkedList<Book> books = new LinkedList();
books.add(new Book(101, "The Hobbit", "J.R.R. Tolkien"));
books.add(new Book(102, "1984", "George Orwell"));
for (Book b : books) {
System.out.println("ID: " + b.id + ", Title: " + b.title + ", Author: " + b.author);
}
}
}
```

Output:

ID: 101, Title: The Hobbit, Author: J.R.R. Tolkien and ID: 102, Title: 1984, Author: George Orwell

19) **Add, Insert, Remove, and Display**

```
import java.util.*;
public class VectorIntegers {
public static void main(String[] args) {
Vector<Integer> v = new Vector<>();
v.add(100);
v.add(200);
v.add(300);
v.add(400);
v.add(500);
v.add(2, 250);
v.remove(1);
Enumeration<Integer> e = v.elements();
while (e.hasMoreElements()) {
System.out.println(e.nextElement());
}
}
}
```

Output: 100, 250, 300, 400, 500

20) **Vector of Strings**

```
import java.util.*;
public class VectorStrings {
public static void main(String[] args) {
Vector<String> v = new Vector<>(Arrays.asList("Harshu", "Thejas", "Dhanya", "Yakshith"));
System.out.println("Contains 'Harshu': " + v.contains("Harshu"));
v.set(1, "Jenny");
System.out.println("After replacement: " + v);
v.clear();
System.out.println("After clearing: " + v);
}
}
```

Output:

```
Contains 'Harshu': true
After replacement: [Harshu,Jenny,Dhanya,Yakshith]
After clearing: []
```

21) **Copy Elements**

```
import java.util.*;
public class VectorCopy {
public static void main(String[] args) {
Vector<String> v1 = new Vector<>(Arrays.asList("Circle", "Square", "Triangle"));
Vector<String> v2 = new Vector<>();
v2.addAll(v1);
System.out.println("Vector 1: " + v1);
System.out.println("Vector 2: " + v2);
}
}
```
Output:

```
Vector 1: [Circle, Square, Triangle]
Vector 2: [Circle, Square, Triangle]
```

22) **Compare two Vectors**

```
import java.util.*;
public class VectorCompare {
public static void main(String[] args) {
Vector<Integer> v1 = new Vector<>(Arrays.asList(10, 20, 30));
Vector<Integer> v2 = new Vector<>(Arrays.asList(10, 20, 30));
Vector<Integer> v3 = new Vector<>(Arrays.asList(10, 20, 40));
System.out.println("v1 equals v2: " + v1.equals(v2));
System.out.println("v1 equals v3: " + v1.equals(v3));
}
}
```
Output:

```
v1 equals v2: true
v1 equals v3: false
```

23) **Method to return sum**

```
import java.util.*;
public class VectorSum {
public static int sum(Vector<Integer> v) {
int total = 0;
for (int num : v) total += num;
return total;
}
public static void main(String[] args) {
Vector<Integer> v = new Vector<>(Arrays.asList(10, 20, 30));
System.out.println("The sum is: " + sum(v));
}
}
```
Output: The sum is: 60

24) **Push, Pop, Peek, and Check if Empty**

```
import java.util.*;
public class StackOperations {
public static void main(String[] args) {
Stack<Integer> stack = new Stack<>();
stack.push(1);
stack.push(2);
stack.push(3);
stack.push(4);
stack.push(5);
stack.pop();
System.out.println("Top element after pop: " + stack.peek());
System.out.println("Is stack empty? " + stack.isEmpty());
}
}
```
Output:

```
Top element after pop: 4
Is stack empty? False
```

25) **Reverse a String**

```java
import java.util.*;
public class ReverseStringStack {
public static void main(String[] args) {
Scanner sc = new Scanner(System.in);
System.out.print("Enter a string: ");
String str = sc.nextLine();
Stack<Character> stack = new Stack<>();
for (char c : str.toCharArray()) stack.push(c);
String rev = "";
while (!stack.isEmpty()) rev += stack.pop();
System.out.println("Reversed string: " + rev);
}
}
```

```
Input: Hello
Reversed String: OlleH
```

26) **Check for Balanced Parentheses**

