* 1. Create a service call with 1 input parameter. Put it in fault.c, AT THE VERY END!! It will perform the following operations inside the kernel:
     1. When the parameter is 0
        1. Initialize a page fault (miss) counter, no returned value
        2. save the PID of the current process
        3. Start counting misses
     2. When the parameter is 1
        1. save the counter in a local variable
        2. zero out the counter
        3. zero out the saved PID
        4. return the value of the counter to the user
  2. Modify fault.c as follows:
     1. insert a global integer variable for counting page faults
     2. insert a global variable to save the PID of the process that will be “watched”.
     3. determine if the current page fault was caused by the PID you saved in A.1.b). This simple if statement will be somewhere inside a function that already exists in fault.c. You will have to figure out where that is.
     4. if the current PID == the saved PID, increment your global counter variable , if they do not match, do nothing
  3. Create 2 user programs that do the *syscall*’s & create page faults.
     1. Version 1
        1. Write a program that defines three distinct arrays (A, B and C) of 100,000 integers each.
        2. Initialize the vectors anyway you want to
        3. Start timing.
        4. Loop through the vectors multiplying the entries and putting the answers in a third vector.
        5. End counting faults.
        6. Print the number of faults and the total CPU time it took to count them.
     2. Version 2
        1. Modify the program to contain a SINGLE struct array that keeps the corresponding numbers from the previous vectors A, B and C ADJACENT to each other (that is, A, B and C will all be in a structure so it they are adjacent in memory.)
        2. Repeat the steps from C1b to C1f above.
        3. Submit the SOURCE version of all Linux files that you changed or created, plus the source of your testing programs (from B & C above). Your testing programs MUST be on your TJW Linux machine and the root password for your machine must be sent to the TA so he/she can compile and run the test program.
* Put your kernel function at the END of  
   /usr/src/redhat/BUILD/kernel-2.6.18/linux-2.6.18.s390x/arch/s390/mm/fault.c
* Keep your testing program in your Linux home directory.

**NOTES:**

1. If you need your own variables for process-specific data within the kernel, you must manage this data carefully. The PCB is NOT a good place to put it, but if you DO put any data there, you will have to re-compile the ENTIRE kernel.
2. None of the above programs are allowed to poll for completion of any operation
3. Be sure to modify all 5 files listed in my “files to modify” slide.