Daniel Guo
Github: yiluun

ID: 913290045
CSC415 Operating Systems

Assignment 1 – Command Line Arguments

Description:
This assignment is to write a C program that accepts arguments via the command line and then displays each of those arguments to the terminal along with how many arguments there are.

Approach / What I Did:
I installed VirtualBox along with the complete Ubuntu image provided by the professor. I then installed the VirtualBox Extension Pack and imported the image. Upon logging into the system, I had to wait for background updates since it locked me out of updating other things. I then did some basic updates and began to get familiarized with commands. After reading the first assignment's README file, I installed Visual Code and cloned my repository via the command line (after authenticating my GitHub token).

I started the actual assignment by searching up how to print out arguments from the command line to first understand what that meant. I figured that arguments were the name of the file, followed by inputs from the user separated by a space by looking at the sample execution in the README file. I figured out that you can make a C file by creating a file with the extension ".c" and created a file

```
student@student-VirtualBox:~/csc415-assignment-1-command-line-yiluun$ touch Guo_Daniel_HW1_main.c
```

Afterwards, I looked up skeleton code for the main function understanding that **#include <studio.h>** gives access to the printf() function, **argc** in **int main(int argc, char *argv[])** refers to the number of arguments entered into the command line on program execution, and **argv** is a pointer that points to the arguments. Then I learned that format specifiers were a bit like variables in other languages, by acting as a placeholder in the printf() function. With this I was able to use my programming knowledge of for loops to iterate through the arguments and print it out.

