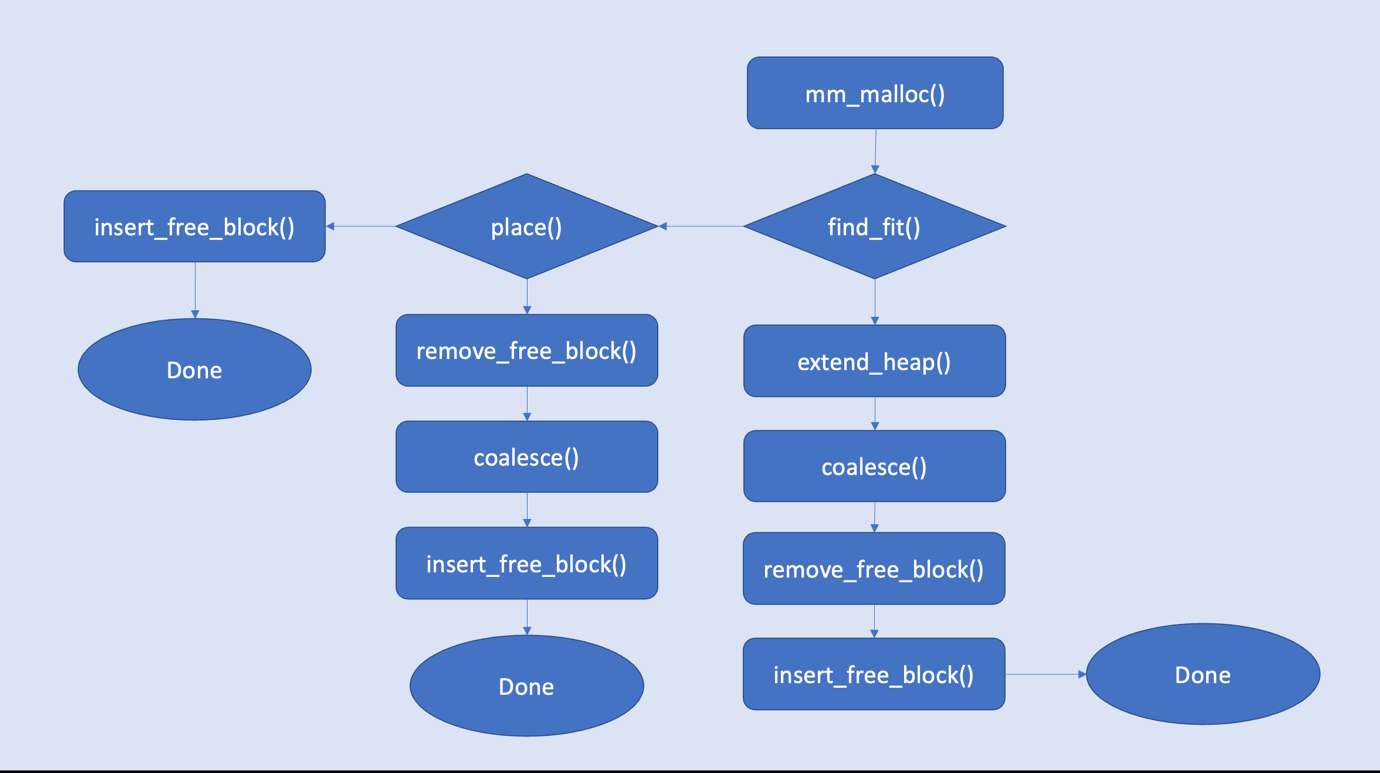
**System Programming Project 3**

담당 교수 : 김영재 교수님

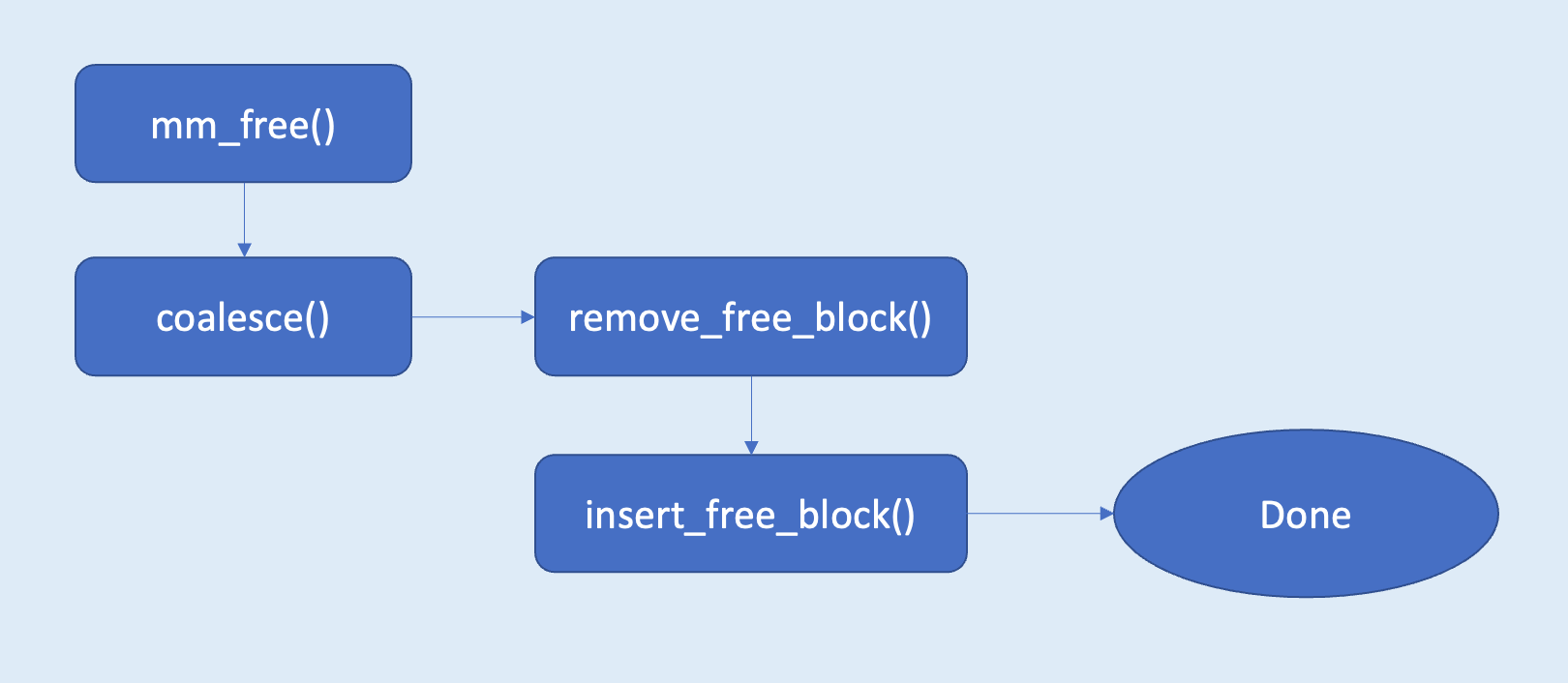
이름 : 박성환

학번 : 20160169

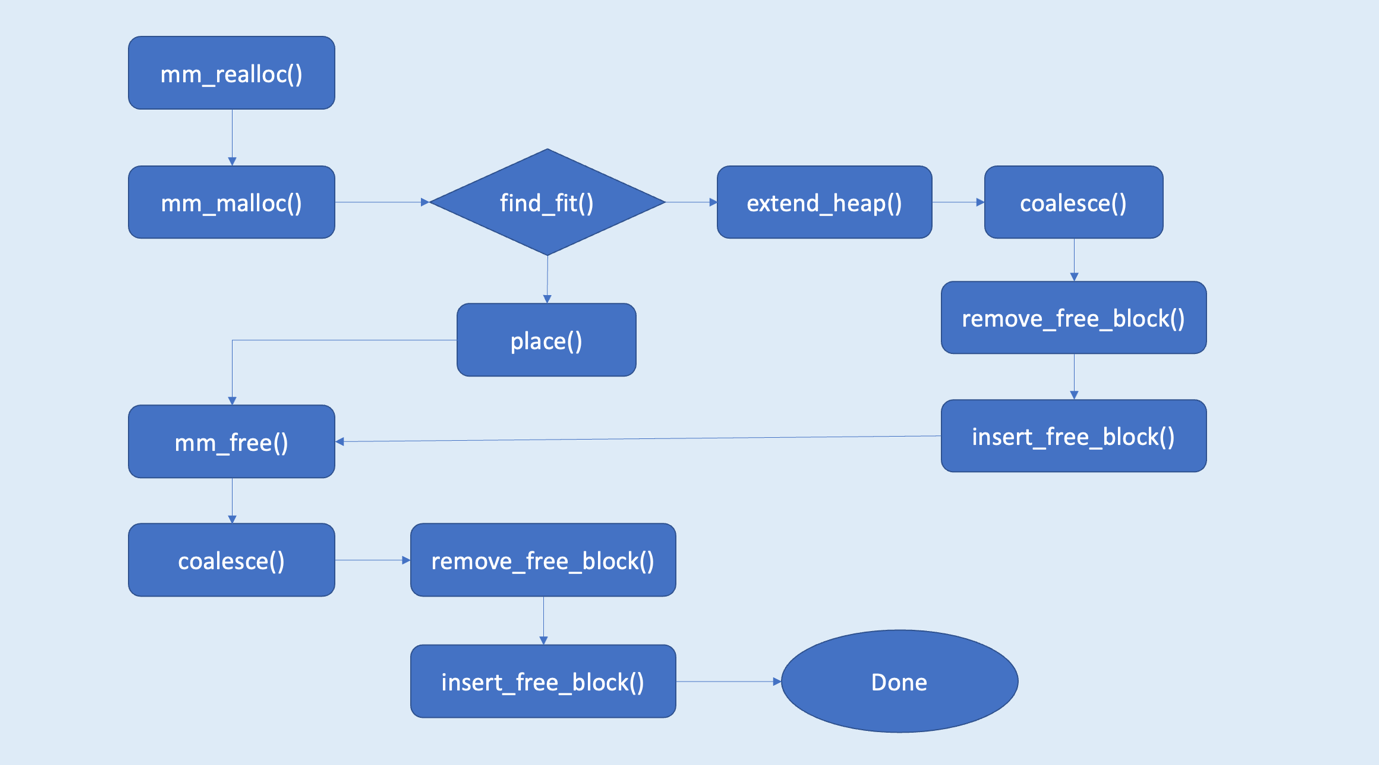
1. **개발 목표**
   1. Dynamic Memory Allocator를 구현한다. mm\_init(), mm\_malloc(), mm\_realloc(), mm\_free() 함수를 중점적으로 구현하고 segregated list를 사용하여 성능을 높일 수 있도록 한다.
2. **Flow** **chart**
