# Programming for Bioinformatics | BIOL7200

## Week 2 Exercise

August 25, 2020

The goal of these exercises is to get you used to working with some basic UNIX commands and their options. Try and think about what each command does, and where it can be used. Reading the man page for each of these is **recommended**.

We’re going to cover the following commands:

|  |  |
| --- | --- |
| Command | Explanation |
| who and finger | Look up information about users |
| top | See the status of running system processes |
| free | See the amount of free and used memory |
| wget | Download files |
| curl | Download files |
| du | See the amount of used storage space |
| df | See the amount of free storage space |
| ps | Lookup information for specific processes |
| vim | Another text editor |
| fg | Foreground |
| which | Find location of a binary on your PATH |
| ln | Create link |
|  |  |

### Instructions for submission

* Prepare two solution sheets for this exercise - one for submission and another for your own reference. **Only submit the submission sheet**. The solution sheet you create for your reference should help you going forward - make it as detailed/brief as you'd like for your own learning style. For the submission sheet, copy the question and write the correct answers below the question.

### Exercises

1. Using who and finger
   1. Use who to see yourself logged into the machine
   2. Find ***all*** the logins on the machine
   3. Use finger to find information about yourself on your Linux VM/whatever you are using
2. Monitoring system usage with top and free
   1. Use top to look at what’s running on the computer
   2. Find the CPU and memory usage of a program; how much memory is left?
   3. Order the processes by CPU usage, ascending and descending
   4. Do the same for memory
3. Looking at hard disk space
   1. Use df to find the free space on your machine’s hard disk
   2. Use du to find the total size of some directory; make the output easy to read, e.g. ‘315M’
4. Finding running processes
   1. Find all the processes that you have running using ps
   2. Find **all** processes that are running on the machine
   3. Find all of the processes being run by the user ‘root’ in user format
5. Murder most foul
   1. Find the PID of your login shell using ps
   2. Kill your login shell with a command
   3. Open a new shell
6. Managing jobs
   1. Start editing a file with vim, then put it in the background using Ctrl + Z
   2. Find it using the jobs command
   3. Restore it to the foreground using fg
   4. Start a new job in the background using & after the command
   5. Move the new job to the foreground using fg
   6. What is the difference between the commands Ctrl + c and Ctrl + z
7. Finding the location of binaries, linking them and alias
   1. Find out where the awk command is located using which
   2. Find out if it is a link or a real binary. Follow this through until you have found the real binary.
   3. Create a symbolic link to the ls command in your home directory called listIt using the ln command
   4. Create an alias for ls -a
8. Hard vs Symbolic links
   1. Create a file randomFile.txt and type the following in it:  
      This is some text.  
      This is some more text.  
      And some more...
   2. Now create a symbolic link for randomFile.txt; name the link symLink.txt
   3. Now create a hard link for randomFile.txt; name the link hardLink.txt
   4. Create another file anotherFile.txt. Put this text in it:  
      1234  
      5678  
      9101
   5. Now delete randomFile.txt; which of the two links still works?
   6. Rename anotherFile.txt to randomFile.txt. Which of the two still works? If both work, is there a difference between the two? If only one works, why?
9. Downloading UCSC Genome Browser files with wget Toy exercise – the objective here is only to learn how to use wget. You don’t need to wait for each file to finish downloading; you can quit the download if you have determined that you are using the correct command. Once you see a file getting downloaded, kill the download process if you want. Read wget manual for helpful options.  
   1. What is the UCSC Genome Browser? What information does it store? What organisms does it focus on? What is the FTP site for accessing the databases?
   2. Navigate to the ENCODE repository on the UCSC FTP site.
      1. What is an ENCODE repository?
      2. Download a file (pick a small one of your choice) from the ENCODE repository on the UCSC Genome Browser with wget and briefly explain what data it contains.
   3. Download all of the Pol2b binding data available in one line with wget
   4. The database folder located in hg19 in goldenPath is an important folder. We are going to fetch four files: knownGene.txt.gz, knownGene.sql, kgXref.txt.gz and kgXref.sql. Copy their link addresses and paste in a file. Now use wget to download the files by reading the URLs from the file you just created. In other words, this time the input URL comes from a file rather than from you giving it in the command, and there are multiple URLs. **Hold on to these files; we will use them later during the course.**
   5. Test if the URL http://www.ncbi.nlm.nih.gov/SNP/snp\_ref.cgi?searchType=adhoc\_search&type=rs&rs=rs12345 is correct using wget. [Hint: Have you seen error 404 while accessing a website? See if HTTP Status codes can help].
10. Write one-liners to perform the following actions with wget or curl
    1. Download the file taxdb.tar.gz from ftp://ftp.ncbi.nlm.nih.gov/blast/db/
    2. Download all files that start with “blast” from ftp://ftp.ncbi.nlm.nih.gov/blast/documents/
    3. Download all “ppt” files from ftp://ftp.ncbi.nlm.nih.gov/blast/demo/
11. The PATH to enlightenment

The PATH environmental variable tells the shell where to look for executables. When you use the ls command, it works because the shell knows to look for it on the PATH .

* 1. Print your PATH environmental variable to the screen. What are the directories in it? Some of them should look familiar at this point.
  2. Make a directory call bin in your home directory and put a symbolic link to some already existing executable in the newly created bin directory. Call the link something different from the actual executable or this is all for naught.
  3. Add you bin directory to your PATH environmental variable. Keep in mind what character separates the different directories in the PATH environmental variable. If you mess up your PATH , then just leave the shell and make a new one.
  4. Try running the executable through the symbolic link. Did it work? Use which to see the alias.
  5. Add your new bin directory to your PATH in your login file. Make sure you can still access all of the programs on your PATH.
  6. Congratulations! You now have a centralized folder that will be on your PATH every time you log in. You can install executables here if you don’t have permission to install things.
  7. If you’re on a system with multiple users, what might be an advantage of install an executable in one of the directories on the default PATH ?

1. The C compilers

There are various C compilers available today. They pretty much do the same thing: turn C code into a program, but they do it with widely varying efficiency.

1. Find out what your cc command points to. In other words, what is your default C compiler?
2. Compile the haveSomeNumbers.c file from Canvas and run the result (you’re going to need the -o flag. Read about it before using it). Add the results to your PATH and run it that way.
3. C is beyond the scope of this course, but try to figure out what the haveSomeNumbers.c file does anyway. This program is very simple. Many, many bioinformatics programs are written in C.
4. Look up comparisons of gcc and icc online. What are the advantages and disadvantages of each? What other C compilers exist?