Assignment No. 5
Multinthreading

Question 1

Write a Java program to create and run a thread by extending the Thread class. The thread should print "Hello from Thread" five times.

HelloQue1.java > ...

```java
//Write a Java program to create and run a thread by extending the Thread class.
//The thread should print "Hello from Thread" five times.

public class HelloQue1 extends Thread {

    public void run() {
        for (int i = 0; i < 5; i++) {
            System.out.println("Hello from Thread");
        }
    }

    Run main | Debug main
    public static void main(String[] args) {
        HelloQue1 myThread = new HelloQue1();
        myThread.start();
    }
}
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\Shweta\Documents\MultithreadingJAVA> javac HelloQue1.java
PS C:\Users\Shweta\Documents\MultithreadingJAVA> java HelloQue1
Hello from Thread
Hello from Thread
Hello from Thread
Hello from Thread
Hello from Thread
PS C:\Users\Shweta\Documents\MultithreadingJAVA>
```

Question 2

Write a Java program to create and run a thread by implementing the Runnable interface. The thread should print numbers from 1 to 5.

```java
//Write a Java program to create and run a thread by implementing the Runnable in
public class RunnableQue2 implements Runnable {
    public void run() {
        for (int i = 1; i <= 5; i++) {
            System.out.println(i);
        }
    }
    public static void main(String[] args) {
        RunnableQue2 mRunnable = new RunnableQue2();
        Thread thread = new Thread(mRunnable);
        thread.start();
    }
}
```

Console | Eclipse IDE for Java Developers 2025-09 Release

```
<terminated> RunnableQue2 [Java Application] C:\Users\Shweta\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jr
1
2
3
4
5
```

Question 3

Write a Java program where the main thread prints "Main Thread Running" and a child thread prints "Child Thread Running". Run them simultaneously.

```java
//Write a Java program where the main thread prints "Main Thread Running" ar
public class Que3{
    static class ChildThread extends Thread {
        public void run() {
            System.out.println("Child Thread Running");
        }
    }
    public static void main(String[] args) {
        ChildThread child = new ChildThread();
        child.start();
        System.out.println("Main Thread Running");
    }
}
```

Console × Eclipse IDE for Java Developers 2025-09 Release

```
<terminated> Que3 [Java Application] C:\Users\Shweta\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.
Main Thread Running
Child Thread Running
```

Question 4

Write a Java program to demonstrate the use of setName() and getName() methods for threads.

```java
//Write a Java program to demonstrate the use of setName() and getName() methods for threads.
public class Que4 {
        public static void main(String[] args) {
            Thread thread1 = new Thread(new Runnable() {
                public void run() {
                    System.out.println("Hello from " + Thread.currentThread().getName());
                }
            });

            Thread thread2 = new Thread(new Runnable() {
                public void run() {
                    System.out.println("Hello from " + Thread.currentThread().getName());
                }
            });

            thread1.start();
            thread2.start();}}
```

Console × Eclipse IDE for Java Developers 2025-09 Release

```
<terminated> Que4 [Java Application] C:\Users\Shweta\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.8.v2025
Hello from Thread-0
Hello from Thread-1
```

Question 5

Write a Java program to demonstrate the use of setPriority() and getPriority() methods by creating two threads with different priorities.

HelloQue1.java | RunnableQue2.java | Que3.java | Que4.java | *Quee5.java

```java
//Write a Java program to demonstrate the use of setPriority() and getPriority() methods by
public class Quee5 {
    static class MyRunnable implements Runnable {
        public void run() {
            System.out.println("Running thread = " + Thread.currentThread().getName() +
                    " priority = " + Thread.currentThread().getPriority());
        }
    }
    public static void main(String[] args) {
        MyRunnable task = new MyRunnable();
        Thread lowPrior = new Thread(task);
        Thread highPrior = new Thread(task);
        lowPrior.setName("LowPriority");
        lowPrior.setPriority(Thread.MIN_PRIORITY);
        highPrior.setName("HighPriority");
        highPrior.setPriority(Thread.MAX_PRIORITY);
        lowPrior.start();
        highPrior.start();
```

Console

```
<terminated> Quee5 [Java Application] C:\Users\Shweta\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21
Running thread = HighPriority priority = 10
Running thread = LowPriority priority = 1
```

