INCOMPLETE DOCUMENT

outputs unfinished

Rexiel Scarlet

2/10/2024

Contents

1 Fibonacci Series

Write a program to print Fibonacci series up to 'n' times (read 'n' as command line argument)

1.1 Code

```
package org.projects.prog1;

public class Fibo {
  public static void main(String args[]) {
    int limit = Integer.parseInt(args[0]);
    for (int i = 0; i < limit; i++) {
      System.out.print(fibo(i) + " ");
    }
  }

  static int fibo(int num) {
    return (num <= 1) ? num : fibo(num - 1) + fibo(num - 2);
  }
}
```

1.2 Output

0 1 1 2 3 5 8 13

2 Operations On Complex Numbers

Write a program to create a class Complex that has two members real and imaginary and methods to initialize and print the complex number. Create another class ComplexOperations and provide static methods to add, multiply, and compare two complex numbers.

2.1 Code

```java
package org.projects.prog2;

public class Complex {
  int real, img;

  Complex(int real, int img) {
    this.real = real;
    this.img = img;
  }

  void display() {
    // formatting for correctly displaying complex numbers
    System.out.printf("%d %si\n", real, (img > 0) ? "+ " + img : "- " + -img);
  }

  static Complex add(Complex n1, Complex n2) {
    int real = n1.real + n2.real;
    int img = n1.img + n2.img;
    return new Complex(real, img);
  }

  static Complex multiply(Complex n1, Complex n2) {
    int real = (n1.real * n2.real) - (n1.img * n2.img);
    int img = (n1.real * n2.img) + (n1.img * n2.real);
    return new Complex(real, img); // result = (ac - bd) + (ad + bc)i
  }

  static void cmp(Complex n1, Complex n2) {
    if (n1.real == n2.real && n1.img == n2.img) {
      System.out.println("The complex numbers are equal");
    } else {
      System.out.println("The complex numbers are not equal");
    }
  }

  public static void main(String args[]) {
    Complex c1 = new Complex(2, 3),
        c2 = new Complex(1, -2),
        c3 = add(c1, c2),
        c4 = multiply(c1, c2);

    c1.display();
    c2.display();
    c3.display();
    c4.display();
    cmp(c1, c2);
    cmp(c1, c1);
  }
}
```

2.2 Output

```
2 + 3i
1 - 2i
3 + 1i
8 - 1i
The complex numbers are not equal
The complex numbers are equal
```

3 Package Implementation

Create a package called pack which contains a class Prime with a member function checkPrime() to check whether a number is prime or not. Create another package called mypack containing a class Matrix with the member functions to: read a matrix, display the matrix, find the sum of diagonal elements of the matrix and check whether the sum is prime or not by accessing the method checkPrime() from Prime class.

3.1 Code

Prime.java (package pack)

```java
package org.projects.prog3.pack;

public class Prime {
  public static boolean checkPrime(int num) {
    for (int i = 2; i <= num / 2; i++) {
      if (num % i == 0) {
        return false;
      }
    }

    return true;
  }
}
```

Matrix.java (package mypack)

```java
package org.projects.prog3.mypack;

import java.util.Scanner;
import org.projects.prog3.pack.Prime;

public class Matrix {
  public static final int size = 3;
  public int matrix[][] = new int[size][size];

  public void accept() {
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter " + size * size + " matrix elements");
    int diagSum = 0;
    for (int i = 0; i < size; i++) {
      for (int j = 0; j < size; j++) {
        matrix[i][j] = sc.nextInt();
      }
    }
  }

  public void display() {
    System.out.println("Given Matrix:");
    for (int i = 0; i < size; i++) {
      for (int j = 0; j < size; j++) {
        System.out.print(matrix[i][j] + " ");
      }
      System.out.println();
    }
  }
```

```
  public int diagSum() {
    int sum = 0;
    for (int i = 0; i < size; i++) {
      for (int j = 0; j < size; j++) {
        if (i == j || size - (i + 1) == j) {
          sum += matrix[i][j];
        }
      }
    }
    return sum;
  }

  public static void main(String args[]) {
    Matrix mat = new Matrix();
    mat.accept();
    mat.display();

    if (Prime.checkPrime(mat.diagSum())) {
      System.out.println("Diagonal is prime");
    } else {
      System.out.println("Diagonal is not prime");
    }
  }
}
```

3.2 Output

Enter 9 matrix elements
1 2 3
0 1 0
0 1 2
Given Matrix:
1 2 3
0 1 0
0 1 2

Diagonal is prime

4 Constructor Overloading

Write a program to create a class Solid with data members: length, breadth, height. Provide 3 constructors having one parameter (for cube), two parameters (for square prism), and three parameters (rectangular prism). Also provides functions to calculate area and volume. Find the area & volume of a cube, a square prism, and a rectangular prism using the above class.

