



COMP 201 - Spring 2023
Assignment 0 - Getting Started with Unix and C
Assigned: 8 March 2023 23:59, Due: 16 March 2023 23:59

Mert Cokelek (mcokelek21@ku.edu.tr) is the lead TA for this assignment.

1 Introduction

In this warm-up assignment, we will become more familiar with: C fundamentals, building simple C programs, basic Git commands, and working through GitHub classroom. You should accept this assignment on GitHub Classroom using the link provided in this document and complete a simple C program involving some basics.

2 Logistics

This is an individual project. All hand-ins are electronic. Clarifications and corrections will be announced on Blackboard.

3 Handout Instructions

I Create a GitHub (github.com) account if you do not have any.

II Fill the following Google Form so that we can match your GitHub username with you:

<https://forms.gle/feFtJxhVn2vhJLfb6>

III Accept the assignment using the following link:

https://classroom.github.com/a/VemH_G9d

IV Clone the GitHub repository created for you to a Linux machine in which you plan to do your work using the following command:

```
$ git clone https://github.com/COMP201-Spring-2023/assignment-0-USER.git
```

(Replace USER with the GitHub username that you use to accept the assignment)

Note: We advise you to work on our linuxpool servers. See Section 8 for details.

V **[IMPORTANT]** After cloning the repository, you are required to **write** the following honor code in a new file called "honor.txt" and **commit & push** it: "I hereby declare that I have completed this assignment individually, without support from anyone else." (other than the TAs :))

You can use the following command to create "honor.txt" file and write the honor code in it:

```
$ echo "I hereby declare that I have completed this assignment individually,  
without support from anyone else." > honor.txt
```

See Section 6 in order to learn about commit & push.

4 Task

4.1 Files

In this assignment include several files:

- `main.c`: the entry point to the C program
- `mylibrary.c`: a large open source mock library
- `mylibrary.h`: the corresponding header file of `mylibrary.c`
- **`myprogram.c`**: a simple library with 5 functions
- `myprogram.h`: the corresponding header file of `myprogram.c`
- `Makefile`: the make script
- `README.md`: the repository readme file
- `test`: a directory containing automated tests

You only need to modify `myprogram.c`. In particular, any section needing modification and code writing is marked with a `//TODO` comment. For each `//TODO`, you need to implement the corresponding function by writing a short block of code that satisfies the functionality that `//TODO` comment asks for.

4.2 Functions To Be Implemented

- `sum(n)`: calculates $\sum_{i=1}^n i = 1 + 2 + \dots + n$
- `product_of_even_numbers(array, count)`: calculates the multiplication (product) of even numbers in the given array with a length of `count`.
- `max_of_numbers(array, count)`: returns the maximum number in the given array with length of `count`.
- `reversed_number(number)`: reverses the digits of the given number. For example, if the number is 12345, return 54321.
- `alphabet_index(index)`: return the alphabet (as an uppercase letter) of given `index` assuming. For index 0, return 'A'; for index 1, return 'B'; ... for index 25, return 'Z'. If the index is out of range, return space ' '.

5 Testing your work

5.1 How to run

To run the program, you must first build it. The provided Makefile streamlines this process, all you need to do is type `make` on command-line and hit enter, and the program will be compiled and the binary file `assignment0` will be created (assuming there are no errors).

You can run this binary file by typing `./assignment0`, which will start the program. The program start point is the function `main()` inside `main.c`. Feel free to read that file, but please do not modify anything except `myprogram.c`.

5.2 Autograding

You can manually test your code by running the program and providing it with manual inputs, or type `make test` on command-line and hit enter to run the automated tests that are included in the repository.

If any of the automated tests included in the package fail, you will receive an error in the form below:

```
$ make test
1c1
< 16
---
> 28
Test test/2-1.test failed
make: *** [test/2-1.test] Error 1
```

The error specifies which test has failed (you can look inside the `test/` directory to see expected input/output) and also tells you what output was expected and what was generated by the program. For example, in the figure above, the expected output was 16, but the generated output was 28.

It goes without saying that most of the tests should fail when you begin the assignment, as the corresponding code is not written yet.

6 Git Instructions (Hand-in and Version-Control)

You will submit your work using GitHub. You should use git commands effectively in terms of committing your work frequently. See Section 7 (Evaluation/Effective use of version control points). ***Here are the steps to submit your work (you are expected to follow these steps for every meaningful change as described in Section 7):***

- I Add the changes to the index: `$ git add .`
- II Commit the changes you make: `$ git commit -m "commit message"`
Note: Please do not forget to replace `commit message` with an appropriate commit message describing the commit.
- III Push (upload) your work to GitHub servers: `$ git push origin main`

7 Evaluation

Your score will be computed out of a maximum of 100 points based on the following distribution:

(60p) Correctness points. We will evaluate your functions using the `make test` to auto-grade. You will get full credit (12 points) for a function if it passes all of the tests performed by `make test`, and no credit otherwise.

