Stack
Queue
Binary Heap

Outline

- Stack
- Queue
- Deque
  - Programming Assignment 1
- Priority Queue
  - Binary Heap
    - Programming Assignment 2

ADT: Stack

- Objects: a set of objects
- Operations:
  - Push(S,x): insert x into stack S
  - Pop(S): remove the last inserted object
  - Last(S): return the last inserted object
  - Empty(S): return if the stack is empty
  - Full(S): return if the stack is full

ADT: Queue

- Objects: a set of objects
- Operations:
  - Enqueue(Q,x): insert x into queue Q
  - Dequeue(Q): remove the first inserted object
  - First(Q): return the first inserted object
  - Empty(Q): return if the queue is empty
  - Full(Q): return if the queue is full

ADT: Deque

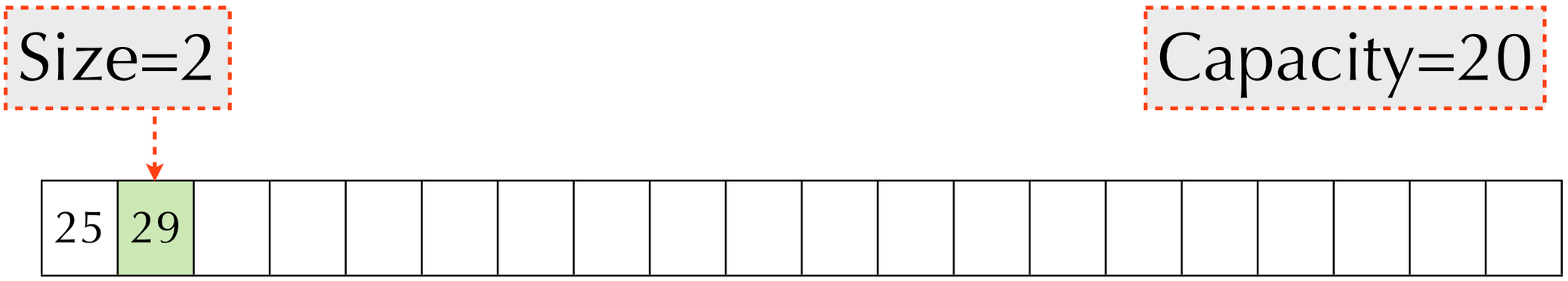
- Objects: a set of objects
- Operations:
  - Push, Pop, Enqueue, Dequque, Top, First, Last, Empty, Full.
- A deque is a stack.
- A deque is a queue.
- A list is a deque.

ADT: Priority Queue

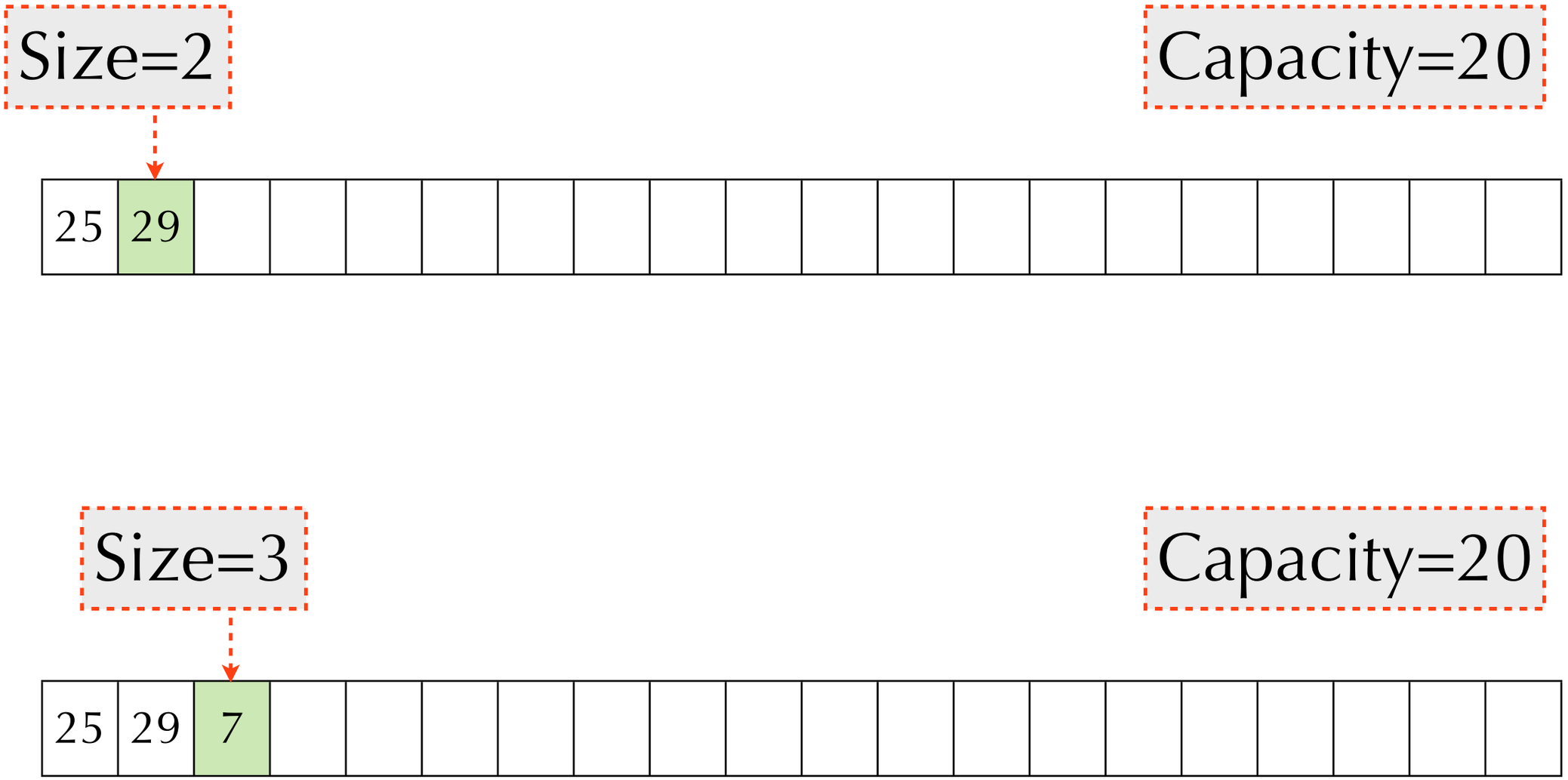
- Objects: a set of <key,object>
- Operations:
  - Insert(PQ,x): insert x into PQ
  - ExtractMin(PQ): remove the object of lowest key
  - Min(PQ): return the object of lowest key
  - DecreaseKey(PQ,k,obj): Decrease the key value of obj to k
  - Empty(PQ): return if PQ is empty
  - Full(PQ): return if PQ is full

Array Based Stack

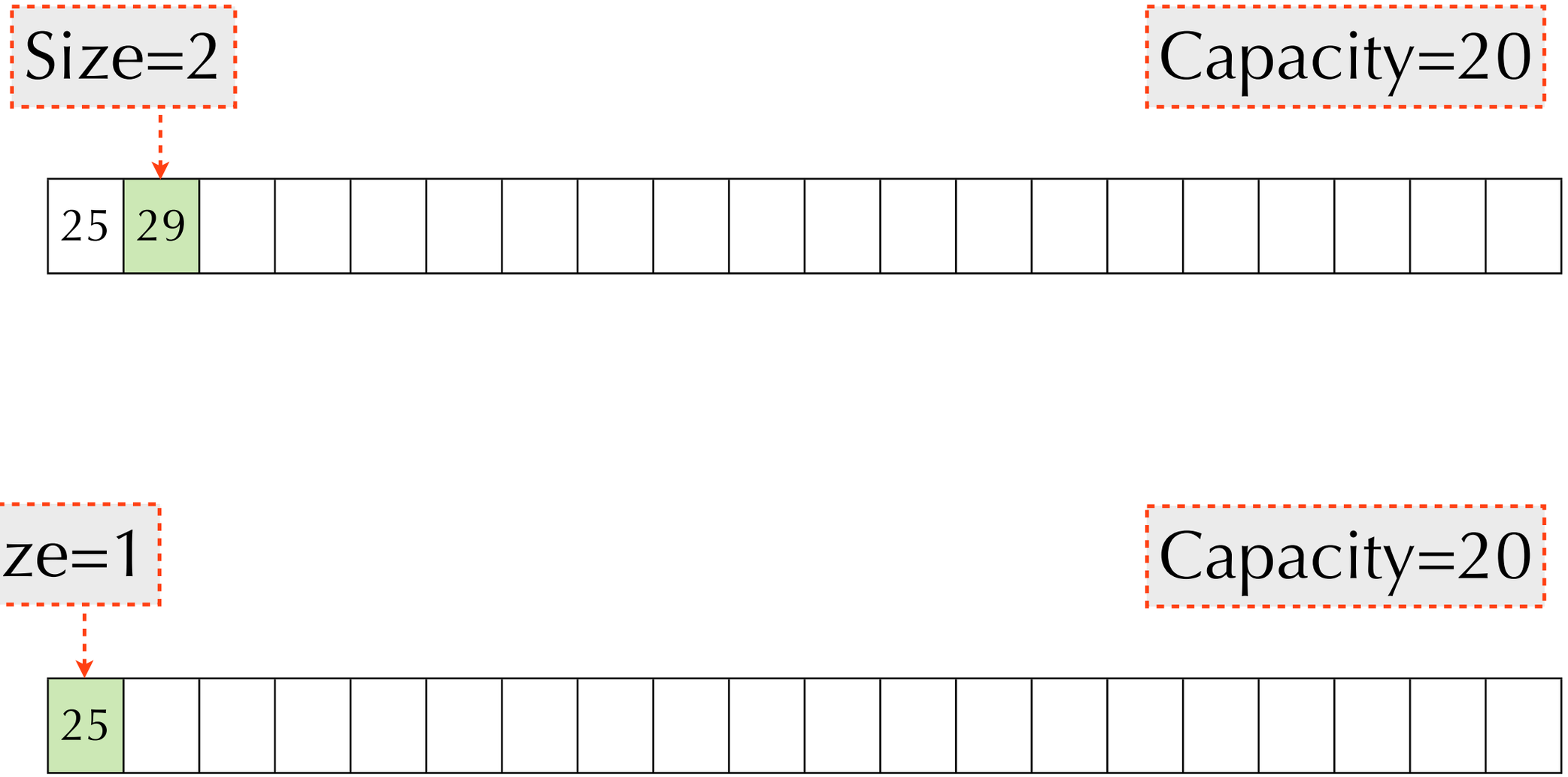
- Using an array to implement a stack
  - Other information: capacity, size.