   1. mm\_malloc



* + 1. mm\_malloc()를 호출하면 우선 find\_fit() 함수로 사용 가능한 free block을 탐색한다.
    2. find\_fit() 함수로 탐색에 성공하면 place() 함수를 통해 탐색한 free block에메모리를 할당한다. place() 함수에서 free block을 전부 사용할지 또는 split할지 정한다.
    3. split을 한 경우에는 해당 block을 segregated list에서 제거한 후 coalesce() 함수로 합친 free block을 segregated list에 다시 삽입한다.
    4. find\_fit() 함수로 탐색에 실패하면 extend\_heap() 함수로 heap 공간을 늘린다. 새로 할당 받은 free block을 coalesce하고 segregated list에 삽입한다.
  1. mm\_free



* + 1. mm\_free() 함수를 호출하면 mm\_free 함수 내에서 header와 footer를 해제한다.
    2. coalesce() 함수로 합친 free block을 segregated list에 삽입한다.
  1. mm\_realloc



* + 1. mm\_realloc()을 호출하면 우선 mm\_malloc()이 호출된다. 위에서 설명한mm\_malloc()의 과정이 그대로 실행되고 mm\_free()가 호출된다.
    2. mm\_free()를 호출하면 위에서 설명한 mm\_free()의 과정이 실행된다.

