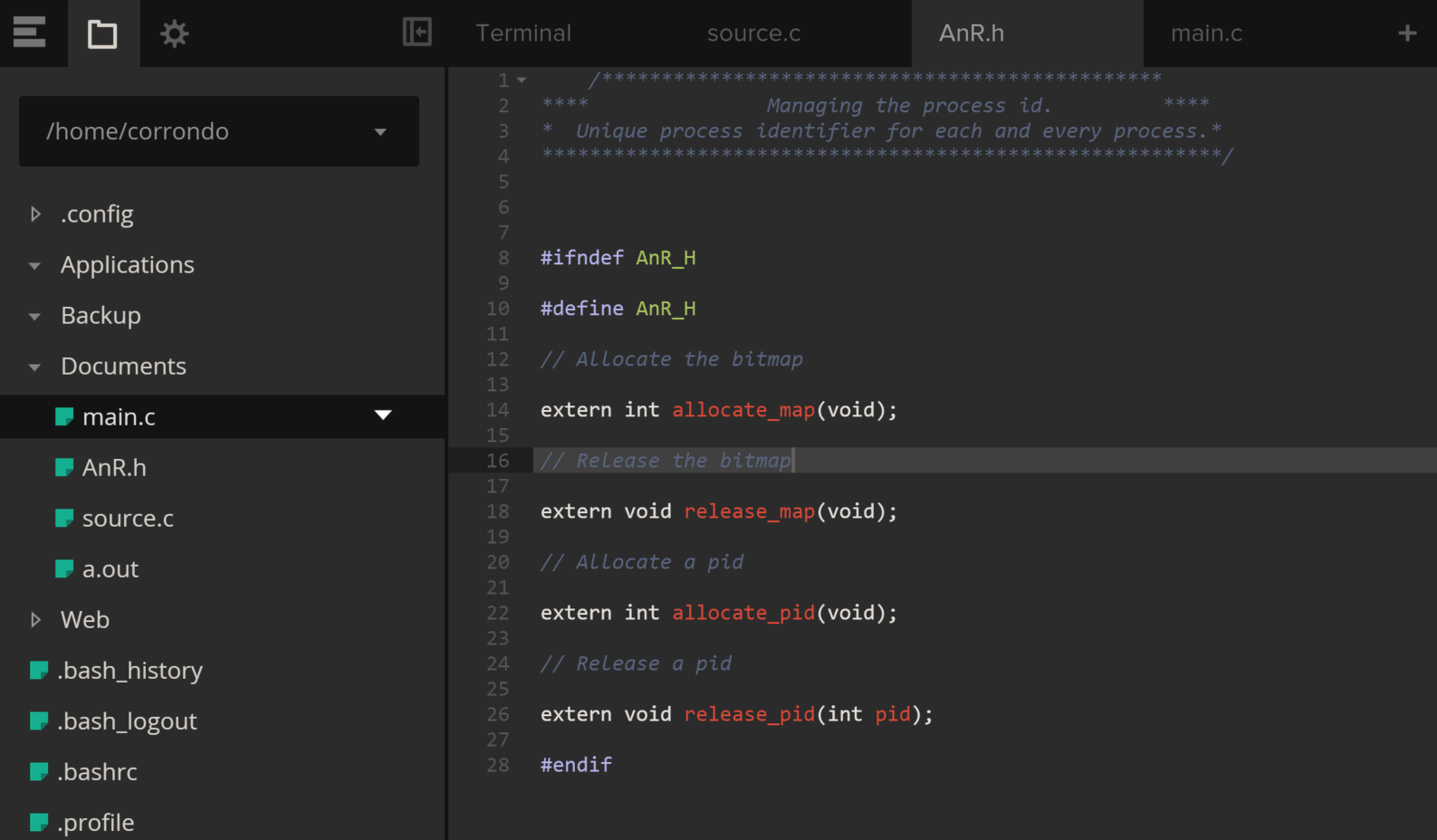
Student name: Federico Watkins

CSIS 3810: Operating Systems

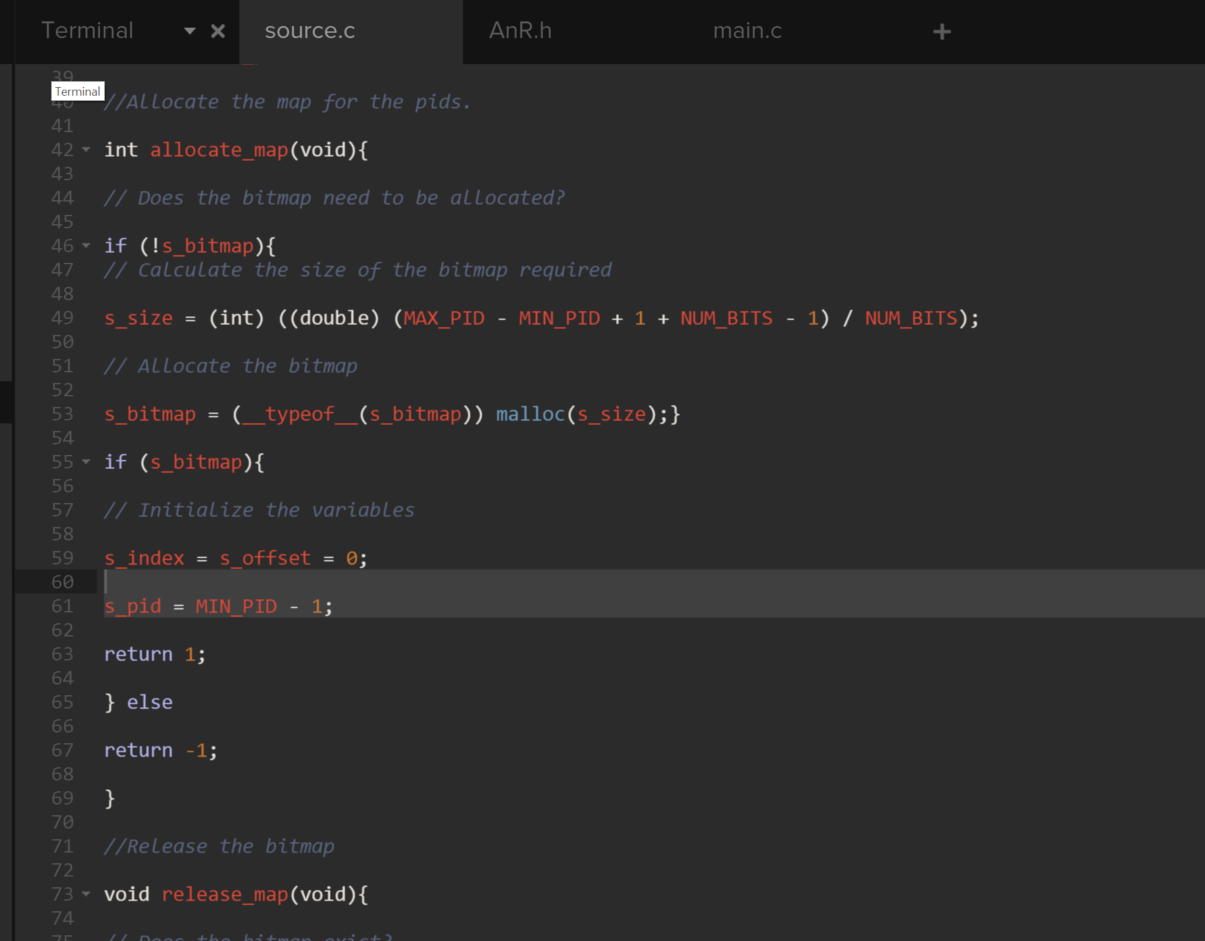
Chapter: 3-4

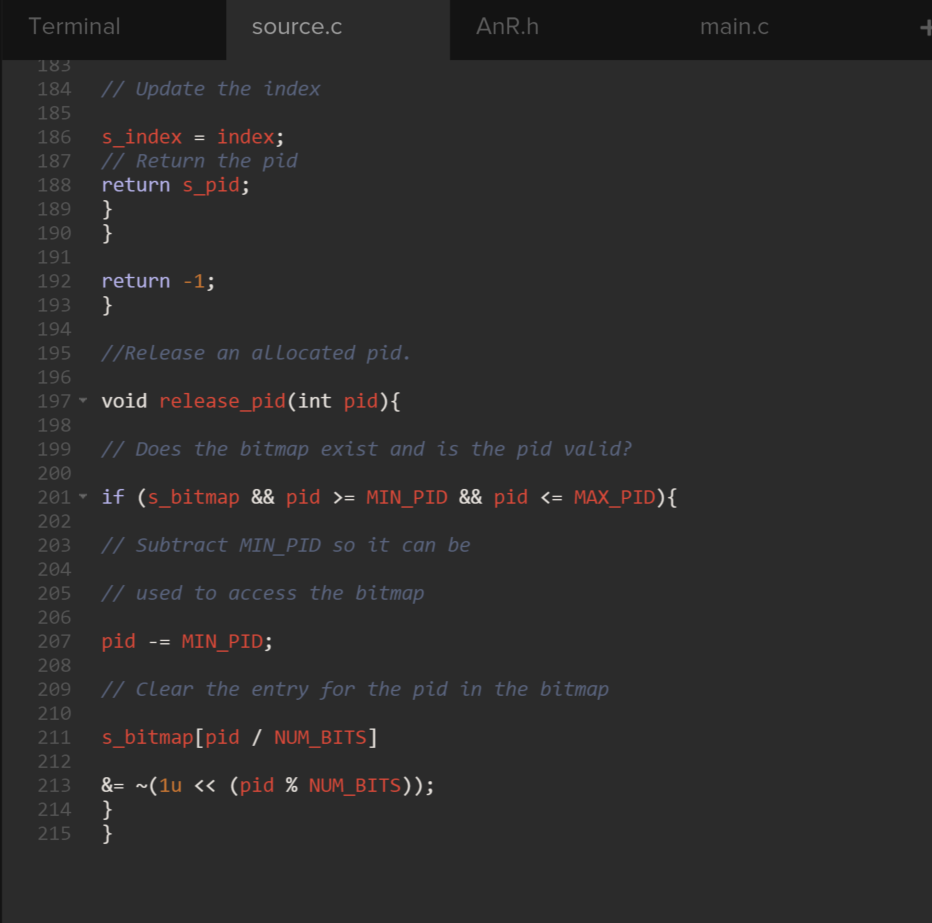
Assignment: 2

First I implement and import a main function header file “main.c” to start execution of the thread using its main method. Second implement a starting function for my threads. The function will take one value as an argument. In this function I will allocate the pid, sleep time and release time and will also release my thread. Another header file “bitmapPid-AR.h” is used to allocate and release the bitmap and pid. Finally I define another source file “ImpMet.c” to implement the methods (Please see attached source code for even more details).

