

# CSE 489/589 Programming Assignment 1

## Instructions for installing/using the Template

### What is this document about?

This page provides instructions for the template of Programming Assignment 1.

### Do I need to use this template?

**YES.** It is mandatory to use this template.

### What is the template?

The template is a collection of a basic directory structure along with some source code and scripts/programs that are required for you complete PA1.

### Why a template?

In the past offerings of this course, students lost points because they failed to follow certain naming/submission conventions that we specify in the assignment description and expect while grading. The template exists to avoid such cases.

### Can I just use parts of the template (and do the rest manually)?

**NO.** If you use scripts/programs not included in the template (or do things manually), we cannot guarantee that your submission will adhere to the PA1 specification and hence may result in the loss of points.

### What Operating Systems (OS) will this work on?

We have tested the scripts on Linux based OSes only. They will NOT run natively on Windows. Although there is no reason for them to not work on environments that try to emulate Linux on Windows (e.g. cygwin), we will not be able to provide you support for them. Same goes for Mac OS X. However, feel free to post questions about this on the course blog. Other students facing the same issue might be able to help you.

### What if I have questions about this template?

Piazza is the central place to ask questions for everything related to the course, including this template. Make sure to use the correct category for PA1 when asking questions about this template to ensure a quick response.

### Will this work on my personal linux machine?

Most likely, yes. However, given there are so many different flavors of Linux and configurations available, we cannot test on each one.

## Installation

→ Create a new directory/folder to house the assignment. For the purpose of this assignment, we will use the name `cse489589_assignment1`. You should create this folder directly inside the directory mentioned in section 3.4 at <https://goo.gl/bqf2E1> on one of the dedicated hosts, and NOT at any other path or sub-directory.

```
$ mkdir cse489589_assignment1
```

→ Change into the assignment directory.

```
$ cd cse489589_assignment1
```

→ Download the starter script.

```
$ wget --no-check-certificate https://ubwins.cse.buffalo.edu/cse-489_589/assignment1_init_script.sh
```

→ Give the starter script execute permission.

```
$ chmod +x assignment1_init_script.sh
```

→ Run the starter script

◆ This needs to be run only once. Running the script again can potentially overwrite your existing work.

```
$ ./assignment1_init_script.sh
```

→ The script will prompt you for some answers.

- ◆ Enter your UBIT username (without the @buffalo.edu part).
- ◆ Enter your Full name.
- ◆ Choose the programming language: C or C++.
- ◆ Confirm the details.
- ◆ Wait for script to finish; On success, it will display a message.

→ When the script exits (successfully)

- ◆ You should see a directory structure, with some scripts and a folder created in the current directory.
- ◆ If the script fails for some reason, it will try to give you a hint about the possible reason. Fix the problem, delete everything inside the current directory (except the startup script itself) and try again.

*The instructions above are meant to be executed once, before you start working on the assignment. Once the template is in place, you should not have to execute any of the steps mentioned above, again.*

## About

→ The script will create a directory named after the UBIT username supplied in the earlier step. This directory is where all source code and Makefile will reside.

→ The directory already contains the following folders/files:

- ◆ src [folder]
  - <your-UBIT-name>\_assignment1.c / <your-UBIT-name>\_assignment1.cpp
  - logger.c / logger.cpp
- ◆ include [folder]
  - global.h
  - logger.h
- ◆ object [folder]
- ◆ logs [folder]
- ◆ Makefile [file]

→ You should add any new source (\*.c/\*.cpp) files that you need inside the src folder only. If you keep any new \*.c/\*.cpp files outside this, they will not be picked up by the Makefile.

→ You should add any new header (\*.h) files that you need inside the include folder only. If you keep any new \*.h files outside this, they will not be picked up by the Makefile.

→ You should NOT modify the logger.c / logger.cpp, logger.h, or the Makefile, in any way. You will be assigned a zero (0) grade if any of these files are modified.

→ You can declare constants in global.h and include it later in your source files.

## Usage

→ For the template to work and to avoid any naming convention mistakes, **you should NOT modify** the *Name and/or Paths* of any of the directories/folders/files that are included in the template. You are free to add new files, in accordance with the instructions above.

→ You can of course add your code inside the `<your-UBIT-name>_assignment1.c` /`<your-UBIT-name>_assignment1.cpp` file included below the `/*Start Here*/` comment line. Do NOT change the code lines above that (it is needed for the logger to work correctly). Further, the grader expects the `main()` function to reside in the file named `<your-UBIT-name>_assignment1.c` /`<your-UBIT-name>_assignment1.cpp`. You should not modify the the first line of the `main()` function in any way.

→ Building your program.

- ◆ Change into the directory named after you UBIT name.

```
$ cd <your-UBIT-name>
```

- ◆ Build the program

```
$ make
```

- ◆ This will create an executable named *assignment1* in the same directory.
- ◆ To clean up the executable and object files

```
$ make clean
```

→ Adding a new source (`.c/.cpp`) or a header (`.h`) file

- ◆ In case of a source (`.c/.cpp`) file, add it inside the *src* folder
- ◆ In case of a header (`.h`) file, add it inside the *include* folder.

## Packaging and Submission

→ Packaging your code for submission

- ◆ Change to `cse489589_assignment1` directory, then execute

```
$ ./assignment1_package.sh
```

- ◆ This will create a tarball, named as `<your-UBIT-name>_pa1.tar`, in the same directory.

**Note that this step does NOT submit your assignment, just packages it.**

**If you are a UB student taking CSE 4/589, you need to use the `submit_cse489/submit_cse589` (available on CSE servers) script to submit this tarball.**

→ Submission (if you are a UB student taking CSE 4/589)

- ◆ Make a submission on a CSE server using the `submit_cse489/submit_cse589` script (see <https://wiki.cse.buffalo.edu/services/content/submit-script>).
- ◆ Take a snapshot as a proof, along with the `$ date` command immediately following the submission. See PA1 recitation slides for an example.

## Grader

→ The automated grader expects the output to adhere to the format/syntax described in the assignment description document.

→ We are providing the same grader to you so that you can verify the correct execution of your code before submission.

→ The grader can run on any machine (e.g., your personal laptop).

→ The grader resides in the outermost directory inside a folder named **grader**.

→ The grader reads configuration from a configuration file located in **grader/grader.cfg**. The configuration details will be provided to you by the course staff.

If you are a UB student taking CSE 4/589, do not modify this file except for adding the port number which will be provided to you by the instructor.

→ To run the grader

- ◆ Change to the cse489589\_assignment1/grader directory, then execute

**\$ ./grader\_controller -h**

- ◆ The -c option requires the path to the grader.cfg file.
- ◆ The -s option requires the path to the tarball created by the packaging script.
- ◆ The -t option supports the following tests:

- startup
- author
- ip
- port
- \_list
- refresh
- send
- broadcast
- block
- blocked
- unblock
- logout
- buffer
- exit
- statistics
- exception\_login
- exception\_send
- exception\_block
- exception\_unblock
- exception\_blocked
- bonus

The 15 points for LOGIN in the Grading Rubric come from the sum of points from the grader tests "\_list" and "buffer".

- ◆ Requires: Python 2.7.x

→ To update the grader

- ◆ Change to cse489589\_assignment1 directory, then execute

**\$ ./assignment1\_update\_grader.sh**

---

Published by [Google Drive](#) – [Report Abuse](#) – Updated automatically every 5 minutes

---