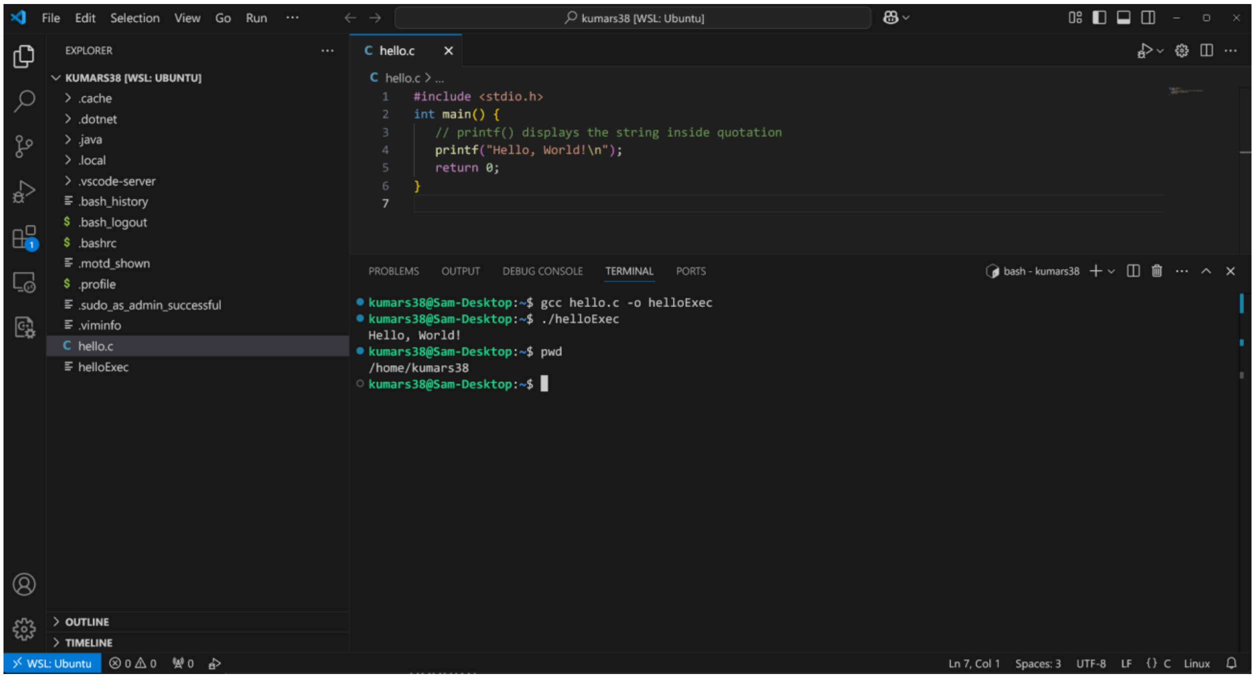
Thursday Jan 16 (Intro to Terminal/Unix Commands)

I would suggest a workspace like this:



```c
#include <stdio.h>
int main() {
   // printf() displays the string inside quotation
   printf("Hello, World!\n");
   return 0;
}

```

```
kumars38@Sam-Desktop:~$ gcc hello.c -o helloExec
kumars38@Sam-Desktop:~$ ./helloExec
Hello, World!
kumars38@Sam-Desktop:~$ pwd
/home/kumars38
kumars38@Sam-Desktop:~$
```

- (For Windows users) Connected to WSL as indicated by blue rectangle, bottom-left
- Explorer is selected on the left sidebar, so we can view all files in our Linux home directory, create new directories, etc.
- Using VS Code's interface to edit any text files (ex. hello.c)
- Using VS Code's built-in terminal with WSL: works the same as launching a Windows terminal (cmd/Powershell) and then typing "ubuntu". On Mac, the terminals are the same already.

For today, we'll be working in the terminal to gain familiarity with Linux commands. Refer to commandLines.pdf (posted on Avenue) for a basic commands and descriptions.

Tips:

- use the up arrow to refer to previously typed commands in your terminal
- type `clear` to clear the terminal window

Exercise:

1. Open the terminal
2. Use `pwd` to see your current location (called the working directory)
   - Should see `/home/(username)`
3. Use `ls` to print the files in your working directory
   - Try `ls -l`
   - Try `ls -la`
   - Try `ls -l -a`
   - what are the differences?
4. Use `mkdir` to create a new directory called “week2-exercise”
5. Use `cd` to navigate to the new directory
   - `cd .` refers to your current directory
   - `cd ..` refers to one directory above (this is the directory that contains your current directory)
   - `cd dir` moves you to the directory called ‘dir’
6. Verify you're in the right place using pwd again
   - Should see `/home/(username)/week2-exercise`
7. Use `echo "Hello" > test.txt` to create a text file
   - echo prints the specified text
   - `>` specifies input to a file
8. Use `cat` to read the contents of test.txt
   - Try echo `"Hi" > test.txt` followed by `cat`
   - Try echo `"Hello" >> test.txt` followed by `cat`
   - Any differences?
9. Use nano to edit the contents of test.txt within the terminal, change “Hello” to “Good evening”, save your changes (press Ctrl+X, Y, enter)

- Can use `vim` also instead of nano (:wq, enter to exit)
- Confirm the changes using `cat test.txt`

10. Use cp to copy test.txt to a new directory called step10

11. Move into that directory using cd

12. Verify the file is here and valid using `ls` and `cat`

13. Use `rm` to remove the file from `the week2-exercise` folder

    - Based on your pwd, which path should you supply to rm?

14. Go back to the `week2-exercise` folder, create an empty file called test2.txt

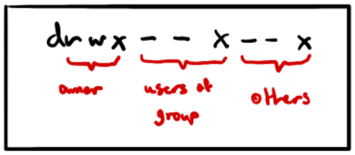
    - How can we make an empty file?

15. View permissions based on `ls -la`

ls -la, good list view:

- = regular file
d = directory
r = read
w = write
x = execute

ex. Permissions:



left side of ls -la

ex. chmod g+w yyyy
↳ gives the group write permissions to yyyy

16. Using the `chmod` command and the following info, modify the permissions of test2.txt to make it **read-only** for *all* users
    - u, g, o, a: refer to user (owner), group, others, or all
    - + r/w/x: adds the permission to the selected group(s)
    - - r/w/x: removes the permission from selected group(s)
    - = r/w/x: sets the permission for selected group(s), removing other perms
    - `Example: chmod` **`o+w`** `test2.txt` adds write privileges to **other users**
17. Confirm the new permissions using `ls -la`
18. Try to edit the file in nano, what happens?
19. Create a file called sum.c which contains the following code

```c
#include <stdio.h>
int main() {
    int number1, number2, sum;
    printf("Enter two integers, separated by a space: ");
    scanf("%d %d", &number1, &number2);
    // calculate the sum
    sum = number1 + number2;
    printf("%d + %d = %d\n", number1, number2, sum);
    return 0;
}
```

- Hint: start with nano `sum.c`

20. Use gcc to compile the file to an executable file called “sum”
    - Recall:
    - `gcc (myfile.c) -o (executable_file_name)`
21. Execute “sum”, should work as expected
    - `./(executable_file_name)`
22. Change the permissions for “sum”, removing **execute** privileges from **owner**
    - (Refer to step 16)
23. Execute “sum” again, what happens?
24. Navigate back to `/home/(username)` directory, remove the `week2-exercise` directory using `rmdir`
    - Note the error message, the directory must be empty
25. `Try rm -r week2-exercise`
    - `-r` flag means recursively delete
26. Clear terminal and exit