Exercise 11

Info

This is an in-class exercise. An exercise page like this one will contain a brief description but is intended to be supplemented by discussion during our meeting time. Complete the exercise to the best of your ability in the time given. Feel free to talk with other students as you work, and do not be afraid to ask questions if you get stuck. Aim to complete as much as possible during our meeting, and submit on Gradescope to check your solution. You are encouraged to work at home to complete what you do not get through today.

Learning Objectives

Reinforces concepts learned in today's lesson:

- Dynamic memory allocation.
- Use valgrind to analyze the memory usage of existing code and identify and remove errors in the code.

Part 1

Pull the starter code for this exercise from the public repo by taking the following steps:

1. Log into an undergraduate cluster computer. Update the course public repo with a `git pull` command.
2. Confirm that you can see the template files for today's exercise by typing `ls exercises/ex11` – you should see files named `pairwise_sum.c` and `primes.c` inside.
3. Navigate to your personal repo for the course and make a directory for today's exercise. Copy both `*.c` source files from the public class repository (in the */exercises/ex11/* directory) into your personal repo's directory for this exercise.

Part 2

You'll work on `pairwise_sum.c` first. To compile the code, use this command:

```
gcc -std=c99 -pedantic -Wall -Wextra pairwise_sum.c -o pairwise_sum
```

Note the use of `-g`, which enables debugging. Then run the program using valgrind:

```
valgrind  --leak-check=full  --show-leak-kinds=all  ./pairwise_sum
```

Read the valgrind output and try to determine what is causing the errors that are reported. Modify `pairwise_sum.c` so that it still performs the desired task (described in comments at the top) but without any memory leaks or invalid reads or writes.

Part 3

Do the same thing you did in Part 2, but for `primes.c`. Since `primes.c` requires the math library, you will have to add the `-lm` flag for the compile command:

```
gcc -std=c99 -pedantic -Wall -Wextra primes.c -o primes -lm -g
```

The valgrind command is essentially the same:

```
valgrind  --leak-check=full  --show-leak-kinds=all  ./primes
```

Read the valgrind output and try to determine what is causing the errors that are reported. Modify `primes.c` so that it still performs the desired task (described in comments at the top) but without any memory leaks or invalid reads or writes.

Reminder

Remember to add and commit to your local repo copy as your work. Push to your remote repo when finished. Also scp and submit a zip of `pairwise_sum.c` and `primes.c` to Gradescope to check your solution. Use `exit` to logout from your ugrad account when finished. If you continue to work on the program after class, make sure to keep your repo updated as well!