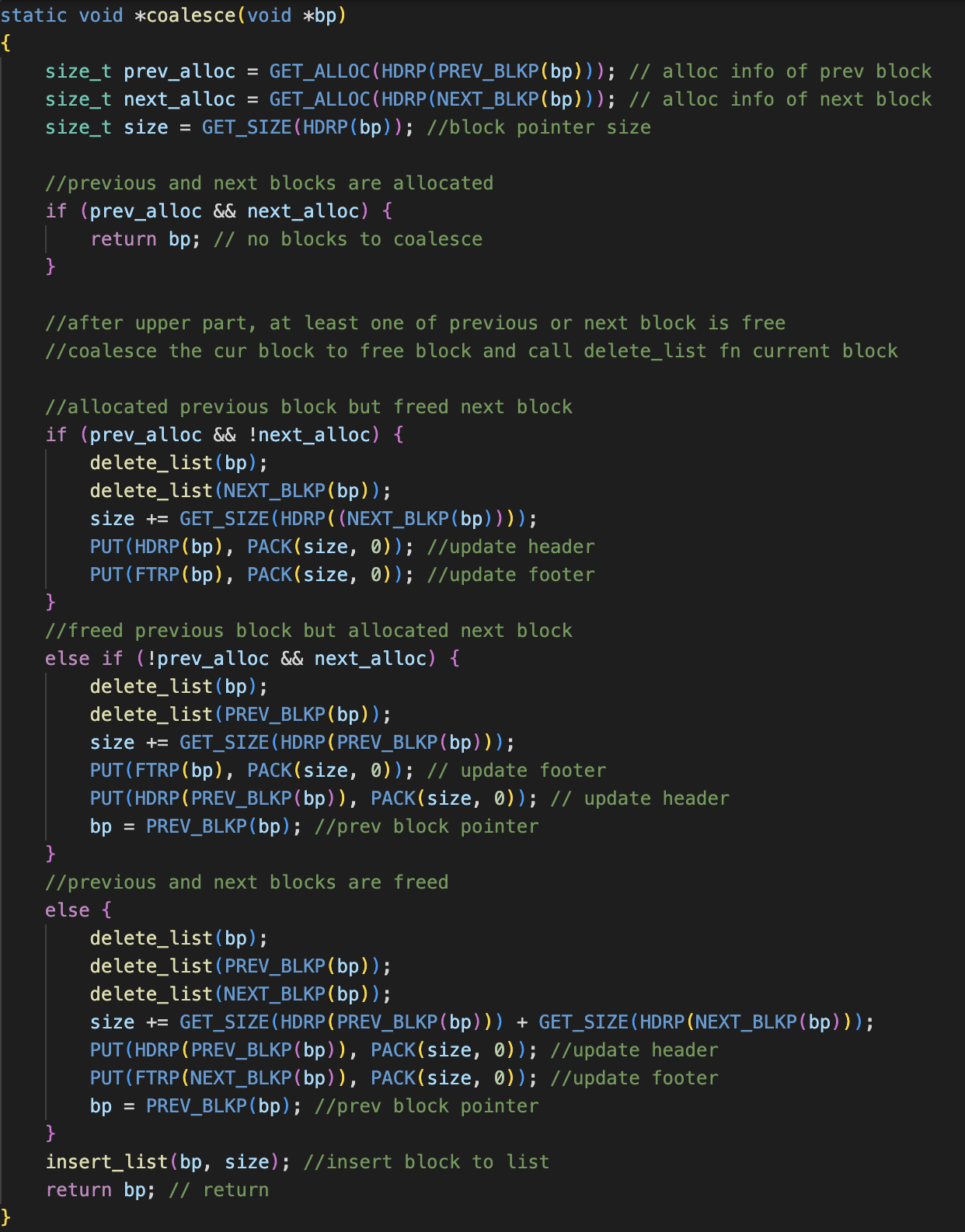