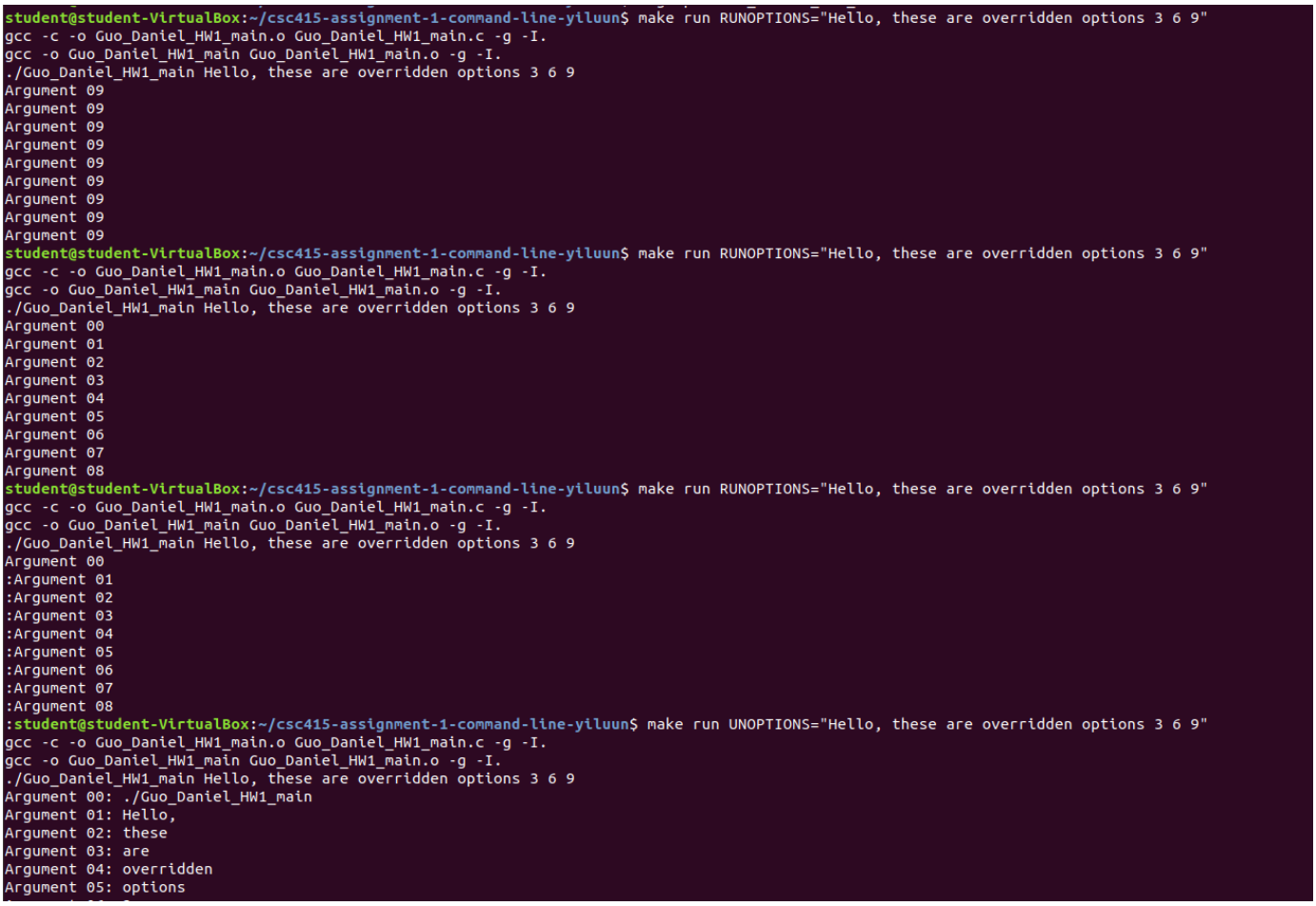
Guo_Daniel_HW1_main.c U

Guo_Daniel_HW1_main.c > main(int, char * [])

```c
#include <stdio.h>

int main(int argc, char* argv[]){

    // for loop to iterate through the number of arguments stored in argc
    for (int i = 0; i < argc; i++) {
        printf("Argument %02d: %s\n", i, argv[i]);
    }

    return 0;
}
```

Issues and Resolutions:

My first issue was that I didn't know how to write in C code, but that was easily solved when I understood how to access the command line arguments. Utilizing the command line was relatively easy, but I did have to look up how to create files, open files, etc. I resolved my issues by testing things one at a home as shown in the screenshot below. First I learned how to iterate through the arguments, but I accidentally used **argc** instead of **i** in the printf() function. Then I tried to figure out how to add a colon, followed by the arguments themselves. This took the longest time looking up. I figured out that you can just write an entire string literal with the format specifiers (**%d** and **%s**) acting as placeholders. After you close the string literal, you can add the values that are being represented by the specifiers behind a comma (**i** and **argv[i]**).



Compilation:

```
student@student-VirtualBox:~/csc415-assignment-1-command-line-yiluun$ make
gcc -c -o Guo_Daniel_HW1_main.o Guo_Daniel_HW1_main.c -g -I.
gcc -o Guo_Daniel_HW1_main Guo_Daniel_HW1_main.o -g -I.
student@student-VirtualBox:~/csc415-assignment-1-command-line-yiluun$
```

Execution:

```
student@student-VirtualBox:~/csc415-assignment-1-command-line-yiluun$ make run RUNOPTIONS="Hello, these are overridden options 3 6 9"
./Guo_Daniel_HW1_main Hello, these are overridden options 3 6 9
There were 9 arguments on the command line.
Argument 00: ./Guo_Daniel_HW1_main
Argument 01: Hello,
Argument 02: these
Argument 03: are
Argument 04: overridden
Argument 05: options
Argument 06: 3
Argument 07: 6
Argument 08: 9
student@student-VirtualBox:~/csc415-assignment-1-command-line-yiluun$
```

The following sites were what I used for help in completing this assignment:

- https://www.freecodecamp.org/news/the-c-beginners-handbook/#introduction-to-c
- https://techvidvan.com/tutorials/command-line-arguments-in-c/#:~:text=These%20command%20line%20arguments%20are,list%20of%20command%20line%20arguments.
- https://www.includehelp.com/c-programs/input-an-integer-value-and-print-with-padding-by-zeros.aspx
- https://stackoverflow.com/questions/34598089/how-can-you-print-multiple-variables-inside-a-string-using-printf
- https://www.tutorialspoint.com/format-specifiers-in-c