

JAVA MOODLE PROGAMMING

1. Final Variable

- Final variable is a variable that is declared in a class and is not redeclared in any other class.
- Final variable is a variable that is declared in a class and is not redeclared in any other class.

2. Final Method

- Final method is a method that is declared in a class and is not overridden in any other class.
- Final method is a method that is declared in a class and is not overridden in any other class.

3. Final Class

- Final class is a class that is declared in a class and is not extended by any other class.
- Final class is a class that is declared in a class and is not extended by any other class.

Source Code Program for Final Variable, Final Method and Final Class

```
1 // Final Variable
2 // Final Method
3 // Final Class
4 // Source Code Program for Final Variable, Final Method and Final Class
```

1. Final Variable

Line	Code	Output
1	int x = 10;	10
2	int y = 20;	20
3	int z = 30;	30
4	int w = 40;	40
5	int v = 50;	50
6	int u = 60;	60
7	int t = 70;	70
8	int s = 80;	80
9	int r = 90;	90
10	int q = 100;	100

CS23333-Object Oriented Programming Using Java 2023

CS23333-Object Oriented Programming Using Java 2023

1. Final Variable

- Final variable is a variable that is declared in a class and is not redeclared in any other class.
- Final variable is a variable that is declared in a class and is not redeclared in any other class.

2. Final Method

- Final method is a method that is declared in a class and is not overridden in any other class.
- Final method is a method that is declared in a class and is not overridden in any other class.

3. Final Class

- Final class is a class that is declared in a class and is not extended by any other class.
- Final class is a class that is declared in a class and is not extended by any other class.

Source Code Program for Final Variable, Final Method and Final Class

```
1 // Final Variable
2 // Final Method
3 // Final Class
4 // Source Code Program for Final Variable, Final Method and Final Class
```

1. Final Variable

Line	Code	Output
1	int x = 10;	10
2	int y = 20;	20
3	int z = 30;	30
4	int w = 40;	40
5	int v = 50;	50
6	int u = 60;	60
7	int t = 70;	70
8	int s = 80;	80
9	int r = 90;	90
10	int q = 100;	100

CS23333-Object Oriented Programming Using Java 2023

CS23333-Object Oriented Programming Using Java 2023

1. Final Variable

- Final variable is a variable that is declared in a class and is not redeclared in any other class.
- Final variable is a variable that is declared in a class and is not redeclared in any other class.

2. Final Method

- Final method is a method that is declared in a class and is not overridden in any other class.
- Final method is a method that is declared in a class and is not overridden in any other class.

3. Final Class

- Final class is a class that is declared in a class and is not extended by any other class.
- Final class is a class that is declared in a class and is not extended by any other class.

Source Code Program for Final Variable, Final Method and Final Class

```
1 // Final Variable
2 // Final Method
3 // Final Class
4 // Source Code Program for Final Variable, Final Method and Final Class
```

1. Final Variable

Line	Code	Output
1	int x = 10;	10
2	int y = 20;	20
3	int z = 30;	30
4	int w = 40;	40
5	int v = 50;	50
6	int u = 60;	60
7	int t = 70;	70
8	int s = 80;	80
9	int r = 90;	90
10	int q = 100;	100

Copy assignment
Test assignment
Test answer

Test
Question
Answer
Feedback

Unit 8
Topic 8
Question 1

Consider the following code snippet. Which of the following is the correct output of the program?

```
public class Main {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
        int sum = 0;
        for (int i = 0; i < arr.length; i++) {
            sum += arr[i];
        }
        System.out.println(sum);
    }
}
```

Options:

- A) 55
- B) 54
- C) 53
- D) 52

Correct Answer: A

Test
Question
Answer
Feedback

Unit 8
Topic 8
Question 2

The given code snippet is a Java program. Which of the following is the correct output of the program?

```
public class Main {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
        int sum = 0;
        for (int i = 0; i < arr.length; i++) {
            sum += arr[i];
        }
        System.out.println(sum);
    }
}
```

Options:

- A) 55
- B) 54
- C) 53
- D) 52

Correct Answer: A

Test
Question
Answer
Feedback

Unit 8
Topic 8
Question 3

Which of the following is the correct output of the program?

```
public class Main {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
        int sum = 0;
        for (int i = 0; i < arr.length; i++) {
            sum += arr[i];
        }
        System.out.println(sum);
    }
}
```

