

ASSIGNMENT

B. LIKHITHA

Assignment 1

Title: Store and Display Elements Using ArrayList

Problem Statement:

Create an ArrayList to store 5 student names and display all names.

Requirements:

- Use ArrayList
- Use add() and get() methods

Expected Outcome:

All student names should be printed one by one.

The screenshot shows a code editor interface with a tab labeled "Assignment1.java". The code is as follows:

```
Assignment1.java
1~ import java.util.*;
2
3~ class Assignment1 {
4~   public static void main(String[] args) {
5~     ArrayList<String> names = new ArrayList<>();
6
7~     names.add("Likhitha");
8~     names.add("Anusha");
9~     names.add("Ravi");
10~    names.add("Sita");
11~    names.add("Manish");
12
13~    for (int i = 0; i < names.size(); i++) {
14~      System.out.println(names.get(i));
15~    }
16~  }
17 }
```

On the right side, there is a "Run" button and an "Output" panel. The output shows the names being printed:

```
Likhitha
Anusha
Ravi
Sita
Manish
== Code Execution Successful ==
```

Assignment 2

Title: Add and Remove Elements from ArrayList

Problem Statement:

Create an ArrayList of integers. Add 5 numbers and remove one number.

Requirements:

- Use add()
- Use remove()

Expected Outcome:

Updated list should be displayed after removal.

```
Assignment2.java
1~ import java.util.*;
2
3~ class Assignment2 {
4~     public static void main(String[] args) {
5~         ArrayList<Integer> numbers = new ArrayList<>();
6
7~         numbers.add(10);
8~         numbers.add(20);
9~         numbers.add(30);
10~        numbers.add(40);
11~        numbers.add(50);
12
13~        numbers.remove(2); // removes 30
14
15~        System.out.println(numbers);
16    }
17 }
```

[10, 20, 40, 50]
== Code Execution Successful ==

Assignment 3

Title: Find Size of Collection

Problem Statement:

Create an ArrayList of city names and find how many cities are stored.

Requirements:

- Use size() method

Expected Outcome:

Total number of cities should be displayed.

```
Assignment3.java
1~ class Assignment3 {
2~     public static void main(String[] args) {
3~         ArrayList<String> cities = new ArrayList<>();
4
5~         cities.add("Hyderabad");
6~         cities.add("Delhi");
7~         cities.add("Mumbai");
8
9~         System.out.println("Total cities: " + cities.size());
10
11    }
12 }
```

Total cities: 3
== Code Execution Successful ==

Assignment 4

Title: Iterate Collection Using for-each Loop

Problem Statement:

Store 5 course names in an ArrayList and display them using a for-each loop.

Requirements:

- Use enhanced for loop

Expected Outcome:

All course names printed successfully.

```
Assignment4.java
1- import java.util.*;
2
3- class Assignment4 {
4-     public static void main(String[] args) {
5-         ArrayList<String> courses = new ArrayList<>();
6
7-         courses.add("Java");
8-         courses.add("Python");
9-         courses.add("C++");
10-        courses.add("HTML");
11-        courses.add("DBMS");
12
13-        for (String course : courses) {
14-            System.out.println(course);
15-        }
16    }
17 }
```

Output

Java
Python
C++
HTML
DBMS

==== Code Execution Successful ===

Assignment 5

Title: Check Element Exists in List

Problem Statement:

Create an ArrayList of fruits and check whether "Apple" exists.

Requirements:

- Use contains() method

Expected Outcome:

Program should print whether Apple is present or not.

```
Assignment5.java
1- import java.util.*;
2
3- class Assignment5 {
4-     public static void main(String[] args) {
5-         ArrayList<String> fruits = new ArrayList<>();
6
7-         fruits.add("Apple");
8-         fruits.add("Banana");
9-         fruits.add("Mango");
10
11-        if (fruits.contains("Apple"))
12-            System.out.println("Apple is present");
13-        else
14-            System.out.println("Apple is not present");
15    }
16 }
```

Output

Apple is present

==== Code Execution Successful ===

Assignment 6

Title: Store Unique Elements Using HashSet

Problem Statement:

Create a HashSet and add duplicate numbers.

