Program #3
Due: Tuesday Feb 20th, 2018 at 11:59:00 pm

Instructor	Dr. Stephen Perkins
Office Location	ECSS 4.702
Office Phone	(972) 883-3891
Email Address	stephen.perkins@utdallas.edu
Office Hours	Tuesday and Thursday 1:30pm – 4:00pm and by appointment

TAs

CS/SE 3377.002 - Vatsalkumar Patel - drp140230@utdallas.edu

CS/SE 3377.501 - Alexander Lee Hayes - alh170730@utdallas.edu

CS/SE 3377.502 – Subrahmanyam Oruganti – osx160430@utdallas.edu

Purpose

Demonstrate the ability to utilize the UNIX ***make*** build system. Demonstrate the ability to clone code from a GIT repository. Demonstrate the ability to build and install source distributed in an *autoconf* format. Demonstrate the ability to call code not in the standard UNIX path. Demonstrate the ability to grab shell output from within C++ code. Demonstrate the ability to utilize both C and C++ program interfaces within the same code. Demonstrate the ability to utilize both C++ programs and standard UNIX utilities (gawk) to perform work.

Assignment

Overview:

You are to download/build/install a very specific version of the program ***gawk***. Then, using your just installed gawk, you are to write a simple ***gawk*** program that will calculate and output the sum of some specific columns in a table that is maintained in a file. Finally, you are to create a C++ program that calls your specific gawk to perform its work. Your C++ program must capture the results of the work (the column sums) and display that info. Then, it must add up all the Colum sums and provide a total sum. You must use a Makefile to compile your program.

Specifics:

- You are to use the ***git*** source repository tool to clone the source of the latest version of gawk. This will create a folder that has the source code to gawk version 4.1.60 (the latest version is a moving target). The latest version is not the version we will use. You are then to use ***git*** to revert the version from 4.1.60 to an older version of 3.1.8. (See notes below)

- You are to create the Makefile for building gawk by utilizing the ***autoconf*** tools. (see notes below).

- You are then to compile the sources and install the binaries using ***make***. (see below)

- You are to validate that you can run your version of gawk and display version 3.1.8.

- You are to write a ***gawk*** program that will parse a file that contains 6 columns of numbers. The ***gawk*** program should sum up column 1 and column 4 and return both numbers on a single line (just 2 integers and nothing else). Your ***gawk*** program should be stored in a file named **gawk.code** and the table of numbers should be stored in a file called **numbers.txt**. You create your own numbers table. See the Example numbers.txt file and the example ***gawk*** output below. From the shell, you may call ***gawk*** with this command (using the gawk you compiled):
  ```
  <full path to gawk's bin folder>/gawk –f gawk.code numbers.txt
  ```
  See the man page for gawk to get the details of the –f flag (and more)

- You are to write a C++ program that will call your gawk program and collect/display the results of the gawk output. The first call to ***gawk*** should pass in the –version option. You should print out the results to screen and it should display version 3.1.8. The second call to ***gawk*** should call the program stored in **gawk.code** that operates on the **number.txt** file. This call will return the 2 integers representing the column sums. From within your C++ program, you are to collect the results of the call and print these integer values to the standard ouput. You are then to add the two numbers together and print out the sum of the two numbers to standard output.
  - You will need to use the *popen/pclose* functions to perform your shell call to gawk and to gather the results back into your C++ program. This is a C interface in Unix… so make sure you realize you are not working with C++ objects. See "man 3 popen" for details (the 3 means section three of the manual).
  - For getting numbers out of C-style strings (the results of the gawk command), you might want to use the C++ *stringstream* type. You will need to google how to do this.

  For debugging purposes, you should also print the strings that are going to be used with *popen/pclose.* See below for example output.

- You must use ***make*** to compile your code. You must create a Makefile that will create your executable by default. You also need to create a target named "clean" that will remove object files and any other files that are not necessary. You should execute 'make clean' before you crate your tar file for submission.

numbers.txt

Your numbers.txt file should contain 6 columns of numbers and at least 4 rows (you can have as many more rows/columns as you like). Each number is separated by a space. For example:

```
1 2 3 4 5 6
11 22 33 44 55 66
2 3 4 5 6 7
111 222 333 444 555 666
```

gawk output

Your gawk program should only output two numbers (they should be on a single line). The numbers should represent the sum of numbers in column 1 and column 4. As an example with the table above:

```
125 497
```