Options:

- A) 55
- B) 54
- C) 53
- D) 52

Correct Answer: A

Test
Question
Answer
Feedback

Unit 8
Topic 8
Question 4

Which of the following is the correct output of the program?

```
public class Main {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
        int sum = 0;
        for (int i = 0; i < arr.length; i++) {
            sum += arr[i];
        }
        System.out.println(sum);
    }
}
```

Options:

- A) 55
- B) 54
- C) 53
- D) 52

Correct Answer: A

Test
Question
Answer
Feedback

Unit 8
Topic 8
Question 5

Which of the following is the correct output of the program?

```
public class Main {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
        int sum = 0;
        for (int i = 0; i < arr.length; i++) {
            sum += arr[i];
        }
        System.out.println(sum);
    }
}
```

Options:

- A) 55
- B) 54
- C) 53
- D) 52

Correct Answer: A

Copy assignment
Test assignment
Test answer

Test
Question
Answer
Feedback

Unit 8
Topic 8
Question 1

Consider the following code snippet. Which of the following is the correct output of the program?

```
public class Main {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
        int sum = 0;
        for (int i = 0; i < arr.length; i++) {
            sum += arr[i];
        }
        System.out.println(sum);
    }
}
```

Options:

- A) 55
- B) 54
- C) 53
- D) 52

Correct Answer: A

Test
Question
Answer
Feedback

Unit 8
Topic 8
Question 2

The given code snippet is a Java program. Which of the following is the correct output of the program?

```
public class Main {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
        int sum = 0;
        for (int i = 0; i < arr.length; i++) {
            sum += arr[i];
        }
        System.out.println(sum);
    }
}
```

Options:

- A) 55
- B) 54
- C) 53
- D) 52

Correct Answer: A

Test
Question
Answer
Feedback

Unit 8
Topic 8
Question 3

Which of the following is the correct output of the program?

```
public class Main {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
        int sum = 0;
        for (int i = 0; i < arr.length; i++) {
            sum += arr[i];
        }
        System.out.println(sum);
    }
}
```

Options:

- A) 55
- B) 54
- C) 53
- D) 52

Correct Answer: A

Test
Question
Answer
Feedback

Unit 8
Topic 8
Question 4

Which of the following is the correct output of the program?

```
public class Main {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
        int sum = 0;
        for (int i = 0; i < arr.length; i++) {
            sum += arr[i];
        }
        System.out.println(sum);
    }
}
```

Options:

- A) 55
- B) 54
- C) 53
- D) 52

Correct Answer: A

Test
Question
Answer
Feedback

Unit 8
Topic 8
Question 5

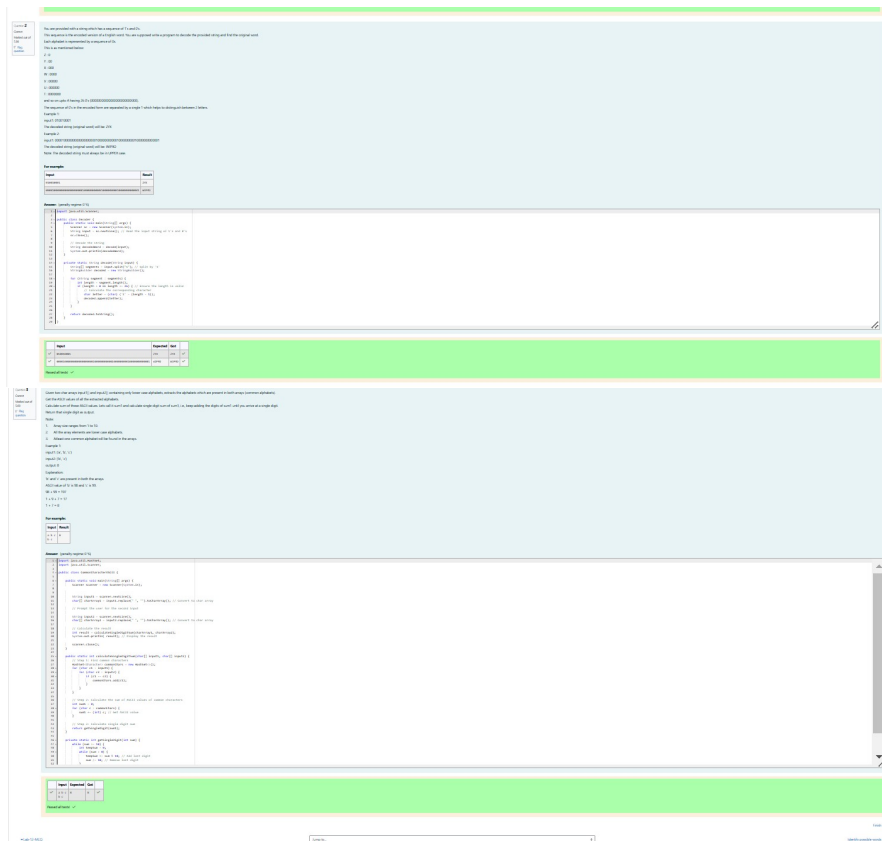
Which of the following is the correct output of the program?

```
public class Main {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
        int sum = 0;
        for (int i = 0; i < arr.length; i++) {
            sum += arr[i];
        }
        System.out.println(sum);
    }
}
```