Requirements:

- Use HashSet
- Add duplicate values

Expected Outcome:

Duplicates should not be stored.

```

Assignment6.java

1- import java.util.*;
2
3- class Assignment6 {
4-     public static void main(String[] args) {
5-         HashSet<Integer> set = new HashSet<>();
6
7-         set.add(10);
8-         set.add(20);
9-         set.add(10); // duplicate ignored
10
11        System.out.println(set);
12    }
13 }
14

```

Output: [20, 10]
==== Code Execution Successful ===

Assignment 7

Title: Display HashSet Elements Problem Statement:

Store 5 colors in a HashSet and display them.

Requirements:

- Use for-each loop

Expected Outcome:

All colors printed (order not important).

```

Assignment7.java

1- import java.util.*;
2
3- class Assignment7 {
4-     public static void main(String[] args) {
5-         HashSet<String> colors = new HashSet<>();
6
7-         colors.add("Red");
8-         colors.add("Blue");
9-         colors.add("Green");
10        colors.add("Yellow");
11        colors.add("Pink");
12
13-        for (String c : colors) {
14-            System.out.println(c);
15-        }
16    }
17 }
18

```

Output: Red
Pink
Blue
Yellow
Green
==== Code Execution Successful ===

Assignment 8

Title: Basic Key-Value Storage Using HashMap

Problem Statement:

Create a HashMap to store employee ID and employee name.

Requirements:

- Use put() and get()

Expected Outcome:

Employee details printed correctly.



```
Assignment8.java
1~ import java.util.*;
2
3~ class Assignment8 {
4~     public static void main(String[] args) {
5~         HashMap<Integer, String> emp = new HashMap<>();
6
7~         emp.put(101, "Ravi");
8~         emp.put(102, "Sita");
9~         emp.put(103, "Kiran");
10
11~        System.out.println(emp.get(101));
12~        System.out.println(emp.get(102));
13~        System.out.println(emp.get(103));
14~    }
15~ }
```

Ravi
Sita
Kiran
== Code Execution Successful ==

Assignment 9

Title: Display All Keys and Values from HashMap

Problem Statement:

Store 3 country codes and country names in a HashMap.

Requirements:

- Use keySet() or entrySet()

Expected Outcome:

All keys and values printed.



```
Assignment9.java
1~ import java.util.*;
2
3~ class Assignment9 {
4~     public static void main(String[] args) {
5~         HashMap<String, String> map = new HashMap<>();
6
7~         map.put("IN", "India");
8~         map.put("US", "United States");
9~         map.put("UK", "United Kingdom");
10
11~        for (Map.Entry<String, String> entry : map.entrySet()) {
12~            System.out.println(entry.getKey() + " → " + entry.getValue());
13~        }
14~    }
15~ }
```

IN ? India
UK ? United Kingdom
US ? United States
== Code Execution Successful ==

Assignment 10

Title: Remove Entry from HashMap

Problem Statement:

Remove one employee entry from a HashMap.

Requirements: • Use remove() method

Expected Outcome:

Remaining entries printed.

The screenshot shows a Java code editor interface. On the left, the code file 'Assignment10.java' is displayed with the following content:

```
1 import java.util.*;
2
3 class Assignment10 {
4     public static void main(String[] args) {
5         HashMap<Integer, String> emp = new HashMap<>();
6
7         emp.put(1, "Ravi");
8         emp.put(2, "Kiran");
9         emp.put(3, "Likhitha");
10
11         emp.remove(2);
12
13         System.out.println(emp);
14     }
15 }
16
```

On the right, the 'Output' panel shows the execution results:

```
{1=Ravi, 3=Likhitha}
== Code Execution Successful ==
```

Assignment 11

Title: Use LinkedList to Store Elements

Problem Statement:

Store 5 numbers using LinkedList and display them.