C++ output

Your C++ program output should look like this:

```
gawk at: /people/cs/s/sxp127930/UTDallas/Courses/Spring2015/Programs/Program3/bin/gawk
Shellcmd1: /people/cs/s/sxp127930/UTDallas/Courses/Spring2015/Programs/Program3/bin/gawk  --version
Shellcmd2: /people/cs/s/sxp127930/UTDallas/Courses/Spring2015/Programs/Program3/bin/gawk  -f gawk.code numbers.txt

The first call to gawk returned:

GNU Awk 3.1.8
Copyright (C) 1989, 1991-2010 Free Software Foundation.

This program is free software; you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation; either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program. If not, see http://www.gnu.org/licenses/.

The second call to gawk returned: 125 497

The sum of Column 1 is: 125
The sum of Column 4 is: 497
The Sum of the two numbers is: 622
```

Deliverables

You must submit your homework through ELearning. You must include a tarball that contains your Makefile, your .cc files, your .h files (if any), your gawk.code file, your numbers.txt file. All source files and Makefiles need to have your name, email, and course number commented at the top. You also need to capture the output of your C++ program by redirecting it to a file. You must submit this output file as well.

DO NOT submit the gawk subfolders as part of your tarball (the gawk, bin, libexec, and share). You will need to use the **---exclude** flag for tar (see the man page). You might consider making a target in your Makefile called 'tarball' that will perform a clean and then a tar command that includes the **--exclude** flags.

This MUST compile and run on cs1.utdallas.edu

Getting the source for GAWK version 3.1.8

See the ***Notes*** section about typing the commands below. We will be covering source code management (SCM) tools in detail later on. For now, this exercise is intended to get you familiar with some of their capabilities. We will be downloading the source code for GNU's version of awk which is called ***gawk*** (see the man page for ***gawk***). The particular SCM tool we will use is called ***git***. ***git*** will allow us to get a complete copy of the source directory for a large and well established project and place that copy in a folder on our system.

To get the copy (called a clone), you need to perform these steps from the shell:

1. Create a new folder for this homework assignment and change your current working directory to be this new folder:
   ```
   mkdir Program3
   cd Program3
   ```

2. Execute the ***pwd*** command and note the full path for your Program3 folder. You will need it later. Then, execute the following ***git*** command:
   ```
   git clone git://git.savannah.gnu.org/gawk.git
   ```

3. Command #2 will create a folder in your current folder called gawk. Change your working directory into this directory:
   ```
   cd gawk
   ```

4. By default, the above command gave you the most recent version of the gawk source code. However, I want you to use a very specific older version of gawk. So, we need to issue a command to git to tell it to give us a specific code revision (in this case 3.1.8). This command will print out a list of all the revisions (each revision has a tag with its name).
   ```
   git tag -l
   ```

5. Now, we need to tell ***git*** to use a specific tag:
   ```
   git checkout tags/gawk-3.1.8
   ```

6. When complete, you have a folder containing the uncompiled source code for the gawk program version 3.1.8

Compiling the source for GAWK version 3.1.8

The source code for gawk is large and complex. Gawk is designed to run on almost any UNIX system. Because there are so many UNIX systems out there, the developers have decided to use a set of GNU tools called ***autoconf*** to help make sure the code works on any UNIX system. While you do not need to worry about the details of ***autoconf*** , you do need to know how to compile project that are created with this toolset (many are). Here are the steps:

7. Determine where you want the executables to be installed. For this assignment, you use the FULL path to the folder you created in step 1.

8. While you are still in the gawk folder, execute this command to create the Makefile you need to compile gawk:

   ```
   ./configure --prefix="<FULL PATH FROM STEP1>"
   ```

   As an example using my account (you must use your own):

   ```
   ./configure --prefix="/people/cs/s/sxp127930/UTDallas/Courses/Spring2015/Programs/Program3"
   ```

9. Once this command finished (it will generate a lot of output), you should have a Makefile. You may now issue this command to compile all the files and create the gawk executable:

   ```
   make
   ```

10. Once you successfully build the executable, you need to tell the system to install it. This will typically install things like the executable, the man pages, support files. You can do this via:

    ```
    make install
    ```

11. If successful, you will see new folders called *bin, libexec, share* in the folder you created in step #1 above.

12. To test, change directories up one level to get out of the gawk subfolder:

    ```
    cd ..
    ```

    This should put you back into the folder you created in step 1. Issue this command:

    ```
    ./bin/gawk –version
    ```

    Verify it shows version 3.1.8. If so, you have successfully downloaded, compiled, and installed a version of gawk that is different than the one normally found on the system.

Notes

- Unlike Windows, UNIX does not work all that well with file names and folder names that contain spaces. When you create folders and files, it is best to utilize _ characters in place of spaces (or just keep file/folder names with no spaces). If you run into commands that do not work properly, make sure your path does not have folders with a space in the name.

- Sometimes PDF files contain hidden symbols and characters. If you "copy/paste" commands out of a PDF file and into a shell, they shell command may have invisible characters and may fail to run. To avoid this issue, please manually type any commands listed in this document.

- No late homework is accepted.

See class notes for more details!