Options:

- A) 55
- B) 54
- C) 53
- D) 52

Correct Answer: A



MINI PROJECT

HOSPITAL MANAGEMENT SYSTEM

CODE:

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.table.DefaultTableModel;
import java.sql.*;

public class HospitalManagementSystem {

    private static final String URL = "jdbc:mysql://localhost:3306/hospital_management";
```

```

private static final String USER = "root"; // Replace with your MySQL username
private static final String PASSWORD = "Maria@2006"; // Replace with your MySQL password

private JFrame frame;
private JTable patientsTable;
private JTable doctorsTable;
private JTable appointmentsTable;
private DefaultTableModel patientsModel;
private DefaultTableModel doctorsModel;
private DefaultTableModel appointmentsModel;

// Method to establish a connection to the MySQL database
public static Connection getConnection() throws SQLException {
    return DriverManager.getConnection(URL, USER, PASSWORD);
}

// Main method to launch the application
public static void main(String[] args) {
    EventQueue.invokeLater(() -> {
        try {
            HospitalManagementSystem window = new HospitalManagementSystem();
            window.frame.setVisible(true);
        } catch (Exception e) {
            e.printStackTrace();
        }
    });
}

// Constructor to initialize the application window
public HospitalManagementSystem() {
    initialize();
}

// Initialize the GUI
private void initialize() {
    frame = new JFrame();
    frame.setBounds(100, 100, 800, 600);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    JPanel panel = new JPanel();
    panel.setLayout(new BorderLayout());
    frame.getContentPane().add(panel, BorderLayout.CENTER);

    JTabbedPane tabbedPane = new JTabbedPane();
    panel.add(tabbedPane, BorderLayout.CENTER);

    // Patients Tab
    JPanel patientsPanel = new JPanel(new BorderLayout());
    tabbedPane.addTab("Patients", null, patientsPanel, null);

    patientsModel = new DefaultTableModel(new String[]{"ID", "Name", "Age", "Gender", "Phone"},
0);
    patientsTable = new JTable(patientsModel);
    JScrollPane patientsScrollPane = new JScrollPane(patientsTable);
    patientsPanel.add(patientsScrollPane, BorderLayout.CENTER);

    JButton addPatientButton = new JButton("Add Patient");

```

```

addPatientButton.addActionListener(e -> showAddPatientDialog());
patientsPanel.add(addPatientButton, BorderLayout.SOUTH);

// Doctors Tab
JPanel doctorsPanel = new JPanel(new BorderLayout());
tabbedPane.addTab("Doctors", null, doctorsPanel, null);

doctorsModel = new DefaultTableModel(new String[]{"ID", "Name", "Specialty", "Phone"}, 0);
doctorsTable = new JTable(doctorsModel);
JScrollPane doctorsScrollPane = new JScrollPane(doctorsTable);
doctorsPanel.add(doctorsScrollPane, BorderLayout.CENTER);

JButton addDoctorButton = new JButton("Add Doctor");
addDoctorButton.addActionListener(e -> showAddDoctorDialog());
doctorsPanel.add(addDoctorButton, BorderLayout.SOUTH);