Push 7

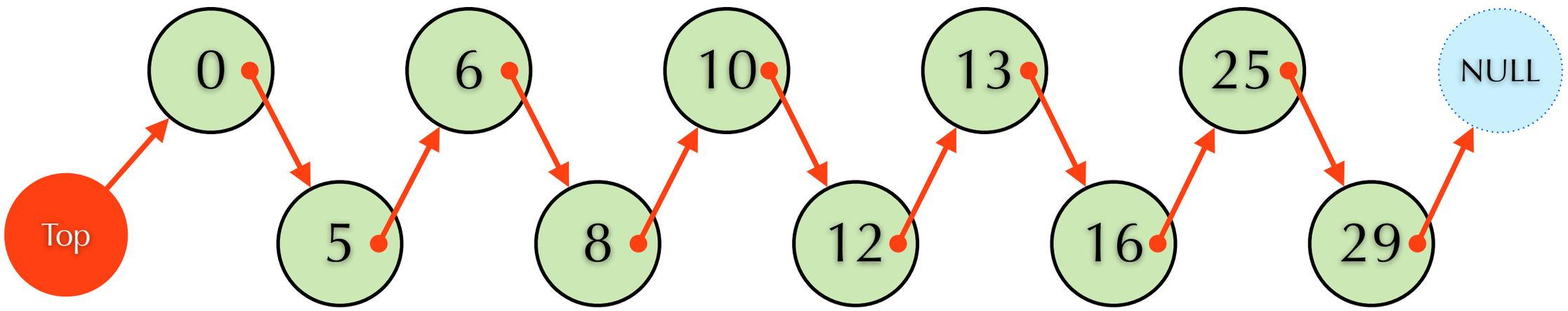


Pop

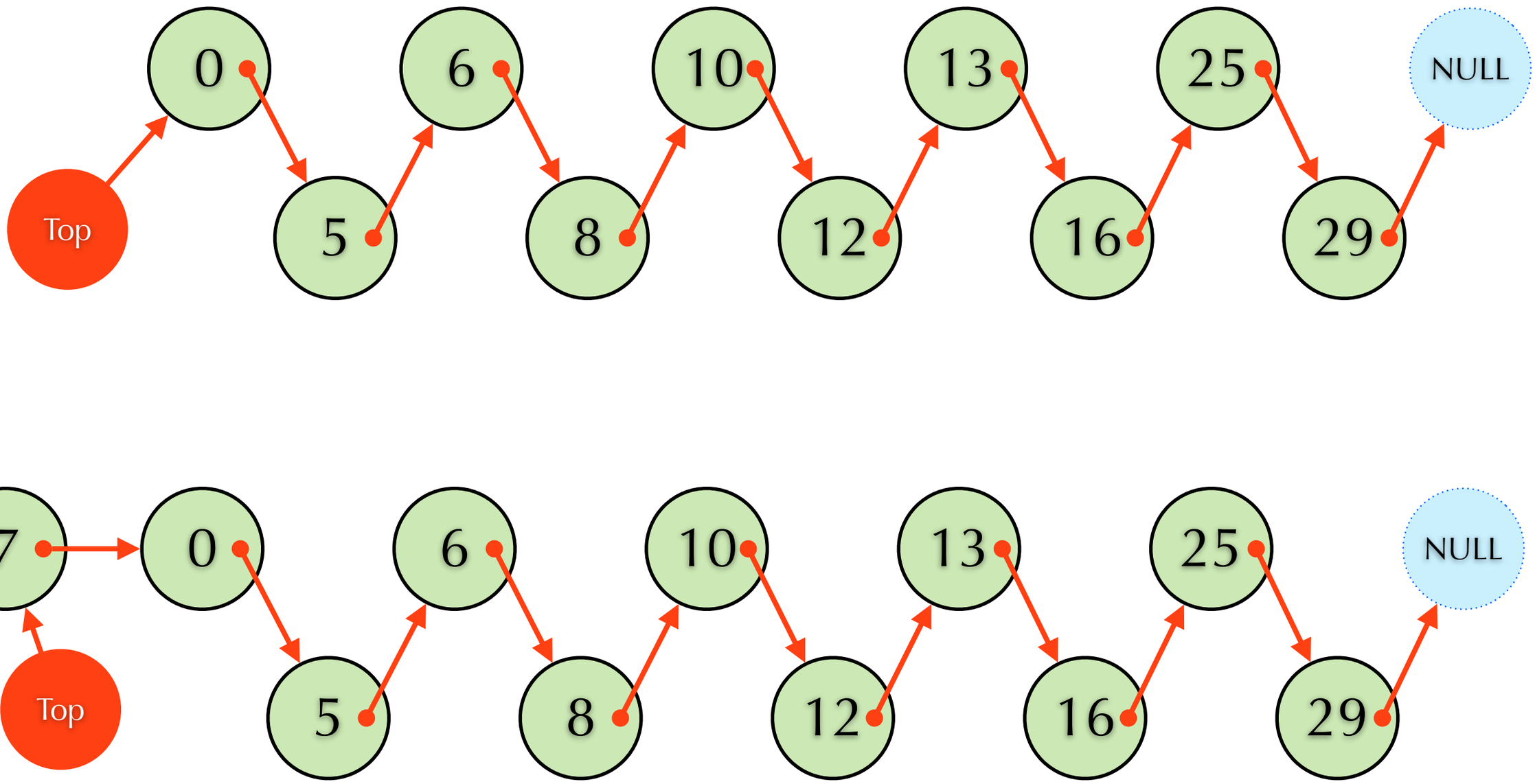


Linked-List Based Stack

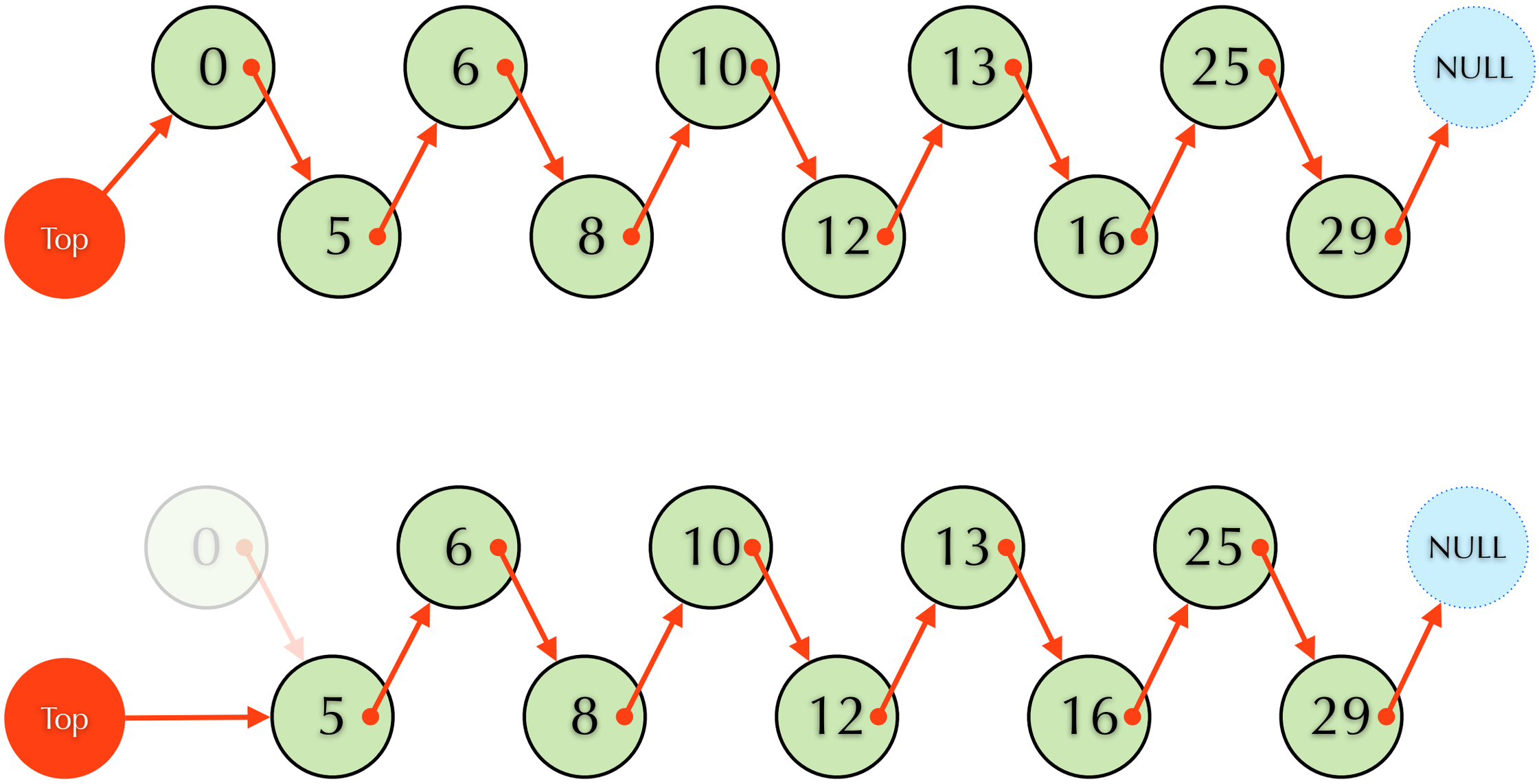
- Push = Prepend!
- Pop = Delete Head!