(30p) Points for "Effective use of version control". You are required to push your changes to the repository frequently. If you only push the final version, even if it is implemented 100% correctly, you will lose a fraction of the grade because you are expected to learn to use Version Control Systems effectively. You do not have to push every small piece of change to GitHub, but every meaningful change should be pushed. For example, each of the functions coded and tested can be one commit. **For each function, there should be at least one commit (with proper commit message) that includes just modifications on that function. Here, 30 points are divided between functions. For each function with at least one proper commit, you will get 6 points.**

(10p) *Style points.* Finally, we've reserved 10 points for a subjective evaluation of the style of your solutions and your commenting. Your solutions should be as clean and straightforward as possible. Your comments should be informative, but they need not be extensive.

8 How to use linuxpool.ku.edu.tr linux servers ¹

I Connect to KU VPN (If you are connected to the KU network, you can skip this step.)

See for details: <https://help.ku.edu.tr/ithelp/vpn-access>

II Connect to linuxpool.ku.edu.tr server using SSH (Replace USER with your Koç University username):

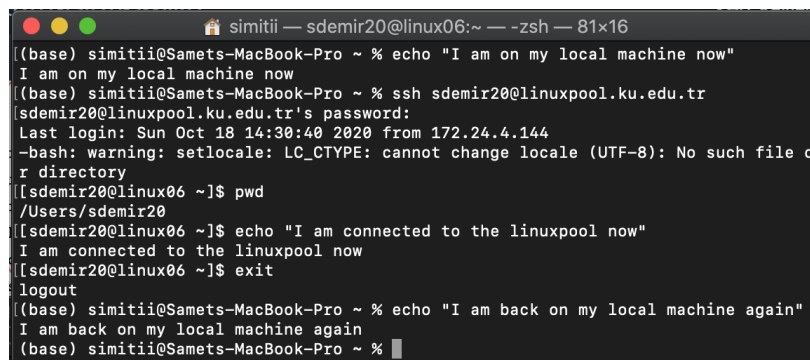
```
$ ssh USER@linuxpool.ku.edu.tr
```

(It will ask your password, type your Koç University password.)

III When you are finished with your work, you can disconnect by typing: `$ exit`

Your connection to the server may drop sometimes. In that case, you need to reconnect.

We suggest you to watch the following video about the usage of SSH, which is used to connect remote servers, and SCP, which is used to transfer files between remote servers and your local machine: <https://www.youtube.com/watch?v=rm6pewTcSro>



```
simitii — sdemir20@linux06:~ — zsh — 81x16
((base) simitii@Samets-MacBook-Pro ~ % echo "I am on my local machine now"
I am on my local machine now
((base) simitii@Samets-MacBook-Pro ~ % ssh sdemir20@linuxpool.ku.edu.tr
sdemir20@linuxpool.ku.edu.tr's password:
Last login: Sun Oct 18 14:30:40 2020 from 172.24.4.144
-bash: warning: setlocale: LC_CTYPE: cannot change locale (UTF-8): No such file or
r directory
[[sdemir20@linux06 ~]$ pwd
/Users/sdemir20
[[sdemir20@linux06 ~]$ echo "I am connected to the linuxpool now"
I am connected to the linuxpool now
[[sdemir20@linux06 ~]$ exit
logout
((base) simitii@Samets-MacBook-Pro ~ % echo "I am back on my local machine again"
I am back on my local machine again
((base) simitii@Samets-MacBook-Pro ~ %
```

Figure 1: How to connect and disconnect using SSH

9 Academic Integrity

All work on assignments must be done individually unless stated otherwise. You are encouraged to discuss with your classmates about the given assignments, but these discussions should be carried out in an abstract way. That is, discussions related to a particular solution to a specific problem (either in actual code or in the pseudocode) will not be tolerated. In short, turning in someone else's work, in whole or in part, as your own will be considered as a violation of academic integrity. Please note that the former condition also holds for the material found on the web as everything on the web has been written by someone else. See [Koç University - Student Code of Conduct](#).

¹For details, please see the guide on linuxpool that we have announced on Blackboard

10 Late Submission Policy

You may use up to 7 grace days (in total) over the course of the semester for the assignments. That is you can submit your solutions without any penalty if you have free grace days left. Any additional unapproved late submission will be punished (1 day late: 20% off, 2 days late: 40% off) and **no submission after 2 days (48 hours) will be accepted.**

11 Useful Links

- If you are a Windows user, you can study this short tutorial on how to use Git on Windows:
<https://www.computerhope.com/issues/ch001927.htm>
- 35 basic Linux commands that every user should know:
<https://www.hostinger.com/tutorials/linux-commands>
- You can also take a look at Unix shell commands on page 15 and 16:
<http://cslibrary.stanford.edu/107/UnixProgrammingTools.pdf>