Question 6

Write a Java program where one thread prints numbers from 1 to 10, and another thread prints numbers from 11 to 20.

```java
//Write a Java program where one thread prints numbers from 1 to 10, and a
public class Que6 {
    static class Range1To10 implements Runnable {
        public void run() {
            for (int i = 1; i <= 10; i++) {
                System.out.println(i);
            }
        }
    }
    static class Range11To20 implements Runnable {
        public void run() {
            for (int i = 11; i <= 20; i++) {
                System.out.println(i);
            }
        }
    }
    public static void main(String[] args) {
        Thread thread1 = new Thread(new Range1To10());
        Thread thread2 = new Thread(new Range11To20());
        thread1.start();
        thread2.start();
    }
}
```

Console

<terminated> Que6 [Java Application] C:\Users\Shweta\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.j

```
1
2
3
4
5
11
6
7
8
9
10
12
13
14
15
```

Question 7

Write a Java program to demonstrate the use of the sleep() method by pausing a thread for 1 second after printing each number.

```java
//Write a Java program to demonstrate the use of the sleep() method by pausing a thread
public class Que7 {
        static class SleepTask implements Runnable {
        public void run() {
            for (int i = 1; i <= 5; i++) {
                System.out.println("Number = " + i);
                try {
                    Thread.sleep(1000);
                } catch (InterruptedException e) {
                    System.out.println("Thread interrupted");
                    Thread.currentThread().interrupt();
                    return;
                }
            }
        }
    }
    public static void main(String[] args) {
        SleepTask task = new SleepTask();
        Thread sleepyThread = new Thread(task);
        sleepyThread.start();
    }
}
```

Console

```
<terminated> Que7 [Java Application] C:\Users\Shweta\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64
Number = 1
Number = 2
Number = 3
Number = 4
Number = 5
```

Question 8

Write a Java program where the main thread waits for a child thread to finish using the join() method.

```java
public class Que8 {
    public static void main(String[] args) throws InterruptedException {
        Thread childThread = new Thread(new Runnable() {
            public void run() {
                System.out.println("Childthread started");
                try {
                    Thread.sleep(2000);
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
                System.out.println("Childthread finished");
            }
        });
        childThread.start();
        System.out.println("Mainthread waiting for childthread to finish");
        childThread.join();
        System.out.println("Mainthread finished");
    }
}
```

Console

```
<terminated> Que8 [Java Application] C:\Users\Shweta\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_
Main thread waiting for childthread to finish
Childthread started
Childthread finished
Main thread finished
```

Question 9

Write a Java program to check whether a thread is alive or not using the isAlive() method.

```java
public class Que9{
    public static void main(String[] args) throws InterruptedException {
        Thread myThread = new Thread(() -> {
            System.out.println("Thread started");
            try {
                Thread.sleep(1000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
            System.out.println("Thread finished");
        });
        System.out.println("Before start thread alive? " + myThread.isAlive());
        myThread.start();
        System.out.println("After start thread alive? " + myThread.isAlive());
        myThread.join();
        System.out.println("After join thread alive? " + myThread.isAlive());
    }
}
```

Console

```
terminated> Que9 [Java Application] C:\Users\Shweta\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.8
Before start thread alive? false
After start thread alive? true
Thread started
Thread finished
After join thread alive? false
```

Question 10

Write a Java program to create two threads:

- **Thread 1 prints "Good Morning" 5 times.**
- **Thread 2 prints "Welcome" 5 times.**
  Run both threads simultaneously.

```java
/*Write a Java program to create two threads: • Thread 1 pri
• Thread 2 prints "Welcome" 5 times.
Run both threads simultaneously.
*/
public class Que10{
    static class AThread extends Thread {
        public void run() {
            for (int i = 0; i < 5; i++) {
                System.out.println("Good Morning");
            }
        }
    }
    static class BThread extends Thread {
        public void run() {
            for (int i = 0; i < 5; i++) {
                System.out.println("Welcome");
            }
        }
    }
    public static void main(String[] args) {
        AThread AThread = new AThread();
        BThread BThread = new BThread();
        AThread.start();
        BThread.start();
    }
}
```

Console

```
<terminated> Que10 [Java Application] C:\Users\Shweta\.p2\pool\plugins\org.eclipse.ju
Good Morning
Good Morning
Good Morning
Good Morning
Good Morning
Welcome
Welcome
Welcome
Welcome
Welcome
```

Question 11

Write a Java program where one thread prints even numbers from 2 to 20, and another thread prints odd numbers from 1 to 19.

```java
//Write a Java program where one thread prints even numbers from 2
public class Que11{
    static class EThread extends Thread {
        public void run() {
            for (int i = 2; i <= 20; i += 2) {
                System.out.println("Even := " + i);
            }
        }
    }
    static class OThread extends Thread {
        public void run() {
            for (int i = 1; i <= 19; i += 2) {
                System.out.println("Odd := " + i);
            }
        }
    }
    public static void main(String[] args) {
        EThread EThread = new EThread();
        OThread OThread = new OThread();
        EThread.start();
        OThread.start();
    }
}
```