Push 7



Pop



Stack: Implementation

- Use array or linked-list
- Array
  - Pro: easier to code, fast
  - Con: space efficiency might be low
- Linked-list:
  - Pro: space efficiency is good
  - Con: slightly harder to code, slow

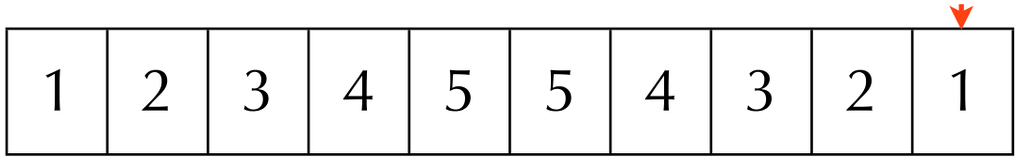
Improve Space Efficiency

- Initial a dynamic array with capacity X
- Double capacity if size=capacity
- Halve capacity if 4size<capacity & capacity>X
- Space efficiency:
  Waste at most max(X,3size) cells
- Tradeoff: Need more time to maintain a proper capacity.

Double Capacity

Size=10

Capacity=10



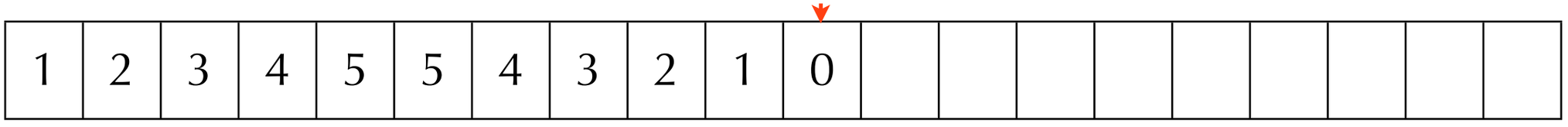
Worst case:

11 writes, 1 allocation, 1 free.

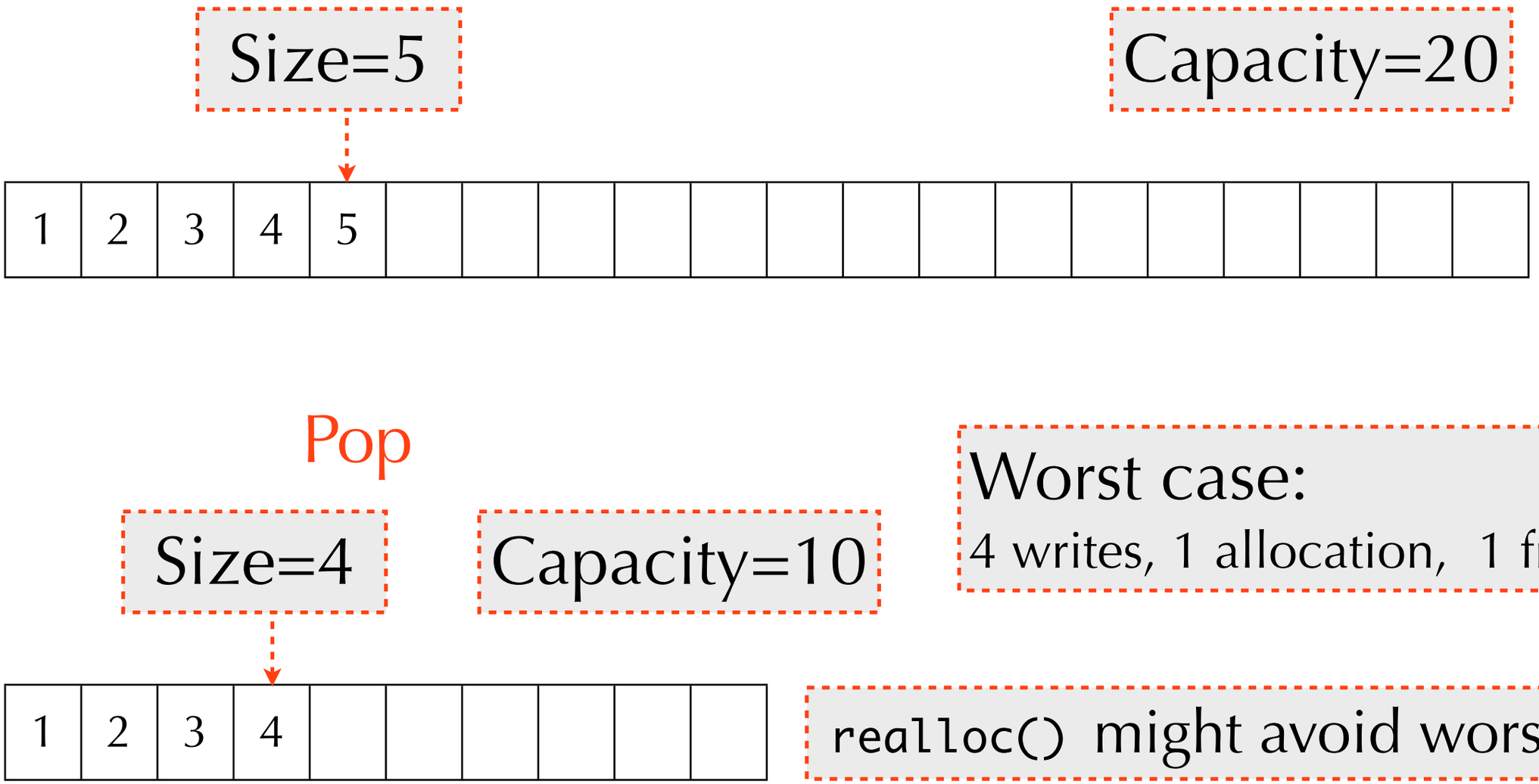
Push 0

Size=11

Capacity=20



Halve Capacity

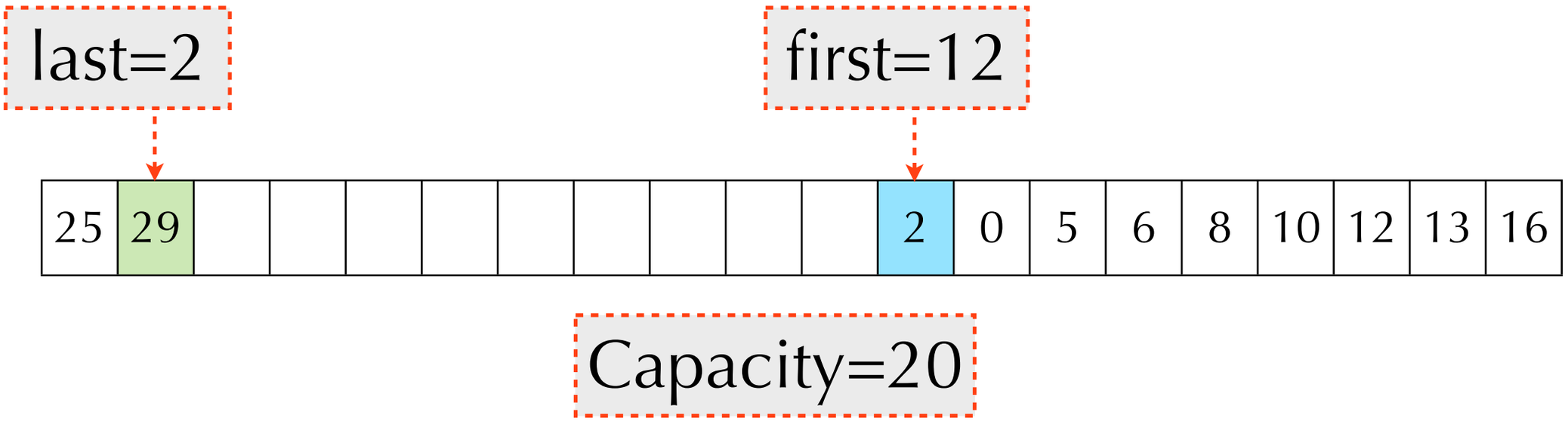


Homework 3.1

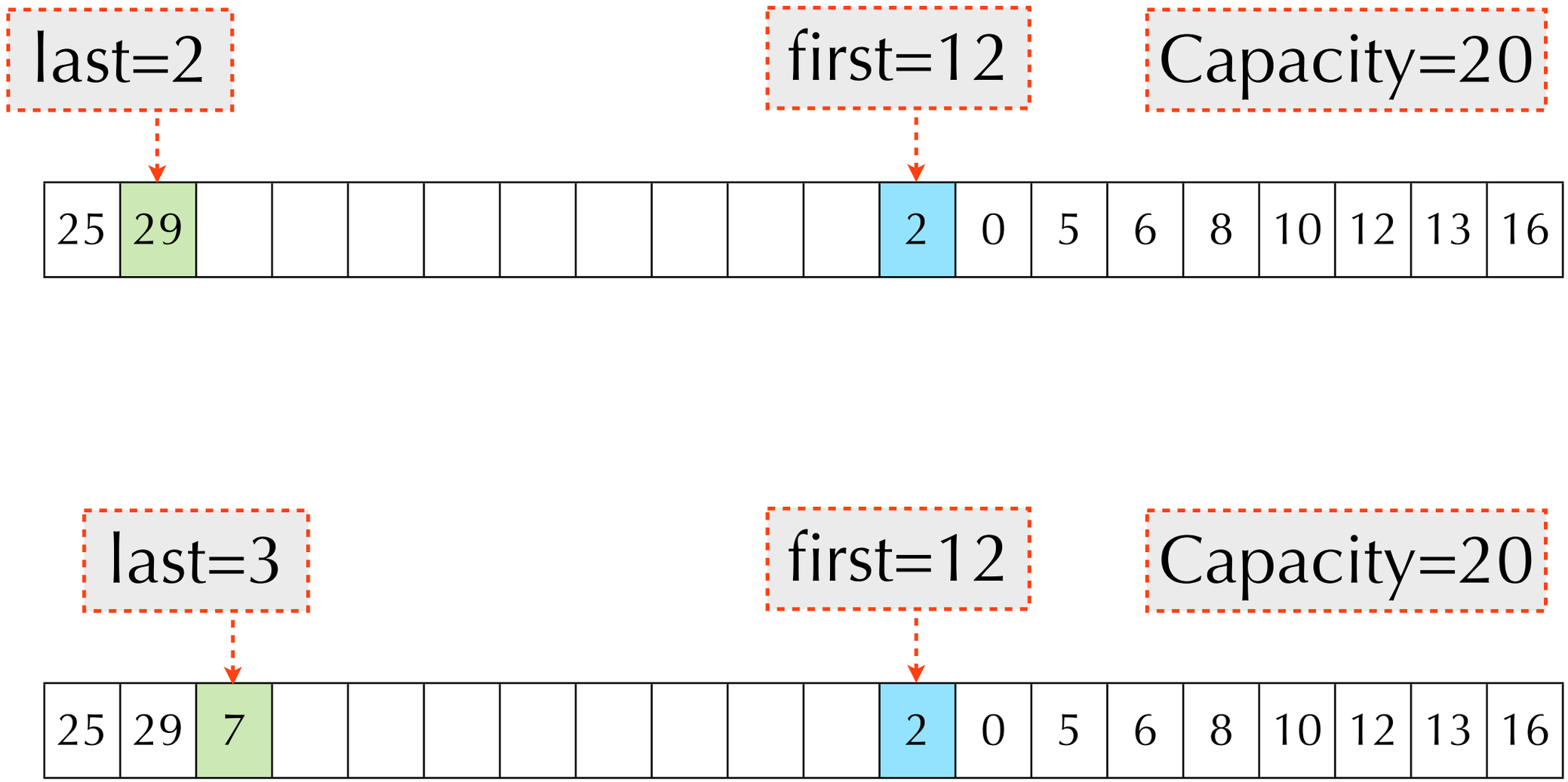
- a) Why linked-list based implementation is usually slower than array based implementation?
- b) How many time is required to finish m stack operations if the stack is implemented by a dynamic array with the resize strategy on previously mentioned?

