**Lab #4 Design Document**

Ryan Goulding, Justin Estaris, Francis Hackenberger

a. We chose to use a first fit memory allocation algorithm, we chose this algorithm because of its greedy nature. The first fit algorithm allows us to process the dispatch list with no regard to what order the processes are in, it’s also greedy in the sense that it doesn’t care what came before or after the current process it’s processing.  
  
b. Our structure for queueing is called Node\_t, it’s a singly linked list that contains a reference to the next node in the queue and a process struct containing the process information. The proc struct is used to contain a processes details as integers such as: memory needed, printers neede, etc. Memory is assigned to the resource struct. The resource struct is a doubly linked list that manages which processes from the Node\_t queue are using memory, if they’re allocated, and where in the memory they’re stored.

c. Our program structure is organized as follows: hostd run’s the main program and contains the method that sorts through the dispatch file. Queue contains the two methods that are used to alter the queue. The first method is pop, it removes the last item from the stack and the second is push, it adds an item to the top of the stack. Utility contains methods and structs that relate to the memory management. The major methods in utility are alloc\_mem and free\_mem, the first allocates memory and the later frees it.

d. A multilevel dispatching scheme is used because of its approach to separating the different processing queues types, it allows the programmer to easily organize the different types of queues.