4.1 Code

```java
package org.projects.prog4;

public class Solid {
  int length, breadth, height;

  public Solid(int side) {
    length = side;
    breadth = side;
    height = side;
  }

  public Solid(int side, int height) {
    length = side;
    breadth = side;
    this.height = height;
  }

  // its really easy to mispell length/breadth/height and get into odd errors
  // ideally during exams shorten these to just l, b, h;
  public Solid(int length, int breadth, int height) {
    this.length = length;
    this.breadth = breadth;
    this.height = height;
  }

  int area() {
    return 2 * ((length * breadth) + (breadth * height) + (height * length));
  }

  int volume() {
    return length * breadth * height;
  }

  void display() {
    System.out.println(String.format("Area: %d\nVolume: %d", area(), volume()));
  }

  public static void main(String args[]) {
    Solid cube = new Solid(2), sPrism = new Solid(2, 3), cuboid = new Solid(2, 3, 4);

    cube.display();
    sPrism.display();
    cuboid.display();
  }
}
```

4.2 Output

Area: 24
Volume: 8
Area: 32
Volume: 12
Area: 52
Volume: 24

5 Abstract Class Implementation

Create an abstract class called Figure which contains three data members (length, breadth and height). Include an abstract method to find the area. Figure class also contains concrete methods to read the data members and to display them. Derive two classes Rectangle and Triangle from Figure and override area() to find the area of a rectangle and triangle.

5.1 Code

```
package org.projects.prog5;

import java.util.Scanner;

abstract class Figure {
  int length, breadth, height;

  abstract int area();

  void accept() {
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter length, breadth, height");
    length = sc.nextInt();
    breadth = sc.nextInt();
    height = sc.nextInt();
  }

  void display() {
    System.out.printf(
        "Length: %d Breadth: %d Height: %d Area: %d", length, breadth, height,
        ↪  area());
  }
}

class Rectangle extends Figure {
  int area() {
    return length * breadth;
  }
}

class Triangle extends Figure {
  int area() {
    return (breadth * height) / 2;
  }
}

public class Main {
  public static void main(String args[]) {
    Rectangle rec = new Rectangle();
    Triangle tri = new Triangle();
    System.out.println("Rectangle");
    rec.accept();
    rec.display();
    System.out.println("Triangle");
    tri.accept();
    tri.display();
  }
}
```

5.2 Output

Rectangle
Enter length, breadth, height
2 3 0
Length: 2, Breadth: 3, Height: 0
Area: 6
Triangle
Enter length, breadth, height
0 2 4
Length: 0, Breadth: 2, Height: 4
Area: 4

6 Inheritance In Java

Write a program to create a class called Rectangle with length, breadth and area. Provide functions to find the area and get back to the area. Create a new class Box by extending the Rectangle class add two new members, height and volume, and also new functions to calculate and get back the volume.

6.1 Code

```java
package org.projects.prog6;

class Rectangle {
  int length, breadth;

  public Rectangle(int length, int breadth) {
    this.length = length;
    this.breadth = breadth;
  }

  public int area() {
    return length * breadth;
  }
}

public class Box extends Rectangle {
  int height;

  public Box(int length, int breadth, int height) {
    super(length, breadth);
    this.height = height;
  }

  public int area() {
    return 2 * ((length * breadth) + (breadth * height) + (height * length));
  }

  public int volume() {
    return length * breadth * height;
  }

  public static void main(String args[]) {
    Rectangle rec = new Rectangle(2, 3);
    System.out.println("Area of Rectangle: " + rec.area());

    Box box = new Box(2, 3, 2);
    System.out.printf("Area of Box: %d\nVolume of Box: %d\n", box.area(),
    ↪  box.volume());
  }
}
// OUTPUT
/*
Area of Rectangle: 6
Area of Box: 32
Volume of Box: 12
*/
```

6.2 Output

7 Abstract Class With Inheritance

Write a program to create an abstract base class Account with 3 members account holder name, account number and balance amount. Provide a constructor to initialize data members, a function to deposit cash to an account, and an abstract function, withdrawal.

