The objective of this assignment is for you to set up the infrastructure you need for subsequent assignments and to write and execute basic MIPS and C programs. Help is available through **piazza**, during **TA office hours** (posted on Canvas syllabus page) and by sending **email** to ece2035-help@ece.gatech.edu. There are also links to tutorials, primers, and installation guides on our course web site: http://www.ece.gatech.edu/academic/courses/ece2035/assignments/ and on Canvas (Files>Additional Resources).

In working this assignment, it is helpful to first complete the Tutorial which may be found under Assignments on Canvas.

HW1-1: The goal of this part is to use a Linux/Unix environment and become familiar with its facilities. Toward this end, you must either acquire/access and run an appropriate Linux/Unix distribution that fits your computing environment or connect to a machine in the Klaus 1448 Linux cluster. The following are some the options available:

- Unix underlying Mac OS X
- Linux Bash shell on Windows 10 (Anniversary Update): installation guide
- Running native Ubuntu (e.g., 18.04.3 LTS)
- VMware on Windows 8 or 10: installation guide
- Virtual Box on Windows 8.1
- Dual booting
- Red Hat on Linux machines in Klaus room 1448

Once you do this, then perform the following exercises.

1. Create and edit a text file that contains the following:
   - Tell one interesting fact about yourself.
   - Which Linux/Unix distribution are you using (e.g., Ubuntu 18.04.3)?
   - The method you are using to run Linux/Unix (e.g., one of the options above).
2. Find a suitable application to capture a screenshot which shows your text file open in an editor, running under Linux/Unix.
3. Submit your screenshot in **jpeg format** in a file of **size no greater than 200K**.

In order for your solution to be properly received and graded, there are a few requirements.

1. The file must be named **`HW1-1.jpg`**.
1. The file must be less than 200K bytes.
2. Your solution must be properly uploaded to the Canvas site (under Assignments) before the scheduled due date.

HW1-2: The goal of this part of the project is to modify a short C program, compile it using the GNU C Compiler `gcc`, and run it. A program shell `HW1-2-shell.c` is provided. You must copy/rename it to `HW1-2.c` and modify it to compute the following.

The shell program has a global variable x which is initialized with a hexadecimal number. Modify the program to determine whether x is a Harshad number in base 16 and if so, whether the factors involved are mirrors of each other (in base 16). Recall that a 16-Harshad number is a positive integer that is divisible by the sum of its digits when written in base 16. In other words, your program should determine if `x = p*q`, where `p` is the sum of hexadecimal digits of `x`, and `q` has the same digits as `p` in reverse order. An example of such a number is 0x6A5. In base 16, 6+A+5 = 15 which evenly divides 0x6A5. So 0x6A5 is a 16-Harshad number and its factors (0x6A5 = 15 * 51) are mirrors of each other.

Example	Result (1 of 3 cases):
`0x6A3: 6+A+3 = 13; 0x643/13` has a remainder	not a 16-Harshad number (case 1)
`0x6A4 = 14 * 55`	16-Harshad w/out mirror factors (case 2)
`0x6A5 = 15 * 51`	16-Harshad w mirror factors (case 3)

Your program should indicate which case is true for any given x, using one of the 3 possible print statements provided in the shell code. Be sure to try multiple test cases, but please return x to the original test value when submitting your code. Here are some other numbers to try: 0x1AAF, 0x1A5F, 0xE29, 0x1B00, 0x1D2F, 0x1A90, 0xE30.

Compiling and running your code:

You should open a "terminal window" to run `gcc` under Linux/Unix (type `man gcc` for compiler usage or look up GCC online documentation on the internet). If you do not have gcc installed, you can use "sudo apt-get install gcc" to install it. See these links for quick tutorials on apt-get and installing packages:

- https://www.howtogeek.com/261449/how-to-install-linux-software-in-windows-10s-ubuntu-bash-shell/
- www.howtogeek.com/63997/how-to-install-programs-in-ubuntu-in-the-command-line/

Note that in the terminal window, you can enter any of the Linux commands (such as `ls`, `cd`, `cp`; for reference see http://ece2035.ece.gatech.edu/assignments/Linux_Cmd_Cheatsheet.pdf).

Use the Linux command `cd` to change your current working directory to the directory in which you placed the shell program. For example,

```
> cd ~/Documents/2035/hw1
```

or

```
> cd /mnt/c/Users/GBurdell/2035/hw1
```

You can list the files in that directory using

```
> ls -la
```

You can copy a file using `cp` or rename a file using `mv` (move a file to a new file). For example:

```
> cp HW1-2-shell.c HW1-2.c
```

Using the ascii text editor of your choice modify the `HW1-2.c` program to compute the span as described above.

Once you write your program, you can compile and run it using the Linux command line:

```
> gcc HW1-2.c -g -Wall -o HW1-2
> ./HW1-2
```

You should become familiar with the compiler options specified by these flags.

In order for your solution to be properly received and graded, there are a few requirements.

1. The file must be named **`HW1-2.c`**.
1. Your name and the date should be included in the header comment.
2. *Do not* `#include` *any additional libraries.*
3. In the starting *shell* program, it is especially important not to remove or modify any print statements since they will be used in the grading process.
4. Your solution must include proper documentation and appropriate indentation.
5. Your solution must be properly uploaded to Canvas before the scheduled due date.

HW1-3: The goal of this part is for you to install MiSaSiM, modify a short assembly program `HW1-3-shell.asm`, simulate, test and debug it in MiSaSiM. The MiSaSiM simulator can be installed according to the instructions at http://lindawills.ece.gatech.edu/misasim/. Copy or rename the shell program to `HW1-3.asm` and modify it to determine whether or not the value at location `XLoc` is a 16-Harshad number with mirror factors and if so, put a 1 in the memory location labeled `Result`, otherwise put a 0 in that location. For this exercise, it might be helpful to open the "Options" menu (the wrench on the toolbar) and select "hexadecimal" as the display base.

Two constraints apply for this assignment only:

- Do not write values to registers $0, $29, $30, or $31.
- Do not use helper functions or function calls (`JAL` instruction).

In order for your solution to be properly received and graded, there are a few requirements.

1. The file must be named **`HW1-3.asm`**.
2. Your name and the date should be included in the beginning of the file.
3. The starting *shell* program should not be modified except for the replacement of the comment "# write your code here..."
4. Your program must store its result at the memory location labeled `Result` when it returns. This answer is used to check the correctness of your code.
5. Your program must return to the operating system via the **`jr`** instruction. *Programs that include infinite loops or produce simulator warnings or errors will receive zero credit.*
6. Your solution must include proper documentation.
7. Your solution must be properly uploaded to Canvas before the scheduled due date**.**

Honor Policy: In all programming assignments, you should design, implement, and test your own code. **Any submitted assignment containing non-shell code that is not fully created and debugged by the student constitutes academic misconduct.** The only exception to this is that you may use code provided in tutorial-0.zip or in the examples on the course website/Canvas as a starting point for your programs (http://ece2035.ece.gatech.edu/examples/index.html or Canvas>Files>Lecture Slides>Code Examples).