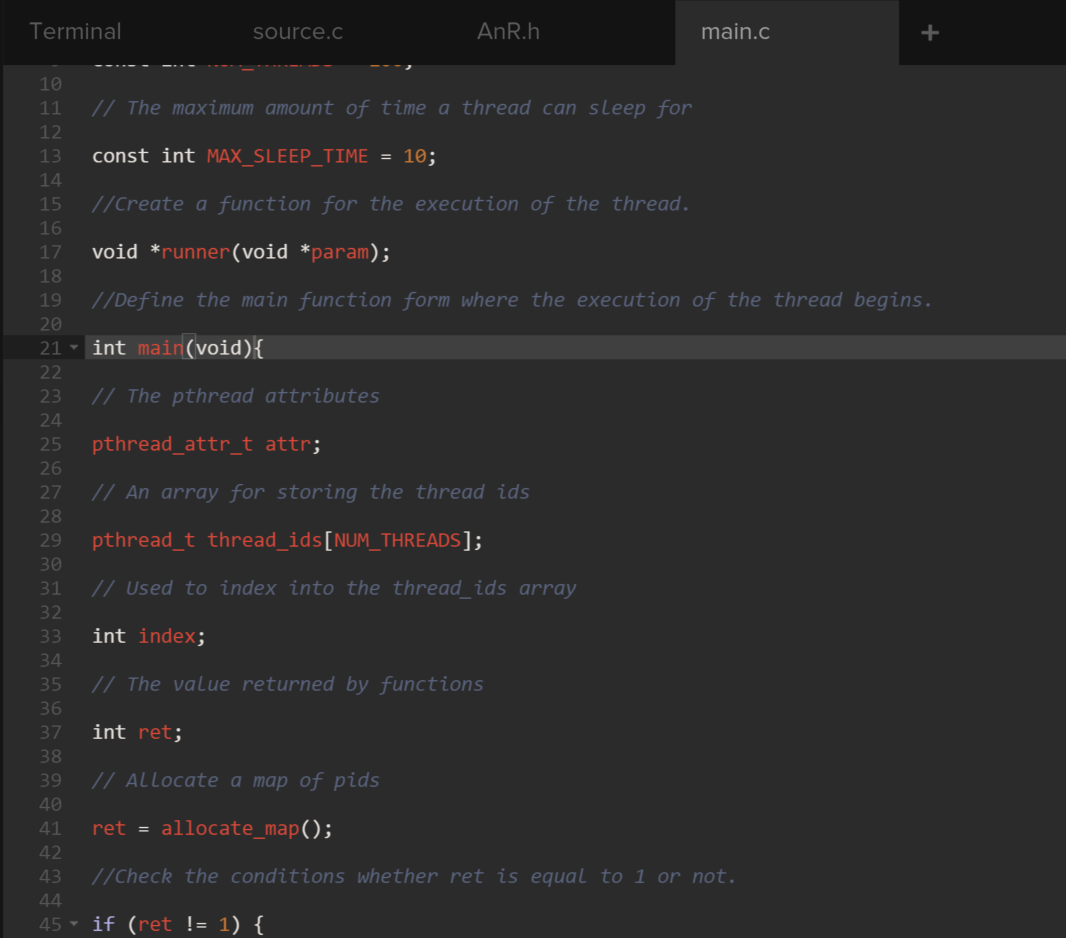
1. 

Create a header file and define allocate and release methods.