// Appointments Tab
JPanel appointmentsPanel = new JPanel(new BorderLayout());
tabbedPane.addTab("Appointments", null, appointmentsPanel, null);

appointmentsModel = new DefaultTableModel(new String[]{"ID", "Patient Name", "Doctor
Name", "Date"}, 0);
appointmentsTable = new JTable(appointmentsModel);
JScrollPane appointmentsScrollPane = new JScrollPane(appointmentsTable);
appointmentsPanel.add(appointmentsScrollPane, BorderLayout.CENTER);

JButton addAppointmentButton = new JButton("Add Appointment");
addAppointmentButton.addActionListener(e -> showAddAppointmentDialog());
appointmentsPanel.add(addAppointmentButton, BorderLayout.SOUTH);

// Load existing data from the database
loadPatients();
loadDoctors();
loadAppointments();
}

// Load patients from the database into the table
private void loadPatients() {
    try (Connection conn = getConnection();
        Statement stmt = conn.createStatement();
        ResultSet rs = stmt.executeQuery("SELECT * FROM patients")) {

        while (rs.next()) {
            int id = rs.getInt("id");
            String name = rs.getString("name");
            int age = rs.getInt("age");
            String gender = rs.getString("gender");
            String phone = rs.getString("phone");
            patientsModel.addRow(new Object[]{id, name, age, gender, phone});
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

// Load doctors from the database into the table
private void loadDoctors() {

```

```

try (Connection conn = getConnection();
    Statement stmt = conn.createStatement();
    ResultSet rs = stmt.executeQuery("SELECT * FROM doctors")) {

    while (rs.next()) {
        int id = rs.getInt("id");
        String name = rs.getString("name");
        String specialty = rs.getString("specialty");
        String phone = rs.getString("phone");
        doctorsModel.addRow(new Object[]{id, name, specialty, phone});
    }
} catch (SQLException e) {
    e.printStackTrace();
}

}

// Load appointments from the database into the table
private void loadAppointments() {
    try (Connection conn = getConnection();
        Statement stmt = conn.createStatement();
        ResultSet rs = stmt.executeQuery("SELECT a.id, p.name AS patient_name, d.name AS
doctor_name, a.appointment_date " +
        "FROM appointments a JOIN patients p ON a.patient_id = p.id JOIN doctors d ON
a.doctor_id = d.id")) {

        while (rs.next()) {
            int id = rs.getInt("id");
            String patientName = rs.getString("patient_name");
            String doctorName = rs.getString("doctor_name");
            Date appointmentDate = rs.getDate("appointment_date");
            appointmentsModel.addRow(new Object[]{id, patientName, doctorName,
appointmentDate});
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

// Show dialog to add a new patient
private void showAddPatientDialog() {
    JTextField nameField = new JTextField(20);
    JTextField ageField = new JTextField(20);
    JTextField genderField = new JTextField(20);
    JTextField phoneField = new JTextField(20);

    JPanel panel = new JPanel(new GridLayout(4, 2));
    panel.add(new JLabel("Name:"));
    panel.add(nameField);
    panel.add(new JLabel("Age:"));
    panel.add(ageField);
    panel.add(new JLabel("Gender:"));
    panel.add(genderField);
    panel.add(new JLabel("Phone:"));
    panel.add(phoneField);

    int option = JOptionPane.showConfirmDialog(null, panel, "Add Patient",
JOptionPane.OK_CANCEL_OPTION, JOptionPane.PLAIN_MESSAGE);

```

```

if (option == JOptionPane.OK_OPTION) {
    String name = nameField.getText();
    int age = Integer.parseInt(ageField.getText());
    String gender = genderField.getText();
    String phone = phoneField.getText();

    try (Connection conn = getConnection());
        PreparedStatement stmt = conn.prepareStatement("INSERT INTO patients (name, age,
gender, phone) VALUES (?, ?, ?, ?)") {

        stmt.setString(1, name);
        stmt.setInt(2, age);
        stmt.setString(3, gender);
        stmt.setString(4, phone);
        stmt.executeUpdate();

        patientsModel.setRowCount(0); // Clear table
        loadPatients(); // Reload patients
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

// Show dialog to add a new doctor
private void showAddDoctorDialog() {
    JTextField nameField = new JTextField(20);
    JTextField specialtyField = new JTextField(20);
    JTextField phoneField = new JTextField(20);