Array Based Queue

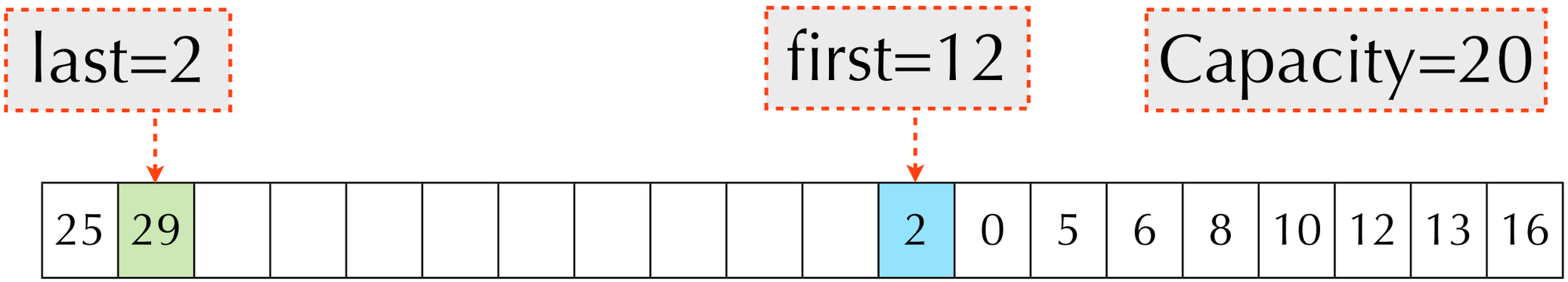
- Using an array to implement a queue
  - Other information: capacity, first, last

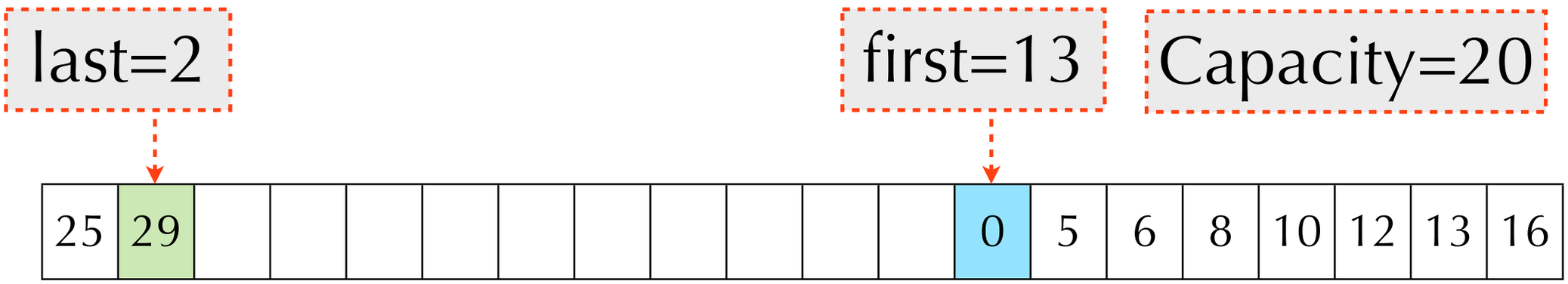


Enqueue 7


last=2
first=12
Capacity=20
25
29
2
0
5
6
8
10
12
13
16
last=3
first=12
Capacity=20
25
29
7
2
0
5
6
8
10
12
13
16

