Software Engineering – Assignment #1

Release Date: 12 September 2025

Due Date: 11:59 PM, 3 October 2025

In this assignment, you will produce 3 Git repositories and 2 OCI-compliant (Docker) images. Each sub-assignment is worth 20 points.

1 Git Assignments

1.1 Branching for a Feature

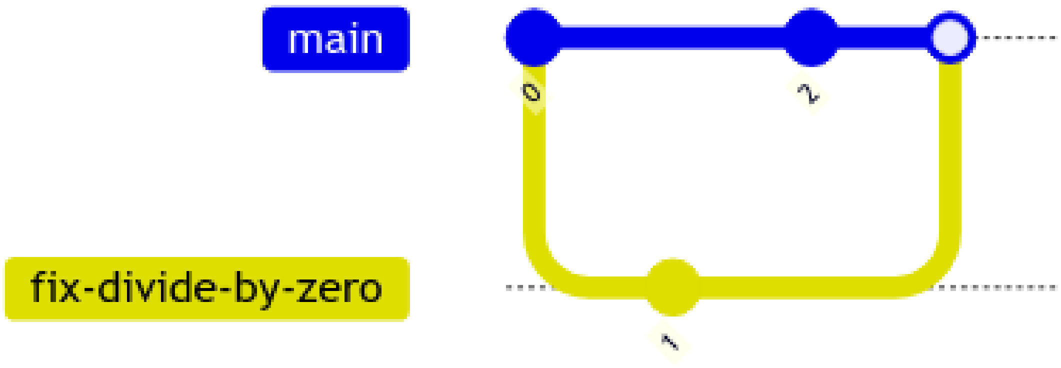
In the directory `./ex01`, you will find a simple calculator implemented in C. There are two problems with this program:

1. this program only accepts integers, but this should be able to handle real numbers; and
2. the program will crash when I try to divide by zero, but I want it to terminate gracefully (i.e. the `main` function should return zero) with an error message.

Your assignment is to complete the following:

1. Create a branch called `fix-divide-by-zero`, and fix the problem #2 there.
2. Checkout the `main` branch and fix the problem #1 there.
3. Then, merge `fix-divide-by-zero` into `main`, creating a merge commit.

By the end of the assignment, the Git commit history should match the one shown below:



1.2 Fixing a Mistake

In the directory `./ex02`, you will find a program that greets you, written in JavaScript. However, there is a typo in the last commit.

Amend the commit to address the typo, so that the `greet` function properly greets the user. There should be no additional commit created in the Git history.

1.3 Cherry-picking for Changes

In the directory `./ex03`, you will find an API server written in Python FastAPI framework. This repository contains two branches: main and develop. In develop branch, you can find new API endpoints implemented in a commit.

You, as the backend developer, is asked to ship some features as soon as possible. From the commits in develop branch, **cherry-pick the `ping` feature and the `greet` feature** into main branch. Do not squash commits.

2 Containerisation Assignments

2.1 Introduction to Containerisation

Write a program (in any language) that receives input from the standard input and outputs something to the standard output. Then, package the program in an OCI image so that running a container will automatically run the program you wrote. Provide three (3) example inputs and expected outputs for each input as text files.

Your input/output must not produce error when the following POSIX shell command is executed:

```
cat input.txt \
    | docker run --interactive --rm <your-container>:latest \
    | diff -q - output.txt
```

Upload your image to Docker Hub and turn in the image name.

2.2 Producing Production-ready Containers

In the directory `./ex05`, you will find a minimal HTTP server written in C. To compile this program, you need `GNU libmicrohttpd` library and its development headers.

Package this program into an OCI image so that running a container will automatically run the web server provided. Note that the included `Makefile` already contains the valid compilation command.

Upload your image to Docker Hub and turn in the image name.

Hint: In Debian-based distributions such as Debian and Ubuntu, the development package for libmicrohttpd is called `libmicrohttpd-dev`.

3 Submission Guidelines

Submit 1 gzipped tarball that contains the following:

1. Unmodified Git directories for sub-assignment 1, 2, and 3
2. Docker Hub image URL for sub-assigment 4 and 5
3. Three pairs of input and expected output text files
4. `metadata.json` file that contains the information above

For your convenience, an interactive archive generator is included for you. Run `./submit.sh` shell script to generate the submission archive. The generation process should look something like:

```
$ ./submit.sh
Enter your student ID: 1234567890
Enter your full name (as on LearnUs): John Doe
---- ex01 ----
Enter the path to your git repository: ./ex01
...
---- ex02 ----
Enter the path to your git repository: ./ex02
...
---- ex03 ----
Enter the path to your git repository: ./ex03
...
---- ex04 ----
Enter your Docker Hub username: johndoe
Enter your Docker Hub repository name: ex04
Confirm that your image is available as docker.io/johndoe/ex04:latest (y/n): y
Enter the path to input file #0: ./ex04/input1.txt
Enter the path to expected output file #0: ./ex04/output1.txt
Enter the path to input file #1: ex04/input2.txt
Enter the path to expected output file #1: ex04/output2.txt
Enter the path to input file #2: ex04/input3.txt
Enter the path to expected output file #2: ex04/output3.txt
---- ex05 ----
Enter your Docker Hub username: johndoe
Enter your Docker Hub repository name: ex05
Confirm that your image is available as docker.io/johndoe/ex05:latest (y/n): y
Creating final submission tarball...
...
Removing intermediate files...
...
Submission tarball created: submit_1234567890.tar.gz
Please upload this file to LearnUs before the deadline.
```