**Report**

As part of your report, answer the following question about Task 2:

1. How do your system calls work?  Provide a pseudocode algorithm for each and elaborate on any

noteworthy steps or problematic areas.

As part of your report, answer the following questions about Task 3:

1. Explain how you manage the memory.

**When a process starts we create a Bitmap and pass in the number of physical pages we have available to use. We use this Bitmap to keep track of used/unused pages within our memory. The actual process is then assigned to our PCB, which contains a unique process id (PID), the parents PID if there is any, the current thread the process is using, and the next/previous processes within the PCB. Our PCB is constructed using a doubly linked list, which we found to be the fastest and easiest type of list to keep managed. The PCB is responsible for handling all of the appending/removing of the required processes, as well as retrieving the processes PID when we need them. Using the Bitmap, and PCB has allowed us to always know how much memory we have to use, as well as giving us the ability to retrieve the required active processes running at anytime.**

1. Describe your memory allocation and deallocation scheme.

**Allocation:**

**Get the address space of the process.**

**Get process PID.**

**Check to see if there is room for the process using first fit, worst fit, best fit, using the Bitmap as reference to available memory locations.**

**Create a new PCB element and assign required information, PID, thread, previous/next processes, and set the Process to Valid.**

**Append the process to the PCB list.**

**Write the PID to register 2.**

**Advance the program counter**

**Deallocation:**

**Zero out the target page table using bzero and the target’s location in the table.**

**Get the targets starting location in memory and clear the Bitmap to the end of the target’s memory usage.**

**Delete the target process’s page table.**

**Get the target’s PID from the PCB.**

**Remove the target from the PCB using its PID.**

**Get the target’s parent from the PCB and wake it up (If it exists).**

**Call currentThread->Finish() to kill the target thread.**

3. How do you start a user process?  Walk through the procedure step by step.

As part of your report, answer the following question about Task 4:

1. Did any particular memory allocation scheme prove more difficult to implement, debug, or test than

the others?  If so, why do you think this happened?

***The Memory allocation schemes where relatively easy to implement compared to the rest of this project. To test we had to figure out how to create a “hole” in memory so that we could test that memory allocation for best and worst fit where working correctly. But once we figured out this it was smooth sailing from there.***

Task 6 report:

1. What problems did you encounter in the process of completing this assignment?  How did you

solve them?  If you failed to complete any tasks, list them here and briefly explain why.

2. What sort of data structures and algorithms did you use for each task?  Did speed or efficiency

impact your choice at all?  If so, how?  Be honest.