***LAB-02***

***DATA STRUCTURES & ALGORITHM***

**TASKS:**

1. package lab2;

import java.util.Vector;

public class VectorExample {

public static void main(String[] args) {

Vector<Integer> numbers = new Vector<>(10);

// Initializing the Vector with 10 integers

for (int i = 1; i <= 10; i++) {

numbers.add(i);

}

// Displaying all integers

System.out.println("Integers in the Vector: " + numbers);

// Calculating the sum of integers

int sum = 0;

for (int number : numbers) {

sum += number;

}

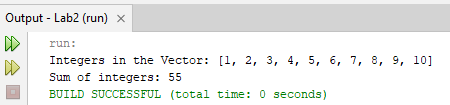
// Displaying the sum

System.out.println("Sum of integers: " + sum);

}

}

**Output:**



2. package lab2;

import java.util.ArrayList;

import java.util.Collections;

import java.util.Scanner;

public class StringArrayListMenu {

public static void main(String[] args) {

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

Scanner scanner = new Scanner(System.in);

int choice;

do {

// Display the menu

System.out.println("\n--- String List Menu ---");

System.out.println("1. Add a String");

System.out.println("2. Display All Strings");

System.out.println("3. Display the Largest String");

System.out.println("4. Exit");

System.out.print("Please enter your choice: ");

// Validate user input for choice

while (!scanner.hasNextInt()) {

System.out.print("That's not a valid number. Please enter your choice: ");

scanner.next(); // Clear invalid input

}

choice = scanner.nextInt();

scanner.nextLine(); // Consume the newline character

switch (choice) {

case 1:

System.out.print("Enter the string you'd like to add: ");

String input = scanner.nextLine();

stringList.add(input);

System.out.println("String added!");

break;

case 2:

if (stringList.isEmpty()) {

System.out.println("The list is currently empty.");

} else {

System.out.println("All Strings: " + stringList);

}

break;

case 3:

if (!stringList.isEmpty()) {

// Find the longest string using a custom comparator

String largestString = Collections.max(stringList, (s1, s2) -> s1.length() - s2.length());

System.out.println("The largest String is: " + largestString);

} else {

System.out.println("The list is empty. Please add some strings first.");

}

break;

case 4:

System.out.println("Thank you for using the String List Menu. Goodbye!");

break;

default:

System.out.println("Oops! That's not a valid choice. Please try again.");

}

}

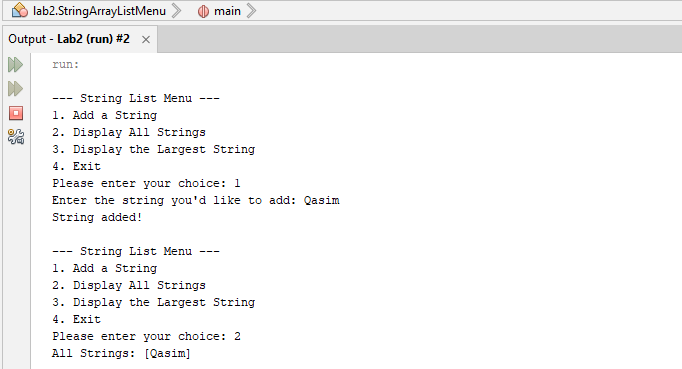
while (choice != 4);

scanner.close();

}

}

**Output:**



3. package lab2;

import java.util.ArrayList;

import java.util.Collections;

import java.util.Comparator;

class Employee {

private int empId;

private String empName;

private String empGender;

private int yearOfJoining;

public Employee(int empId, String empName, String empGender, int yearOfJoining) {

this.empId = empId;

this.empName = empName;

this.empGender = empGender;

this.yearOfJoining = yearOfJoining;

}

public int getYearOfJoining() {

return yearOfJoining;

}

@Override

public String toString() {

return "Employee{" +

"empId=" + empId +

", empName='" + empName + '\'' +

", empGender='" + empGender + '\'' +

", yearOfJoining=" + yearOfJoining +

'}';

}

}

class YearOfJoiningComparator implements Comparator<Employee> {

@Override

public int compare(Employee e1, Employee e2) {

return Integer.compare(e1.getYearOfJoining(), e2.getYearOfJoining());

}

}

public class EmployeeManagement {

public static void main(String[] args) {

ArrayList<Employee> employees = new ArrayList<>();

employees.add(new Employee(1, "Alice", "Female", 2018));

employees.add(new Employee(2, "Bob", "Male", 2020));

employees.add(new Employee(3, "Charlie", "Male", 2019));

employees.add(new Employee(4, "Diana", "Female", 2021));