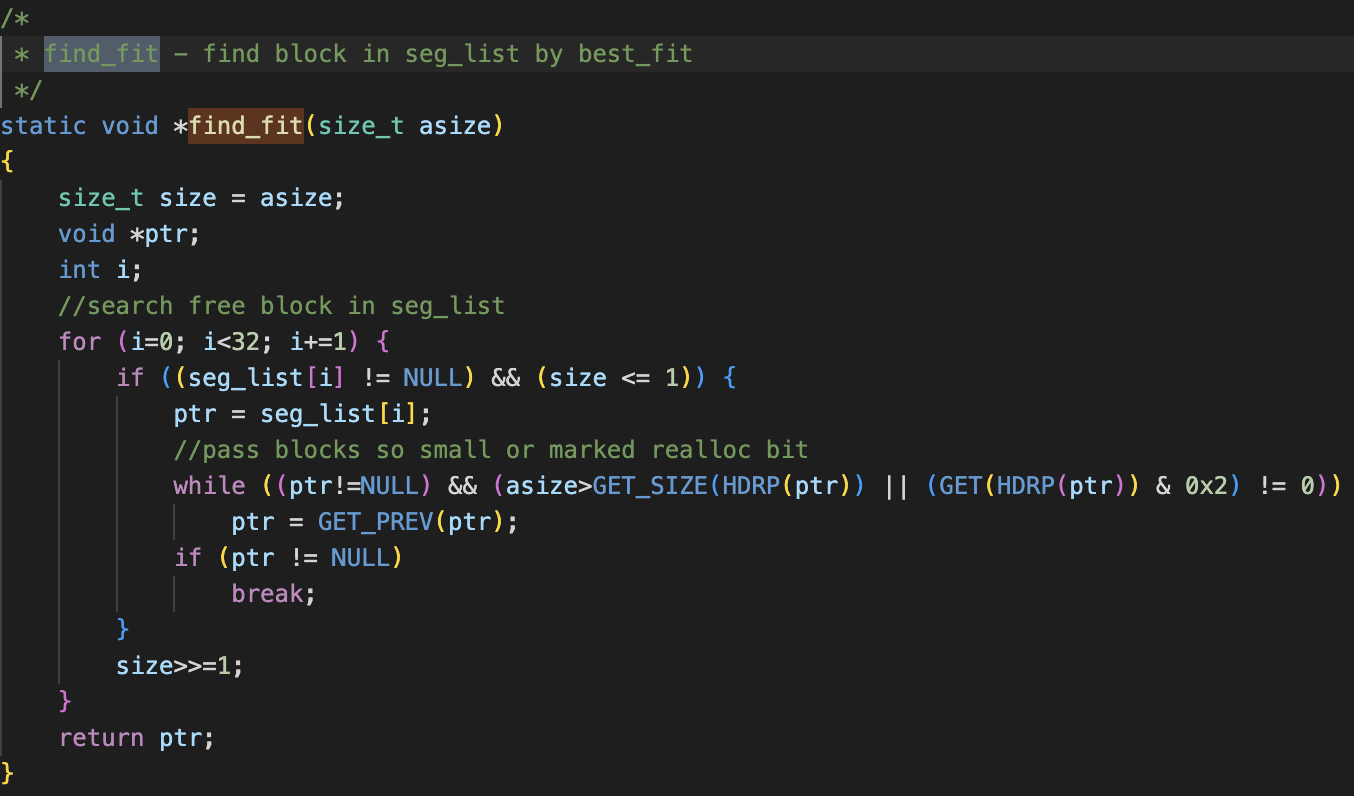
Lab11. Malloc Lab

