Assignment 6 – Virtual Memory Management Assignment
Due Sunday April 29th by 11:59pm
Submit using the following link: https://classroom.github.com/g/clC9zX53

Description:
Another important task of an operating system is memory management. In this assignment, you will build a program that simulates two page-replacement policies in virtual memory management. In general, your simulator should manage the pages of each process in the system according to their page requests.

You should be building up the memory management on top of your scheduling simulator from assignment 4. This means that with the added events of process arrival, execution, and termination, there are also events for page requests and page faults. You can assume that the overhead of page faults (loading a page from disk to main memory + replacement if needed) is negligible. Your output should be similar to your previous simulator indicating the occurrence of the different events.

Design Details:
1. Your system has the following general settings:
   a. Only paging is used.
   b. You can have unlimited disk space.
   c. Your main memory is large enough to hold the resident sets of all processes in the system.
   d. The fetching policy is on-demand.
   e. The resident set size is of a fixed-size of 4 for all processes.
   f. The placement policy is First Fit.
   g. The replacement policies that you will be evaluating are the LRU and FIFO.
   h. You can ignore cleanup for this assignment.

2. In the Process class, you will need to store the information about the pages needed by the process. Specifically,
   a. Generate a random stream of page requests, one request for every unit of time that the job will be running. This could be done in the Generator class if you’d like.
   b. You will need a page table to keep track of the pages in the memory.
   c. Keep track of the number of page faults, to print out at the end of the simulation.

3. Write a class called MemoryManager to simulate the memory manager for an operating system. The memory manager should have access to the processes’ page tables to translate the “logical” page numbers to their corresponding “physical” frame numbers.

4. Create a driver class that will start the simulation and print results into a final results file named results.txt. Make sure to display the address translations in your output as well.