Programming Language Concepts

Programming Language Theory

Topics

- Memory Management
  - Static Management
  - Dynamic Management w/ Stack
  - **Dynamic Management w/ Heap**
  - Scope Rule Implementation

Heap

- Why we need ***Heap*** for memory management?
    - We already have stack, and it seems natural to manage memory for procedures.
- Some languages have statements which allow ***explicit memory allocation***.

Explicit Memory Allocation

- With explicit memory allocation, there is no guarantee of LIFO.
- In the example, pointer variable p is the first one allocated, and also the first one deallocated.
- If we use the stack, we can't deallocate p before q, since q is at the top.

```c
int *p, *q;
p = malloc(sizeof(int));
q = malloc(sizeof(int));
*p = 1;
*q = 2;
free(p);
free(q);
```

Allocation

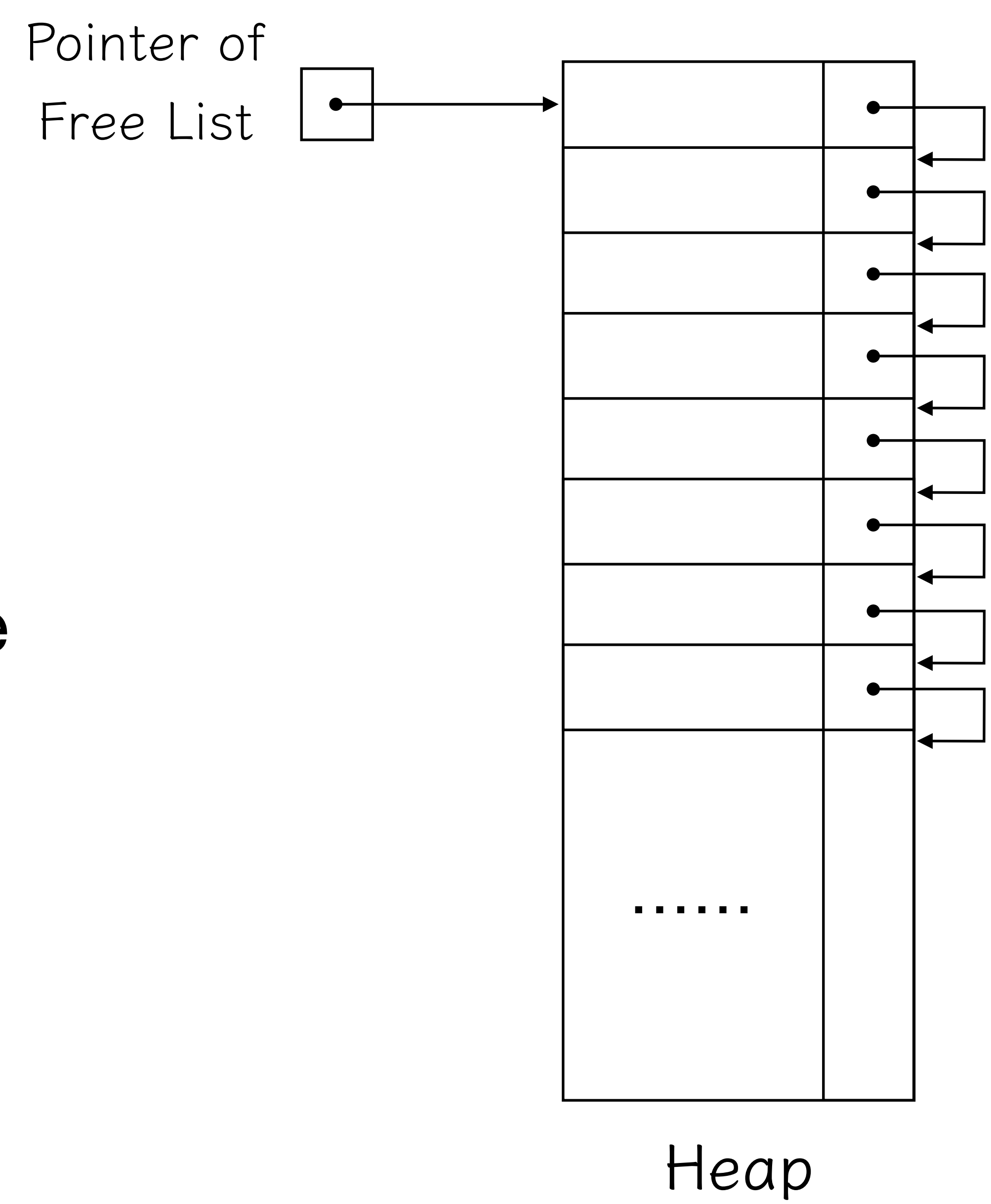
Deallocation

Heap Management

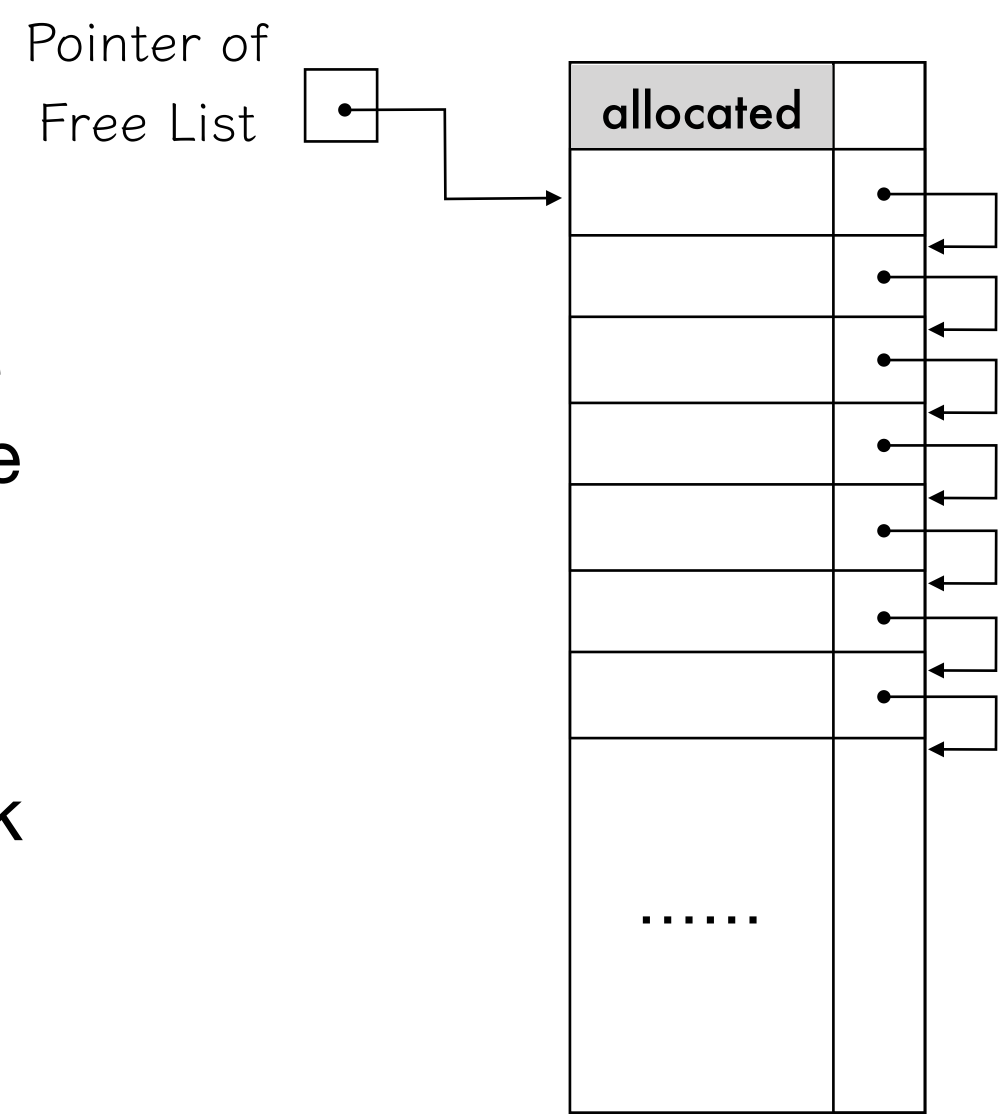
- Heap management methods fall into two main categories,
- based on whether the memory blocks are considered,
    - *Fixed Length Blocks*
    - *Variable Length Blocks*

Fixed Length Blocks

- Divide the heap to multiple fixed length blocks.
- Using a free list to maintain the list of free blocks.
- For each request, the first free block will be allocated.
- The pointer of the free list points to the first block of the list.



