Kresimir Tokic  
CMSC412 Final Project  
10/5/20  
  
This project simulates various paging algorithms as set forth by the project rubrics. I put everything into one Main.java file. When you execute the file, it expects two command line arguments, one for the number of physical frames and the second the size of virtual memory. The initial on-screen prompts and user experience flow force you to either generate a random reference string or input your own manually one integer at a time. The program will check that the user’s input a valid integer between the specified values of 0 to 9. Once a reference string has been successfully created, the full menu will appear at which point the user can create new reference strings, display the string or run the various paging algorithms. This user experience control will make sure the user can not attempt to simulate a paging algorithm on an empty or invalid reference string. The user will remain in a loop until they select 0 to exit the program. Each Menu item is executed from its own method, each paging algorithm is contained within its respective method, with the exception of OPT which uses two methods, the first is the primary execution of the algorithm and the second predicts how the frame that will not be used. During the execution of a paging algorithm the on-screen output will display the element of the reference string, what is populated in the physical frames and running total of page faults. The user will be prompted to press a key to continue through each element of the reference string. Upon completion of the algorithm the total number of page faults will be displayed. My implementations of these algorithms are not the most efficient possible. I’m not sure how the pros would do it but based on my reading it seems many operating systems are using combinations of these algorithms. I would assume the pros have fine-tuned these to their best possible time complexities. I’m also curious what the most ideal data type in Java would be each of the algorithms. Some of the ones I used are unsorted and others are sorted. In some algorithms the unsorted data type is fine but others I found easier to track and trouble shoot if it were sorted.