```java
import java.io.*;
import java.net.Socket;
import java.nio.charset.StandardCharsets;
import java.util.zip.CRC32;
import java.util.zip.Checksum;
import java.util.zip.GZIPOutputStream;
import java.util.Base64;

public class client {
    public static void main(String[] args) {
        try {
            System.out.println("Connecting to the server...");

            // Create a socket and connect to the server on localhost, port 8080
            Socket socket = new Socket("localhost", 8080);

            System.out.println("Connected to the server.");

            // Create output stream for communication with the server
            ObjectOutputStream objectOutputStream = new ObjectOutputStream(
                    new GZIPOutputStream(socket.getOutputStream()));

            // Specify the folder path on the client side
            String desktopPath = System.getProperty("user.home") + "/Desktop" + "/Files";
            File folder = new File(desktopPath);

            // List files in the folder
            File[] files = folder.listFiles();
            if (files == null) {
                System.err.println("No files found in the folder: " + desktopPath);
                return;
```

```
32              }
33
34              // Send the number of files to expect
35              objectOutputStream.writeInt(files.length
   );
36              objectOutputStream.flush();
37
38              // Send up to 20 files at a time
39              int filesToSend = Math.min(20, files.
   length);
40              for (int i = 0; i < filesToSend; i++) {
41                  File file = files[i];
42
43                  // Read the file content
44                  String fileContent = readFileContent(
   file);
45
46                  // Compute checksum for the content
47                  long checksum = computeChecksum(
   fileContent);
48
49                  // Send the content and checksum to
   the server
50                  byte[] compressedContent = compress(
   fileContent);
51                  objectOutputStream.writeObject(
   compressedContent);
52                  objectOutputStream.writeLong(checksum
   );
53                  objectOutputStream.flush();
54              }
55
56              // Close the streams and socket
57              objectOutputStream.close();
58              socket.close();
59
60          } catch (IOException e) {
61              System.err.println("Connection failed.
   Make sure the server is running and check your
   firewall settings.");
62              e.printStackTrace();
```

```
63          }
64      }
65
66      private static byte[] compress(String content)
    throws IOException {
67          ByteArrayOutputStream byteArrayOutputStream
     = new ByteArrayOutputStream();
68          try (GZIPOutputStream gzipOutputStream = new
     GZIPOutputStream(byteArrayOutputStream)) {
69              gzipOutputStream.write(content.getBytes(
    StandardCharsets.UTF_8));
70          }
71          return byteArrayOutputStream.toByteArray();
72      }
73
74      private static String readFileContent(File file
    ) throws IOException {
75          StringBuilder fileContent = new
    StringBuilder();
76          try (BufferedReader fileReader = new
    BufferedReader(new FileReader(file))) {
77              String line;
78              while ((line = fileReader.readLine
    ()) != null) {
79                  fileContent.append(line).append("\n"
    );
80              }
81          }
82          return fileContent.toString();
83      }
84
85      private static long computeChecksum(String data
    ) {
86          Checksum checksum = new CRC32();
87          checksum.update(data.getBytes(
    StandardCharsets.UTF_8), 0, data.length());
88          return checksum.getValue();
89      }
90 }
91
```