23CDR007

ANISHKUMAR A

1)

```
package JAVA;

class MyThread extends Thread {
    public void run() {
        System.out.println("Thread is Running");
    }
}
public class Main {
    public static void main(String[] args) {
        MyThread t = new MyThread();
        t.start();
    }
}
```

OUTPUT:

```
Thread is Running
```

2)

```
package JAVA;
class NumberThread extends Thread {
    public void run() {
        try {
            for (int i = 1; i <= 3; i++) {
                System.out.println(i);
                Thread.sleep(1000); // pause for 1 second (1000 ms)
```

```
            }
        } catch (InterruptedException e) {
            System.out.println("Thread interrupted");
        }
    }
}

public class Main {
    public static void main(String[] args) {
        NumberThread t = new NumberThread();
        t.start(); // start the thread
    }
}
```

OUTPUT:

```
1
2
3
```

3)

```
package JAVA;
class ChildThread extends Thread {
    public void run() {
        System.out.println("Child Thread: Running...");
        try {
            Thread.sleep(2000); // simulate some work (2 sec delay)
        } catch (InterruptedException e) {
            System.out.println("Child Thread interrupted");
        }
        System.out.println("Child Thread: Finished");
    }
}

public class Main {
    public static void main(String[] args) {
```

```
        ChildThread t = new ChildThread();
        t.start(); // start child thread

        try {
            t.join(); // main thread waits until child finishes
        } catch (InterruptedException e) {
            System.out.println("Main Thread interrupted");
        }

        System.out.println("Main Thread Resumed");
    }
}
```

OUTPUT:

```
Child Thread: Running...
Child Thread: Finished
Main Thread Resumed
```

4)

```
package JAVA;
//Program to name threads and display their names
class WorkerThread extends Thread {
 public WorkerThread(String name) {
    super(name); // set custom thread name
 }

 public void run() {
    System.out.println("Running: " + getName());
 }
}

public class Main {
 public static void main(String[] args) {
    WorkerThread t1 = new WorkerThread("Worker-1");
    WorkerThread t2 = new WorkerThread("Worker-2");
```

```
    t1.start();
    t2.start();
 }
}
```

OUTPUT:

```
<terminated> Main [Java Application] C:\Users\Anish\.
Running: Worker-1
Running: Worker-2
```

5)

```
package JAVA;
//Shared BankAccount class
class BankAccount {
 private int balance = 0;

 // synchronized deposit method to prevent race conditions
 public synchronized void deposit(int amount) {
    balance += amount;
 }

 public int getBalance() {
    return balance;
 }
}

//Thread class for depositing money
class DepositThread extends Thread {
 private BankAccount account;
 private int amount;
```

```
 public DepositThread(BankAccount account, int amount) {
    this.account = account;
    this.amount = amount;
 }

 public void run() {
    account.deposit(amount);
 }
}

public class Main {
 public static void main(String[] args) {
    BankAccount account = new BankAccount();

    // Two threads depositing money
    DepositThread t1 = new DepositThread(account, 1000);
    DepositThread t2 = new DepositThread(account, 1000);

    t1.start();
    t2.start();

    try {
       t1.join();
       t2.join();
    } catch (InterruptedException e) {
       System.out.println("Main Thread Interrupted");
    }

    System.out.println("Final Balance: " + account.getBalance());
 }
}
```

OUTPUT:

```
<terminated> Main [Java Application] C:\Users\Anish\.p2\pool\pl
Final Balance: 2000
```

6)

```
package JAVA;
class TablePrinter {
 public void printTable(int number) {
    synchronized (this) { // synchronized block
       System.out.println("Table of " + number + ":");
       for (int i = 1; i <= 10; i++) {
          System.out.print(number * i + " ");
          try {
             Thread.sleep(200); // small delay to simulate work
          } catch (InterruptedException e) {
             System.out.println("Thread interrupted");
          }
       }
       System.out.println("\n"); // new line after table
    }
 }
}
class TableThread extends Thread {
 private TablePrinter printer;
 private int number;

 public TableThread(TablePrinter printer, int number) {
    this.printer = printer;
    this.number = number;
 }

 public void run() {
```

```
    printer.printTable(number);
 }
}

public class Main {
 public static void main(String[] args) {
    TablePrinter printer = new TablePrinter();
    TableThread t1 = new TableThread(printer, 2);
    TableThread t2 = new TableThread(printer, 5);

    t1.start();
    t2.start();
 }
}
```

OUTPUT:

```
<terminated> Main [Java Application] C:\Users\Anish\.p2\p
Table of 2:
2 4 6 8 10 12 14 16 18 20

Table of 5:
5 10 15 20 25 30 35 40 45 50
```

7)

```
// Thread 1 → prints Title & Author
class BookInfoThread1 extends Thread {
   public void run() {
      System.out.println("Title: Java Programming");
      System.out.println("Author: James Gosling");
   }
}

// Thread 2 → prints Publisher & Year
class BookInfoThread2 extends Thread {
   public void run() {
```

```
        System.out.println("Publisher: Sun Press");
        System.out.println("Year: 2021");
    }
}

// Thread 3 → prints Price & ISBN
class BookInfoThread3 extends Thread {
    public void run() {
        System.out.println("Price: 450");
        System.out.println("ISBN: 123-4567");
    }
}

public class Main {
    public static void main(String[] args) {
        BookInfoThread1 t1 = new BookInfoThread1();
        BookInfoThread2 t2 = new BookInfoThread2();
        BookInfoThread3 t3 = new BookInfoThread3();

        // Start all threads
        t1.start();
        try {
            t1.join();  // ensure Title & Author print first
        } catch (InterruptedException e) {}

        t2.start();
        try {
            t2.join();  // ensure Publisher & Year print second
        } catch (InterruptedException e) {}

        t3.start();
    }
}
```

