For the first phase, I have mostly just created the structure of what will(hopefully) expand into a robust mock operating system. I'm also a bit unsure of what a standard format for documentation would be so I hope this is sufficient.

Overview of different structures

-PCB

For the PCB struct, I have added all the fields I know I will need with hopefully relevant data types. The instruction text is left as a single array of characters because I was unable to figure out how to get an array of character pointers passed around effectively. This is something I would like to implement down the road. The PID field is how I am linking a PCB to its related Process.

-Process

The process struct I left as light as possible since I know there will be much more overhead for this struct later. Right now its only its process ID and a pointer to its PCB.

-Potentially registers

I was planning to add registers as their own struct, but i'm not positive what data will be needed for a register, so for the time being I left registers as an array of long pointers, one for each register. Looking into the "register" key-word in the C language as it seems relevant.

Overview of different methods

-Constructors

Basic constructor functions created to add initial values for things like PID, allocated memory, zero initialized arrays. Additionally the number of processes and next PID are changed for each process. A better function(hashing potentially) will be implemented down the road for more robust process ID's.

-UpdateState()

Simple way to pass a new state to a PCB. In the future, I plan to expand this method to change a processes queue or position in a queue when a state is changed.

-K&R methods

I copied a few basic methods from K&R's The C Programming language. Extremely simple functions(itoa, reverse) that I noted in the comments so I hope this is no issue.

-KillProcess

This was an area of difficulty for me. I created an extremely simple method to kill a process, but I still have memory leak issues so this method will absolutely be updated soon.

Pitfalls/Struggles/Planned expansions

-Had a lot of issues trying to figure out how to structure the content of my file. Not the structs or logic, but specifically how to separate the program into different files or potentially headers. I am extremely inexperienced with C but figured it was the perfect choice for this project so I'm learning as I go. Planning to set up a meeting with you to discuss ways to organize my code for scalability.

-There is a definite memory leak issue with my program and am having trouble making sense of the Valgrind output to determine exactly what is happening. Another topic to discuss with you during a short meeting at the beginning of the next iteration/sprint. The Valgrind script as well as its output will be included.

-For the first phase, I chose to implement the user specifications via command line arguments. This is something I plan to update to more of a running function to allow the user to interact with the operating system, not execute it once.

-Like many others, this project was a ton of brand new content in a language I'm not especially confident in. I have done my best to structure my code and objects(structs) to allow for easy additions/expansions like adding a queue/virtual memory address/etc.

-Initial values for things like memory or the max instructions/registers are chosen to be far in excess of what will probably be relevant values. I have #define'd a few values at the top for easy adjustment once I have a better idea of what amounts of memory will be appropriate but also minimalistic.

-Templates are still something I was having a bit of trouble understanding. The "seeds" made sense, but in practice it seemed easier to add a randomized number cycles to the end of each instruction. I'm sure this is not the best way to go about this so it will be something to clear up in the next iteration.