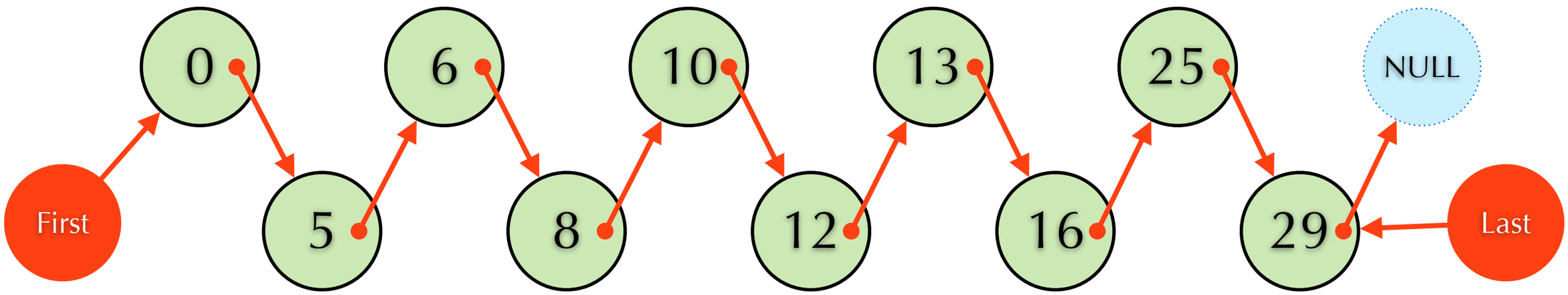
Dequeue


last=2
first=12
Capacity=20
25
29
2
0
5
6
8
10
12
13
16

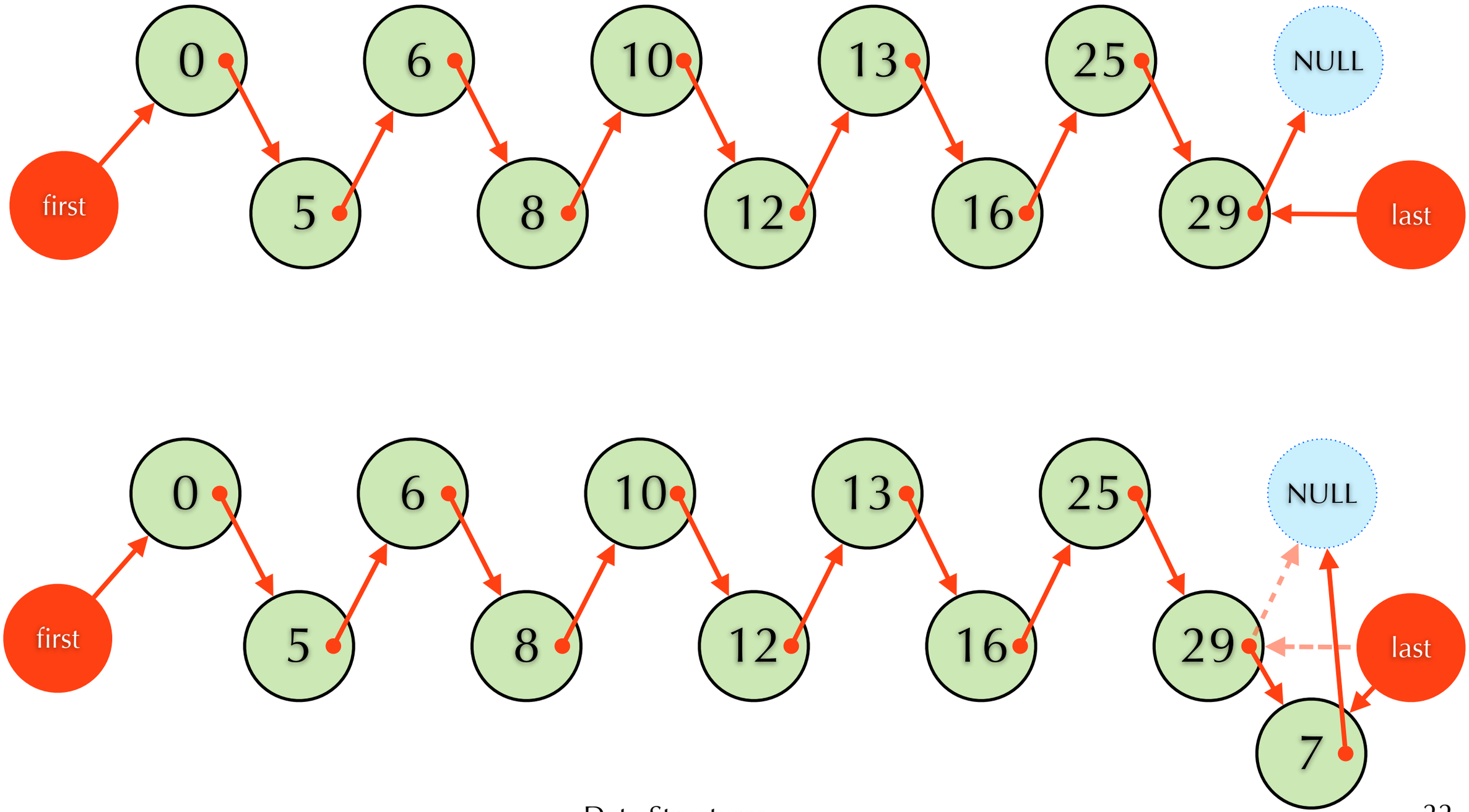

last=2
first=13
Capacity=20
25
29
0
5
6
8
10
12
13
16

Linked-List Based Queue

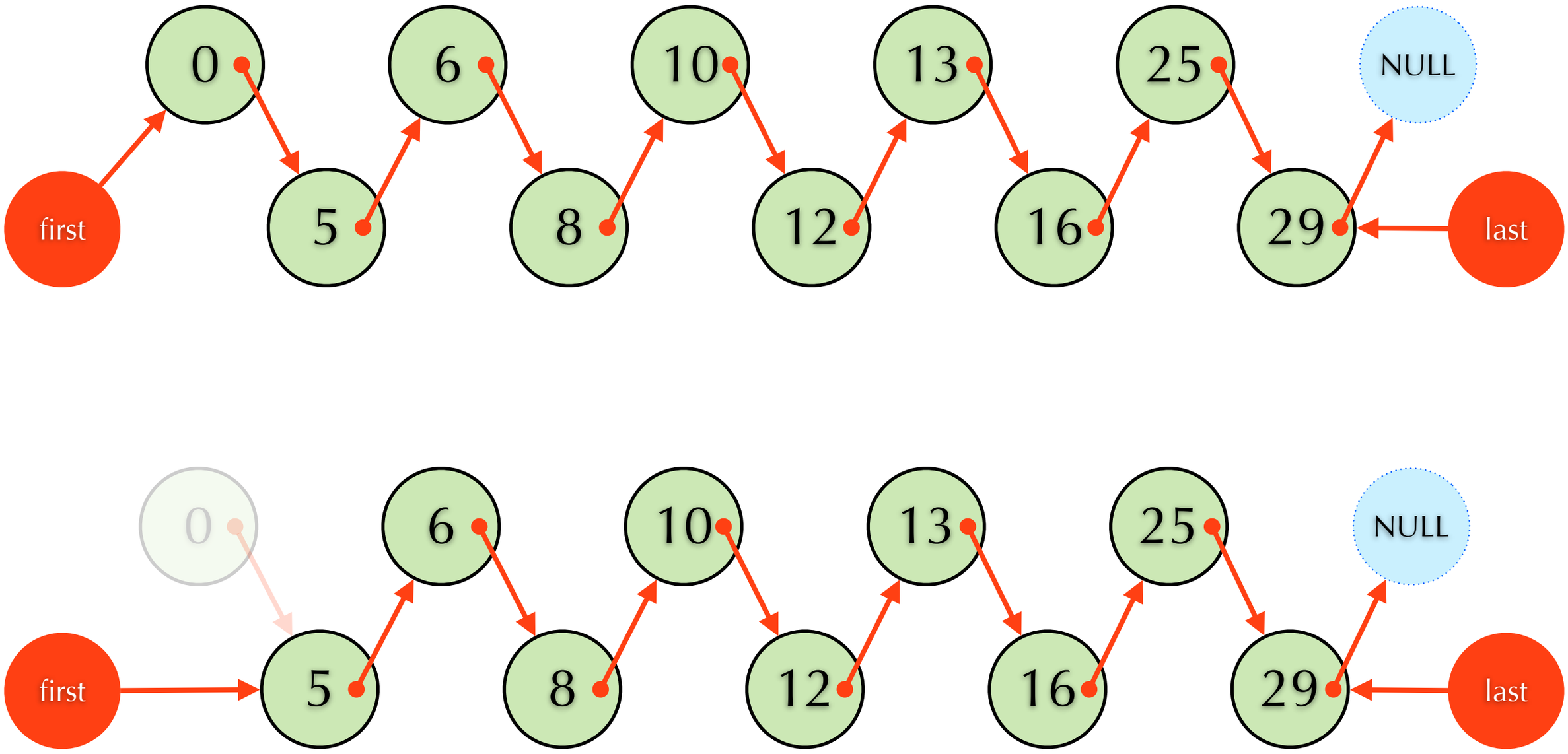
- Enqueue = Append!
- Dequeue = Delete Head!



Enqueue 7



Dequeue



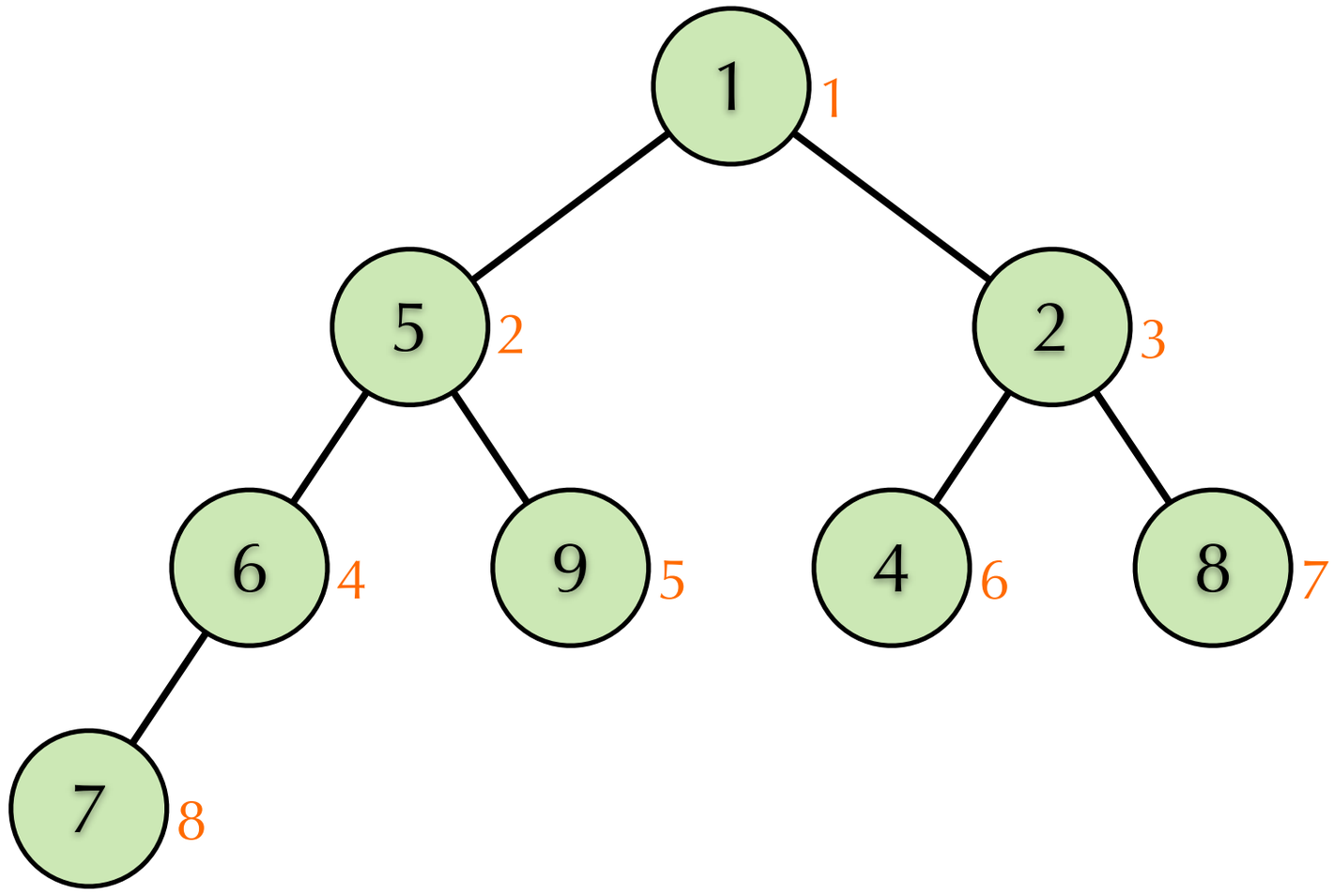
Queue: Remainder

- Array
  - When you apply the space saving trick, beware of the circular structure
- Linked-list:
  - Initialization first and last
  - Enqueue object into an empty queue
  - Dequeue when there is only one object in the queue

Binary Heap (Min)

