# 📚 COP4610 Assignment 2: Multi-Threaded TCP Shell Server - Testing Guide

## 🔄 How to Use the Provided Sample Codes - *for manual tesing*:

1. **Understand** very thoroughly the behavior and request/response patterns of the instructor-provided sample executables: `sample_client` and `sample_server`. **Grant executable permissions to these files if not set already:**

   🛠️ **Granting Permissions**

   ```
   # Check current file permissions
   ls -l <filename>

   # If you see:
   # -rw-------  OR encounter "Permission denied" on execution

   # Run the following commands to grant permissions:
   chmod +rwx sample_client
   chmod +rwx sample_server

   # Always run the server first in one terminal
   ./sample_server

   # Next, run the client in another terminal
   ./sample_client
   ```

   🔍 **Key behaviors to observe:**

   - Server startup messages and connection handling
   - Client connection and command prompt
   - Command execution and output formatting
   - HELP command response
   - EXIT command behavior
   - Server handling multiple clients simultaneously
   - Server/Client graceful shutdown with Ctrl+C

## 🤖 How to use the Autograder

The autograder script is available to facilitate automated testing throughout your development process **at any stage**. A correct implementation will earn **90 out of 100 points** through the autograder. The remaining **10 points** will be awarded based on:

- Adherence to submission guidelines
- Code structure and quality
- Code documentation

*** Please note that instructor will use the same autograder for the final grading.**

## 1. Required Files and Directory Structure

Ensure that the following files are located in the same directory:

```
Assignment_2/
├── shell_server.h          # Header file (provided - DO NOT MODIFY)
├── autograder_socket.sh    # Autograder (provided - DO NOT MODIFY)
├── client.c                # Student client implementation
└── server.c                # Student server implementation
```

## 2. Executing the Autograder

Run the autograder script using the following command:

```
# Run autograder
./autograder_socket.sh
```

The script will compile your code, run the test cases, and provide a detailed score report.

---

# 🤖 How to use the Batch-autograder

The batch autograder, **used by the instructor**, processes all student submissions at once. It utilizes the provided framework files, extracts the required files from your submitted `.zip` file, and performs grading accordingly.

⚠️ ****Backup Files Before Running Batch Grader**. The script deletes existing files to rebuild the environment from scratch.

## 1. Required Files and Directory Structure

Ensure that the following files are located in the same directory:

```
Assignment_2/
├── shell_server.h           # Header file (provided - DO NOT MODIFY)
├── autograder_socket.sh     # Autograder (provided - DO NOT MODIFY)
├── batchgrader_socket.sh    # Batch-autograder (provided - DO NOT MODIFY)
├── client.c                 # Student client implementation
└── server.c                 # Student server implementation
```

## 2. Executing the Batch-autograder

Run the autograder script using the following command:

```
# Run batch-autograder
./ batchgrader_socket.sh
```

---