```java
import java.util.*;
public class BalancedParentheses {
public static void main(String[] args) {
String expr = "((a+b) * (c-d))";
Stack<Character> stack = new Stack<>();
boolean valid = true;
for (char c : expr.toCharArray()) {
if (c == '(') stack.push(c);
else if (c == ')') {
if (stack.isEmpty()) {
valid = false;
break;
}
stack.pop();
}
}
if (!stack.isEmpty()) valid = false;
System.out.println(valid ? "Parentheses are balanced." : "Parentheses are not balanced.");
}
}
```

```
Output: Parentheses are balanced.
```

27) **Decimal to Binary**

```java
import java.util.*;
public class DecimalToBinary {
public static void main(String[] args) {
int num = 25;
Stack<Integer> stack = new Stack<>();
while (num > 0) {
stack.push(num % 2);
num /= 2;
}
System.out.print("Binary representation: ");
while (!stack.isEmpty()) System.out.print(stack.pop());
System.out.println();
}
}
```

Output: Binary representation: 11001

28) **Create HashSet of Strings**

```java
import java.util.*;
public class HashSetCities {
public static void main(String[] args) {
HashSet<String> cities = new HashSet<>();
cities.add("Hyderabad");
cities.add("Pune");
cities.add("Mumbai");
cities.add("Chennai");
cities.add("Mumbai");
System.out.println("Cities in the set:");
for (String city : cities) {
System.out.println(city);
}
}
}
```

Output: Cities in the set:, Pune, Mumbai, Hyderabad, Chennai

29) **Remove, Check, and Clear**

```java
import java.util.*;
public class HashSetOperations {
public static void main(String[] args) {
HashSet<String> set = new HashSet<>(Arrays.asList("Java", "Python", "C++"));
set.remove("Python");
System.out.println("Does 'Java' exist? " + set.contains("Java"));
set.clear();
System.out.println("Set after clearing: " + set);
}
}
```

Output: Does 'Java' exist? true and Set after clearing: []

30) **Method to Return Max Element**

```java
import java.util.*;
public class HashSetMax {
public static int getMax(HashSet<Integer> set) {
return Collections.max(set);
}
public static void main(String[] args) {
HashSet<Integer> nums = new HashSet<>(Arrays.asList(10, 5, 30, 25));
System.out.println("Maximum number in the set: " + getMax(nums));
}
}
```

Output: Maximum number in the set: 30

31) **Custom Objects with equals() and hashCode()**

```java
import java.util.*;
class Student {
```

```
int id;
String name;
Student(int id, String name) {
this.id = id; this.name = name;
}
public boolean equals(Object o) {
if (this == o) return true;
if (!(o instanceof Student)) return false;
Student s = (Student) o;
return id == s.id && name.equals(s.name);
}
public int hashCode() {
return Objects.hash(id, name);
}
}
public class LinkedHashSetStudents {
public static void main(String[] args) {
LinkedHashSet<Student> set = new LinkedHashSet<>();
set.add(new Student(10, "Rahul"));
set.add(new Student(20, "Suman"));
set.add(new Student(30, "Ankit"));
set.add(new Student(10, "Rahul"));
System.out.println("Students in LinkedHashSet:");
for (Student s : set) {
System.out.println("ID: " + s.id + ", Name: " + s.name);
}
}
}
```

Output: Students in LinkedHashSet:, ID: 10, Name: Rahul, ID: 20, Name: Suman, ID: 30, Name: Ankit

32) **Merge two LinkedHashSets**

```
import java.util.*;
public class LinkedHashSetMerge {
public static void main(String[] args) {
LinkedHashSet<String> set1 = new LinkedHashSet<>(Arrays.asList("X", "Y", "Z"));
LinkedHashSet<String> set2 = new LinkedHashSet<>(Arrays.asList("W", "Z", "A"));
set1.addAll(set2);
System.out.println("Merged LinkedHashSet: " + set1);
}
}
```

Output: Merged LinkedHashSet: [X, Y, Z, W, A]

33) **Add countries in random order and print sorted**

```
import java.util.*;
public class TreeSetCountries {
public static void main(String[] args) {
TreeSet<String> countries = new TreeSet<>();
countries.add("Germany");
countries.add("France");
countries.add("Spain");
System.out.println("Sorted countries: " + countries);
}
}
```

Output: Sorted countries: [France, Germany, Spain]