20210815 정재현

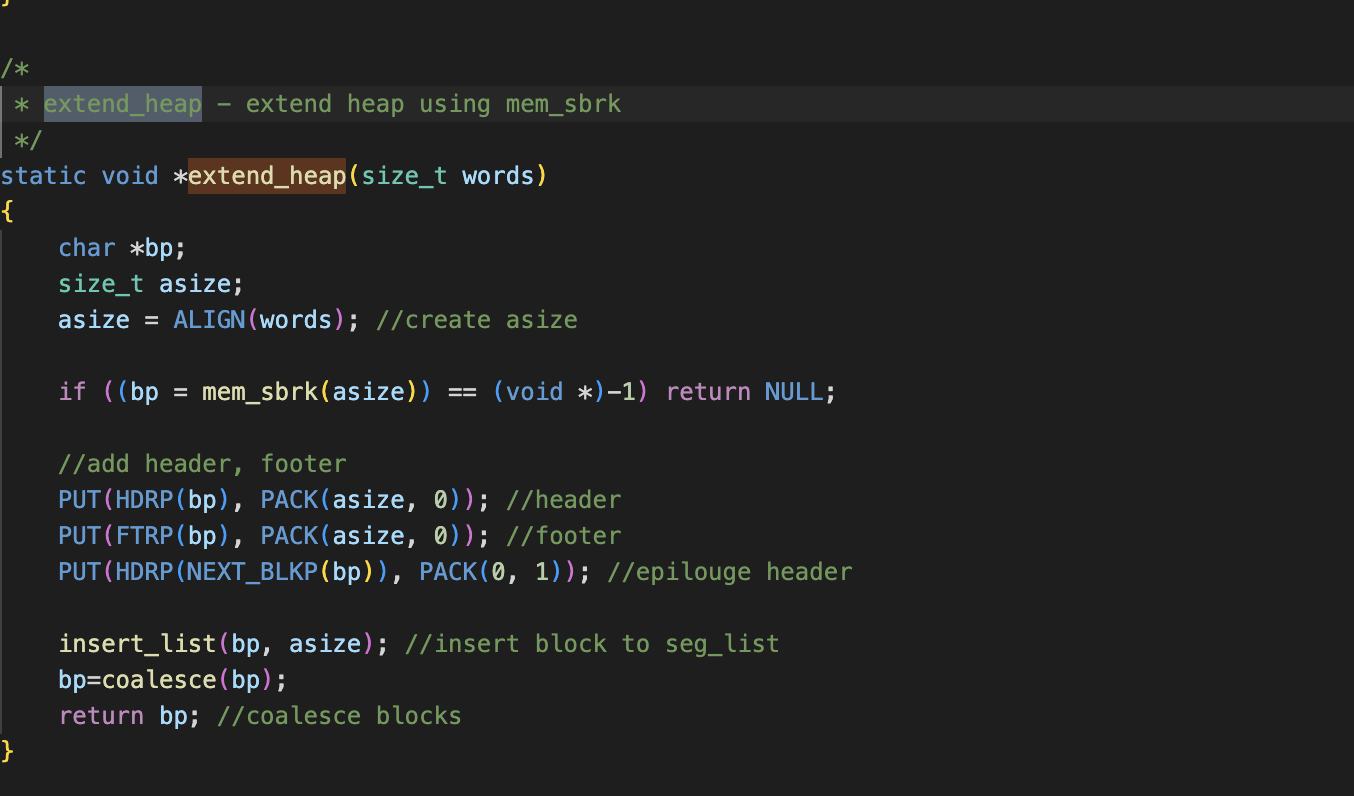
이번 LAB 과제는 Dynamic memory allocation인 malloc을 구현해보는 것을 목표로 한다. 나는 이를 구현하기 위해 segregated와 implicit, explicit 중 segregated free list를 이용여 구현해보았다. Search시 find\_fit 함수를 사용하였다. 기본적으로 mm.c에는 mm\_init, mm\_malloc, mm\_free, mm\_realloc로 구성되어 있었고, 이후 나는 find\_fit, place, extend\_heap, coalesce, insert\_list, delete\_list를 구현하였다.