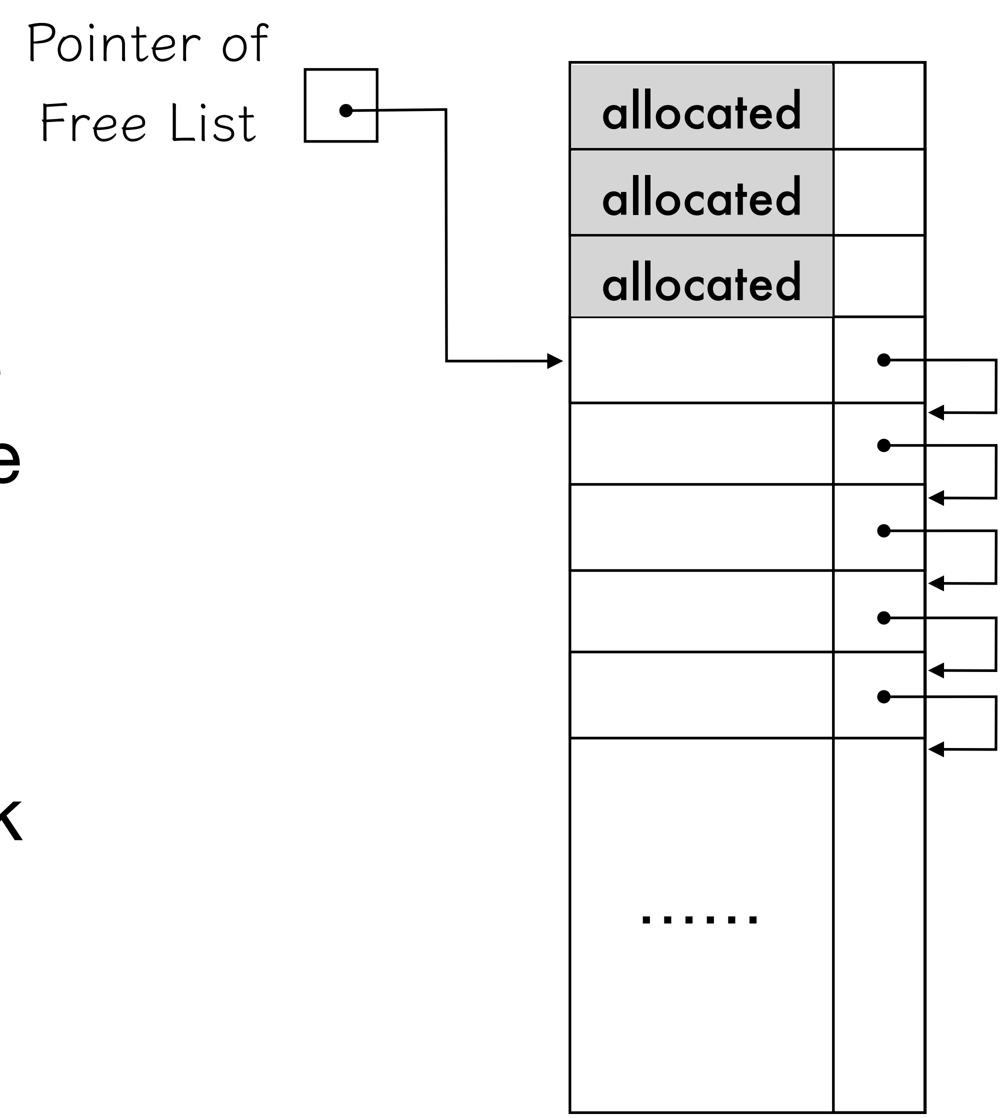
Fixed Length Blocks



- When there is a request, the first block is assigned and the block is removed from the free list.
- When a block is freed (or deallocated), the block is back to the free list.

Fixed Length Blocks

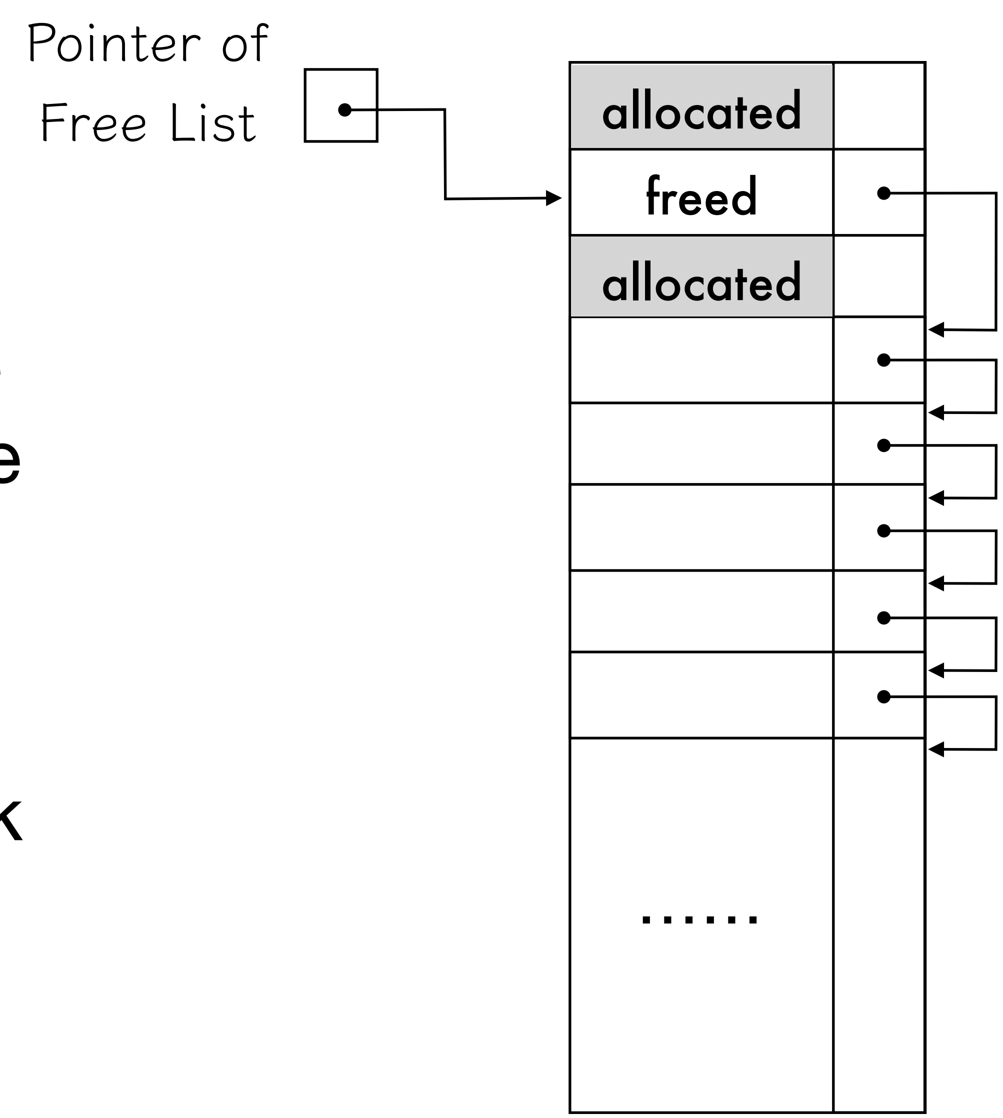
- When there is a request, the first block is assigned and the block is removed from the free list.
- When a block is freed (or deallocated), the block is back to the free list.



