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.

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.

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.

Question 4

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

Question 5

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

Question 6

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

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.

Question 8

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

Question 9

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

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.

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.

Question 12

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

Question 13

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

Question 14

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

Question 15

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

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`).

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).

Question 18

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

Question 19

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

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.**