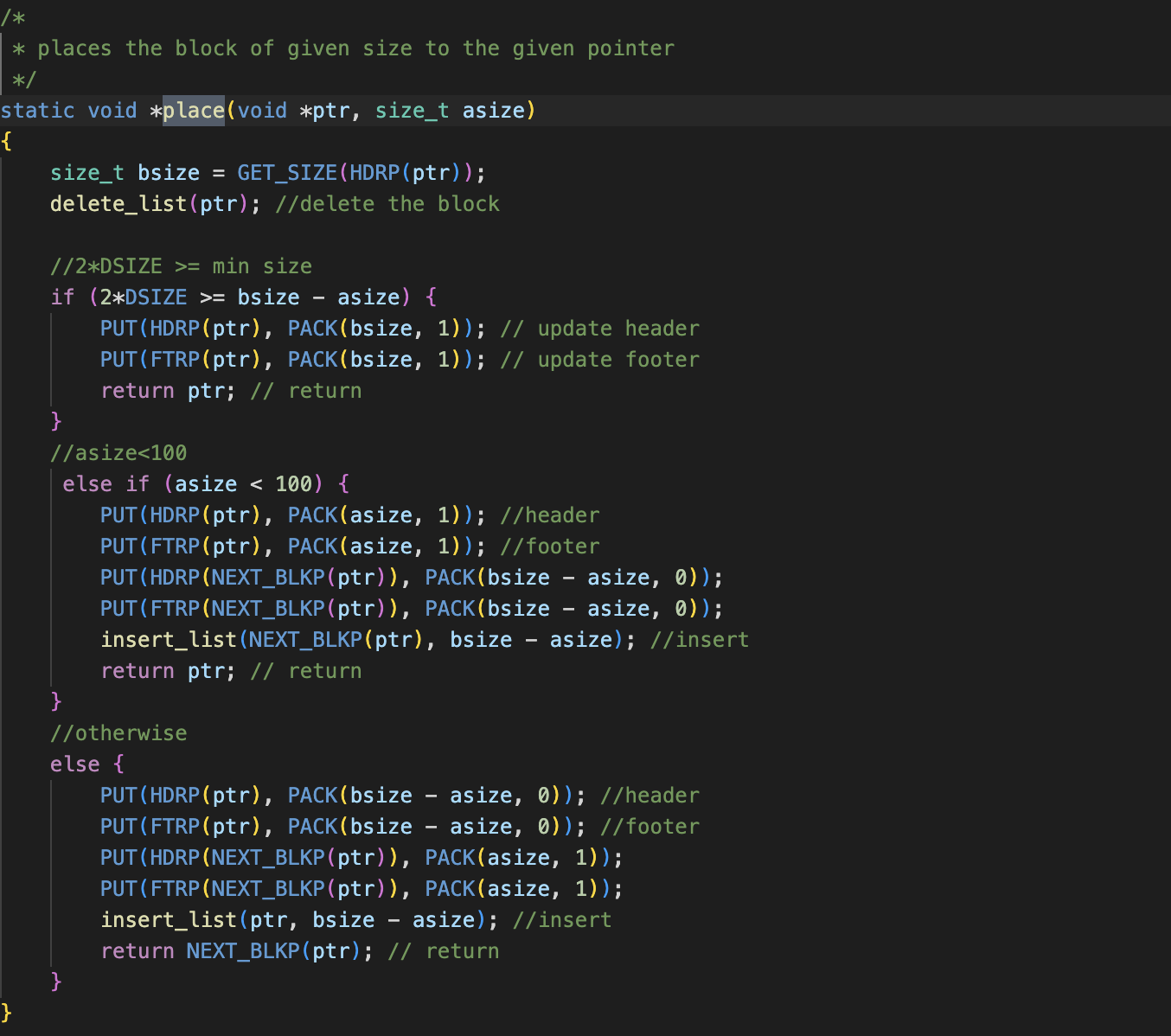
Coalesce의 경우 케이스를 나누어 앞뒤 블록의 free 여부에 따라 4가지의 케이스로 나누어 연결시켜 coalesce 기능을 구현할 수 있었다.