    JPanel panel = new JPanel(new GridLayout(3, 2));
    panel.add(new JLabel("Name:"));
    panel.add(nameField);
    panel.add(new JLabel("Specialty:"));
    panel.add(specialtyField);
    panel.add(new JLabel("Phone:"));
    panel.add(phoneField);

    int option = JOptionPane.showConfirmDialog(null, panel, "Add Doctor",
JOptionPane.OK_CANCEL_OPTION, JOptionPane.PLAIN_MESSAGE);
    if (option == JOptionPane.OK_OPTION) {
        String name = nameField.getText();
        String specialty = specialtyField.getText();
        String phone = phoneField.getText();

        try (Connection conn = getConnection());
            PreparedStatement stmt = conn.prepareStatement("INSERT INTO doctors (name, specialty,
phone) VALUES (?, ?, ?)") {

            stmt.setString(1, name);
            stmt.setString(2, specialty);
            stmt.setString(3, phone);
            stmt.executeUpdate();

            doctorsModel.setRowCount(0); // Clear table
            loadDoctors(); // Reload doctors
        } catch (SQLException e) {

```

```

        e.printStackTrace();
    }
}

// Show dialog to add a new appointment
private void showAddAppointmentDialog() {
    JComboBox<String> patientComboBox = new JComboBox<>();
    JComboBox<String> doctorComboBox = new JComboBox<>();
    JTextField dateField = new JTextField(20);

    try (Connection conn = getConnection();
        Statement stmt = conn.createStatement()) {

        // Load Patients into ComboBox
        ResultSet rsPatients = stmt.executeQuery("SELECT id, name FROM patients");
        while (rsPatients.next()) {
            patientComboBox.addItem(rsPatients.getInt("id") + ": " + rsPatients.getString("name"));
        }

        // Load Doctors into ComboBox
        ResultSet rsDoctors = stmt.executeQuery("SELECT id, name FROM doctors");
        while (rsDoctors.next()) {
            doctorComboBox.addItem(rsDoctors.getInt("id") + ": " + rsDoctors.getString("name"));
        }

    } catch (SQLException e) {
        e.printStackTrace();
    }

    JPanel panel = new JPanel(new GridLayout(4, 2));
    panel.add(new JLabel("Select Patient:"));
    panel.add(patientComboBox);
    panel.add(new JLabel("Select Doctor:"));
    panel.add(doctorComboBox);
    panel.add(new JLabel("Appointment Date (YYYY-MM-DD):"));
    panel.add(dateField);

    int option = JOptionPane.showConfirmDialog(null, panel, "Add Appointment",
        JOptionPane.OK_CANCEL_OPTION, JOptionPane.PLAIN_MESSAGE);
    if (option == JOptionPane.OK_OPTION) {
        String[] patientParts = patientComboBox.getSelectedItem().toString().split(": ");
        String[] doctorParts = doctorComboBox.getSelectedItem().toString().split(": ");
        int patientId = Integer.parseInt(patientParts[0]);
        int doctorId = Integer.parseInt(doctorParts[0]);
        String appointmentDate = dateField.getText();

        try (Connection conn = getConnection();
            PreparedStatement stmt = conn.prepareStatement("INSERT INTO appointments (patient_id,
            doctor_id, appointment_date) VALUES (?, ?, ?)")) {

            Date date = Date.valueOf(appointmentDate); // Convert the input date string to a Date
            object

            stmt.setInt(1, patientId);
            stmt.setInt(2, doctorId);
            stmt.setDate(3, date);

```



```

        stmt.executeUpdate();

        appointmentsModel.setRowCount(0); // Clear table
        loadAppointments(); // Reload appointments
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
}
}

```

SNAPSHOTS:

Sign Up Here


UserName

Select UserType

Select

Email Address

Password



Doctor Details

Doctor ID	<input type="text"/>
Full Name	<input type="text"/>
Father's Name	<input type="text"/>
Email-id	<input type="text"/>
Contact no	<input type="text"/>
Address	<input type="text"/>
Qualifications	<input type="text"/>

Patient Detail's

Patient ID	<input type="text"/>
Name	<input type="text"/>
Father's Name	<input type="text"/>
Address	<input type="text"/>
Contact No.	<input type="text"/>
Email-id	<input type="text"/>
Age	<input type="text"/>