7.1 Code

```
package org.projects.prog7;

// this program does not require a main function
// since abstract classes can not be instantiated
public abstract class Account {
  String holder_name;
  long number;
  float balance;

  Account(String name, long num) {
    balance = 0;
    holder_name = name;
    number = num;
  }

  public void deposit(int amount) {
    balance += amount;
  }

  abstract void withdraw(int amount);
}
```

7.2 Output

8 Interface Implementation

Write a program to create a class Employee with data members name, code and basic pay and with functions to initialize and print information. Create an interface Salary with a function salary calculation. By inheriting the Employee class and Salary Interface create a new class SalarySlip which overrides the salary calculation method to calculate the net salary of an employee from basic pay. Provide a function to print the Salary Slip of the employee in the SalarySlip class.

8.1 Code

```
package org.projects.prog8;

class Employee {
  String name, code;
  public int basicPay;

  public Employee(String name, String code, int basicPay) {
    this.name = name;
    this.code = code;
    this.basicPay = basicPay;
  }

  public void display() {
    System.out.println("Name: " + name);
    System.out.println("Code: " + code);
    System.out.println("Basic Pay: " + basicPay);
  }
}

interface Salary {
  public abstract int calculate();
}

public class SallarySlip extends Employee implements Salary {
  int allowances, bonuses, deductions;

  public SallarySlip(
      String name, String code, int basicPay, int allowances, int bonuses, int
      ↪  deductions) {
    super(name, code, basicPay);
    this.allowances = allowances;
    this.bonuses = bonuses;
    this.deductions = deductions;
  }

  public int calculate() {
    return basicPay + allowances + bonuses - deductions;
  }

  public void display() {
    System.out.printf("Employee: %s\nCode: %s\nFinal Sallary: %d\n", name, code,
    ↪  calculate());
  }

  public static void main(String args[]) {
    SallarySlip slip = new SallarySlip("Bob Rose", "001", 10000, 4000, 3000, 1500);
    slip.display();
  }
```

```
}

// OUTPUT
/*
Employee: Bob Rose
Code: 001
Final Sallary: 15500
*/
```

8.2 Output

9 Multithreading In Java

Write a Java program that implements a multithreaded program has three threads. First thread generates a random integer every 1 second and if the value is even, second thread computes the square of the number and prints. If the value is odd the third thread will print the value of cube of the number.

9.1 Code

```
package org.projects.prog9;

import java.util.Random;

public class Multithread {
  public static int random = 0;

  public static Runnable threadMain =
      new Runnable() {
        public void run() {
          Random rand = new Random();
          for (int i = 0; i < 4; i++) {
            try {
              Thread.sleep(1000);
            } catch (Exception e) {
              // ignore
            }
            random = rand.nextInt(10);
            if (random % 2 == 0) {
              new Thread(threadEven).start();
            } else {
              new Thread(threadOdd).start();
            }
          }
        }
      };

  public static Runnable threadEven =
      new Runnable() {
        public void run() {
          System.out.println("Square: " + random * random);
        }
      };

  public static Runnable threadOdd =
      new Runnable() {
        public void run() {
          System.out.println("Cube: " + random * random * random);
        }
      };

  public static void main(String[] args) {
    new Thread(threadMain).start();
    /* alternative
      Thread th = new Thread(threadMain);
      th.start();
    */
  }
}
```

9.2 Output

10 Exception Handling

Write an exception handling program using try, catch, throw, throws and finally.

10.1 Code

```
package org.projects.prog10;

public class Errors {
  static void division() throws InterruptedException {
    Thread.sleep(100);
    throw new ArithmeticException("division by zero");
  }

  public static void main(String args[]) {
    try {
      division();
    } catch (InterruptedException e) {
      System.out.println(e.getMessage());
    } catch (ArithmeticException e) {
      System.out.println(e.getMessage());
    } finally {
      System.out.println("Hello Mars!");
    }
  }
}
// output
/*
division by zero
Hello Mars!
*/
```

10.2 Output

11 Validation using Swing

Write a swing program for validating the form having a numeric field, character field, phone number, and email id.