Requirements:

- Use add() method

Expected Outcome:

All elements printed.

```
Assignment11.java
1- import java.util.*;
2
3- class Assignment11 {
4-     public static void main(String[] args) {
5-         LinkedList<Integer> list = new LinkedList<>();
6
7-         list.add(5);
8-         list.add(15);
9-         list.add(25);
10-        list.add(35);
11-        list.add(45);
12
13-        for (int n : list) {
14-            System.out.println(n);
15-        }
16    }
17 }
```

Output

```
5
15
25
35
45
== Code Execution Successful ==
```

Assignment 12

Title: Check Collection Is Empty

Problem Statement:

Create an ArrayList and check whether it is empty.

Requirements:

- Use isEmpty()

Expected Outcome:

Program should print true or false.

```
Assignment12.java
1- import java.util.*;
2
3- class Assignment12 {
4-     public static void main(String[] args) {
5-         ArrayList<String> items = new ArrayList<>();
6
7-         System.out.println("Is list empty? " + items.isEmpty());
8
9-         items.add("A");
10
11        System.out.println("Is list empty? " + items.isEmpty());
12    }
13
14
15 }
```

Output

```
Is list empty? true
Is list empty? false
== Code Execution Successful ==
```

Assignment 13

Title: Clear All Elements from Collection

Problem Statement:

Add elements to an ArrayList and remove all elements.

Requirements:

- Use clear() method

Expected Outcome:

List should be empty.

```
Assignment13.java
1- import java.util.*;
2
3- class Assignment13 {
4-     public static void main(String[] args) {
5-         ArrayList<String> data = new ArrayList<>();
6
7-         data.add("One");
8-         data.add("Two");
9-         data.add("Three");
10
11        data.clear();
12
13        System.out.println(data); // empty list
14    }
15 }
16
```

Output: []
==== Code Execution Successful ===

Assignment 14

Title: Store Different Data Types Using Wrapper Classes

Problem Statement:

Store integer values using wrapper class in an ArrayList.

Requirements:

- Use Integer wrapper class

Expected Outcome:

Values stored and printed successfully.

```
Assignment14.java
1- import java.util.*;
2
3- class Assignment14 {
4-     public static void main(String[] args) {
5-         ArrayList<Integer> nums = new ArrayList<>();
6
7-         nums.add(10);
8-         nums.add(20);
9-         nums.add(30);
10
11        for (Integer n : nums) {
12            System.out.println(n);
13        }
14    }
15 }
16
```

Output: 10
20
30
==== Code Execution Successful ===

Assignment 15

Title: Convert Collection to Array

Problem Statement:

Convert an ArrayList of strings to an array.

Requirements:

- Use `toArray()`

Expected Outcome:

Array elements printed.

The screenshot shows a Java code editor interface. On the left, the code file `Assignment15.java` is displayed with line numbers from 1 to 19. The code creates an `ArrayList` named `list` containing three elements: "Red", "Blue", and "Green". It then uses the `toArray` method to convert this list into an array `arr`, and finally prints each element of the array using a `for` loop and `System.out.println`. On the right, the `Output` panel shows the printed results: "Red", "Blue", and "Green", followed by the message "Code Execution Successful".

```
Assignment15.java
1- import java.util.*;
2
3- class Assignment15 {
4-     public static void main(String[] args) {
5-         ArrayList<String> list = new ArrayList<>();
6
7-         list.add("Red");
8-         list.add("Blue");
9-         list.add("Green");
10
11        String arr[] = list.toArray(new String[0]);
12
13        for (String s : arr) {
14            System.out.println(s);
15        }
16    }
17 }
18
19
```

Output

Red
Blue
Green
==== Code Execution Successful ===