1. **Create a class called Employee that includes three pieces of information as instance variables: first name, last name, and monthly salary. Your class should have a constructor that initializes the three instance variables. Provide a setter and getter method for each instance variable. If the monthly salary is not positive, set it to 0.0. Write a test application named EmployeeTest that demonstrates the Employee class's capabilities. Create two Employee objects and display each object's yearly salary. Then give each Employee a 10% raise and display each Employee's yearly salary again.**

**Answer:**

class Employee {

private String firstName;

private String lastName;

private double monthlySalary;

public Employee(String firstName, String lastName, double monthlySalary)

{

this.firstName = firstName;

this.lastName = lastName;

if (monthlySalary > 0) {

this.monthlySalary = monthlySalary;

}

else {

this.monthlySalary = 0.0;

}

}

public void setData(String firstName, String lastName, double monthlySalary) {

this.firstName = firstName;

this.lastName = lastName;

if (monthlySalary > 0) {

this.monthlySalary = monthlySalary;

} else {

this.monthlySalary = 0.0;

}

}

public String getData() {

return "Name: " + firstName + " " + lastName + "\nMonthly Salary: " + monthlySalary;

}

public double getYearlySalary() {

return monthlySalary \* 12;

}

public void giveRaise() {

this.monthlySalary += this.monthlySalary \* 0.10;

}

public double getMonthlySalary() {

return monthlySalary;

}

}

public class EmployeeTest {

public static void main(String[] args) {

Employee emp1 = new Employee("Pragati", "Shendage", 3000.00);

Employee emp2 = new Employee("Prajyot", "Shendage", 4000.00);

System.out.println("Employee 1 Details:");

System.out.println(emp1.getData());

System.out.println("Yearly Salary: " + emp1.getYearlySalary());

System.out.println("\nEmployee 2 Details:");

System.out.println(emp2.getData());

System.out.println("Yearly Salary: " + emp2.getYearlySalary());

emp1.giveRaise();

emp2.giveRaise();

System.out.println("\nAfter 10% raise:");

System.out.println("Employee 1 Details:");

System.out.println(emp1.getData());

System.out.println("Yearly Salary: " + emp1.getYearlySalary());

System.out.println("\nEmployee 2 Details:");

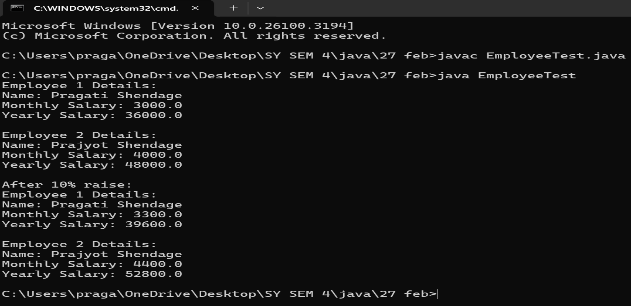
System.out.println(emp2.getData());

System.out.println("Yearly Salary: " + emp2.getYearlySalary());

}

}

**Output:**



1. **Implement a Java program to print the area of a rectangle by creating a class named 'Area' that has two methods. The first method, named 'setDim', takes the length and breadth of the rectangle as parameters. The second method, named 'getArea', returns the area of the rectangle. The length and breadth of the rectangle are entered through the keyboard.**

**Answer:**

import java.util.Scanner;

class Area {

double length;

double breadth;

public void setDim(double length, double breadth) {

this.length = length;

this.breadth = breadth;

}

public double getArea() {

return length \* breadth;

}

}

public class RectangleArea {

public static void main(String[] args)

{

Area rectangle = new Area();

Scanner sc = new Scanner(System.in);

System.out.print("Enter the length of the rectangle: ");

double length = sc.nextDouble();

System.out.print("Enter the breadth of the rectangle: ");

double breadth = sc.nextDouble();

rectangle.setDim(length, breadth);

double area = rectangle.getArea();

