Divya Shrivastava
CMSC 312 – Introduction to Operating Systems
Professor Bartosz Krawczyk
12/10/2019

CMSC 312 Project Documentation

When working on this project, I aimed to complete all of the criteria in the B grade section. In order to do this, I started off this project by creating a process for each program file and Process Control Block (PCB). My PCB is not its own class. Instead, each part of the PCB is created as the process itself is created. After, I had created a single scheduler – first-come first serve. However, after a few iterations, I changed the original scheduler to be Shortest Job First, and I implemented another scheduler – Round Robin (RR). I made a comparison such that my code displays how the two schedulers would run the processes, comparing them. Eventually, I added process priorities to RR scheduler by randomly generating values to the process ID's and then re-ordering them accordingly. Throughout this project, I have been implementing I/O interrupts and handlers to be able to run my project without any errors.

After implementing the schedulers, I have added basic memory and operations to keep track of how much memory is available before and after running each process. Next, I implemented Multithreading where four threads would be running simultaneously. I did this by creating other multithreading classes to create the process. After, I implemented piping as an inter-process communication method between the parent process and one child process. I also implemented a variable between these two processes that the child would then read display the variable number, indicating communication between parent and child. Thereafter, I had been working on paging. At first, I created a physical memory class to assign physical memory space to the processes. After, I was able to create an address translator class that accesses backing store class, page table class, translation lookaside buffer class in order to create virtual and physical addresses for each process. Furthermore, these values, along with the page number, frame number, and offset are stored in the TLB and page table. I implemented Second Chance Algorithm, as well.

To tune this project, run the project from the OperatingSystems class. The user will be prompted to enter how many times they would like to process each program file, as well as how many numbers of cycles. Then, the project will run from there. Toward the end of the running, the project will have a GUI pop up with information about each process.

I have implemented the following criteria:

1. Requirement #1: Process Implementation and PCB – file: Process.java, line: 72-80, 94-108; MultiThreading.java
2. Requirement #2: Critical section within each process – file: MultiThreading.java, line: 18, 34, 65; file: MultiThreading2.java, line: 20, 36, 69; file: MultiThreading3.java, line: 22, 37, 66; file: MultiThreading4.java, line: 16, 31, 61
3. Requirement #3: Critical Section Resolving Scheme – file: MutexLock.java, line: 1-24; file: MultiThreading.java, line: 35, 39; file: MultiThreading2.java, line: 37, 41; file: MultiThreading3.java, line: 38, 42; file: MultiThreading4.java, line: 32, 36

4. Requirement #4: Single inter-process communication method – file: Process.java, line: 18, 19, 127-136; file: MultiThreading2.java, line: 89-100; file: MultiThreading3.java, line: 79-97
5. Requirement #5: Single level parent-child relationship - file: Process.java, line: 115-125, 202-217; file: MultiThreading.java, line: 88-112
6. Requirement #6 & #7: Scheduler, 2 Schedulers and Comparison – file: Process.java, line: 182-183, 190-191; file: SJFScheduler.java; file: RRScheduler.java
7. Requirement #8: Process Priorities – file: RRScheduler.java, line: 16-36
8. Requirement #9: Basic memory and operations on it – file: MultiThreading.java, line: 41-61; file: MultiThreading2.java, line: 43-64; file: MultiThreading3.java, line: 44-63; file: MultiThreading4.java, line: 38-58
9. Requirement #10: Memory divided into hierarchy - file: MultiThreading.java, line: 41-61; file: MultiThreading2.java, line: 43-64; file: MultiThreading3.java, line: 44-63; file: MultiThreading4.java, line: 38-58
10. Requirement #11: Paging – file: MultiThreading.java, line: 73; file: MultiThreading2.java, line: 75; file: MultiThreading3.java, line: 74; file: MultiThreading4.java, line: 69; file: PageTable.java; file: PhysicalMemory.java; file: AddressTranslator.java; file: BackingStore.java; file: TLB.java
11. Requirement #12: I/O Interrupts and handlers: Throughout project, everywhere
12. Requirement #13: Multithreading via hardware – file: Process.java, line: 91-166; file: MultiThreading.java, file: MultiThreading2.java; file: MultiThreading3.java; file: MultiThreading4.java
13. Requirement #14: GUI – file: 197-205; file: JTableGUI, line: 1-49
14. Requirement #15: Loading external processes and generating new ones on user request – file: OperatingSystems.java, line: 27-39
15. Extra functionality: Second Chance Algorithm – file: Process.java, line: 194; file: SecondChancePageReplacement.java, line: 1-72