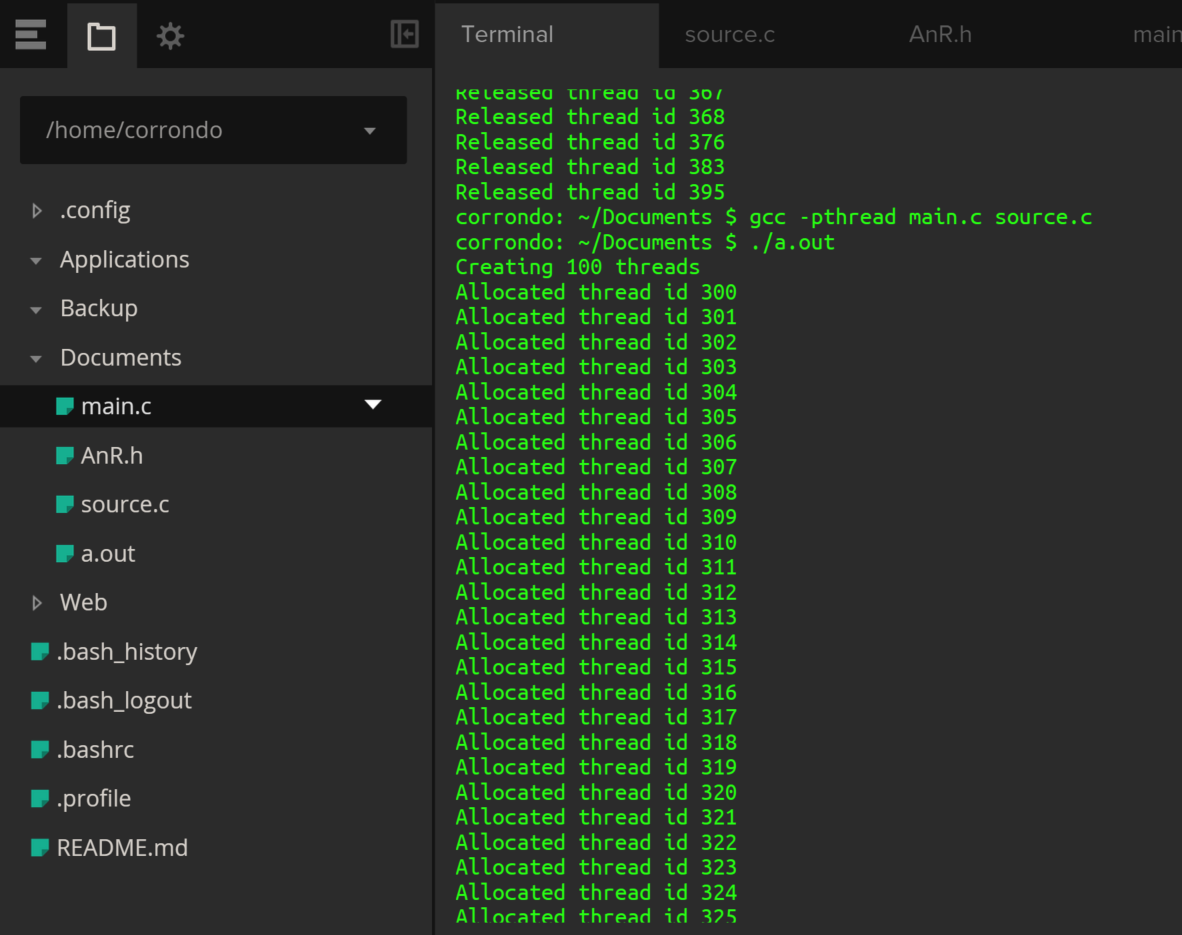
1. 