Fixed Length Blocks

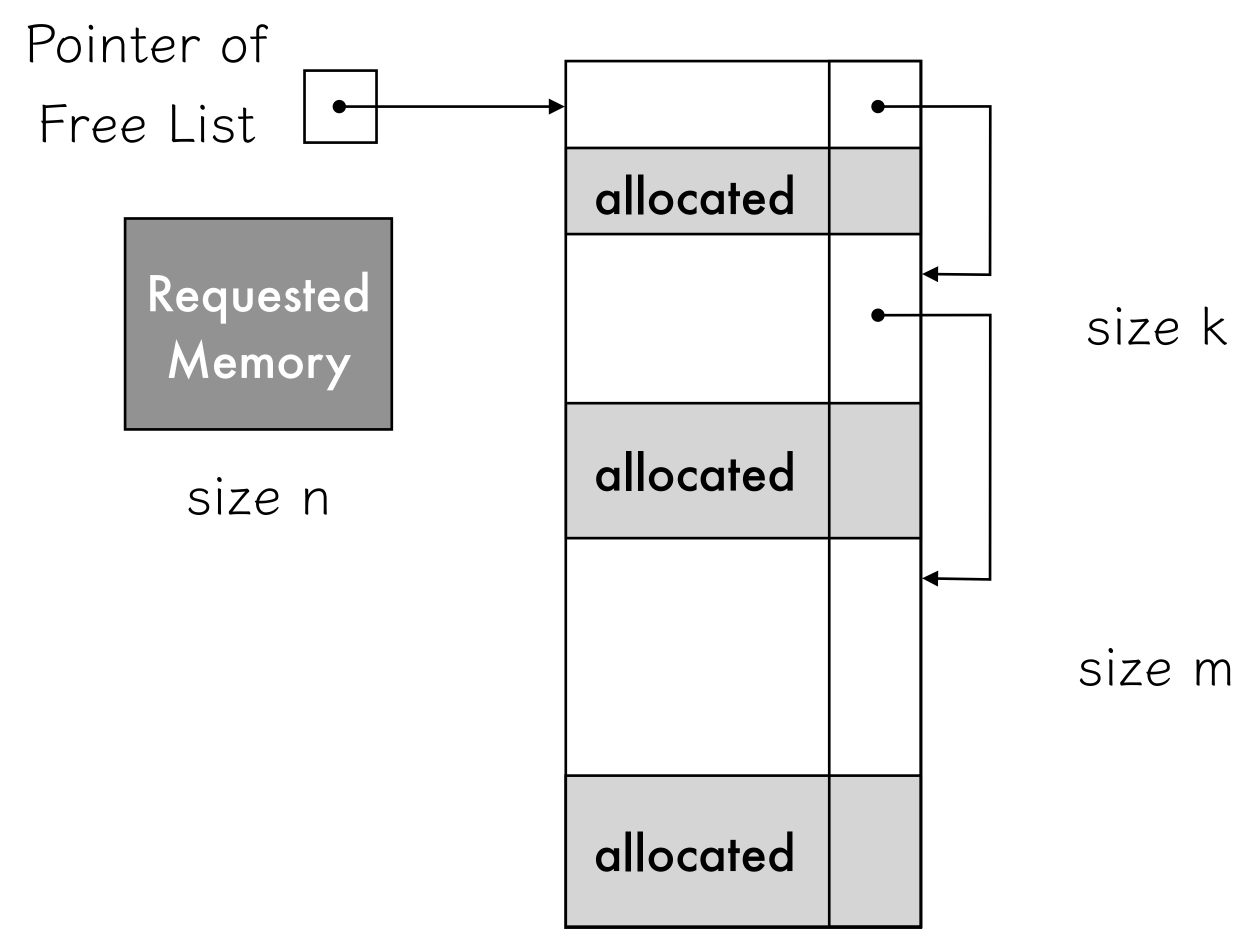
- When there is a request, the first block is assigned and the block is removed from the free list.

- When a block is freed (or deallocated), the block is back to the free list.