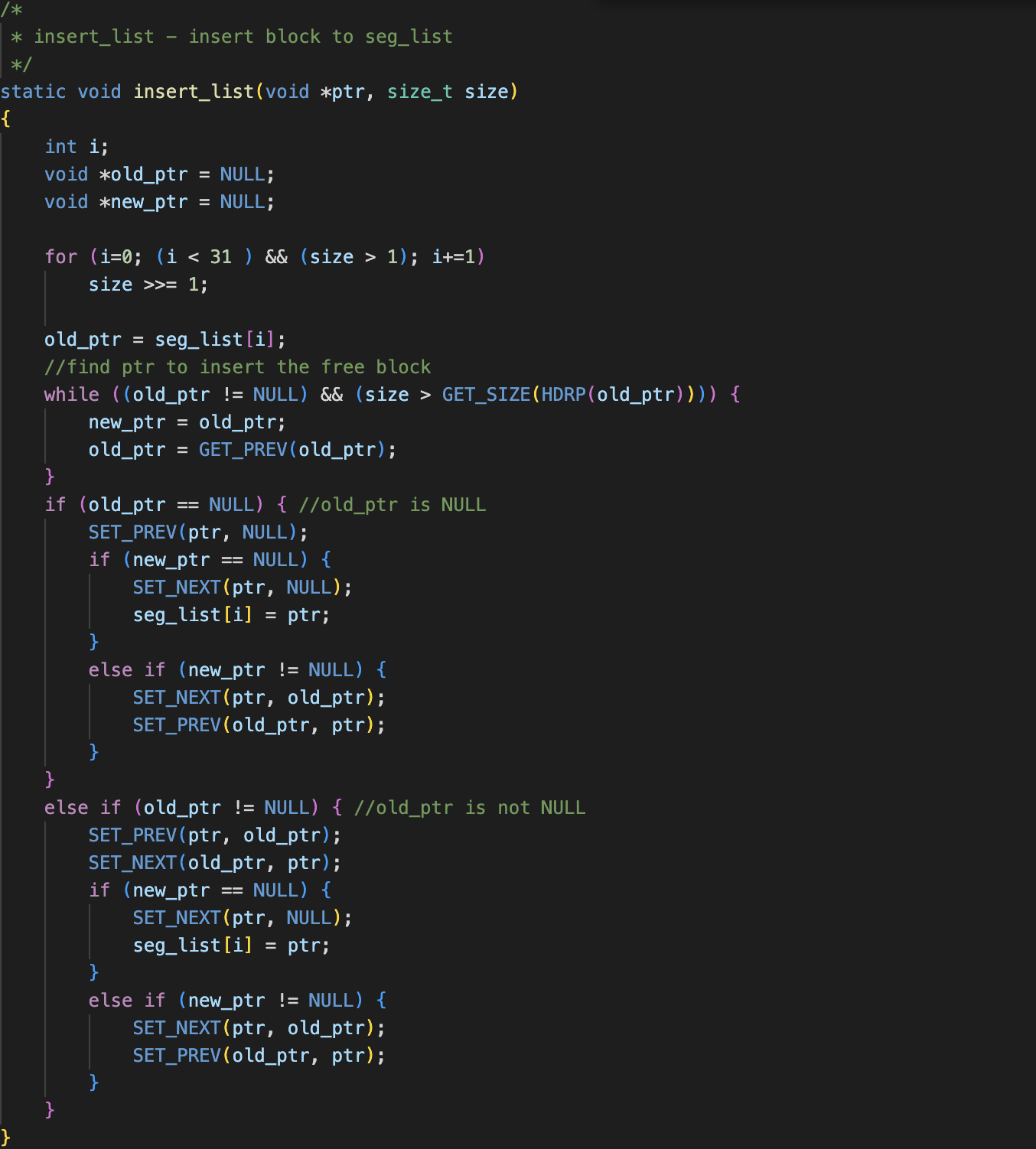
Find\_fit을 구현할 때에는 mm\_init, mm\_malloc, coalesce일 때 업데이트해주었고, 크기 블록이 지정된 크기와 같거나 더 크게 배치할 포인터를 찾은 뒤 반환하도록 구현하였다.



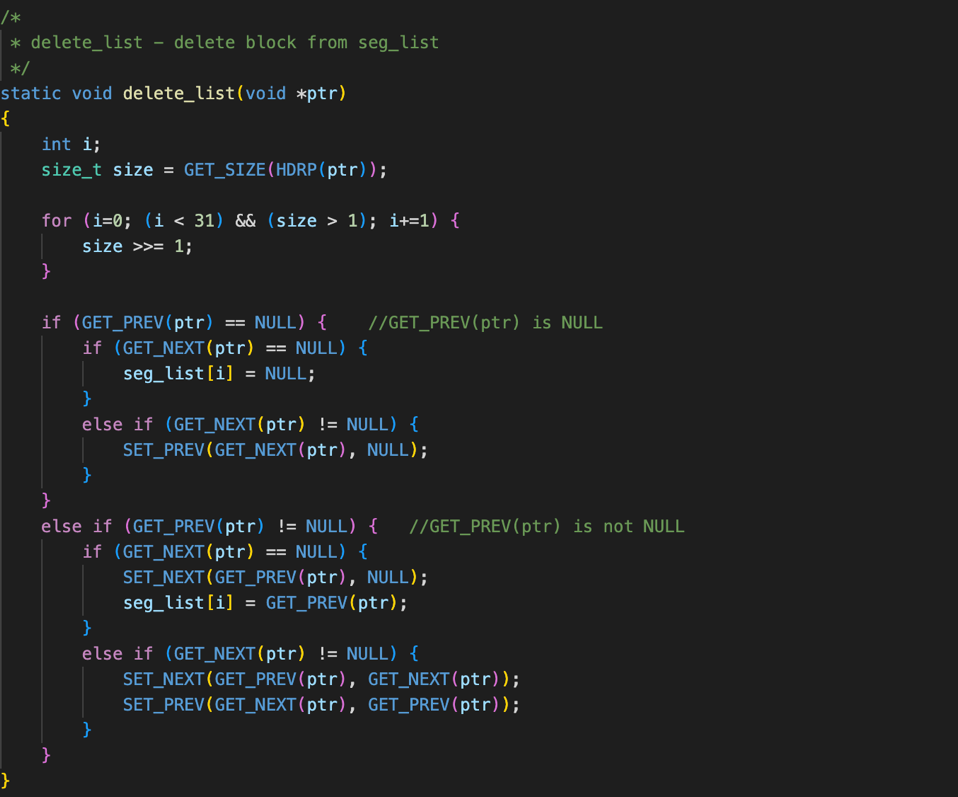
Extend\_heap의 경우는 heap을 초기화하거나, mm\_malloc이 적절한 search를 하지 못했을 때 호출 된다.