Collections.sort(employees, new YearOfJoiningComparator());

for (Employee employee : employees) {

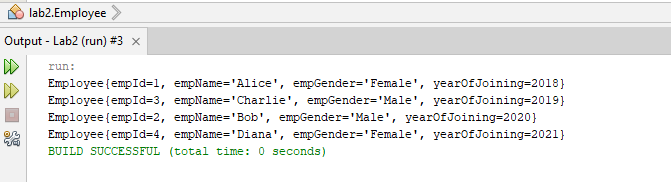
System.out.println(employee);

}

}

}

**Output:**



4. package lab2;

import java.util.Vector;

public class Vector {

public static void main(String[] args) {

Vector<Integer> numbers = new Vector<>(10);

// Initializing the Vector with 10 integers

for (int i = 1; i <= 10; i++) {

numbers.add(i);

}

// Displaying all the integers

System.out.println("Integers in the Vector: " + numbers);

// Calculating the sum of the integers

int sum = 0;

for (int number : numbers) {

sum += number;

}

System.out.println("Sum of integers: " + sum);

// Finding the maximum element in the Vector

int max = numbers.get(0);

for (int number : numbers) {

if (number > max) {

max = number;

}

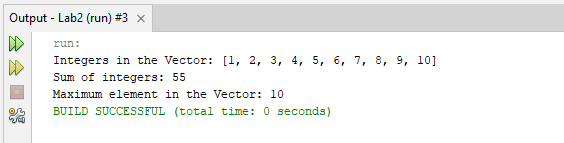
}

System.out.println("Maximum element in the Vector: " + max);

}

}

**Output:**



5. package lab2;

import java.util.ArrayList;

//Find the k-th Smallest Element in a Sorted ArrayList

public class KthSmallestElement {

public static void main(String[] args) {

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

sortedList.add(1);

sortedList.add(3);

sortedList.add(5);

sortedList.add(7);

sortedList.add(9);

int k = 3; // Example value for k

int kthElement = findKthSmallest(sortedList, k);

System.out.println("The " + k + "-th smallest element is: " + kthElement);

}

public static int findKthSmallest(ArrayList<Integer> list, int k) {

if (k <= 0 || k > list.size()) {

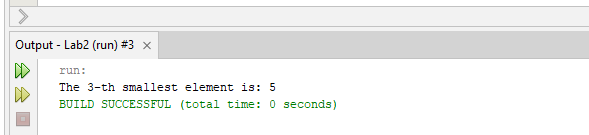
throw new IllegalArgumentException("k is out of bounds");

}

return list.get(k - 1);

}

}

**Output:**

6. package lab2;

import java.util.ArrayList;

public class MergeArraysList {

public static void main(String[] args) {

ArrayList<String> list1 = new ArrayList<>();

list1.add("Apple");

list1.add("Banana");

ArrayList<String> list2 = new ArrayList<>();

list2.add("Orange");

list2.add("Grapes");

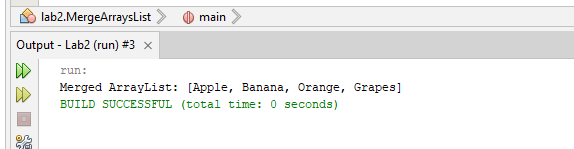
ArrayList<String> mergedList = new ArrayList<>(list1);

mergedList.addAll(list2);

System.out.println("Merged ArrayList: " + mergedList);

}

}

**Output:**

**HOME-TASKS:**

1. package lab2;

import java.util.Collections;

import java.util.Scanner;

import java.util.Vector;

public class Vectors {

public static void main(String[] args) {

// Create a Vector to store integer objects

Vector<Integer> numbers = new Vector<>();

Scanner scanner = new Scanner(System.in);

System.out.println("Enter integers (type 'done' to finish):");

// Read integers from the user

while (scanner.hasNextInt()) {

numbers.add(scanner.nextInt());

}

// Clear the scanner buffer

scanner.nextLine();

// Sort the vector

Collections.sort(numbers);

// Display the largest and smallest numbers

if (!numbers.isEmpty()) {

System.out.println("Sorted numbers: " + numbers);

System.out.println("Largest number: " + numbers.lastElement());

System.out.println("Smallest number: " + numbers.firstElement());

} else {

System.out.println("No numbers were entered.");

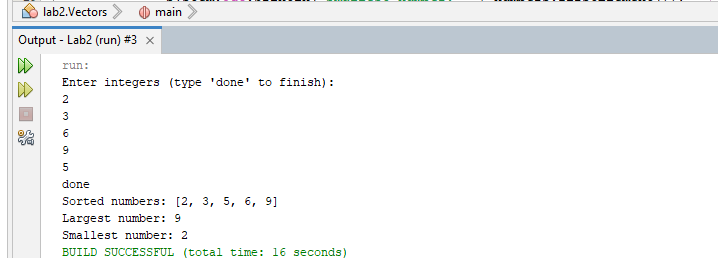
}

scanner.close();