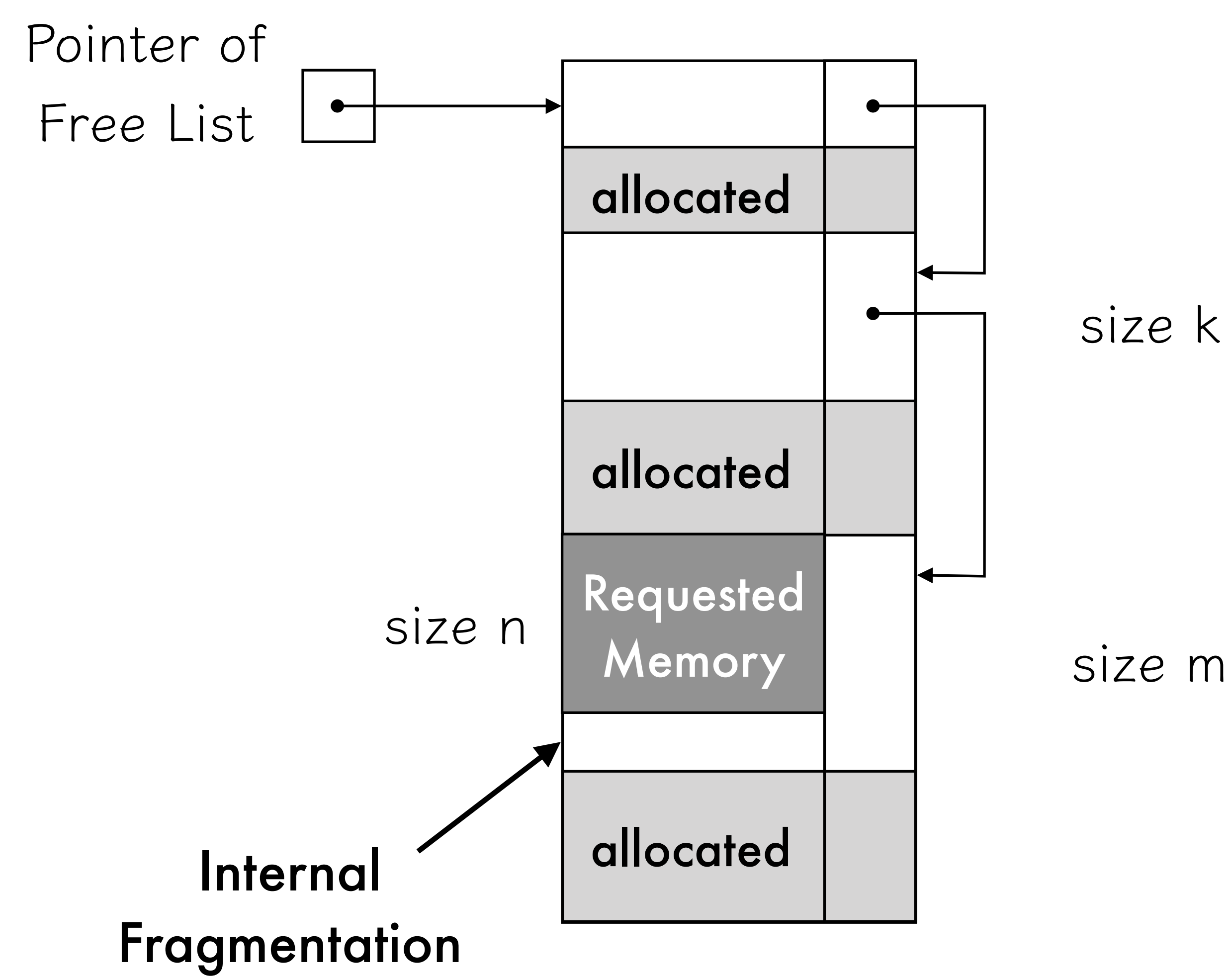
Variable Length Blocks

- Similar to fixed length blocks, it maintains a free list for available blocks.
- The size of blocks can be different.
- When a request for memory of size n, it allocates a free block fits to this size.
  - e.g.) n > k and n < m.
  - The third free block is allocated.



