⇒ Mandatory hands-on

**Exercise 2: E-commerce Platform Search Function**

**Code:**

**package week\_1.Algorithms\_DS;**

**import java.util.Arrays;**

**class ProductSearch {**

**int productId;**

**String productName;**

**String category;**

**public ProductSearch(int id, String name, String category) {**

**this.productId = id;**

**this.productName = name;**

**this.category = category;**

**}**

**}**

**public class SearchDemo {**

**public static void linearSearch(ProductSearch[] products, String keyword) {**

**for (ProductSearch p : products) {**

**if (p.productName.toLowerCase().contains(keyword.toLowerCase())) {**

**System.*out*.println("Found: " + p.productName);**

**}**

**}**

**}**

**public static int binarySearch(String[] productNames, String target) {**

**int left = 0, right = productNames.length - 1;**

**while (left <= right) {**

**int mid = left + (right - left) / 2;**

**int res = target.compareToIgnoreCase(productNames[mid]);**

**if (res == 0)**

**return mid;**

**if (res > 0)**

**left = mid + 1;**

**else**

**right = mid - 1;**

**}**

**return -1;**

**}**

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

**ProductSearch[] products = {**

**new ProductSearch(1, "Laptop", "Electronics"),**

**new ProductSearch(2, "Smartphone", "Electronics"),**

**new ProductSearch(3, "Book", "Stationery")**

**};**

**System.*out*.println("Linear Search:");**

***linearSearch*(products, "phone");**

**String[] names = {"Book", "Laptop", "Smartphone"};**

**Arrays.*sort*(names);**

**System.*out*.println("Binary Search:");**

**int index = *binarySearch*(names, "Laptop");**

**if (index >= 0)**

**System.*out*.println("Found: " + names[index]);**

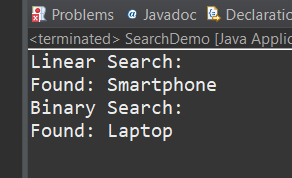
**else**

**System.*out*.println("Not found.");**

**}**

**}**

**Output:**

****

**Exercise 7: Financial Forecasting**

**Code:**

**package week\_1.Algorithms\_DS;**

**public class FinancialForecast {**

**public static double futureValue(double presentValue, double rate, int years) {**

**if (years == 0)**

**return presentValue;**

**return *futureValue*(presentValue \* (1 + rate), rate, years - 1);**

**}**

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

**double presentValue = 10000;**

**double growthRate = 0.05;**

**int years = 5;**

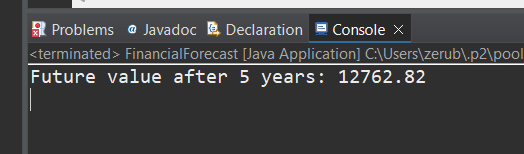
**double result = *futureValue*(presentValue, growthRate, years);**