1. **함수, 매크로 및 전역 변수** 
   1. 
      1. Segregated list에 대한 포인터이다.
      2. mem\_init 함수에서 mem\_sbrk() 함수를 통해 block 사이즈 별로 첫 번째 노드를 할당한다.
   2. 텍스트이(가) 표시된 사진

      자동 생성된 설명
      1. 위는 segregated list에 관련된 매크로이다.
      2. SEG\_FIND로 각각의 segregated list의 첫 번째 block으로 이동한다.
      3. SEG\_INIT\_NULL 매크로는 mem\_init() 함수에서 segregated list의 첫 번째 block을 NULL로 초기화한다.
      4. NEXT\_SEG와 PREV\_SEG를 이용해 segregated list 내에서 앞뒤 노드로 이동한다.
      5. SEG\_NEXT\_OF\_PREV는 현재 포인터가 가리키는 block 이전 block의 다음 block을 가리킨다. SEG\_PREV\_OF\_NEXT는 현재 포인터가 가리키는 block 다음 block의 이전 block을 가리킨다.
   3. 텍스트이(가) 표시된 사진

      자동 생성된 설명
      1. mm\_init 함수에서는 교재의 코드에서 segregated list를 초기화하는 부분(106 ~ 115 line)을 추가했다.
   4. 텍스트이(가) 표시된 사진

      자동 생성된 설명
      1. mm\_free 함수에서는 우선 해당 block의 header와 footer를 해제한다.
      2. 해당 block을 coalesce() 함수로 합친 후 insert\_free\_block() 함수로 segregated list에 삽입한다.
   5. 텍스트이(가) 표시된 사진

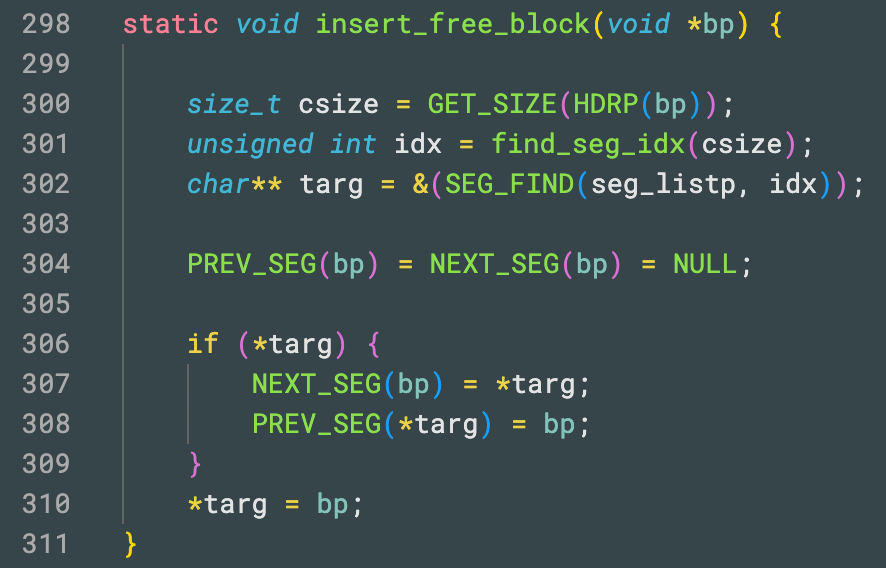
      자동 생성된 설명
      1. mm\_malloc 함수는 교재의 코드를 사용하였다.
   6. 텍스트이(가) 표시된 사진

      자동 생성된 설명
      1. mm\_realloc() 함수는 제공된 함수를 사용했으며 194번 줄에 copySize의 값을 reallocation하기 전의 block 사이즈로 바꾸는 코드를 추가하였다.
   7. 텍스트이(가) 표시된 사진

      자동 생성된 설명
      1. extend\_heap 함수는 교재의 코드에서 221, 222번 줄만 변경하였다. Coalesce() 함수로 앞뒤의 free block과 합쳐진 새 free block을 반환 받고 이를 insert\_free\_block() 함수를 통해 segregated list에 삽입한다.
   8. 텍스트이(가) 표시된 사진

      자동 생성된 설명
      1. coalesce() 함수는 free한 block의 앞뒤에 allocated 되어 있지 않은 block이 있을 경우 block들을 합친다.
      2. 첫 번째 if문은 해당 block 뒤의 block이 free block인 경우이고 234번 줄에 그 block을 segregated list에서 제거하는 코드를 추가하였다.
      3. 두 번째 if문은 해당 block 앞의 block이 free block인 경우이고 해당 block을 앞의 block과 합쳐야 하기 때문에 241번 줄에 해당 block을 segregated list에서 제거하는 코드를 추가하였다.
      4. 마지막 if문은 해당 block의 앞뒤 block이 모두 free block인 경우이고 그 block들을 모두 segregated list에서 제거하는 코드를 추가하였다.
   9. 텍스트이(가) 표시된 사진