11.1 Code

```
package org.projects.prog11;

import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;

public class Swing {
  public static void main(String[] args) {
    JFrame frame = new JFrame("Swing Example");
    frame.setLayout(new GridLayout(3, 2));
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setSize(300, 130);

    JTextField phoneNum = new JTextField();
    frame.add(new JLabel("Phone"));
    frame.add(phoneNum);

    JTextField email = new JTextField();
    frame.add(new JLabel("Email"));
    frame.add(email);

    JButton submit = new JButton("Submit");
    JLabel result = new JLabel();
    result.setHorizontalAlignment(JLabel.CENTER);

    submit.addActionListener(
        new ActionListener() {
          public void actionPerformed(ActionEvent e) {
            String em = email.getText();
            String phone = phoneNum.getText();
            result.setText("invalid");

            if (em.isEmpty() || phone.isEmpty()) {
              return;
            }

            try {
              Long.parseLong(phone);
              if (phone.length() < 10) {
                return;
              }
            } catch (Exception error) {
              return;
            }

            if (!em.contains("@")) {
              return;
            }

            result.setText("Valid");
```

```
          }
        });

    frame.add(submit);
    frame.add(result);
    frame.setVisible(true);
  }
}
```

11.2 Output

12 Text Field in Swing

Write a Swing interactive program in which the number is entered in the textField, if the entered number is even, it should be displayed in List1 otherwise in List2.

12.1 Code

```
package org.projects.prog12;

import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;

public class Swing {
  public static void main(String[] args) {
    JFrame frame = new JFrame("Swing Example");
    frame.setLayout(new GridLayout(4, 2));
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setSize(300, 130);

    JTextField number = new JTextField();
    frame.add(new JLabel("Number: "));
    frame.add(number);

    JLabel list1 = new JLabel();
    list1.setHorizontalAlignment(JLabel.CENTER);
    frame.add(new JLabel("List 1"));
    frame.add(list1);

    JLabel list2 = new JLabel();
    list2.setHorizontalAlignment(JLabel.CENTER);
    frame.add(new JLabel("List 2"));
    frame.add(list2);

    JButton submit = new JButton("Submit");

    submit.addActionListener(
        new ActionListener() {
          public void actionPerformed(ActionEvent e) throws NumberFormatException {
            int num = Integer.parseInt(number.getText());
            number.setText("");
            if (num % 2 == 0) {
              list1.setText(list1.getText() + " " + num);
            } else {
              list2.setText(list2.getText() + " " + num);
            }
          }
        });

    frame.add(submit);
    frame.setVisible(true);
  }
}
```

12.2 Output

13 Traffic Light Simulation

Write a java program to simulate a traffic light. The program lets the user select one of the three lights: red, yellow, or green. On selecting a button, an appropriate message with "Stop", "Ready" or " Go" should appear above the buttons selected color.

13.1 Code

```
package org.projects.prog13;

import java.awt.Color;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;

public class Traffic {
  public static void main(String[] args) {
    JFrame frame = new JFrame("Traffic");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setLayout(new GridLayout(2, 1));
    frame.setSize(300, 130);

    JLabel sign = new JLabel();
    sign.setHorizontalAlignment(JLabel.CENTER);
    frame.add(sign);

    JPanel panel = new JPanel();
    panel.setLayout(new GridLayout(1, 3));

    JButton red = new JButton("STOP");
    red.setBackground(Color.RED);
    panel.add(red);

    JButton yellow = new JButton("READY");
    yellow.setBackground(Color.YELLOW);
    panel.add(yellow);

    JButton green = new JButton("GO");
    green.setBackground(Color.GREEN);
    panel.add(green);

    ActionListener on_click =
        new ActionListener() {
          public void actionPerformed(ActionEvent e) {
            sign.setText(((JButton) e.getSource()).getText());
          }
        };

    red.addActionListener(on_click);
    yellow.addActionListener(on_click);
    green.addActionListener(on_click);

    frame.add(panel);
    frame.setVisible(true);
  }
}
```

13.2 Output

14 Applet In Java

Create an applet using Java, which take a number as input and display table of the number. Use appropriate GUI components and layout in your applet.