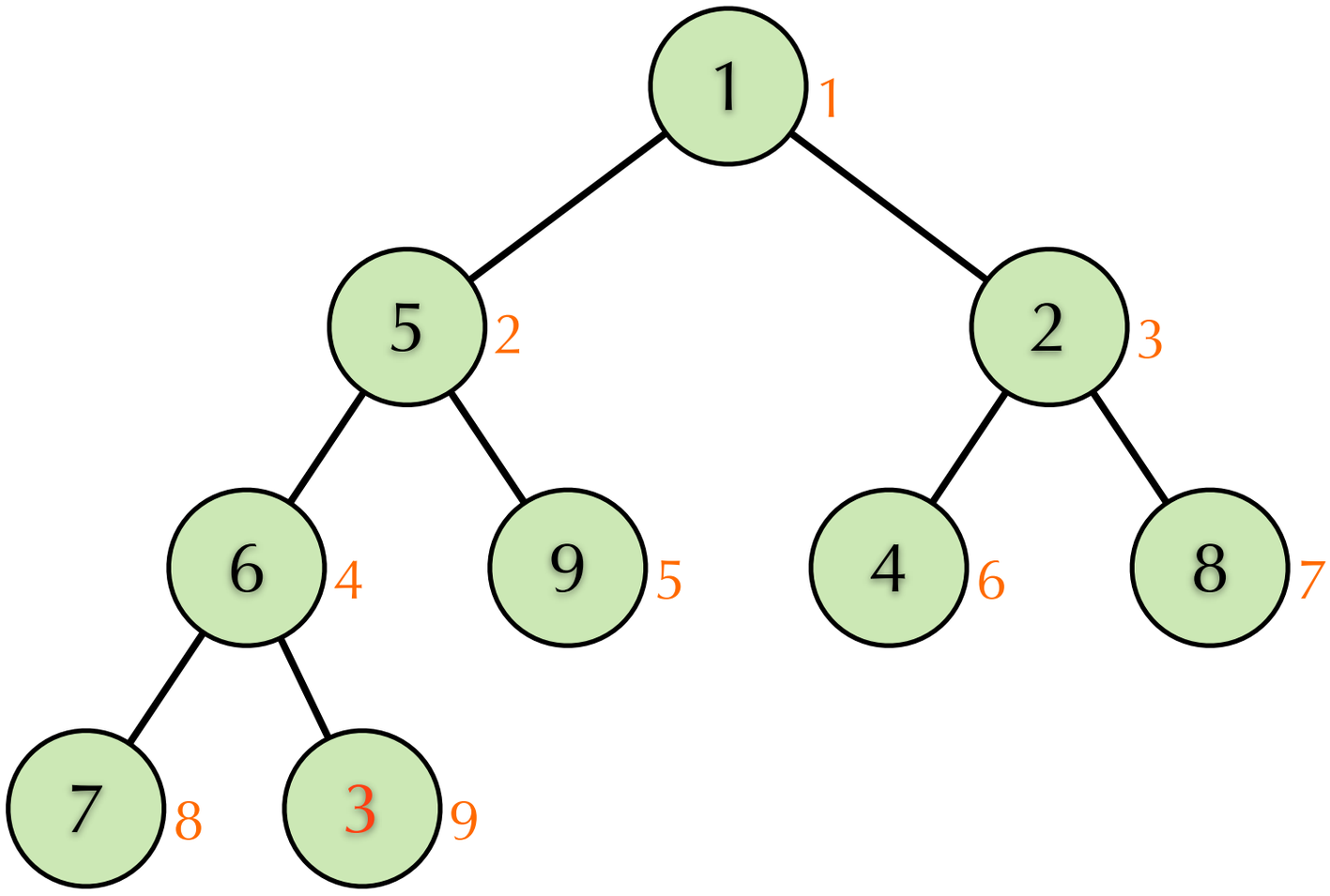
- A binary tree H of n nodes
  - The nodes are in n consecutive positions in H's array representation
- Every node stores a key-object pair.
  - A's key≤B's key if A is B's parent.
- This is a nice structure for finding minimum!
  - The root will has minimum key.