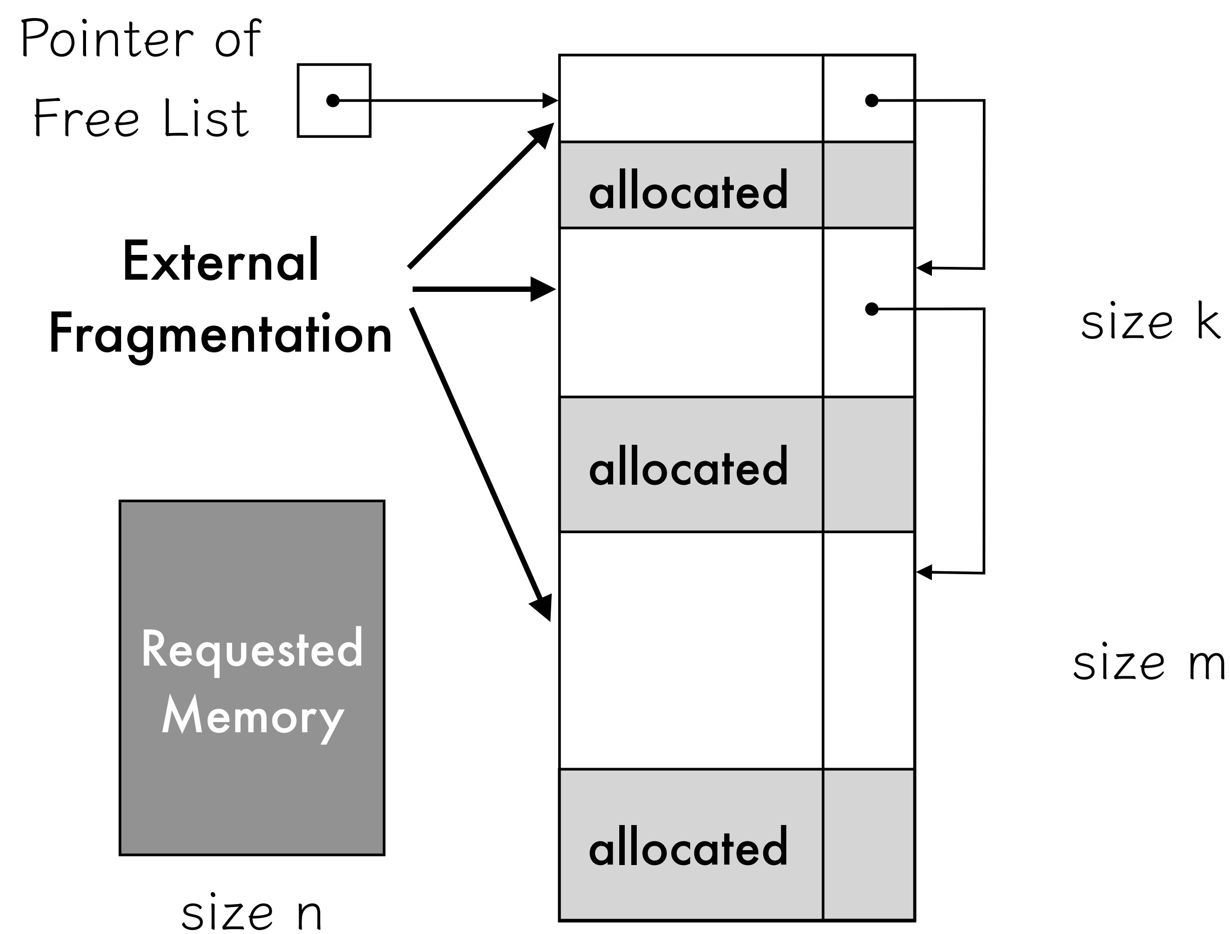
Fragmentation

- Variable length method causes fragmentation.
- Due to fragmentation, memory space is wasted, or it reduces the performance of programs.
- ***Internal Fragmentation***: Allocated block size is greater than the requested size.
    - m > n, then d = m - n is wasted.



Fragmentation

- ***External Fragmentation***
  - Due to the scattered free blocks, requested memory cannot be allocated,
  - even it there exists enough space.
  - m + k > n, but they are not consecutive.



Using Single Free List

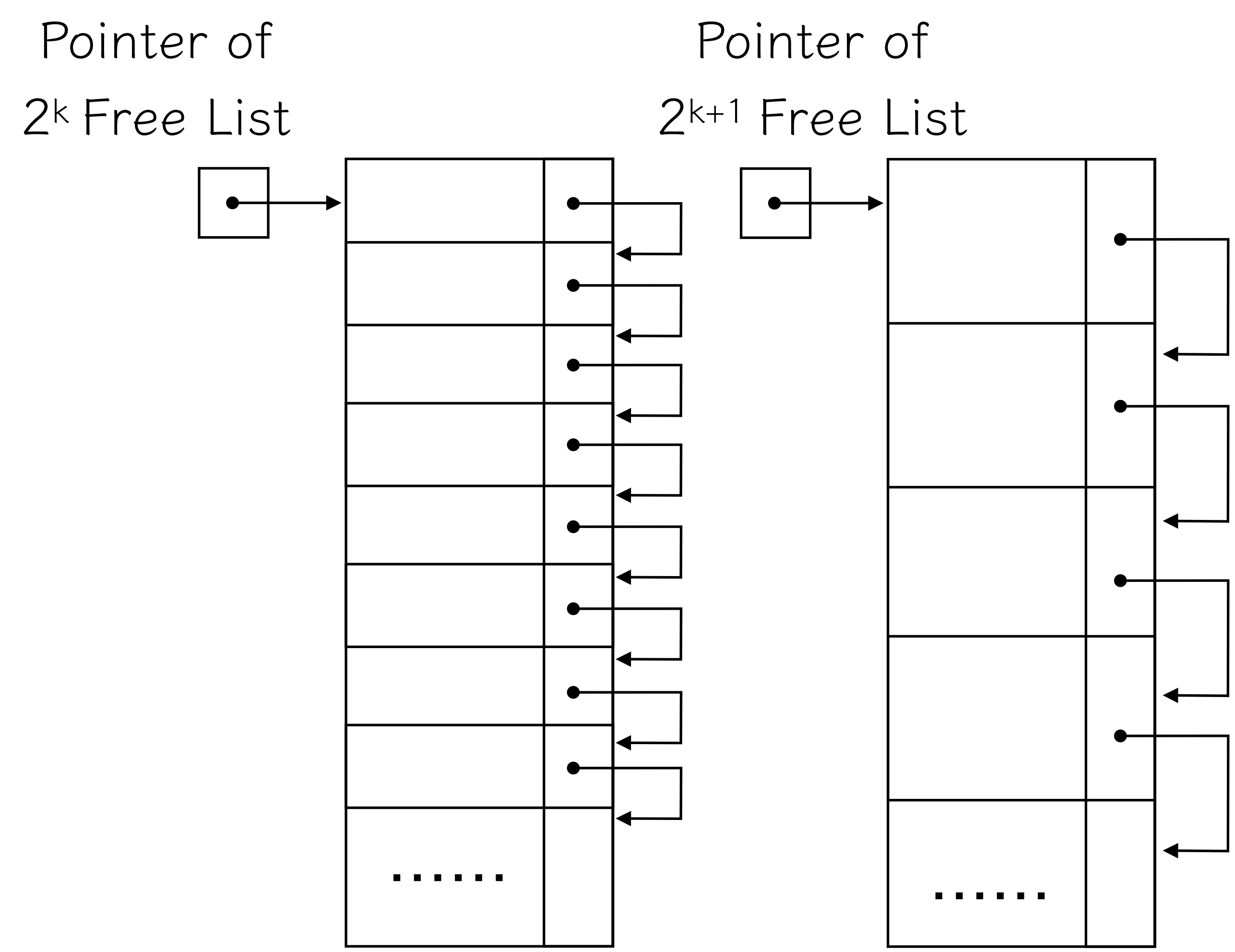
- When there is a request for memory allocation of size *n*,
- Directly use the free list.
    - First Fit: allocate the first block bigger than size *n*.
    - Best Fit: allocate the size $k >= n$ block which has the minimum $d = k - n$.
- Free Memory Compaction.
    - When the end of the heap is reached, move all active blocks to the end.