14.1 Code

```
package org.projects.prog14;

import java.applet.*;
import java.awt.*;
import java.awt.event.*;

public class Tables extends Applet implements ActionListener {
  TextField t1;
  Button t2;
  String[] result;

  public void init() {
    t1 = new TextField(5);
    t2 = new Button("Submit");
    t2.addActionListener(this);

    result = new String[10];

    add(new Label("Enter a number"));
    add(t1);
    add(t2);
  }

  public void paint(Graphics g) {
    for (int i = 0; i < 10; i++) {
      g.drawString(result[i], 20, 40 + 20 * i);
    }
  }

  public void actionPerformed(ActionEvent e) {
    int number = Integer.parseInt(t1.getText());
    for (int i = 1; i <= 10; i++) {
      result[i - 1] = number + " x " + i + " = " + i * number;
    }
  }
}
```

14.2 Output

15 Savings Account

Write a Java program to define Account class. Derive Saving_Account class from Account class. Define appropriate constructors for both classes. Define the following methods in the Saving_Account class:

- Display (): To display account details including account number and balance in the account.
- Deposit (): To deposit money in an account.

15.1 Code

```
package org.projects.prog15;

class Account {
  public String accName;
  public long accNumber;

  Account(String name, long number) {
    accName = name;
    accNumber = number;
  }
}

public class SavingsAccount extends Account {
  private int accBalance;

  SavingsAccount(String name, long number) {
    super(name, number);
    accBalance = 0;
  }

  public void display() {
    System.out.println("Account: " + accName);
    System.out.println("No: " + accNumber);
    System.out.println("Balance: " + accBalance);
  }

  public void deposit(int ammount) {
    accBalance += ammount;
  }

  public static void main(String args[]) {
    SavingsAccount account = new SavingsAccount("Bijo", 1239812398);
    account.deposit(1000);
    account.display();
  }
}
```

15.2 Output

16 Integer Divisions

Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a NumberFormatException. If Num2 were zero, the program would throw an Arithmetic Exception Display the exception in a message dialog box.

16.1 Code

```
package org.projects.prog16;

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class Dialog {
  public static void main(String args[]) {
    JFrame frame = new JFrame("cool frame");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setSize(300, 200);
    frame.setLayout(new GridLayout(2, 1));

    JPanel panel = new JPanel();
    panel.setLayout(new GridLayout(2, 2));

    panel.add(new JLabel("Num 1: "));
    JTextField num1 = new JTextField();
    panel.add(num1);

    panel.add(new JLabel("Num 2: "));
    JTextField num2 = new JTextField();
    panel.add(num2);

    frame.add(panel);

    JButton submit = new JButton("Submit");
    submit.addActionListener(
        new ActionListener() {
          public void actionPerformed(ActionEvent e) {
            JDialog dialog = new JDialog();
            dialog.setSize(300, 50);

            JLabel prompt = new JLabel();
            prompt.setHorizontalAlignment(JLabel.CENTER);
            try {
              int n1 = Integer.parseInt(num1.getText());
              int n2 = Integer.parseInt(num2.getText());
              int result = n1 / n2;

              prompt.setText(n1 + "/" + n2 + " = " + result);
            } catch (NumberFormatException _e) {
              prompt.setText("Invalid Number");
            } catch (ArithmeticException _e) {
              prompt.setText("Division by Zero");
            }

            dialog.add(prompt);
```

```
            dialog.setVisible(true);
          }
        });

    frame.add(submit);
    frame.setVisible(true);
  }
}
```

16.2 Output

17 Simple Interest

Write a Java program to create an applet to find the simple interest on a given amount, rate of interest and duration. Use proper GUI components in your program.

17.1 Code

```
package org.projects.prog17;

import java.applet.*;
import java.awt.*;
import java.awt.event.*;

public class Interest extends Applet implements ActionListener {
  TextField principle, rate, time;
  Label result;
  Button submit;

  public void init() {
    principle = new TextField("principle");
    rate = new TextField("rate%");
    time = new TextField("time");

    add(new Label("Fill the following"));
    add(principle);
    add(rate);
    add(time);

    submit = new Button("Submit");
    submit.addActionListener(this);
    add(submit);

    result = new Label();
    add(result);
  }

  public void actionPerformed(ActionEvent e) {
    int p = Integer.parseInt(principle.getText());
    int r = Integer.parseInt(rate.getText());
    int n = Integer.parseInt(time.getText());

    float si = (p * r * n) / 100;
    result.setText("" + si);
  }
}
```

17.2 Output

18 Mouse Coordinates

Write an applet program to display the mouse coordinates.

