1. 메모리 접근을 위한 시스템 콜

메모리나 하드웨어와 같은 중요한 컴퓨터 시스템 자원은 OS에서 관리 된다.

정확히 말하자면 OS에 속해 있는 **커널(kernel)**에 의해서 관리 된다.

함부로 접근되면 안되는 중요한 자원이기 때문에 OS에서 **시스템 콜(System Call)**을 통해 **커널 모드**로 전환 하여 메모리에 접근을 해야 한다.

C / C++ 에서는 메모리에 접근하여 메모리를 할당 받기 위해서 Malloc을 사용한다.

Malloc은 시스템 콜 함수 mmap와 sbrk함수를 사용하여 만들어 졌다.

즉 , malloc도 시스템 콜을 통해 메모리에 접근하여 할당을 해주는 것이다.

1. mmap 함수와 sbrk

mmap 함수는 원래 파일이나 장치를 메모리에 대응 시켜 주기 위해 사용되는 함수이다.

그런데 MAP\_ANONYMOUS|MAP\_PRIVATE 을 사용하면 파일이나 장치와 상관없는 익명의 메모리를 할당 해준다. mmap함수는 페이지 단위로 할당 이루어지기 때문에 malloc에서 큰 익명의 메모리가 필요할 때 이 함수를 사용한다.

malloc이 page( 4KB ) 보다 큰 메모리 할당 요청을 받았을 때 heap에서 알맞은 블록을 찾아서 할당 해주는 것이 아니라 mmap을 통해서 알맞은 메모리를 새로 가져와서 할당을 해준다. 이 과정에서 heap을 거치지 않는다.

sbrk는 힙의 마지막 부분을 늘려 주어 힙의 크기를 변화 시켜 주는 함수이다. 매개변수로 양수를 주면 늘어나고 음수를 주면 줄어든다.

malloc은 작은 크기의 메모리 할당을 요청 받았을 때 heap에서 사용 가능한 블록 중 할당 요청을 받은 크기보다 큰 블록을 찾아서 할당해준다. 근데 알맞은 블록이 없다면 위에서 말한 sbrk함수를 통해서 힙의 크기를 늘려 새로운 알맞은 블록을 할당 해준다.

1. Dynamic Memory Allocation

테이블이(가) 표시된 사진

자동 생성된 설명

1. Allocator에 대한 사전 지식

블록 단위로 메모리를 관리한다.

블록 구성 :

Header , Footer , Payloading and padding

Header , Footer : 각각 4 byte ( = 32bits (0~3 bit) 할당 여부 , (4~31) Block Size )

각각 블록의 시작 점과 끝점을 나타내는 부분이다. Header와 footer는 각각 4 Byte이므로 블록의 크기는 최소 8 Byte이다.

Payload : 실제 정보들이 들어가는 부분

Padding : 내부 단편화를 감수하고 할당해야 할 메모리보다 많은 메모리를 할당해주는 경우 (선택적으로 사용한다.)

Prev( Previous ) , Next Block pointer : 할당이 해제 된 가용 블록들끼리 포인터들로 연결 시키기 위해 사용하는 블록 포인터 ( 이중 연결리스트 형태 )

테이블이(가) 표시된 사진

자동 생성된 설명

1. Explicit Allocator

@ 응용 프로그램이 직접 메모리를 할당 / 반환하는 방식의 Allocator

@ 할당이 해제 된 가용 블록들을 연결리스트 형태로 관리 ( Prev , Next block pointer 사용 )

@ C 나 C++ 와 같은 low-level 언어에서 사용한다.

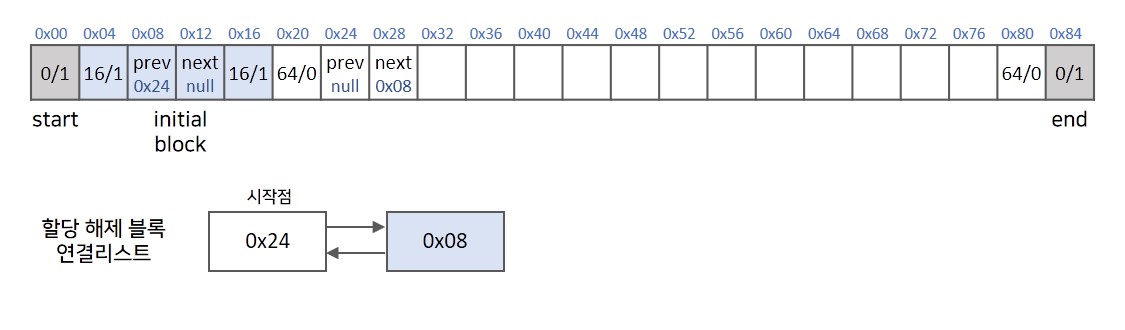
@ Malloc / Calloc / Realloc / Free / New / Delete 등이 이에 속한다.

연결 리스트에 블록을 추가하는 방법

1. Ordered List : 순서대로 추가
2. LIFO( Last-In-First-Out ) : Stack 처럼 작동

LIFO 방식이 성능 적인 면에서 우세 하다.

LIFO를 이용한 Explicit Allocator의 Heap 구조



@ 블록 속 데이터 ( 블록의 Size / 할당 여부[ 1 은 할당된 블록, 0 은 가용블록] )

@ start 와 end는 Heap의 처음과 끝을 나타내는 블록 (사이즈 0 , 할당 1)

@ initial block : Header + Footer + prev & next block pointer 으로 구성 되어 있고 가용 블록 연결 리스트의 끝을 나타내는 블록 (사이즈 16 , 할당 1)

@ 0x20 ~ 0x84 : 할당 해제 된 가용 블록이다. (사이즈 64 , 할당 0) initial 블록과 연결 되어 있다.

메모리 할당 / 해제 과정