Console

```
<terminated> Que11 [Java Application] C:\Users\Shweta\.p2\pool\plugins\org.eclipse.justj.openj
Odd := 1
Odd := 3
Odd := 5
Odd := 7
Even := 2
Even := 4
Even := 6
Even := 8
Even := 10
Even := 12
Even := 14
Even := 16
Odd := 9
```

Question 12

Write a Java program to create three threads. Each thread should print its own message 3 times.

```java
public class Que12{
    static class MymsgThread extends Thread {
        private String msg;
        public MymsgThread(String name, String msg) {
            super(name);
            this.msg = msg;
        }
        public void run() {
            for (int i = 0; i < 3; i++) {
                System.out.println(Thread.currentThread().getName() + "= " + msg);
            }
        }
    }
    public static void main(String[] args) {
        MymsgThread thread1 = new MymsgThread("Thread-1", "Hello from Thread 1");
        MymsgThread thread2 = new MymsgThread("Thread-2", "Hii from Thread 2");
        MymsgThread thread3 = new MymsgThread("Thread-3", "hey from Thread 3");
        thread1.start();
        thread2.start();
        thread3.start();
    }
}
```

Console

```
<terminated> Que12 [Java Application] C:\Users\Shweta\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_
Thread-1= Hello from Thread 1
Thread-1= Hello from Thread 1
Thread-1= Hello from Thread 1
Thread-2= Hii from Thread 2
Thread-3= hey from Thread 3
Thread-3= hey from Thread 3
Thread-3= hey from Thread 3
Thread-2= Hii from Thread 2
Thread-2= Hii from Thread 2
```

Question 13

Write a Java program to demonstrate the difference between calling `run()` directly and calling `start()` on a thread.

```java
//Write a Java program to demonstrate the difference between calling run() directly and
public class Que13 {
    static class MyThread extends Thread {
        public void run() {
            System.out.println("thread := " + Thread.currentThread().getName());
        }
    }
    public static void main(String[] args) {
        MyThread myThread = new MyThread();
        System.out.println("Call run()");
        myThread.run();
        System.out.println("Call start()");
        myThread.start();
    }
}
```

Console

```
terminated> Que13 [Java Application] C:\Users\Shweta\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64
Call run()
thread := main
Call start()
thread := Thread-0
```

Question 14

Write a Java program to create a thread that calculates the sum of numbers from 1 to 100.

```java
//Write a Java program to create a thread that calculates the sum of numbers fr
public class Que14 extends Thread {
        private long sum = 0;
        public void run() {
            for (int i = 1; i <= 100; i++) {
                sum += i;}
            System.out.println("Completed");
        }
        public long getSum() {
            return sum;
        }
        public static void main(String[] args) throws InterruptedException {
            Que14 sumThread = new Que14();
            sumThread.start();
            sumThread.join();
            System.out.println("Sum = " + sumThread.getSum());
        }
    }
```

Console

```
<terminated> Que14 [Java Application] C:\Users\Shweta\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.
Completed
Sum = 5050
```

Question 15

Write a Java program to demonstrate how to stop a thread gracefully using a boolean flag instead of the deprecated `stop()` method.

```java
public class Que15 {
    public static void main(String[] args) throws InterruptedException {
        final boolean[] running = {true};
        Thread tThread = new Thread(new Runnable() {
            public void run() {
                System.out.println("Thread started.");
                while (running[0]) {
                    System.out.println("Thread is running...");
                    try {
                        Thread.sleep(500);
                    } catch (InterruptedException e) {
                        System.out.println("Thread was interrupted.");
                        break;
                    }
                }
                System.out.println("Thread stopped");
            }
        });
        tThread.start();
        Thread.sleep(2000);
        System.out.println("thread stop request");
        running[0] = false;
    }
}
```

Console

```
terminated> Que15 [Java Application] C:\Users\Shweta\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_
Thread started.
Thread is running...
Thread is running...
Thread is running...
Thread is running...
thread stop request
Thread stopped
```

Question 16

Write a Java program where one thread prints the

lowercase alphabet (a to z), and another thread prints the uppercase alphabet (A to Z).

```java
//Write a Java program where one thread prints the lowercase alphabet (a to z), and another thread
public class Que16{
    static class Lcase extends Thread {
        public void run() {
            for (char c = 'a'; c <= 'z'; c++) {
                System.out.print(c + " ");
            }
            System.out.println();
        }
    }

    static class Ucase extends Thread {
        public void run() {
            for (char c = 'A'; c <= 'Z'; c++) {
                System.out.print(c + " ");
            }
            System.out.println();
        }
    }
    public static void main(String[] args) {
        Lcase lower = new Lcase();
        Ucase upper = new Ucase();
        lower.start();
        upper.start();
    }
}
```