18.1 Code

```
package org.projects.prog18;

import java.applet.*;
import java.awt.*;
import java.awt.event.*;

public class MouseProg extends Applet implements MouseMotionListener {
  int x, y;

  public void init() {
    x = 0;
    y = 0;

    addMouseMotionListener(this);
  }

  public void paint(Graphics g) {
    g.drawString("x = " + x + " y = " + y, 20, 20);
  }

  public void mouseMoved(MouseEvent e) {
    x = e.getX();
    y = e.getY();
    repaint();
  }

  public void mouseDragged(MouseEvent e) {}
}
```

18.2 Output

19 Simple Banner

Write an applet program to display Simple Banner.

19.1 Code

```
import java.applet.*;
import java.awt.*;
import java.awt.event.*;

public class Banner extends Applet implements Runnable {
  String bannerText;
  Thread scroll;

  public void init() {
    bannerText = "REALLY COOL BANNER ";
    scroll = new Thread(this);
  }

  public void start() {
    scroll.start();
  }

  public void paint(Graphics g) {
    g.drawString(bannerText, 100, 100);
  }

  public void run() {
    try {
      while (true) {
        Thread.sleep(500);
        bannerText = bannerText.substring(1) + bannerText.charAt(0);
        repaint();
      }
    } catch (Exception e) {
    }
  }
}
```

19.2 Output

20 Grid Layout Manager

Write a program using GridLayoutManager.

20.1 Code

```
package org.projects.prog20;

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class GridLay {
  public static void main(String args[]) {
    JFrame frame = new JFrame("Simple Grid");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setSize(300, 200);
    frame.setLayout(new GridLayout(3, 1));

    JPanel panel = new JPanel();
    panel.setLayout(new GridLayout(1, 2));
    panel.add(new JLabel("Enter Name"));

    JTextField name = new JTextField();
    panel.add(name);

    frame.add(panel);

    JButton submit = new JButton("Submit");
    JLabel hello = new JLabel();

    submit.addActionListener(
        new ActionListener() {
          public void actionPerformed(ActionEvent e) {
            hello.setText("Hello " + name.getText() + "!!");
          }
        });
    frame.add(submit);

    frame.add(hello);

    frame.setVisible(true);
  }
}
```

20.2 Output

21 Priority Threads

Write a multithreading program using different priority threads.

21.1 Code

```
package org.projects.prog21;

public class Threading {
  public static Runnable simple_runnable =
      new Runnable() {
        public void run() {
          String name = Thread.currentThread().getName();
          for (int i = 0; i < 10; i++) {
            System.out.println(name + " " + i);
          }
        }
      };

  public static void main(String args[]) {
    Thread th0 = new Thread(simple_runnable), th1 = new Thread(simple_runnable);

    th0.setPriority(Thread.MIN_PRIORITY);
    th1.setPriority(Thread.MAX_PRIORITY);

    th0.start();
    th1.start();
  }
}
```

21.2 Output

22 Employee Details

Write a JDBC program to add and display the details of an employee

22.1 Code

```
package org.projects.prog22;

import java.sql.*;

public class Employee {
  public static void main(String args[]) {
    try {
      Class.forName("com.mysql.cj.jdbc.Driver");
      String username = "root", password = "coolPass", dbname = "dbone";
      Connection cn =
          DriverManager.getConnection(
              // "jdbc:mysql://localhost:3306/dbone", "root", "coolPass"
              "jdbc:mysql://localhost:3306/" + dbname, username, password);

      Statement state = cn.createStatement();
      ResultSet rs = state.executeQuery("select * from employee");

      while (rs.next()) {
        System.out.println("Id: " + rs.getInt(1));
        System.out.println("Name: " + rs.getString(2));
        System.out.println("Designation: " + rs.getString(3));
        System.out.println();
      }

      cn.close();
    } catch (Exception e) {
    }
  }
}
```

22.2 Output

23 Update In JDBC

Write a JDBC program to add and update the details of a student.