      자동 생성된 설명
      1. find\_fit 함수는 block의 사이즈에 맞는 segregated list를 찾는 함수이다. find\_seg\_idx() 함수를 통해 asize 보다 작은 segregated list 중에서 가장 큰 리스트에 대한 인덱스를 반환 받고 그 리스트부터 시작하여 asize보다 큰 free block을 탐색하여 반환한다. 이를 만족하는 Free block이 없는 경우에는 NULL을 반환한다.
   10. 텍스트이(가) 표시된 사진

       자동 생성된 설명
       1. place 함수는 malloc한 block을 free block에 할당하는 함수이다.
       2. 우선 281 ~ 289번 줄은 split 후 남은 block이 최소 block 크기인 16byte 보다 크거나 같은 경우이다. 일단 현재 bp가 가리키는 block은 할당된 block이기에 segregated list에서 제거하고 footer와 header를 allocated로 설정한다. 그리고 bp를 split한 후 남은 block을 가리키게 하고 해당 block의 footer와 header를 해제한다. 마지막으로 해당 block을 segregated list에 삽입한다.
       3. Split 후 남은 block이 최소 block 크기보다 작을 경우엔 해당 block을 segregated list에서 제거하면 된다.
   11. 
       1. insert\_free\_block 함수는 파라미터로 받은 block을 segregated list에 삽입하는 함수이다.
   12. 텍스트이(가) 표시된 사진

       자동 생성된 설명
       1. remove\_free\_block 함수는 파라미터로 받은 block을 segregated list에 제거하는 함수이다.
   13. 텍스트, 화면이(가) 표시된 사진

       자동 생성된 설명
       1. find\_seg\_idx 함수는 파라미터로 받은 block size 값에 해당하는 segregated list의 번호를 반환한다. 해당 번호는 SEG\_FIND 매크로에서 사용된다.
2. **mm\_check**
   1. 텍스트이(가) 표시된 사진

      자동 생성된 설명
      1. mm\_check 함수는 위와 같다. 위 사진의 주석과 같이 다음 사항들을 확인한다.
         * Segregated list의 모든 free block들이 유효한 heap 영역에 있는지 확인.
         * 모든 free block들이 segregated list에 있는지 확인
         * Segregated list의 모든 block들의 헤더에 free 표시가 되어 있는지 확인
         * Coalescing에 잘 작동하는지 확인
   2. 텍스트이(가) 표시된 사진

      자동 생성된 설명
      1. 우선 traverse 함수는 위와 같다. Segregated list를 순회하며 파라미터로 받은 함수 포인터에 대한 함수를 실행하여 확인하고자 하는 사항을 수행한다.
   3. 텍스트이(가) 표시된 사진

      자동 생성된 설명
      1. check\_blocks\_in\_heap 함수는 traverse 함수에서 작동하는 함수로 seg\_p 파라미터는 traverse 함수로부터 받은 segregated list의 각각의 free block들을 의미한다. 해당 free block이 유효한 heap 영역에 있는지 검사한다.
   4. 텍스트이(가) 표시된 사진

      자동 생성된 설명
      1. check\_free\_block\_mark 함수는 traverse 함수에서 작동하는 함수로 seg\_p 파라미터는 traverse 함수로부터 받은 segregated list의 각각의 free block들을 의미한다. 해당 free block의 header가 allocated로 비트 마스킹이 되어 있으면 오류를 발생시킨다.
   5. 텍스트이(가) 표시된 사진

      자동 생성된 설명
      1. Check\_coalesce 함수는 traverse 함수에서 작동하는 함수로 seg\_p 파라미터는 traverse 함수로부터 받은 segregated list의 각각의 free block들을 의미한다. 해당 free block의 앞 뒤로 인접한 block이 free인 경우 coalescing이 제대로 이루어지지 않은 것이므로 오류를 발생시킨다.
   6. 텍스트이(가) 표시된 사진

      자동 생성된 설명
      1. check\_free\_blocks\_in\_seg 함수는 heap\_listp로부터 시작하여 모든 block들을 순회하며 해당 block이 free block인 경우에 그 block을 segregated list에서 탐색한다. 만약 탐색에 실패하면 free block과 segregated list의 consistency가 위반된 것이므로 오류를 발생시킨다.