The degree of success with this project is about 70%. I could not use the buddy system policy to accomplish the allocation requests and free operation. The main method that I take to handle this project is that I learn how to differentiate the feature between the first fit policy and best fit policy, and have a good understand the coalesce process when we come across free operation. The main frame is that I use the two “struct”: one is the request “ struct”, the other is the free\_list “struct”. These two “struct” help me to track the free element on the linked list, manage the allocation request and free operation. After I finish the main frame of my project, I finish the detailed code of this program.

First Fit 1MB: Total allocations 490 of 500.

First Fit 512KB: Total allocations 374 of 500.

Best Fit 1MB: Total allocations 492 of 500.

Best Fit 512KB: Total allocations 377 of 500.

Buddy System 1MB: Total allocations 0 of 500.(Failed)

Buddy System 512KB: Total allocations 0 of 500.(Failed)

First fit policy tries to find the first suitable free block, while best fit policy leaves the smallest fraction of the free block. So best fit policy would have more allocation than the first fit policy. I failed in implementing the buddy system policy, but theoretically buddy system would have less allocation than the First fit and best Fit because of the internal fragmentation. And also when total initial memory size increases, the total successful allocations also increases, this applies to all three allocation policy.

Email address: Wen\_Liang@student.uml.edu

command for the executable “assign5”:

./assign5 first 512 proj5\_data ./assign5 first 1024 proj5\_data

./assign5 best 512 proj5\_data ./assign5 best 1024 proj5\_data

./assign5 buddy 512 proj5\_data ./assign5 buddy 1024 proj5\_data