}

}

**Output:**



2. package lab2;

import java.util.Scanner;

public class HashCodeGenerator {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter a string: ");

String userInput = scanner.nextLine();

int hashCodeValue = userInput.hashCode();

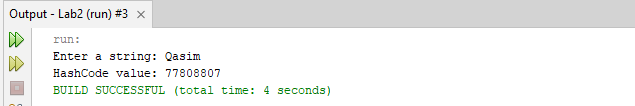
System.out.println("HashCode value: " + hashCodeValue);

scanner.close();

}

}

**Output:**



3. package lab2;

import java.util.HashSet;

import java.util.Set;

//Employee Management System

class Employee {

private String name;

private int id;

public Employee(String name, int id) {

this.name = name;

this.id = id;

}

public String getName() {

return name;

}

public int getId() {

return id;

}

@Override

public int hashCode() {

return name.hashCode() + id;

}

@Override

public boolean equals(Object obj) {

if (this == obj) return true;

if (!(obj instanceof Employee)) return false;

Employee other = (Employee) obj;

return this.name.equals(other.name) && this.id == other.id;

}

@Override

public String toString() {

return "Employee{" + "name='" + name + '\'' + ", id=" + id + '}';

}

}

class EmployeeManagement {

private Set<Employee> employees;

public EmployeeManagement() {

employees = new HashSet<>();

}

public boolean addEmployee(String name, int id) {

Employee employee = new Employee(name, id);

return employees.add(employee);

}

public boolean employeeExists(String name, int id) {

Employee employee = new Employee(name, id);

return employees.contains(employee);

}

public void displayEmployees() {

for (Employee employee : employees) {

System.out.println(employee);

}

}

public static void main(String[] args) {

EmployeeManagement management = new EmployeeManagement();

management.addEmployee("John Doe", 1);

management.addEmployee("Jane Smith", 2);

management.addEmployee("John Doe", 1); // Duplicate, should not be added

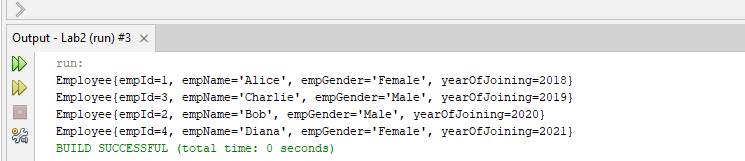
System.out.println("Employee exists: " + management.employeeExists("Jane Smith", 2));

System.out.println("Employee exists: " + management.employeeExists("John Doe", 3));

management.displayEmployees();

}

}

**Output:**

4. package lab2;

class Color {

private int red;

private int green;

private int blue;

// Constructor

public Color(int red, int green, int blue) {

this.red = red;

this.green = green;

this.blue = blue;

}

// Getters

public int getRed() {

return red;

}

public int getGreen() {

return green;

}

public int getBlue() {

return blue;

}

// Override equals method

@Override

public boolean equals(Object obj) {

if (this == obj) return true;

if (obj == null || getClass() != obj.getClass()) return false;

Color color = (Color) obj;

return red == color.red && green == color.green && blue == color.blue;

}

// Override hashCode method

@Override

public int hashCode() {

int result = red;

result = 31 \* result + green;

result = 31 \* result + blue;

return result;

}

// Override toString method for easier display

@Override

public String toString() {

return "Color{" +

"red=" + red +

", green=" + green +

", blue=" + blue +

'}';

}

// Main method to demonstrate the Color class

public static void main(String[] args) {

Color color1 = new Color(255, 0, 0); // Red

Color color2 = new Color(255, 0, 0); // Red

Color color3 = new Color(0, 255, 0); // Green

// Display colors

System.out.println("Color 1: " + color1);

System.out.println("Color 2: " + color2);

System.out.println("Color 3: " + color3);

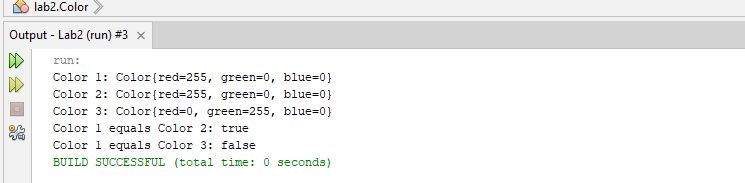
// Check equality

System.out.println("Color 1 equals Color 2: " + color1.equals(color2)); // true

System.out.println("Color 1 equals Color 3: " + color1.equals(color3)); // false

}

}

**Output:**