Kevin Nguyen

1) List what you have done and how you tested them.
    a) I have completed implementing the functions MMU, get_PFN, pagefault_handler, fifo, lru, and clock. I have tested all of them by running my program with different input files and comparing the output with the vm_reference program by doing a diff on the output. For the clock, since vm_reference does not show the output, I did the algorithm by hand and checked that the result is correct.
2) Explain your design of data structures.
    a) Most of the design of the data structures applies to the page replacement algorithms functions.
        i) FIFO: Pages to replace are kept track by using a pseudo queue. Since the indexes are the same as the PFN, the function uses an integer, called fifo_tail, to determine which PFN to return. First, the function will copy the value of fifo_tail to a variable. Then it will increment by 1 then mod by MAX_PFN. Lastly, it will return the previous variable.
        ii) LRU: The data structure used to implement this function is a double linked list. When the program starts, all of the page faults will call the pagefault_handler function. That function will initialize the linked list by adding all of the free frames. Afterwards, when there is a page hit, that page will be moved to the top of the linked list. If there is a page miss, the bottom page will be removed and its PFN will be returned. Also, the new page will be placed at the top of the linked list.
        iii) CLOCK: There is an array that keeps track of all of the reference bits. First the program will initialize all of the bits to 1. If there is a page hit, the function will change the page reference bit to a 1. When there is a miss, the clock function will loop through the array. If the bit is 1 then change the bit to 0 and move to the next index. If the bit is 0 then change the bit to 1, move the clock head, and return the index as the PFN.