23.1 Code

```
package org.projects.prog23;

import java.sql.*;
import java.util.Scanner;

public class Student {
  public static Scanner sc = new Scanner(System.in);

  public static void main(String args[]) {
    try {
      Class.forName("com.mysql.cj.jdbc.Driver");
      String username = "root", password = "coolPass", dbname = "dbone";
      Connection cn =
          DriverManager.getConnection(
              // "jdbc:mysql://localhost:3306/dbone", "root", "coolPass"
              "jdbc:mysql://localhost:3306/" + dbname, username, password);
      System.out.println("1 - add Student");
      System.out.println("2 - update Student");

      switch (sc.nextInt()) {
        case 1:
          addStudent(cn);
          break;
        case 2:
          updateStudent(cn);
          break;
        default:
          System.out.println("Invalid Choice");
          break;
      }

      displayStudents(cn);
      cn.close();
    } catch (Exception e) {
    }
  }

  public static void addStudent(Connection cn) {
    sc.nextLine(); // ignore garbage
    System.out.println("Enter name");
    String name = sc.nextLine();

    System.out.println("Enter roll number, age");
    int roll = sc.nextInt(), age = sc.nextInt();

    System.out.println("Enter Grade");
    char grade = sc.next().charAt(0);

    // to create the table use the following mysql code
    // create table students (
    //   roll int unique not null,
    //   name varchar(20) not null,
```

```
    //   age int not null,
    //   grade char(1) not null
    // );

    try {
      cn.createStatement()
          .execute(
              String.format(
                  "INSERT INTO students VALUES (%d, '%s', %d, '%c')", roll, name,
                  ↪  age, grade));
    } catch (Exception e) {
      System.out.println(e.getMessage());
    }
  }

  public static void updateStudent(Connection cn) {
    System.out.println("Enter rol number to update");
    int roll = sc.nextInt();

    System.out.println("Enter new age");
    int age = sc.nextInt();

    System.out.println("Enter new grade");
    char grade = sc.next().charAt(0);

    try {
      cn.createStatement()
          .executeUpdate(
              String.format(
                  "UPDATE students SET age = %d, grade = '%c' WHERE roll = %d", age,
                  ↪  grade, roll));
    } catch (Exception e) {
      System.out.println(e.getMessage());
    }
  }

  public static void displayStudents(Connection cn) {
    try {
      ResultSet set = cn.createStatement().executeQuery("select * from students");
      while (set.next()) {
        System.out.println(
            String.format(
                "roll: %d, name: %s, age: %d, grade: %s",
                set.getInt(1), set.getString(2), set.getInt(3), set.getString(4)));
      }
    } catch (Exception e) {
    }
  }
}
```

23.2 Output

24 Product Details

Write a JDBC program to add and delete the details of a product.

24.1 Code

```
package org.projects.prog24;

import java.sql.*;
import java.util.Scanner;

public class Product {
  public static Scanner sc = new Scanner(System.in);

  public static void main(String args[]) {
    try {
      Class.forName("com.mysql.cj.jdbc.Driver");
      String username = "root", password = "coolPass", dbname = "dbone";
      Connection cn =
          DriverManager.getConnection(
              // "jdbc:mysql://localhost:3306/dbone", "root", "coolPass"
              "jdbc:mysql://localhost:3306/" + dbname, username, password);

      boolean done = false;
      while (!done) {
        System.out.println("1 - add Product");
        System.out.println("2 - delete Product");
        System.out.println("3 - display");

        switch (sc.nextInt()) {
          case 1:
            addProduct(cn);
            break;
          case 2:
            deleteProduct(cn);
            break;
          case 3:
            displayProducts(cn);
            break;
          default:
            System.out.println("Invalid Choice");
            done = true;
            break;
        }
      }
      cn.close();
    } catch (Exception e) {
    }
  }

  public static void addProduct(Connection cn) {
    sc.nextLine(); // ignore garbage
    System.out.println("Enter Product name");
    String name = sc.nextLine();

    System.out.println("Enter product price");
    float price = sc.nextFloat();
```

```
    // to create the table use the following mysql code
    // create table products (
    //   id int unique AUTO_INCREMENT,
    //   name varchar(20) not null,
    //   price decimal(8, 2) not null,
    // );

    try {
      cn.createStatement()
          .execute(
              String.format("INSERT INTO products (name, price) VALUES ('%s', %f)",
              ↪  name, price));
    } catch (Exception e) {
      System.out.println(e.getMessage());
    }
  }

  public static void deleteProduct(Connection cn) {
    System.out.println("Enter and id to delete");
    int id = sc.nextInt();

    try {
      cn.createStatement().execute(String.format("DELETE FROM products WHERE id =
      ↪  %d", id));
    } catch (Exception e) {
      System.out.println(e.getMessage());
    }
  }

  public static void displayProducts(Connection cn) {
    try {
      ResultSet set = cn.createStatement().executeQuery("select * from products");
      while (set.next()) {
        System.out.println(
            String.format(
                "id: %d, name: %s, cost: %f", set.getInt(1), set.getString(2),
                ↪  set.getFloat(3)));
      }
    } catch (Exception e) {
    }
  }
}
```

24.2 Output