McGill University
School of Computer Science

COMP-206

Mini Assignment #7

Due: April 14, 2019 on myCourses at 23:30

Borrowing Wikipedia's definition: *Common Gateway Interface (CGI) offers a standard protocol for web servers to execute programs that execute like console applications (also called command-line interface programs) running on a server that generates web pages dynamically.*

This last assignment is about creating a simple CGI program that you can invoke remotely through a webpage.

Your mission is to create a C program that converts an int number to its binary representation. This program will run as a server application and will be printing out the results in a dynamically generated html page.
We essentially need 2 entities to be able to expose our program to the outside world:

- an html page that will play the role of an interface between the user and your program
- a c program that implements the logic of converting a number to binary

These 2 entities should reside in a folder called *public_html* that is placed under your *home* directory on *mimi*.

Please carry out the following steps in order to complete your assignment successfully:

1. Create *public_html* directory and grant all users reading and executing rights to it.

2. cd into *public_html* directory and:

    a. create a C file called **binary_converter.c**. Your C program will receive an int number through CGI GET method and will convert this number into its binary representation and send the result back to the browser.

    b. Create an html file called **index.html**. The html file should implement a simple form using GET method. The form will have a label field "Enter an int number", followed by an input field then a submit button called "convert to binary". When this button is clicked, the binary_converter program should run on the server and the result will be printed out to the browser.
    To test your index.html form, open your favorite browser and type the following url (replace UserName by your actual McGill username):
    https://www.cs.mcgill.ca/~UserName/

3. Now let's organize things using a Makefile. Create a makefile that will perform the following operations:
    a. If the user types "make project" the makefile should:
        i. create a new project directory under your home directory called public_html (if it does not already exist)
        ii. copy binary_converter.c and index.html files from your home directory to public_html directory (for this scenario you need to have copies of these files placed under your home folder because public_html folder will be created by the makefile)
        iii. grant all reading and executing rights to the public_html directory
        iv. grant all reading and executing rights to the index.html file

    b. If the user types "make add" the makefile should:
        i. initialize git under public_html directory
        ii. add the binary_converter.c and index.html to the git repository
        iii. commit binary_converter.c and index.html to the git repository with any suitable log message

    c. If the user types "make compile" the makefile will:
        i. compile the c file binary_converter.c and creates a binary called binary_converter.cgi
        ii. grant all reading and executing rights to the generated executable: binary_converter.cgi

    d. If the user only types "make", the makefile will perform all of the above in exactly the same order

4. When you finish your project make sure that the last modifications to your source code file and html file are committed to the git repo (you can run “make add” to achieve this). It is advisable to run the commit command after every modification to your source code to keep track of all the changes that you have done during the development of your project. At the end, use the git log command and output the log results to a file that you call *git_log.txt*.

WHAT TO HAND IN

Everything must be submitted to My Courses before the due date. Remember that you can hand in your assignment up to two days late but there will be a penalty of 5% each day. After that, your assignment will not be accepted. Please hand in the following:

- A single zip file having: **the C file, the html file, the makefile and your git log output written to a file called git_log.txt**

HOW IT WILL BE GRADED

The assignment is worth a total of 20 points.

- 20 points are distributed as follow:
  - 6 points for the C program converting successfully a number to binary representation
  - 4 points for dealing properly with the GET method from within the main function
  - 8 points for the makefile
    - 2 points for make project
    - 2 point for make add
    - 1 point for make compile
    - 2 point for the default make
  - 2 points for the html file
  - 1 point for the log file

GRADING RULES

The following rules are followed by the TA when grading assignments:

- A program must run in order to get a grade (even if it does not run well). If it does not run (does not compile) it will receive a zero. (Make sure to run your programs from Trottier – they sometimes do not run the same from home when logging in using putty.)
- The TA will grade using the mimi.cs.mcgill.ca server.
- All questions are graded proportionally (assuming the program runs at all). This means that if 40% of the question is correct, you will receive 40% of the grade.
- The TA will run the makefile and observe the execution for each case.
- The TA will check whether the c file is compiling without errors and the executable is created.
- The TA will check whether git was being initialized and the files added to the repo.
- The TA will check whether the public_html folder is being created and its content populated with the proper access.
- The TA will check whether the index.html file is accessible over the internet and displays fine in the browser.

- The TA will test the results of the converter functions by entering values through the browser and observing the output.
- The TA will check if the log file is provided and if it contains the results of git log command showing the history of your changes.