Assignment_2/ProducerConsumerLogging.java

```java
import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
import java.time.LocalDateTime;
import java.util.concurrent.*;

//shutdown hook  will be triggered when the application receives a SIGINT (Ctrl+C)

public class ProducerConsumerLogging {

    private static final int BUFFER_CAPACITY = 100;
    private static final BlockingQueue<String> logQueue = new LinkedBlockingQueue<>(BUFFER_CAPACITY);

    private static final String LOG_FILE_PATH = "/Users/Kevin/Code/Design of Operating Systems/Assignment_2/system_logs.txt";

    // Assignment_2/system_logs.txt
    public static void main(String[] args) {
        // Start producer and consumer threads
        ExecutorService executor = Executors.newFixedThreadPool(2);
```

```
20          try {
21              executor.execute(new LogProducer());
22              executor.execute(new LogConsumer());
23
24              // Optional: Shut down gracefully after some time
    (e.g., for demo purposes)
25              Runtime.getRuntime().addShutdownHook(new
    Thread(() -> {
26                  executor.shutdownNow();
27                  System.err.println("Shutdown initiated. Logs
    flushed.");
28              }));
29          } finally {
30              executor.shutdown();
31          }
32      }
33
34      static class LogProducer implements Runnable {
35          @Override
36          public void run() {
37              try {
38                  int logCounter = 1;
39                  while
    (!Thread.currentThread().isInterrupted()) {
40                      String logMessage = String.format(
41                              "[%s] INFO: Log message #%d",
```

```
42                              LocalDateTime.now(),
logCounter++);
43                      logQueue.put(logMessage); // blocks if
queue is full
44                      Thread.sleep(500); // simulate log
generation rate
45                  }
46              } catch (InterruptedException e) {
47                  Thread.currentThread().interrupt();
48                  System.err.println("LogProducer
interrupted");
49              }
50          }
51      }
52
53      static class LogConsumer implements Runnable {
54          @Override
55          public void run() {
56
57              try (BufferedWriter writer = new
BufferedWriter(new FileWriter(LOG_FILE_PATH, true))) {
58                  while
(!Thread.currentThread().isInterrupted()) {
59                      String log = logQueue.take(); // blocks
if queue is empty
```

```
60                      writer.write(log);
61                      writer.newLine();
62                      writer.flush(); // force disk write for
    every line (could be buffered in practice)
63                  }
64              } catch (InterruptedException e) {
65                  Thread.currentThread().interrupt();
66                  System.err.println("LogConsumer
    interrupted");
67              } catch (IOException e) {
68                  System.err.println("Error writing logs to
    file: " + e.getMessage());
69              }
70          }
71      }
72  }
73
74  /*
75   * What are the components of OS? The OS is made up of the
    kernel and system
76   * programs. The OS provides services via system calls,
    command line
77   * interpreter, and gui interfaces
78   * Why are system calls in OS? To complete some operation
    that the operating
```

```
79   * system controls. To provide an interface to the services
     made available by an
80   * operating system.
81   */
```