Semin Bae

Tony Mione

CSE320 / Fall 2024

Oct/30/2024

Report_HW4

Did you find the embedded instructions and hints in the source code helpful or confusing?

- The instructions and hints embedded in the code were very helpful in completing the task. For example, the part where the find_free_block function uses a for loop to iterate over the block list or the part where containerof is used to find the address of the block in the mem_free function gave clear directions when writing the code, making it easier to implement.

What section(s) of the code were most difficult to complete?

- The coalesce function implementation was the most difficult. In particular, the process of writing the logic to merge blocks based on four different cases was complicated. I had to check if the previous and next blocks were in use and then decide whether to merge them with the current block based on that. For example, if both prev and next blocks are not in use, these three blocks need to be merged into one large block. To do this, we need to update the header information of each block using bmgr->set_header. While doing this, I learned how important it is to handle the interrelationships and states between memory blocks.

Did this assignment clarify some of the memory management concepts discussed in the lecture?

- Yes, this assignment greatly helped me understand the memory management concepts covered in the lecture. For example, by implementing the process of appropriately dividing the remaining blocks in the mem_alloc function or adding new memory through extend_heap when necessary, I was able to deeply understand how dynamic memory allocation actually works. In addition, by freeing the current block and merging adjacent blocks in the mem_free function when freeing memory, I was able to experience how important memory management is to system efficiency and performance.

When I actually wrote and ran the code, the memory management concepts I learned theoretically in class became much more realistic. In particular, while implementing the four cases covered in coalescing into code, I realized how delicate memory merging must be. I felt that the logic for handling various cases through NULL checks and if statements is important in the process of deciding whether to merge the current block depending on whether the previous and next blocks are used. I also understood why memory allocation must be precise and the status of each block must be closely checked when expanding the heap area with the sbrk function. This assignment was a great help in understanding the detailed mechanisms of dynamic memory management, including memory allocation, release, and merging.