Example: binary heap



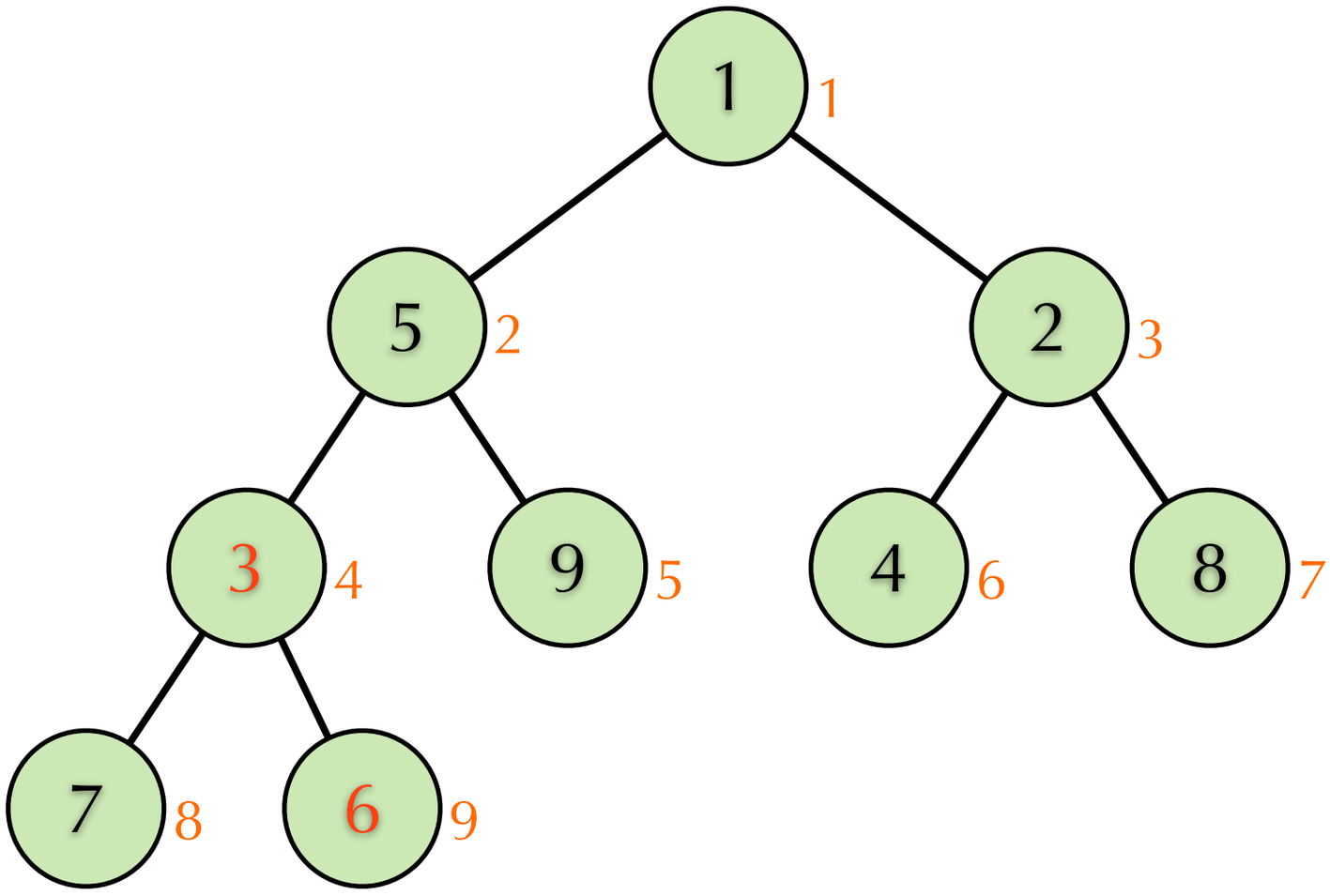
1	5	2	6	9	4	8	7		

Insert 3



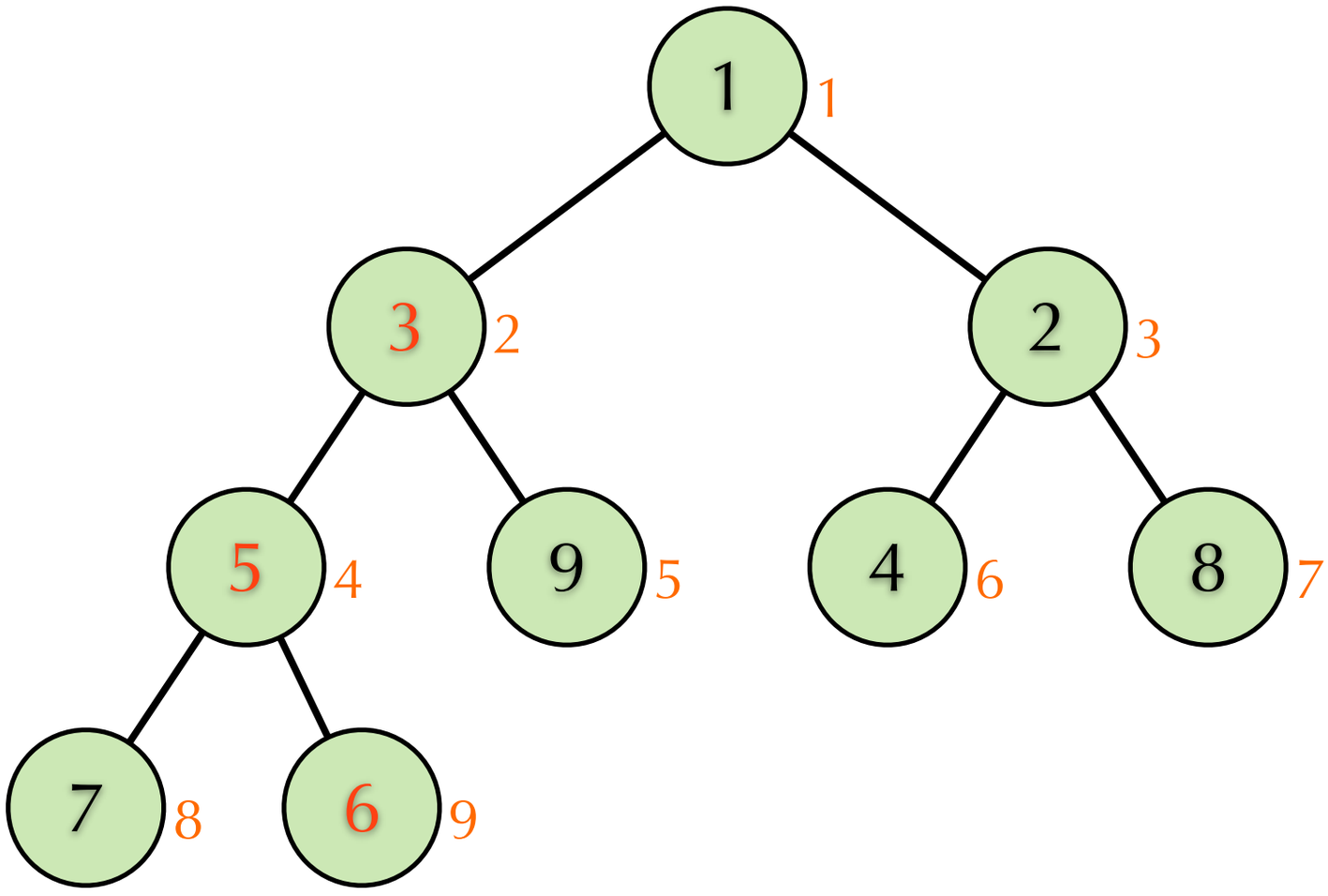
1	5	2	6	9	4	8	7	3	

Insert 3



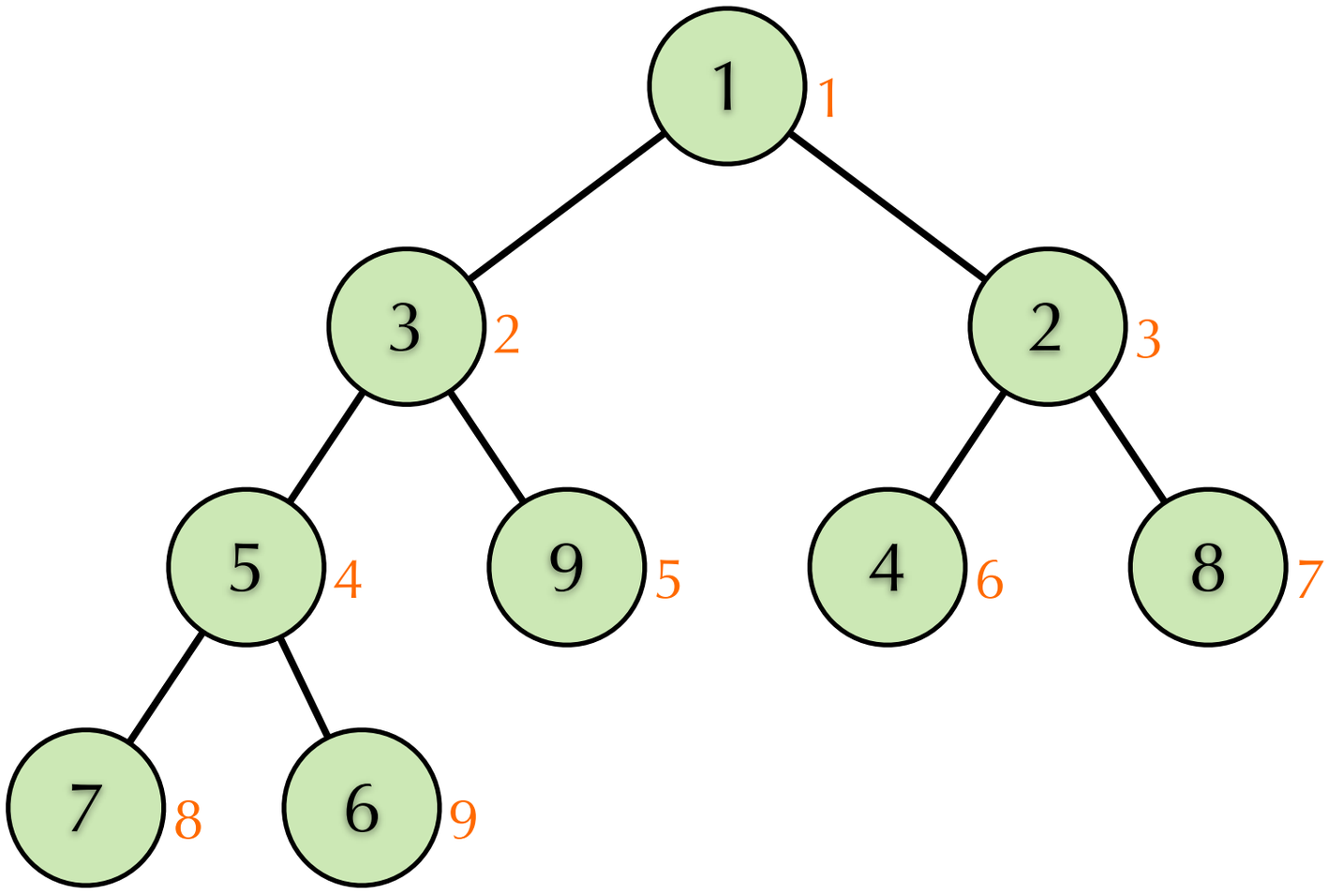
1	5	2	3	9	4	8	7	6	

Insert 3



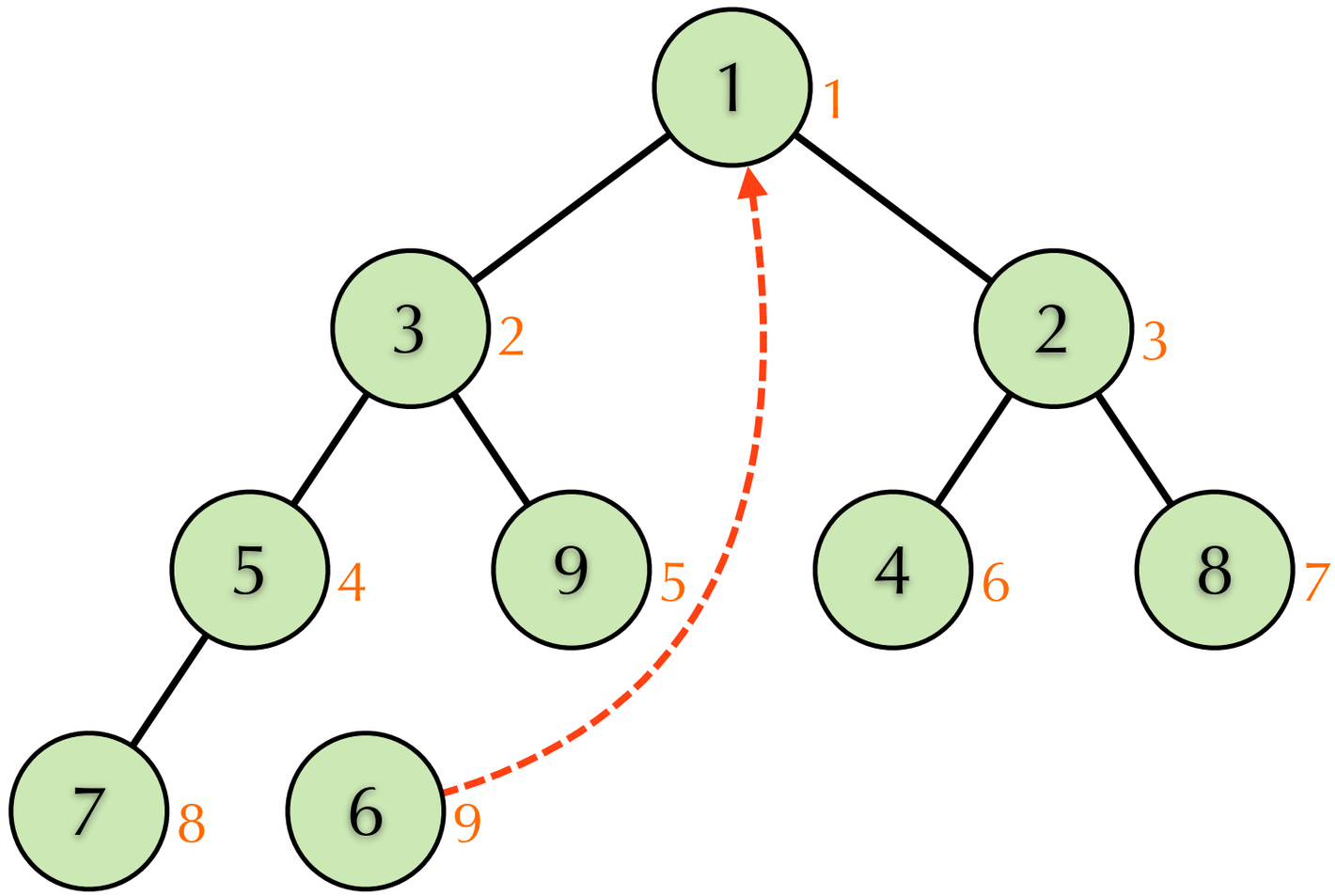
1	3	2	5	9	4	8	7	6	

Extract Minimum



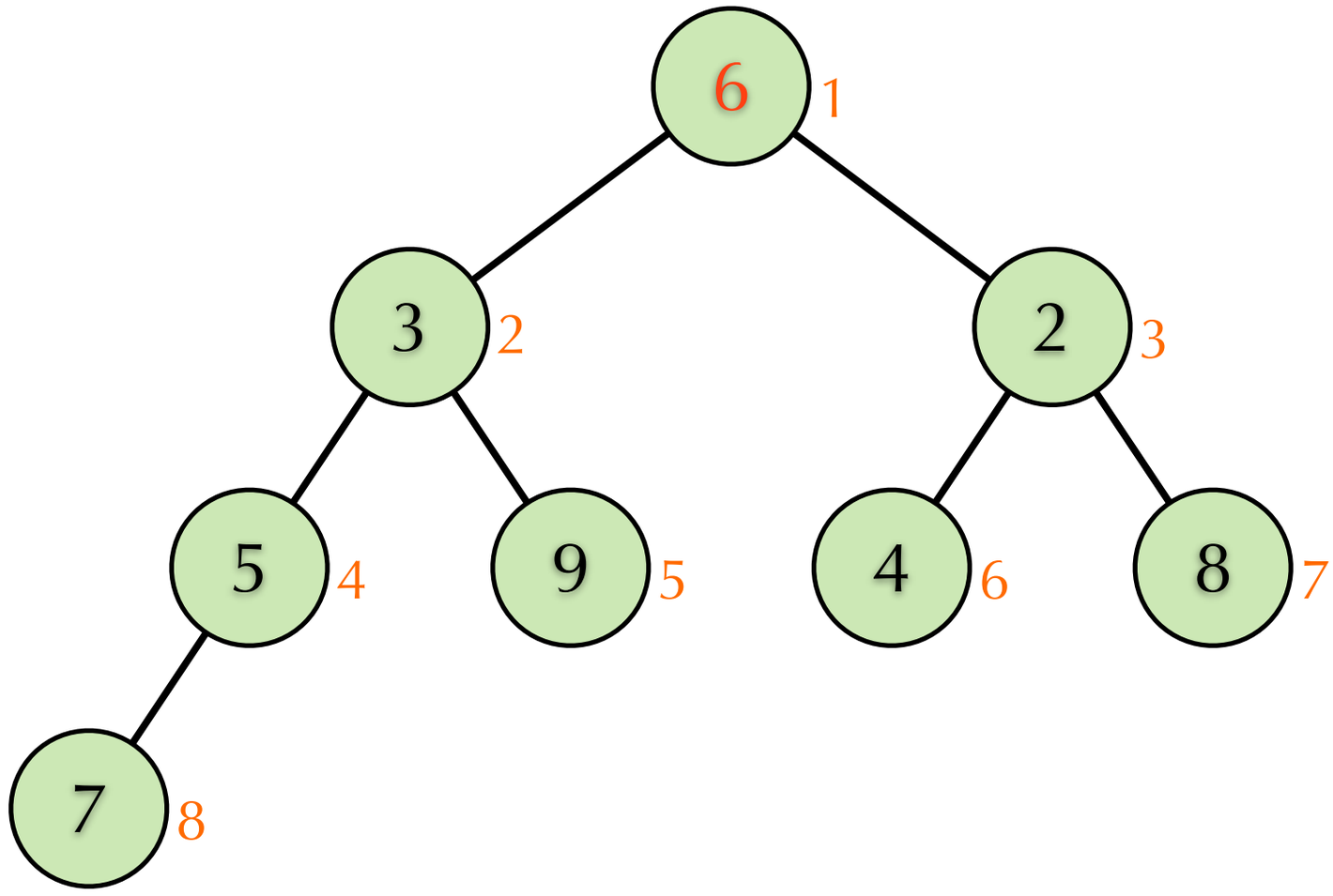
1	3	2	5	9	4	8	7	6	

Extract Minimum



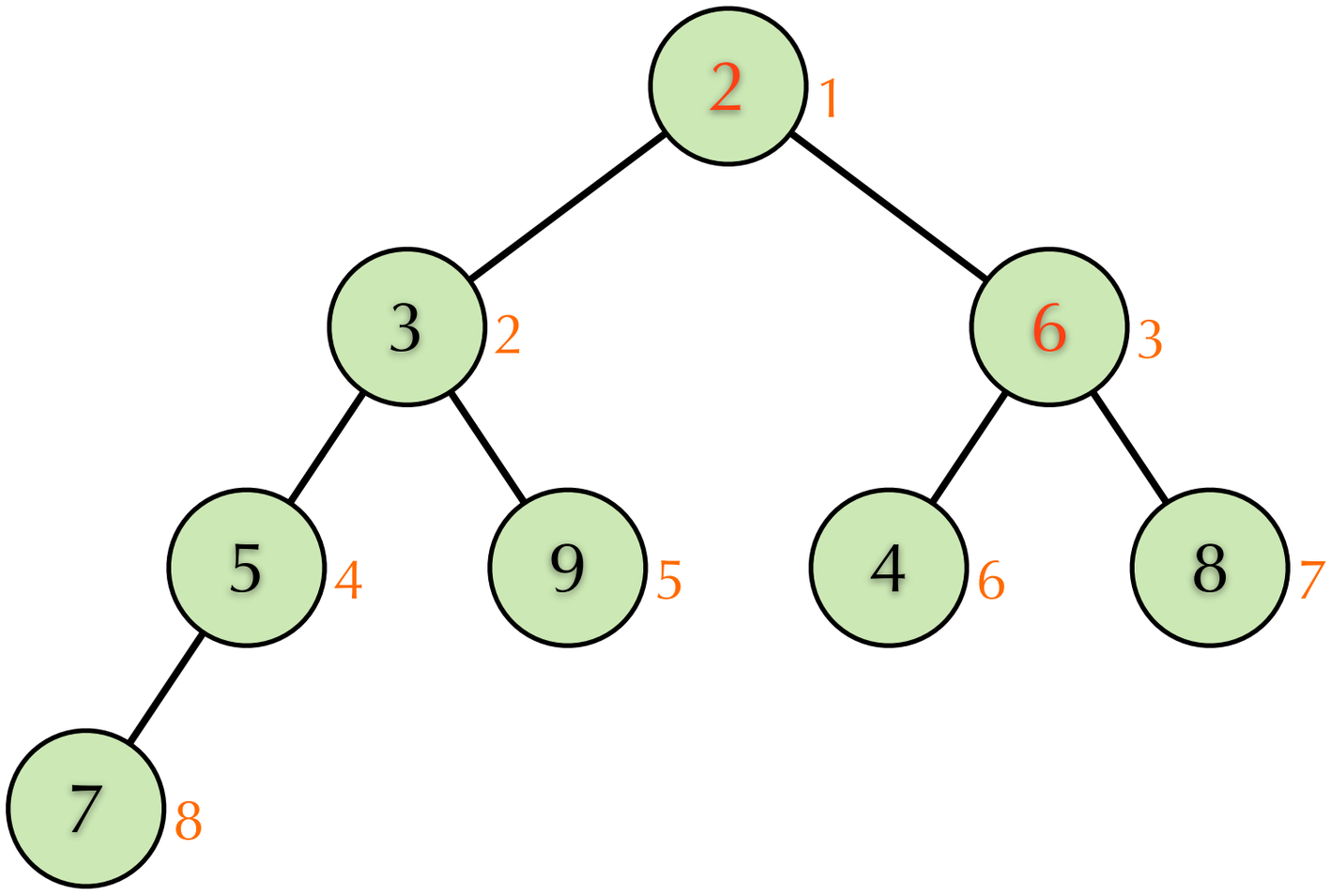
1	3	2	5	9	4	8	7	6	

Extract Minimum



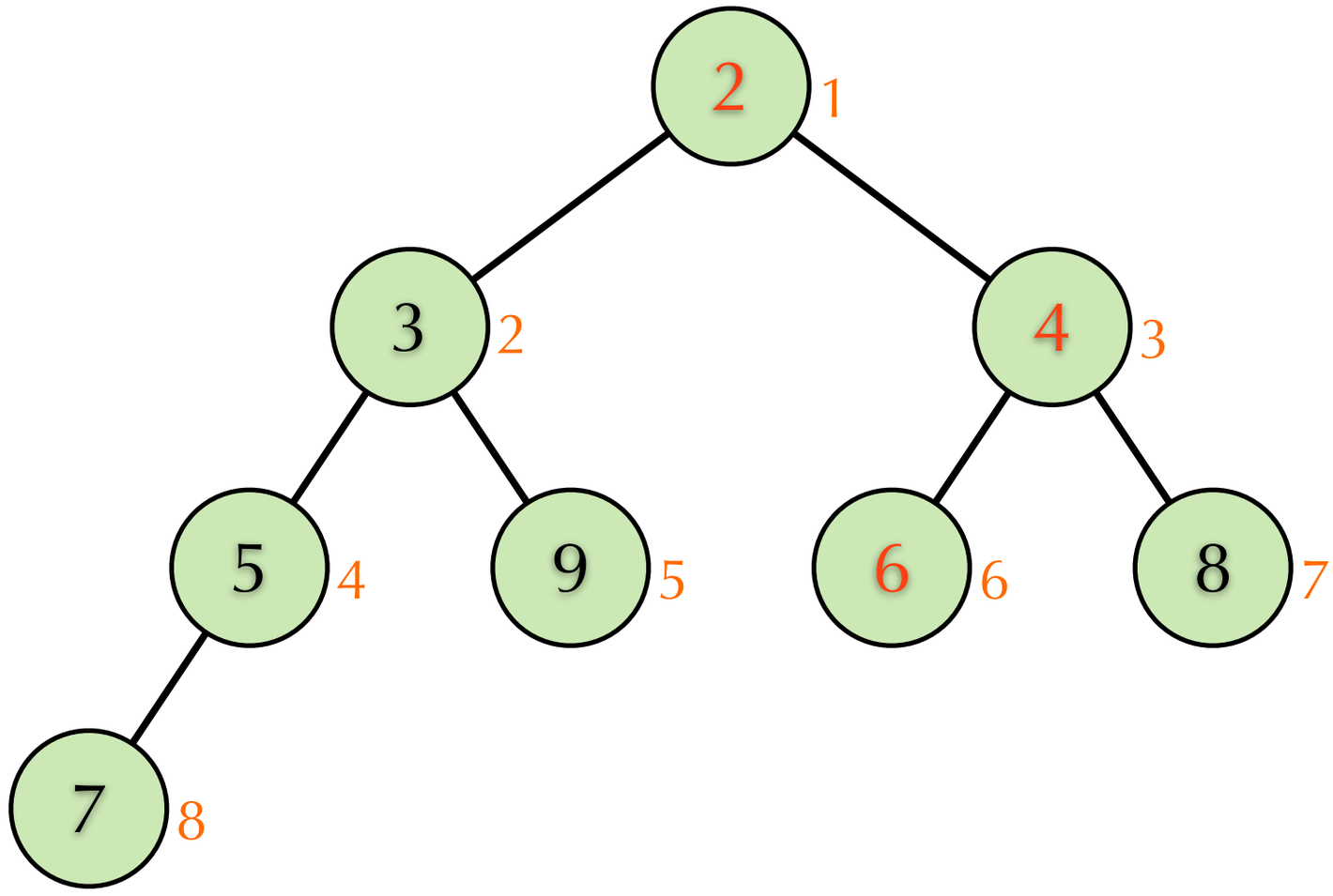
6	3	2	5	9	4	8	7		

Extract Minimum



2	3	6	5	9	4	8	7		

Extract Minimum



2	3	4	5	9	6	8	7		

Remainder

- Beware! Some node can have exactly one child!
- Beware! A node can have children out of bounds of the array.
- Arrays in C are zero-based.

Homework 3.2

- a) Does the space saving trick work for binary heaps? If it works, then how much extra time will it consume?
- b) When implementing binary heap, most programmers choose array based binary tree. Why?
- c) Given an n-element array. How to transform it to a min heap in O(n) time?