CS 215 - Fundamentals of Programming II
Fall 2019 – Using GitKeeper for Submitting Assignments

While there are a plethora of software development environments for the C++ programming language that are implemented on many different platforms, for consistency and experience, programming assignments in CS 215 are to be completed using the C++ compiler (g++ or clang++) in a UNIX environment using the C++11 standard. In addition, there are several source control systems. In this course, the git source control system will be introduced for use in managing personal projects and submitting assignments.

Compiler and Compilation

Programming assignments must compile on the submission system server (Ubuntu 18.04LTS Linux) using the clang++ compiler using the C++11 standard for minimal credit to be given for the project. This does not mean that some (or even all) of the programming assignment must be developed under Linux. Those of you that have other development systems can complete some or all of an assignment on that system. However, when the program is turned in, it must compile using clang++ without any intervention from the instructor. A program that works "perfectly" using, say, the Visual Studio compiler will fail during the automated submission if it does not compile in a UNIX environment using clang++. (Also note that the lab practical exams will require that programs be written using the tools in a UNIX environment.)

All source files will use the following conventional suffix indication:

`.cpp`	user-defined C++ source files
`.h`	user-defined C++ header files

Common compiler options include:

`-o <name>`	creates executable file of given name; default is `a.out` otherwise
`-c`	compile only; creates .o object file(s) of same name(s) as source file(s)
`-std=c++11`	compile to the C++11 standard
`-Wall`	compile with all warnings turned on; this is required to produce the most conforming code possible
`-g`	compile for symbolic debugging

Generally, you will submit a makefile in addition to source and header files that will invoke the compiler appropriately. (Makefiles are covered in class.) Note: when you do run the compiler from the command line, **be very careful that you put the executable name after the -o option**; g++ will overwrite the first file after the -o. It is recommended that this option be listed after the source files.

GitKeeper Submission System

This course uses the GitKeeper submission system. This is a submission system based on the git version control system described in the *An Itty Bit of Git* handout. Assignments are distributed as git repositories that a student clones from the GitKeeper server. To submit an assignment, a student pushes her repository. The GitKeeper server replaces the normal push actions by creating a temporary repository with the submitted code, performing instructor specific grading actions, capturing the results, and emailing these results to the student. Typically, the grading actions will be to compile and run the submission on various test cases, capturing the output of your program (including user prompts). This output is then compared to the expected output as given in an assignment. If the output matches exactly, the system reports success. If it does not match exactly, the system reports failure.

Students are added to the GitKeeper system for a particular class by an instructor. Students will receive an email giving their username and (randomly generated) password on the GitKeeper server (gitkeeper.evansville.edu), if they do not already have one. Logins to the GitKeeper server using `ssh` are allowed for the purposes of changing the password on the account (command `passwd`), but most other commands are disabled. In particular, student accounts are not allowed to list or change directories.

For each assignment, the following steps will happen:

1. When the instructor publishes an assignment, the student will receive an email with the URL to clone for the assignment of the form

   ```
   user@gitkeeper.evansville.edu:/home/user/instr/class/assign.git
   ```

   This email also may contain specific instructions or information pertaining to the assignment.

2. To do the assignment, first change to the directory where the assignment is to be stored, then clone the repository by using the URL given in the email

   ```
   $ git clone user@gitkeeper.evansville.edu:/home/user/instr/class/assign.git
   ```

   Respond with the GitKeeper password from the original account creation email (or whatever it was changed it to).

3. The resulting local repository will be in subdirectory `assign`. Change to this directory and complete the assignment. Be sure to commit changes to the local repository often.

   ```
   $ git add .
   $ git commit -m "finished some function"
   ```

4. To submit an assignment, the local repository must be clean (i.e. all changes committed), then push the repository back to the server. If the repository is not clean, the push will fail by saying everything is up to date; commit the changes and try again.

   ```
   $ git push
   ```

   As noted above, the GitKeeper server will compile, run, and test the submission, and send an email with the result.

The result emails will contain the following information:

1. Whether the assignment compiled without errors. If it does not, the grading script stops, and announces a failure.
2. Otherwise, the time of submission will be given. If the submission is after the due date, it will also state the number of **calendar** days the assignment is late. The actual late penalty is determined by the instructor.
3. The results of one or more tests.
4. Whether the assignment passed or failed the auto-grading. Please note that correct results from the auto-grading is only part of the total score for an assignment.

Assignments may be submitted multiple times, so if a submission fails the auto-grading, fix the errors and try again. Only the last submission will be graded. Submissions after 3 “days” late may or may not be graded at the discretion of the instructor.