CSCE 4600
Spring 2016
Project #2
Due: May 3, 2016 at 11:59pm via BB Learn Dropbox

For this project you may be working in groups of up to three students in the class. Each member of your group **must** sign up on the corresponding project sheet in order to receive a grade. Your task is to develop a simulation of different scheduling disciplines that allocate a set of processes to available processors.

Using the *process generator* that you developed in Homework #2 and used in Project #1, generate a set of 64 processes with different memory and cycle (i.e. runtime) requirements. You may assume that the processes arrive in the system every 50 cycles (e.g., at times 0, 50,100, 150, 200, …etc.). Also, you can assume that once a process has started, it will run to completion (i.e., there is no scheduling). However, a process can only start executing if the required memory is available and can be allocated.

1. Assuming that the combined memory requirement of all 64 processes is less than 20 MB, use the system calls *malloc()* and *free()* to dynamically allocate and de-allocate memory to your processes. Measure the total system time that is required to simulate the execution of your 64 processes.

2. Develop your own memory management system with corresponding function calls *my_malloc()* and *my_free()* to manage a pre-defined block of memory of size 20MB. The function *my_malloc()* will try to find the required chunk of memory within the pre-defined 20MB block and allocate it to an arriving process. Upon completion, the memory partition is returned to the memory pool via the function *my_free(), to be used by other processes.* Compare the performance of your memory manager to the performance of *malloc()* and *free()*. Note that you memory manager does only call the system's *malloc()* once to request the initial 20MB block, hence you are reducing the number of context switches.

3. Assume that the available memory is less than the sum of requirements of all 64 processes. Modify your system such that a process can only execute if the required memory can be allocated, otherwise, the process will have to wait in a special queue for memory to become available. For a given set of 64 processes, analyze the overall completion of your simulation under the assumption that:

   - The system only has 50% of the memory required by the set of 64 processes.

   - The system only has 10% of the memory required by the set of 64 processes.

   Of course, the required memory per process is always guaranteed to be less than system memory.

<u>Deliverables</u>: Write a 5-page report that highlights your implementation of the memory manager and shows the results of the performance analysis. You must discuss all the limitations and assumptions. You must submit well-documented code

Have FUN!!