OSSIM – Objective 2

- Overview
  - Develop functions which initialize the memory and memory management data structures of the simulator
  - simulate the basic functions of the CPU, as well as provide more simulation output.
  - Boot:
    - The simulator will initally call your Boot() function that will load programs from boot.dat that are stored in the format described in intro.doc.
    - Boot() will also initialize the data structures responsible for managing the simulated memory and will call Get_Instr() repeatedly to read instructions from boot.dat and will store them in the simulated memory.
    - Finally, Boot() will call Display_pgm() for each program in boot.dat to output it to simout.

OSSIM – Objective 2

- Overview
  - After Boot() has completed XPGM() will be called which simulates a context switch and then calls Cpu().
  - Cpu() sets the memory address register (MAR) to the value passed to it by XPGM(), this will be 0 initially.
  - Cpu() then calls Fetch() to get the next instruction to execute from memory.
  - Fetch() calls Mu() to determine the physical location in memory of the requested instruction and uses the result to return the instruction to Cpu().
  - Cpu() then handles the instruction accordingly depending on the operation. This entire cycle then repeats until there are no more simulation events.

OSSIM – Objective 2

- Important Variables
  - MEMMAP
    - A pointer to 2* MAXSEGMENTS of type struct segment_type
    - User memory
      - MEMMAP[0] … MEMMAP[MAXSEGMENTS - 1]
    - Kernel Memory
      - MEMMAP[MAXSEGMENTS] … MEMMAP[2 * MAXSEGMENTS - 1]
    - Each segment_type
      - segment length in instructions (seglen)
      - the base address (membase) in memory where the segment begins.

OSSIM – Objective 2

- Important Variables
  - MEM
    - A pointer to MEMSIZE array of data types of type struct instr_type (defined in osdefs.h).

OSSIM – Objective 2

- void Boot(void)
  - This function is called from simulator.c and reads from boot.dat and initializes the memory and memory management data structures.
  - The programs from boot.dat represent the OS and are loaded into the upper half of MEMMAP.

OSSIM – Objective 2

- **Directions:**
  - Read the file boot.dat whose file pointer is PROGM_FILE[BOOT] and whose format is given in intro.doc. You will have to check for PROGRAM on the first line and read in the number of programs in the file. Then read in each segment and store the access bits and number of instructions.
  - With the program and segment data initialize MEMMAP starting at segment MAXSEGMENTS. The size of MEMMAP is 2 * MAXSEGMENTS. The first half is reserved for user memory, while the upper half is reserved for the OS.
  - Call Get_Instr() repeatedly to read instructions from boot.dat and update TotalFree and FreeMem based on the number of instructions read from boot.dat.
  - Call Display_pgm() to display each program.