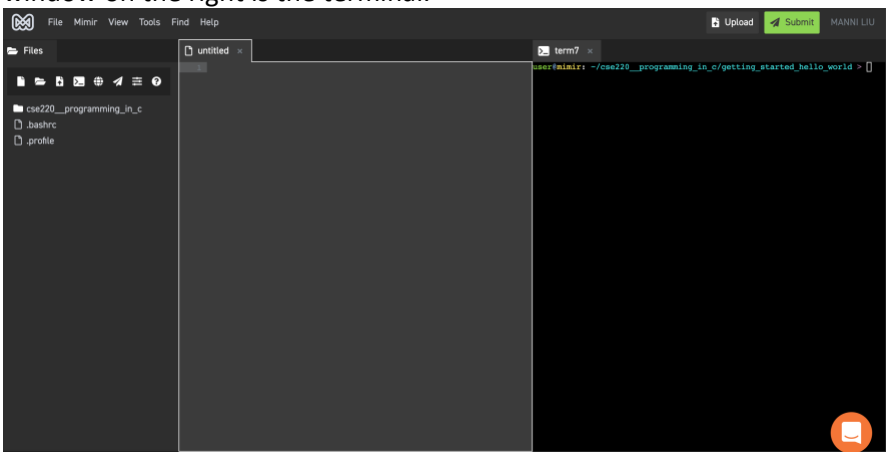
Lab Assignment #1

Purpose: demonstrate a number of unix commands, edit and run your first C program

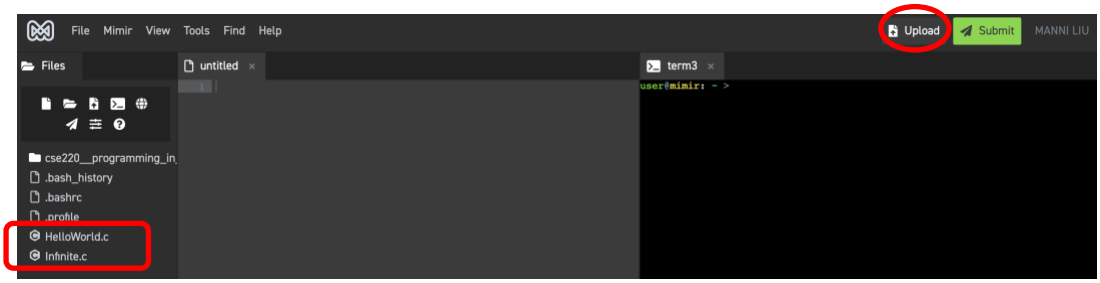
Getting started

1. Log in to your Mimir: https://www.mimirhq.com/
2. Open Mimir IDE:

3. If your screen is showing the following content, then you succeed in entering Mimir IDE. The window on the right is the terminal.



4. Download **HelloWorld.c** and **Infinite.c** from D2L, save them on your local computer first.
5. Upload **HelloWorld.c** and **Infinite.c** into your Mimir IDE using the button on the top-right corner. You will be able to see these two files in your current directory when you succeed.



Some Unix Commands

Open a terminal window on your screen. Type the following commands and observe the output

NOTE: The individual commands designated with font `like this` are to be run on the command line. The underlined sections indicate that you should run a previously learned command to complete the specified task

1. **Where are we?**

Run the following command. What happens?

```
pwd
```

The pwd command prints the current directory. You should get something like: /home/(your_username)

2. **What is in our current directory?**

Run the following command. What happens?

```
ls
```

The ls command lists the current directory. You should see a listing of the files in your directory. Now try it with the –a option.

```
ls –a
```

Is there a difference from the previous output?

3. **How do we create a file?**

Run the following command. What happens?

```
touch myFirstFile
```

The touch command "touch" a file, setting the time and date of the file to the current time and date. If the file does not exist, it creates an empty one. Now type the command ls again with more details:

```
ls -l
```

You should see the file you just created. Touch myFirstFile and list again and note the difference in timestamp.

```
touch myFirstFile
ls –l
```

4. **How do we delete a file?**

The rm command removes a file. Type the following command to remove that file you just created:

```
rm myFirstFile
```

Do a directory listing (ls command) to be sure this worked.

