ECE-C353: Systems Programming
Homework Assignment 4: Writing Signal Handlers

Write a single threaded program called `primes.c` that computes the first 10,000,000 prime numbers. The current progress should be printed to the screen every 10 seconds by using `SIGALRM`. The current progress should also be printed to the screen upon receiving the `SIGUSR1` signal – allowing the user to check the current progress on demand by using the `kill` command to send signal 10 (`SIGUSR1`). To make the assignment a bit more interesting, your program should always block `SIGQUIT`. If `SIGTERM` is received, you should print the current progress followed by the message `Goodbye!` on its own line.

The current number of primes your program has found must be stored in the `DATA` segment of the process's virtual address space using a variable named `num_found`.

Found primes must be stored in sequential order into the array `primes[]`, which must also be located in the `DATA` segment.

At minimum, your program should consist of the following functions:

- `int is_prime(unsigned int num)` – This function accepts an unsigned integer `num` and returns 0 if `num` is not prime and a non-zero value if `num` is prime. Use the following macros in your program to increase readability:

  ```
  #define FALSE 0
  #define TRUE !FALSE
  ```

  Note: There are plenty of existing prime number checking algorithms already in existance. Feel free to use one. I don't expect you to write one from scratch.

- `void handler(int sig)` – This is a custom signal handler you will write to handle `SIGALRM`, `SIGUSR1`, and `SIGTERM`. Your program should receive a `SIGALRM` signal every 10 seconds. This means that you will have to setup the request for the next `SIGALRM` delivery from within your signal handler. Additionally, this signal handler should print the total number of prime numbers found so far as well as the last 5 that were found. Although not reentrant, you may use `printf()` in your signal handler – this will introduce a small chance of a crash if `SIGUSR1` is received while `SIGALRM` is being serviced. Here is an example of what should be printed to the screen:

  ```
  Found 7047368 primes.
  Last 5 primes found:
      123830857 123830881 123830893 123830909 123830929
  ```

- `int main(int argc, char* argv[])` – This is the main function of your program. Here you should:
  - block `SIGQUIT`
  - install your signal handler for the necessary signals
  - kickoff the initial `SIGALARM` signal (see: `man 2 alarm`)
  - iterate in some sort of loop, which should terminate after finding the first 10 million primes.

Note: When updating `num_found` and `primes[]`, any signal that could cause the current progress to be displayed should be blocked (i.e. `SIGALRM`, `SIGUSR1`, `SIGTERM`) and unblocked once the update is complete. This will prevent a situation where the two variables are not synchronized due to interruption during the update – which would result in an inaccurate report.

Deliverables:

You will submit 1 file via BBLearn:

- `abc123_primes.c`

Upload your code (do your own work!) to the BBLearn submission link.

(As always, replace **abc123** with your Drexel ID).