Place의 경우 지정된 크기의 블록을 지정된 포인터에 배치하도록 구현을 하였다.

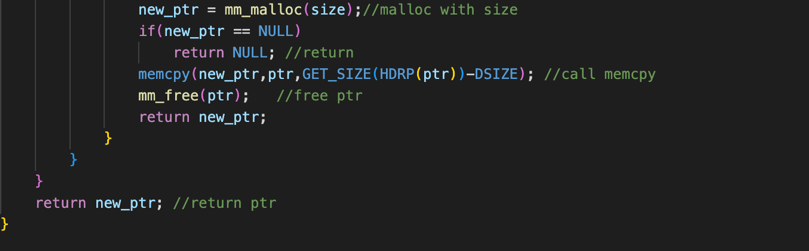


Insert\_list의 경우 segregated free list에 블록을 넣어주는데, seg\_list element들 간의 연결을 업데이트한다.

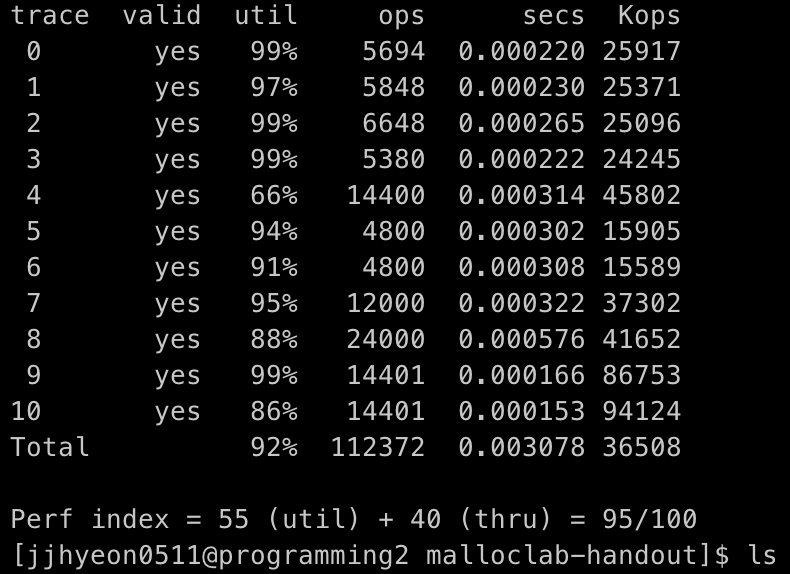


Delete\_list의 경우 segregated free list에 블록을 삭제하는데, 마찬가지로 블록을 삭제하면서 seg\_list element들 간의 연결을 업데이트한다..





Mm\_realloc의 경우 mm\_malloc 및 mm\_free 측면을 고려하여 간단히 구현할 수 있었다..



결국 make를 하고 ./mdriver -V로 확인해본 결과 위와 같은 결과를 얻을 수 있었다.

이번 과제를 통해서 memory allocate를 할 때 segregated free list, explicit, implicit 이 세가지 방법으로 이해할 수 있다는 것을 알았고, 단순히 동적할당이라고 생각했던 과정 안에 메모리 누수나 여러가지 구조와 함수가 있다는 것이 있다는 것을 알 수 있었다. 컴퓨터 메모리와 동적할당을 공부하며 메모리나 heap 구조에 대해 조금 더 알 수 있어서 좋았다. 나중에 기회가 된다면 이번에 구현했던 segregation free list말고도 다른 구현도 해보면서 이해해보고 싶다.