**System.*out*.printf("Future value after %d years: %.2f\n", years, result);**

**}**

**}**

**Output:**

****

⇒ Additional Important hands-on

**Exercise 1: Inventory Management System**

**Code:**

**package week\_1.Algorithms\_DS;**

**import java.util.HashMap;**

**class Product {**

**int productId;**

**String productName;**

**int quantity;**

**double price;**

**public Product(int productId, String productName, int quantity, double price) {**

**this.productId = productId;**

**this.productName = productName;**

**this.quantity = quantity;**

**this.price = price;**

**}**

**public String toString() {**

**return "[" + productId + ", " + productName + ", Qty: " + quantity + ", Price: " + price + "]";**

**}**

**}**

**public class InventoryManagement {**

**HashMap<Integer, Product> inventory = new HashMap<>();**

**public void addProduct(Product p) {**

**inventory.put(p.productId, p);**

**}**

**public void updateProduct(int productId, int newQuantity, double newPrice) {**

**Product p = inventory.get(productId);**

**if (p != null) {**

**p.quantity = newQuantity;**

**p.price = newPrice;**

**}**

**}**

**public void deleteProduct(int productId) {**

**inventory.remove(productId);**

**}**

**public void displayInventory() {**

**for (Product p : inventory.values()) {**

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

**}**

**}**

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

**InventoryManagement system = new InventoryManagement();**

**Product p1 = new Product(101, "Laptop", 10, 50000);**

**Product p2 = new Product(102, "Mouse", 50, 500);**

**system.addProduct(p1);**

**system.addProduct(p2);**

**system.displayInventory();**

**system.updateProduct(101, 8, 48000);**

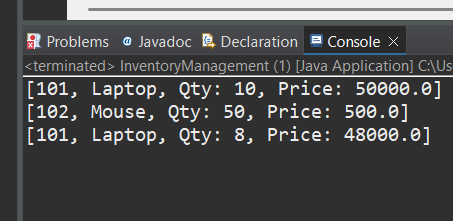
**system.deleteProduct(102);**

**system.displayInventory();**

**}**

**}**

**Output:**



**Exercise 3: Sorting Customer Order**

**Code:**

**package week\_1.Algorithms\_DS;**

**class Order {**

**int orderId;**

**String customerName;**

**double totalPrice;**

**public Order(int id, String name, double price) {**

**this.orderId = id;**

**this.customerName = name;**

**this.totalPrice = price;**

**}**

**public String toString() {**

**return "[" + orderId + ", " + customerName + ", Total: " + totalPrice + "]";**

**}**

**}**

**public class OrderSorting {**

**public static void bubbleSort(Order[] orders) {**

**int n = orders.length;**

**for (int i = 0; i < n - 1; i++) {**

**for (int j = 0; j < n - i - 1; j++) {**

**if (orders[j].totalPrice > orders[j + 1].totalPrice) {**

**Order temp = orders[j];**

**orders[j] = orders[j + 1];**

**orders[j + 1] = temp;**

**}**

**}**

**}**

**}**

**public static void quickSort(Order[] orders, int low, int high) {**

**if (low < high) {**

**int pi = *partition*(orders, low, high);**

***quickSort*(orders, low, pi - 1);**

***quickSort*(orders, pi + 1, high);**

**}**

**}**

**private static int partition(Order[] orders, int low, int high) {**

**double pivot = orders[high].totalPrice;**

**int i = low - 1;**

**for (int j = low; j < high; j++) {**

**if (orders[j].totalPrice < pivot) {**

**i++;**

**Order temp = orders[i];**

**orders[i] = orders[j];**

**orders[j] = temp;**

**}**

**}**

**Order temp = orders[i + 1];**

**orders[i + 1] = orders[high];**

**orders[high] = temp;**

**return i + 1;**

**}**

**public static void printOrders(Order[] orders) {**

**for (Order o : orders) {**

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

**}**

**}**

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

**Order[] orders = {**

**new Order(1, "Alice", 2500),**

**new Order(2, "Bob", 1500),**

**new Order(3, "Charlie", 4000)**

**};**

**System.*out*.println("Bubble Sort:");**

***bubbleSort*(orders);**

***printOrders*(orders);**

**System.*out*.println("Quick Sort:");**

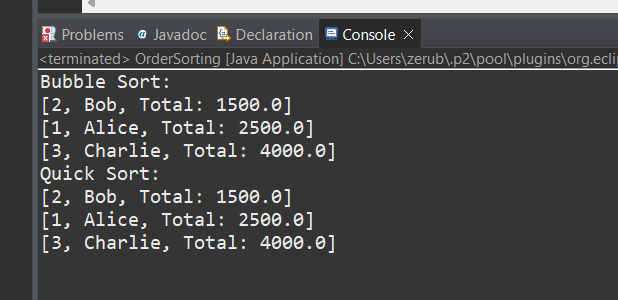
***quickSort*(orders, 0, orders.length - 1);**