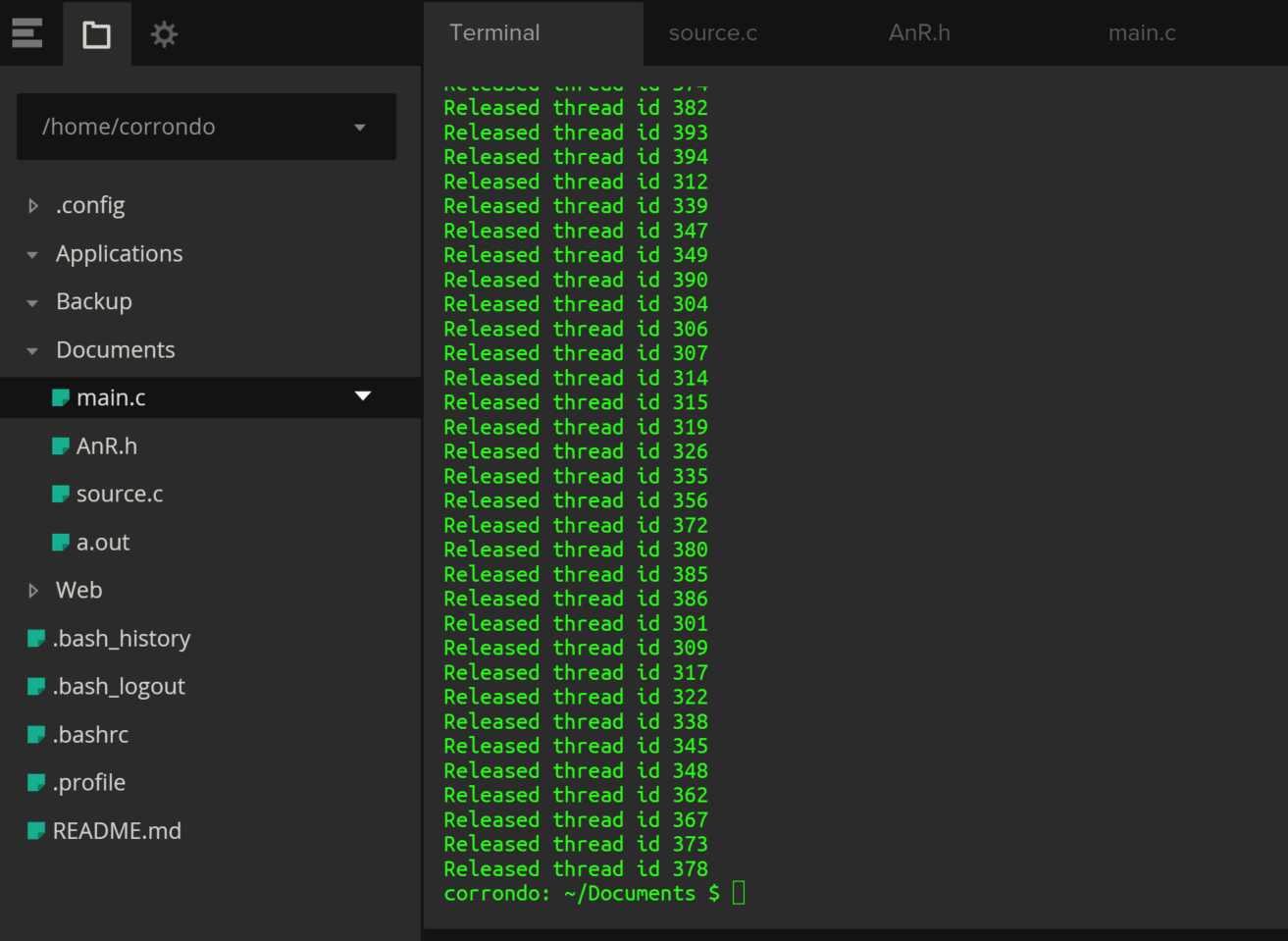


Create a C++ source file to implement allocate and release methods.

1. 

This main.c is used to execute the thread calling its methods

1. 



To ensure that the code is functioning, here I enter the commands “gcc –pthread main.c source.c” to run the files. As shown in the screen clipping, an output file is created (./a.out). Once entered into the command line to run it begins to create 100 threads. It also reports that it is allocating and releasing thread id’s. All id’s are unique for every thread.

References:

Operating System Concepts 9th Edition, Ch. 3-4, p. all

Operating System Concepts 9th Edition ppt. slides

https://computing.llnl.gov/tutorials/pthreads/#Abstract http://www.yolinux.com/TUTORIALS/LinuxTutorialPosixThreads.html

http://www.csl.mtu.edu/cs4411.ck/www/NOTES/process/fork/create.html

---Things learned in Chapter 3:

This chapter 3 clarified what the notion of a process means as it pertains to the operating system in a computer environment. This program of execution, which forms the basis of all computation is responsible for the scheduling, creation and termination of processes. Using shared memory and message passing as a means of exploration, I was exposed to inter-process communication. Such exposure facilitated the understanding of client-server systems communication. Amongst the most interesting points covered are the state in which a process may be found which are new, ready, running, waiting, or terminated. Its possible queues being the request queue and the ready queue. They could be independent or cooperators, etc.

---Things learned in Chapter 4:

Chapter 4 clarified the meaning and purpose of a thread. It explained that a thread ia a fundamental unit if CPU utilization that forms the basis for multithreade­­­­­­­d computer systems. It discussed the APIs of the Pthread, Windows, as well as Java thread libraries. I explored several strategies that provided implicit threading and also examined issues related to multithreading programing. Before examining issues related to multithreading, the chapter introduced three distinct models which were many-to-one, one-to-one, and the many-to-many. Operating system support for threads in Windows and Linux truly highlighted some major differences between the two unique OS’s. At the end it is obvious that chapter 3 and chapter 4 complement each other as a thread is ultimately described as a flow of control within a process. Such flow of control results in increased responsiveness to the user, resource sharing within the process, economy, and scalability factors, such as more efficient use of multiple processing cores