Multiple Free Lists

- Buddy System
  - Have multiple free lists with size power of 2 (i.e. 2^n).
  - For size n request, find a block from the free list of $2^k >= n$ blocks.
  - If there is no available block, then search 2^{k+1} free list next.
- Fibonacci Heap
  - Instead of 2^n free lists, use Fibonacci numbers as block sizes in free lists.
  - $Fib(n) = Fib(n-1) + Fib(n-2)$

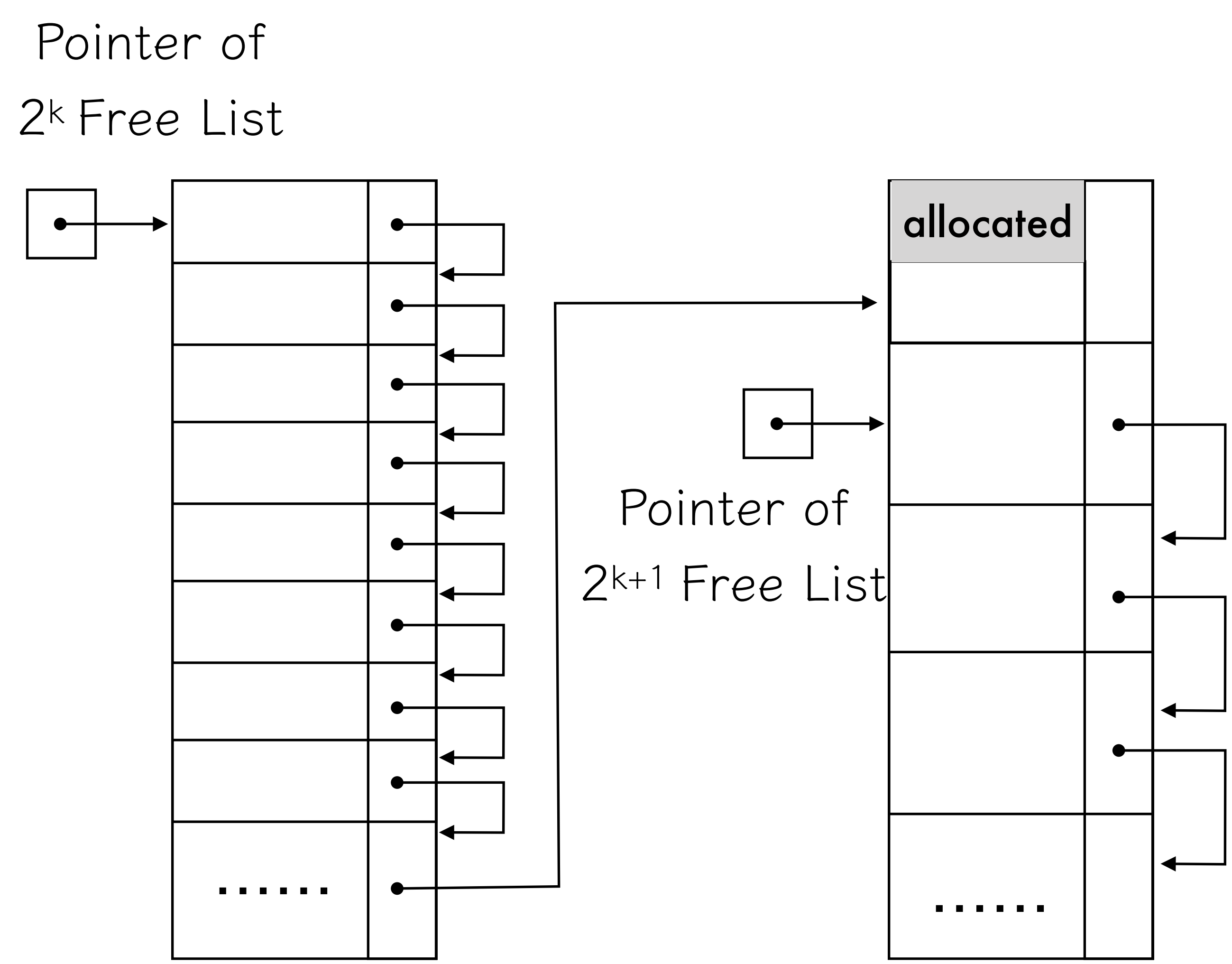
Multiple Free Lists

- **Buddy System**
  - Have multiple free lists with size power of 2 (i.e. 2^n).
  - For size n request, find a block from the free list of 2^k >= n blocks.