OUTPUT:

```
Table of 2:
2 4 6 8 10 12 14 16 18 20

Table of 5:
5 10 15 20 25 30 35 40 45 50
```

8)

```
package JAVA;
import java.io.*;

//Thread 1 → Writes to file
class WriterThread extends Thread {
 private File file;

 public WriterThread(File file) {
    this.file = file;
 }

 public void run() {
    try (FileWriter writer = new FileWriter(file)) {
        System.out.println("Writing to file...");
        writer.write("Hello, Multithreading!");
    } catch (IOException e) {
        e.printStackTrace();
    }
 }
}

//Thread 2 → Reads from file
class ReaderThread extends Thread {
 private File file;

 public ReaderThread(File file) {
```

```
    this.file = file;
  }

  public void run() {
    try (BufferedReader reader = new BufferedReader(new FileReader(file))) {
        System.out.println("Reading from file...");
        String content = reader.readLine();
        System.out.println("File Content: " + content);
    } catch (IOException e) {
        e.printStackTrace();
    }
  }
}

public class Main {
  public static void main(String[] args) {
    File file = new File("sample.txt");

    WriterThread writer = new WriterThread(file);
    ReaderThread reader = new ReaderThread(file);

    try {
        writer.start();
        writer.join();   // wait until writing is finished
        reader.start();  // then read the file
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
  }
}
```

OUTPUT:

```
Writing to file...
Reading from file...
File Content: Hello, Multithreading!
```

9)

```
// Daemon Thread Example
class DaemonTask extends Thread {
    public void run() {
        while (true) {
            System.out.println("Daemon Thread Running...");
            try {
                Thread.sleep(1000); // pause for 1 second
            } catch (InterruptedException e) {
                System.out.println("Daemon Thread Interrupted");
            }
        }
    }
}

public class Main {
    public static void main(String[] args) {
        DaemonTask daemon = new DaemonTask();
        daemon.setDaemon(true); // mark thread as daemon
        daemon.start();

        // Main thread work
        try {
            Thread.sleep(4000); // let daemon run for 4 seconds
        } catch (InterruptedException e) {
            e.printStackTrace();
        }

        System.out.println("Main Thread Finished");
        // When main thread ends, daemon thread also ends automatically
```

```
    }
}
```

OUTPUT:

```
Daemon Thread Running...
Daemon Thread Running...
Daemon Thread Running...
Daemon Thread Running...
Main Thread Finished
```

10)

```
// Deadlock Example
class Resource {
    String name;

    Resource(String name) {
        this.name = name;
    }
}

class Task1 extends Thread {
    private Resource resA;
    private Resource resB;

    Task1(Resource resA, Resource resB) {
        this.resA = resA;
        this.resB = resB;
    }

    public void run() {
        synchronized (resA) {
            System.out.println("Thread-1 locked " + resA.name);
            try { Thread.sleep(100); } catch (InterruptedException e) {}

            System.out.println("Thread-1 waiting for " + resB.name);
            synchronized (resB) {
```

```
            System.out.println("Thread-1 locked " + resB.name);
        }
    }
  }
}

class Task2 extends Thread {
  private Resource resA;
  private Resource resB;

  Task2(Resource resA, Resource resB) {
    this.resA = resA;
    this.resB = resB;
  }

  public void run() {
    synchronized (resB) {
      System.out.println("Thread-2 locked " + resB.name);
      try { Thread.sleep(100); } catch (InterruptedException e) {}

      System.out.println("Thread-2 waiting for " + resA.name);
      synchronized (resA) {
        System.out.println("Thread-2 locked " + resA.name);
      }
    }
  }
}

public class Main {
  public static void main(String[] args) {
    Resource resourceA = new Resource("Resource A");
    Resource resourceB = new Resource("Resource B");

    Task1 t1 = new Task1(resourceA, resourceB);
```

```
        Task2 t2 = new Task2(resourceA, resourceB);

        t1.start();
        t2.start();
    }
}
```

OUTPUT:

```
Thread-2 locked Resource B
Thread-1 locked Resource A
Thread-1 waiting for Resource B
Thread-2 waiting for Resource A
```

11)

```
// Thread Interruption Example
class MyThread extends Thread {
    public void run() {
        try {
            while (true) {
                System.out.println("Thread running...");
                Thread.sleep(500); // small pause
            }
        } catch (InterruptedException e) {
            System.out.println("Thread interrupted!");
        }
        System.out.println("Thread exiting...");
    }
}

public class Main {
    public static void main(String[] args) {
        MyThread t = new MyThread();
        t.start();
```

```
        try {
            Thread.sleep(2000); // main waits 2 seconds
        } catch (InterruptedException e) {
            e.printStackTrace();
        }

        t.interrupt(); // interrupt the thread
    }
}
```

OUTPUT:

```
Thread running...
Thread running...
Thread running...
Thread running...
Thread interrupted!
Thread exiting...
```