This assignment will extend your Linux-style shell code to incorporate memory management (again, only simulated). You will be asked to take a memory space of a given size (specified through commands described below), and apply first, best, and next fit algorithms to a sequence of allocation requests. You can modify your existing simulation code base or optionally start with a sample solution to the shell assignment, coded by Dr. Wilkerson (provided in Canvas).

The core functionality is contained in the stepping through the simulated time (memstep) and viewing of the memory space (memview). The other commands help by setting up the environment (or resetting the environment for a new run (memreset)) such as setting the algorithm (memalg), specifying the total space available (memset), and loading the simulation file (memload). The memview command should provide some visualization of the memory space (however you decide to show it). Note that this will likely be simple ASCII “art” and you might need to consider how it displays based on the size of the memory (for example, one \_ or X displaying for “empty” or “full” would work OK if the size was a few hundred, but not if the size was a few hundred thousand)… The required commands to support are listed below with their arguments.

memreset

memalg <first | best | next>

memset <size in units>

memload <file name to load simulation>

memstep [all | timesteps]

default for memstep should be a single time unit

memview

memrun <configuration file>

To expedite testing, memrun is provided to encapsulate multiple configuration aspects within a single file. This command should automatically reset the simulation space, implement the specified algorithm, set the memory size, and then provide the appropriate simulation data for loading. (All details needed are in the specification file as discussed below.)

This program is to be coded in C++ and submitted in Canvas before the deadline. If you require any additional configuration (e.g., C++ 11 standard), provide a Makefile with your submission. This is an individual assignment – each person should produce and submit their own code solution.

The format for trace data is simulation time, space required, time required. So

1 16 88

Would request 16 units of memory at simulation time 1 requiring it to be held until simulation time 89 (needs it for 88 time units). The memrun file will contain two initial lines one with either best, first, or next for the algorithm, followed by a line with a single integer for the memory space size.