  - If there is no available block, then search 2^{k+1} free list next.



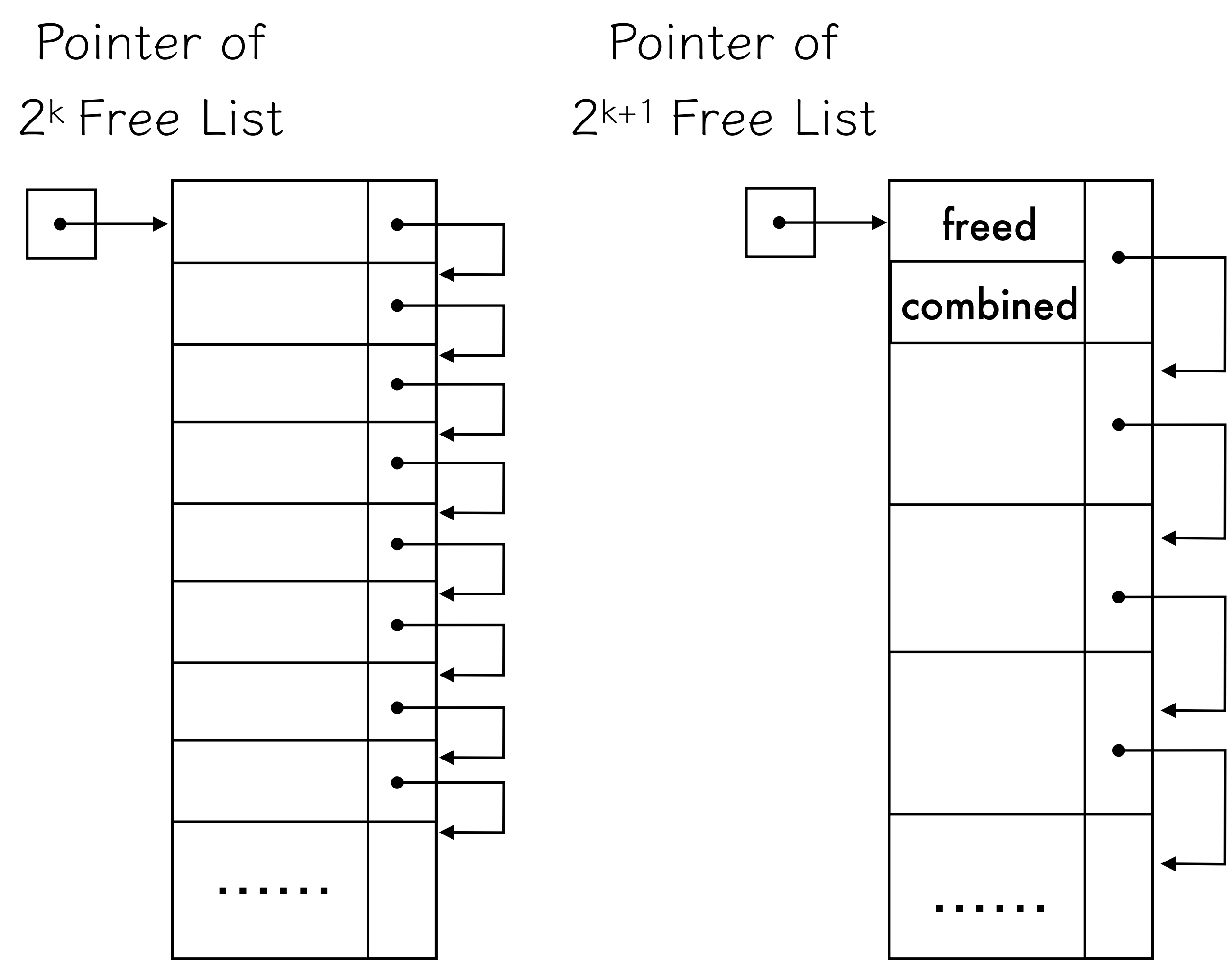
Multiple Free Lists

- When a free block is found in 2^{k+1} free list,
  - Split this block into two 2^k blocks.
  - Allocate one of them, and connect the other to 2^k free list.



Multiple Free Lists

- Next time the allocated block is freed,
  - Find its *buddy* which is resulted by the split, and check it is also free.
  - Combine them and attach it to 2^{k+1} free list again.



Summary

- Dynamic Memory Management w/ Heap
    - Differences between Fixed/Variable Length Blocks.
    - Fragmentation
    - Heap Management w/ Single / Multiple Free Lists