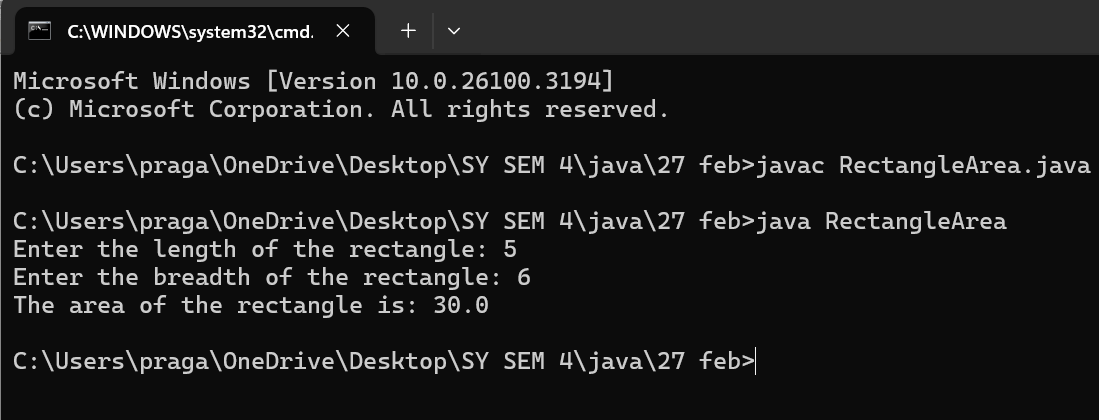
System.out.println("The area of the rectangle is: " + area);

sc.close();

}

}

**Output:**

****

1. **Write a Java program to demonstrate the use of static variables, static blocks, and static methods.**

**Answer:**

class StaticExample {

static int count;

static {

count = 10;

System.out.println("Static block executed. Count initialized to " + count);

}

static void displayCount() {

System.out.println("Static method called. Count: " + count);

}

public static void main(String[] args) {

displayCount();

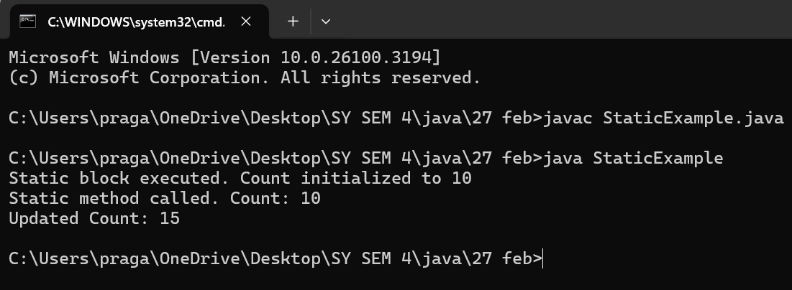
count += 5;

System.out.println("Updated Count: " +count);

}

}

**Output:**

****

1. **Write a Java program to arrange 10 names in alphabetical order.**

**Answer:**

import java.util.Arrays;

import java.util.Scanner;

class AlphabeticalOrder {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

String[] names = new String[10];

System.out.println("Enter 10 names:");

for (int i = 0; i < 10; i++) {

names[i] = scanner.nextLine();

}

Arrays.sort(names);

System.out.println("\nNames in alphabetical order:");

for (String name : names) {

System.out.println(name);

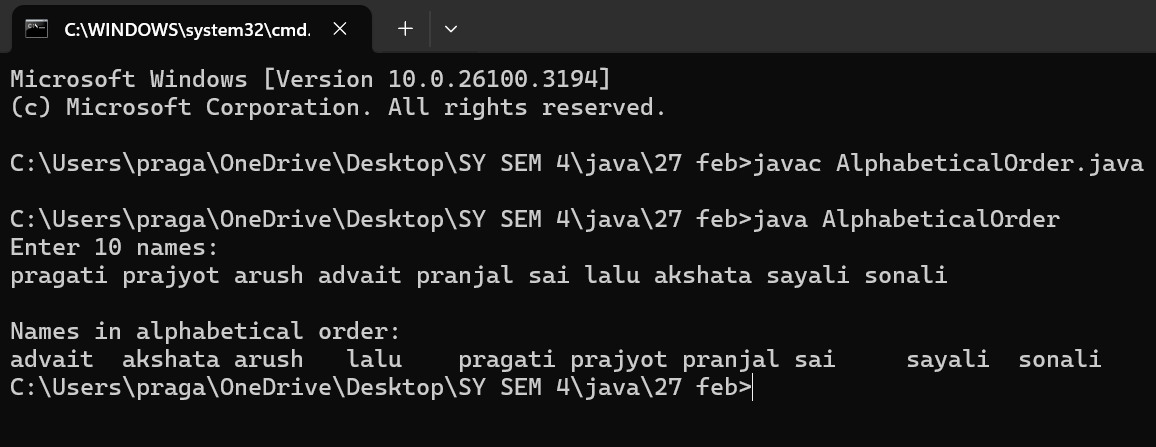
}

scanner.close();