Console

```
<terminated> Que16 [Java Application] C:\Users\Shweta\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.8.v202507
A B a b c C d e D E F G H I J K L f g h i j M N O P Q R k S T l m n U V W o p q r X s t u v w x Y Z
y z
```

Question 17

Write a Java program to demonstrate how multiple threads can access a shared counter variable. Show the problem of race condition (without synchronization).

```java
public class Que17 {
    static class CntTask implements Runnable {
        private static int cnt = 0;
        public static int getcnt() {
            return cnt;
        }
        public void run() {
            for (int i = 0; i < 1000; i++) {
                cnt++;
            }
        }
    }
    public static void main(String[] args) throws InterruptedException {
        CntTask task1 = new CntTask();
        CntTask task2 = new CntTask();
        Thread thread1 = new Thread(task1);
        Thread thread2 = new Thread(task2);
        thread1.start();
        thread2.start();
        thread1.join();
        thread2.join();
        System.out.println("Expected value = 2000");
        System.out.println("Actual value = " + CntTask.getcnt());
        System.out.println("The actual value is less than 2000 due to the race condition");
    }
}
```

Console

```
<terminated> Que17 [Java Application] C:\Users\Shweta\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.
Expected value = 2000
Actual value = 2000
The actual value is less than 2000 due to the race condition
```

Question 18

Write a Java program to demonstrate synchronization by using the `synchronized` keyword on a method that increments a counter.

```java
public class Que18{
    private static int cnt = 0;
    public synchronized void increment() {
        cnt++;
    }
    public static void main(String[] args) throws InterruptedException {
        Que18 demo = new Que18();
        Thread thread1 = new Thread(() -> {
            for (int i = 0; i < 1000; i++) {
                demo.increment();
            }
        });
        Thread thread2 = new Thread(() -> {
            for (int i = 0; i < 1000; i++) {
                demo.increment();
            }
        });
        thread1.start();
        thread2.start();
        thread1.join();
        thread2.join();
        System.out.println("Expected value = 2000");
        System.out.println("Actual value = " + cnt);
        System.out.println("The actual value is now correct due to synchronization");
    }
}
```

Console

```
<terminated> Que18 [Java Application] C:\Users\Shweta\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.
Expected value = 2000
Actual value = 2000
The actual value is now correct due to synchronization
```

Question 19

Write a Java program to create a thread that prints the current time every 2 seconds, five times.

```java
//Write a Java program to create a thread that prints the current time every 2 seconds
import java.time.LocalTime;
public class Que19 {
    public static void main(String[] args) {
        Thread tThread = new Thread(new Runnable() {
            public void run() {
                for (int i = 0; i < 5; i++) {
                    System.out.println("Current time: " + LocalTime.now());
                    try {
                        Thread.sleep(2000);
                    } catch (InterruptedException e) {
                        System.out.println("Thread interrupted.");
                        break;
                    }
                }
            }
        });
        tThread.start();
    }
}
```

Console

```
<terminated> Que19 [Java Application] C:\Users\Shweta\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_
Current time: 16:08:08.132715800
Current time: 16:08:10.143606500
Current time: 16:08:12.153606200
Current time: 16:08:14.163916300
```

Question 20

Write a Java program where two threads run in parallel:

- **The first thread prints `"Learning Java"` 5 times.**
- **The second thread prints `"Multithreading in action"` 5 times.**

```java
 • The second thread prints "Multithreading in action" 5 times.
*/
public class Que20{
    static class ThreadA implements Runnable {
        public void run() {
            for (int i = 0; i < 5; i++) {
                System.out.println("Learning Java");
            }
        }
    }
    static class ThreadB implements Runnable {
        public void run() {
            for (int i = 0; i < 5; i++) {
                System.out.println("Multithreading in action");
            }
        }
    }
    public static void main(String[] args) {
        Thread thread1 = new Thread(new ThreadA());
        Thread thread2 = new Thread(new ThreadB());
        thread1.start();
        thread2.start();
    }
}
```

Console

```
<terminated> Que20 [Java Application] C:\Users\Shweta\.p2\pool\plugins\org.eclipse.justj.openjd
Learning Java
Learning Java
Learning Java
Learning Java
Learning Java
Multithreading in action
Multithreading in action
Multithreading in action
Multithreading in action
Multithreading in action
```