***printOrders*(orders);**

**}**

**}**

**Output:**

****

**Exercise 4: Employee Management System**

**Code:**

**package week\_1.Algorithms\_DS;**

**class Employee {**

**int employeeId;**

**String name;**

**String position;**

**double salary;**

**public Employee(int id, String name, String pos, double sal) {**

**this.employeeId = id;**

**this.name = name;**

**this.position = pos;**

**this.salary = sal;**

**}**

**public String toString() {**

**return "[" + employeeId + ", " + name + ", " + position + ", Salary: " + salary + "]";**

**}**

**}**

**public class EmployeeManagement {**

**Employee[] employees = new Employee[10];**

**int count = 0;**

**public void addEmployee(Employee e) {**

**if (count < employees.length) {**

**employees[count++] = e;**

**}**

**}**

**public void searchEmployee(int id) {**

**for (int i = 0; i < count; i++) {**

**if (employees[i].employeeId == id) {**

**System.*out*.println(employees[i]);**

**return;**

**}**

**}**

**System.*out*.println("Employee not found.");**

**}**

**public void deleteEmployee(int id) {**

**for (int i = 0; i < count; i++) {**

**if (employees[i].employeeId == id) {**

**for (int j = i; j < count - 1; j++) {**

**employees[j] = employees[j + 1];**

**}**

**count--;**

**return;**

**}**

**}**

**}**

**public void displayAll() {**

**for (int i = 0; i < count; i++) {**

**System.*out*.println(employees[i]);**

**}**

**}**

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

**EmployeeManagement ems = new EmployeeManagement();**

**ems.addEmployee(new Employee(1, "John", "Manager", 60000));**

**ems.addEmployee(new Employee(2, "Jane", "Developer", 50000));**

**ems.displayAll();**

**ems.searchEmployee(1);**

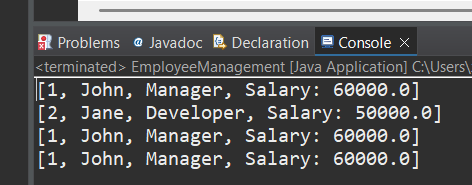
**ems.deleteEmployee(2);**

**ems.displayAll();**

**}**

}

**Output:**

****

**Exercise 5: Task Management System**

**Code:**

**package week\_1.Algorithms\_DS;**

**class Task {**

**int taskId;**

**String taskName;**

**String status;**

**Task next;**

**public Task(int id, String name, String status) {**

**this.taskId = id;**

**this.taskName = name;**

**this.status = status;**

**this.next = null;**

**}**

**public String toString() {**

**return "[" + taskId + ", " + taskName + ", " + status + "]";**

**}**

**}**

**public class TaskManagement {**

**Task head = null;**

**public void addTask(Task newTask) {**

**if (head == null) {**

**head = newTask;**

**} else {**

**Task current = head;**

**while (current.next != null) {**

**current = current.next;**

**}**

**current.next = newTask;**

**}**

**}**

**public void searchTask(int id) {**

**Task current = head;**

**while (current != null) {**

**if (current.taskId == id) {**

**System.*out*.println("Found: " + current);**

**return;**

**}**

**current = current.next;**

**}**

**System.*out*.println("Task not found.");**

**}**

**public void deleteTask(int id) {**

**Task current = head;**

**Task prev = null;**

**while (current != null) {**

**if (current.taskId == id) {**

**if (prev == null) {**

**head = current.next;**

**} else {**

**prev.next = current.next;**

**}**

**return;**

**}**

**prev = current;**

**current = current.next;**

**}**

**}**

**public void displayTasks() {**

**Task current = head;**

**while (current != null) {**

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

**current = current.next;**

**}**

**}**

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

**TaskManagement tm = new TaskManagement();**

**tm.addTask(new Task(1, "Design", "Pending"));**

**tm.addTask(new Task(2, "Coding", "In Progress"));**

**tm.addTask(new Task(3, "Testing", "Pending"));**

**System.*out*.println("All Tasks:");**

**tm.displayTasks();**

**tm.searchTask(2);**

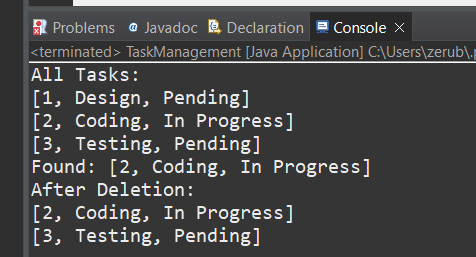
**tm.deleteTask(1);**

**System.*out*.println("After Deletion:");**

**tm.displayTasks();**

**} }**

**Output:**

****

**Exercise 6: Library Management System**

**Code:**

**package week\_1.Algorithms\_DS;**

**import java.util.Arrays;**

**class Book {**

**int bookId;**

**String title;**

**String author;**

**public Book(int id, String title, String author) {**

**this.bookId = id;**

**this.title = title;**

**this.author = author;**

**}**

**}**

**public class LibrarySearch {**

**public static void linearSearch(Book[] books, String title) {**

**for (Book b : books) {**

**if (b.title.equalsIgnoreCase(title)) {**

**System.*out*.println("Found: " + b.title + " by " + b.author);**

**return;**

**}**

**}**

**System.*out*.println("Book not found.");**

**}**

**public static int binarySearch(String[] titles, String target) {**

**int left = 0, right = titles.length - 1;**

**while (left <= right) {**

**int mid = left + (right - left) / 2;**

**int res = target.compareToIgnoreCase(titles[mid]);**

**if (res == 0)**

**return mid;**

**if (res > 0)**

**left = mid + 1;**

**else**

**right = mid - 1;**

**}**

**return -1;**

**}**

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

**Book[] books = {**

**new Book(1, "Java Programming", "Author A"),**

**new Book(2, "Data Structures", "Author B"),**

**new Book(3, "Algorithms", "Author C")**

**};**

**System.*out*.println("Linear Search:");**

***linearSearch*(books, "Algorithms");**

**String[] titles = {"Algorithms", "Data Structures", "Java Programming"};**

**Arrays.*sort*(titles);**

**System.*out*.println("Binary Search:");**

**int index = *binarySearch*(titles, "Data Structures");**

**if (index >= 0)**

**System.*out*.println("Found: " + titles[index]);**

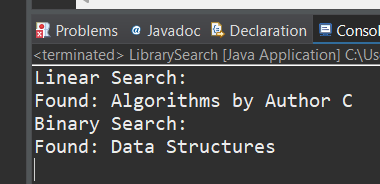
**else**

**System.*out*.println("Not found.");**

**}**

**}**

**Output:**

****