**Compilation instructions**

g++ project4.cpp

./a.out

**Design Write-Up**

I used vectors to represent memory blocks. The vectors contain struct nodes that has a process name, memory size, starting address, and end address. My vector starts out with a single block of free memory and allocates processes into the memory. When RQ is called I look for a hole that is big enough for the new process when found, I assign it to the front of the vector (from index 0) if it is too big, I reject it. When RL is called, I look for the same process name and assign the process name “FREE”. When C is called, I move all the free memory to the end of the vector then I add all the memory of free spaces into a single node (the last node) then update the starting and ending address then delete all the useless nodes. One of the biggest problems I had was erasing the nodes in the vector in a loop. I deleted all the useless nodes in a loop but as I was deleting it, it would mess up the indexes, so I ended up deleting wrong nodes. I resolved this by storing all of the indexes of useless nodes into a separate vector then as I iterate through the vector of all the indexes, I would decrement all of the indexes every time I deleted a node.