}

}

**Output:**

****

**5. Write a Java program to implement a stack and a queue.**

**Answer: Stack:**

import java.util.Scanner;

class Stack {

private int arr[];

private int top;

private int capacity;

public Stack(int size) {

arr = new int[size];

capacity = size;

top = -1;

}

public void push(int item) {

if (top == capacity - 1) {

System.out.println("Stack Overflow! Cannot push " + item);

return;

}

arr[++top] = item;

System.out.println(item + " pushed into stack.");

}

public int pop() {

if (top == -1) {

System.out.println("Stack Underflow! Cannot pop");

return -1;

}

return arr[top--];

}

public void display() {

if (top == -1) {

System.out.println("Stack is empty!");

return;

}

System.out.print("Stack elements: ");

for (int i = 0; i <= top; i++) {

System.out.print(arr[i] + " ");

}

System.out.println();

}

}

public class StackImplementation {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter stack size: ");

int size = scanner.nextInt();

Stack stack = new Stack(size);

while (true) {

System.out.println("\n1. Push\n2. Pop\n3. Display\n4. Exit");

System.out.print("Enter your choice: ");

int choice = scanner.nextInt();

switch (choice) {

case 1:

System.out.print("Enter value to push: ");

int value = scanner.nextInt();

stack.push(value);

break;

case 2:

System.out.println("Popped: " + stack.pop());

break;

case 3:

stack.display();

break;

case 4:

System.out.println("Exiting...");

scanner.close();

return;

default:

System.out.println("Invalid choice, try again.");

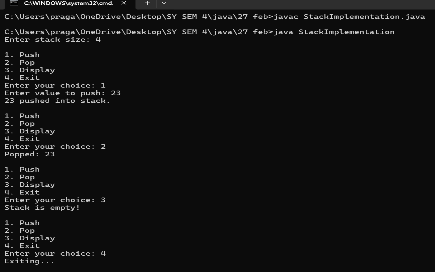
}

}

}

}

**Output:**

****

**Queue:**

**Answer**:

import java.util.Scanner;

class Queue {

private int arr[];

private int front, rear, capacity;

public Queue(int size) {

arr = new int[size];

capacity = size;

front = 0;

rear = -1;

}

public void enqueue(int item) {

if (rear == capacity - 1) {

System.out.println("Queue Overflow! Cannot enqueue " + item);

return;

}

arr[++rear] = item;

System.out.println(item + " enqueued into queue.");

}

public int dequeue() {

if (front > rear) {

System.out.println("Queue Underflow! Cannot dequeue");

return -1;

}

return arr[front++];

}

public void display() {

if (front > rear) {

System.out.println("Queue is empty!");

return;

}

System.out.print("Queue elements: ");

for (int i = front; i <= rear; i++) {

System.out.print(arr[i] + " ");

}

System.out.println();

}

}

public class QueueImplementation {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter queue size: ");

int size = scanner.nextInt();

Queue queue = new Queue(size);

while (true) {

System.out.println("\n1. Enqueue\n2. Dequeue\n3. Display\n4. Exit");

System.out.print("Enter your choice: ");

int choice = scanner.nextInt();

switch (choice) {

case 1:

System.out.print("Enter value to enqueue: ");

int value = scanner.nextInt();

queue.enqueue(value);

break;

case 2:

System.out.println("Dequeued: " + queue.dequeue());

break;

case 3:

queue.display();

break;

case 4:

System.out.println("Exiting...");

scanner.close();

return;

default:

System.out.println("Invalid choice, try again.");

}

}

}

**}**

**Output:**