5. **How do we create a directory?**

The mkdir command creates a new directory. Type this command:

```
mkdir cse220
```

Now do a listing to be sure it created the directory.

Create another directory called activities

6. **How do we move into a directory?**

The cd command changes the current directory. Change into the activities directory:

```
cd activities
```

Print the current working directory by typing pwd. Verify you are in /home/(your_username)/activities

Create a blank file in the current directory by touch mySecondFile

Try to change to the directory summerActivities:

```
cd summerActivities
```

This should **FAIL** since summerActivities does not exist.

7. **How do we move out of a directory?**

Now change into the special directory ..

```
cd ..
```

This special directory (..) represents the parent of the current directory.

Now print the current working directory. You should be in /home/(your_username)

8. **How do we work with the current directory?**

Now change into the special directory .

```
cd .
```

Then print the current directory. Your directory should not have changed.

'.'is a special directory representing the current directory, so changing into '.' keeps you in the same directory

9. **How do we delete a directory?**

The rmdir command removes a directory. Try to remove the directory activities:

```
rmdir activities
```

The rmdir command will only remove an empty directory. If you have content in the directory it will fail with a message like "failed to remove stuff: Directory not empty". In that case, you can remove the directory using this command:

```
rm –r activities
```

Verify activities has been removed.

10. **How do we move beyond our local directories?**

Try typing the following command:

```
cd ~/cse220
```

After each command type pwd to be sure you know what it does

You have now typed a complete path. We separate the directory names with a "/" character. This command takes you to the directory cse220 which is a child of the home directory. You should be able to type this again and it should work, leaving you in the same directory.

```
cd ../cse220
```

This moves you to the directory cse220 as a child of the parent directory. This will be the same directory.

```
cd ../cse220/.././cse220
```

Remember that '..' means parent directory and '.' means the current directory. Do you know what this command is doing?

11. **How do we interact with other directories?**

If you are not already, move into the cse220 directory. Create a directory called lab01. Change into directory lab01. Type the following:

```
ls /home/(your_username)
```

12. **How do we copy files?**

```
This lists the content of directory /home/(your_username). You should
see a file HelloWorld.c. Copy it into your lab01 directory:

cp /home/(your_username)/HelloWorld.c .
```

The cp command copies a file from /home/(your_username)/ into the current directory. List the content of lab01 to make sure you have successfully copied this file.

13. **How do we move or rename files?**

Rename this file to MyHelloWorld.c

```
mv HelloWorld.c MyHelloWorld.c
```

Again, do a directory listing to confirm.

14. **How do we view the contents of a file?**

To display the first 5 lines, type:

```
head -5 MyHelloWorld.c
```

To display the last 7 lines, type:

```
tail -7 MyHelloWorld.c
```

Now list the whole content:

```
cat MyHelloWorld.c
```

Compiling and Running Programs

Copy the file /home/(your_username)/Infinite.c into your lab01 directory.

Compile it (i.e. turn it from written code to machine code) using the following command:

```
gcc –o infinite Infinite.c
```

Remember gcc compiles the program, the -o switch tells it to create an output file named "infinite" instead of "a.out".

A second command (‘./’ + the executable name) executes the program:

```
./infinite
```

Killing Programs

You should notice that your program runs endlessly and you won’t be able to use the terminal anymore. Open another terminal window and list the most CPU intensive processes:

```
top
```

You will likely see infinite right there. Press q to quit top. Type this command in the working terminal window:

```
ps u
```

You should see a list of running processes, each with a process id (PID). Using the PID for the infinite process, stop your infinite program from running:

```
kill (infinitePID)
```

Now do a ps again and infinite should be gone. Look at top again and it should not show that process anymore.

Editing your C program

Now type the following commands to compile and run this program:

```
gcc –o sayHello MyHelloWorld.c
./sayHello
```

Note the ./ means the file to execute is in the current directory. It won't work without it (try it).

Open the file MyHelloWorld.c. This is the C program you ran earlier. It should look like this:

```
#include <stdio.h>

int main() {
      printf("Hello World!!!\n");
      return 0;
}
```

Change the program to print "Hello from *your name*\n" instead of “Hello World!!!\n”.
Save your changes, compile into an executable called sayHello and run.

Completion

Show the TA that you can execute the “sayHello” executable and they will give you credit for the lab.

End the session recording by typing the following:

```
exit
```