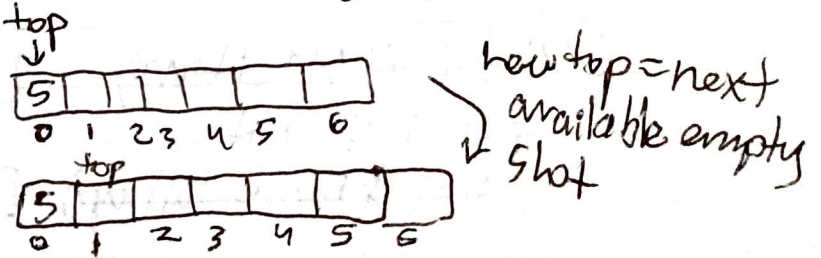
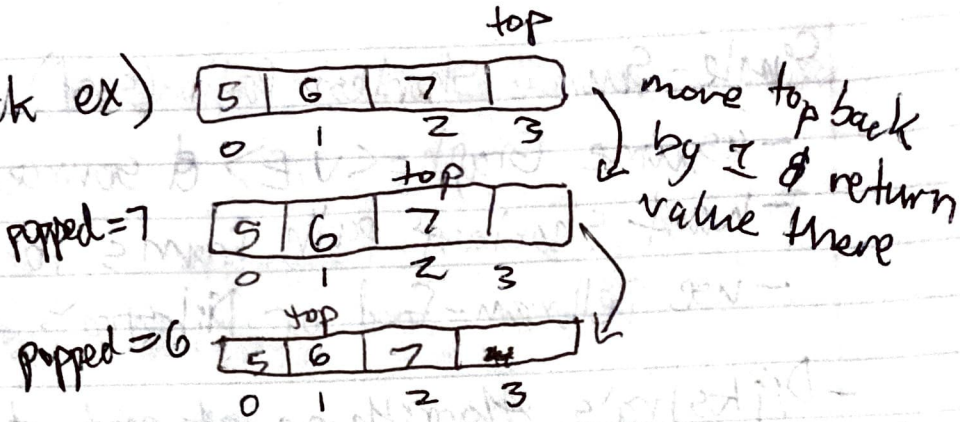
(Lecture 10)

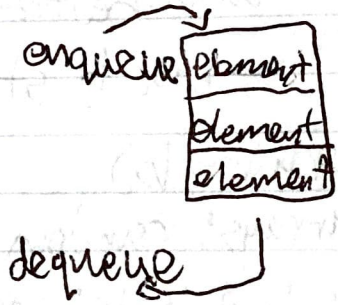
- arrays: random access allowed directly in const time
- linked lists: sequential access (only access in particular order)
- stacks & queues operate as containers for other data types
  - both have fixed operators, distinct ordering to elements
- stacks
  - added & removed by last-in-first-out (LIFO) order
  - capacity = # of stack elements that will fit
    (can be increased as # of elements ↑ by dynam reallocation)
  - pushing adds to top ex)



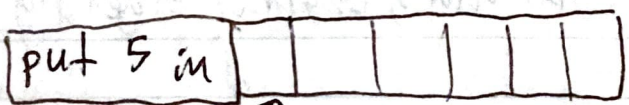
- popping off stack ex)

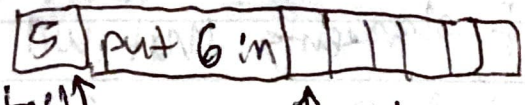


- queues = abstract data type (ADT), need: head, tail, size, & array pointer
  - also need constructor, destructurer
  - empty & full of predicates
  - enqueue inserts into queue
  - dequeues remove from queue
  - FIFO = first in first out order

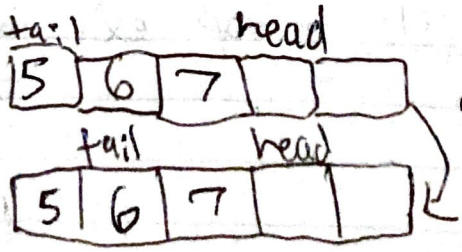


- ex) enqueuing items
  - head = next empty, available slot
  - tail = index of item to dequeue
  - insert at head & move head forward





- ex) dequeuing items (pass item at tail & move tail up)



- empty queque has same head & tail
- full queue if head+1 = tail (mod n)
- unbounded queues from linked lists possible
  - has wrapper, keeps track of head & tail

- stack removes most recently added element
- queue removes least recently added element

- priority queue: every element has priority associated w/ it
  - element of highest priority is dequeued before elements w/ lower priority
  - if 2 elements have same priority, preference given to position of element in queue

unit	size in bits	value	
bit	1	0/1	← smallest
nibble	4	hex digit	
byte	8	ASCII	← smallest addressable
halfword	16		
word	32	native size, register length	
long word	64		

- logical shift moves everything L or R & adds 0 to empty slot
- arithmetic shift left = 0s from right & right = sign bits from left
- << = left shift & >> = right shift

XOR ^:

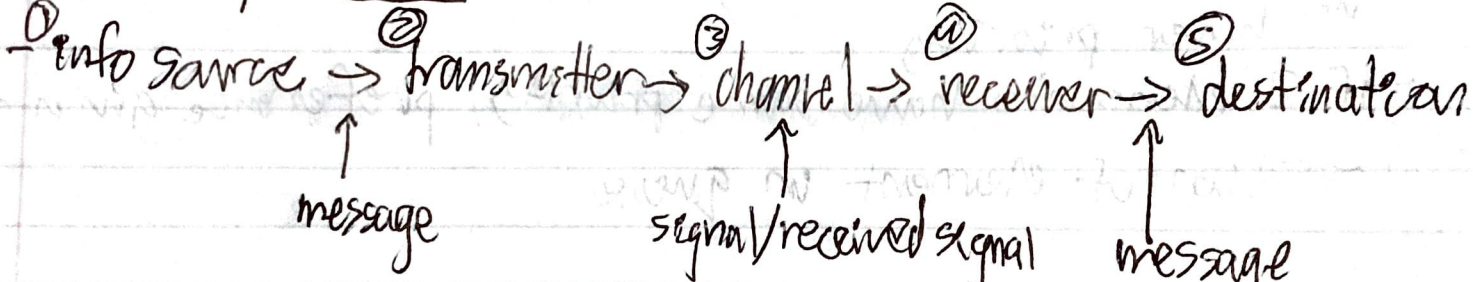
^	1	0
0	0	1
1	1	0

Sets

- Set operations: $A \cap B$ intersection
  $A \cup B$ union
  $A - B = A \cap \overline{B}$ difference
  $\overline{A}$ or A^c complement

- setting bits use OR
- clearing bits use AND
- getting bits use shift right & AND

Data Compression



① produces message to be communicated

② operates on message, produces signal for channel

③ medium to transmit signal

④ performs inverse operation of transmitter

⑤ target of message

Entropy = uncertainty of event's occurrence $H = -\sum_{i=1}^{n} p_i \log_2(p_i)$

↳ average # of Q's needed to guess random symbol picked from message

Huffman: ASCII symbol, make histogram of 256 indices
↳ pq: lower freq = higher priority
↳ tree, code table, dump tree ⇔ reconstruct tree

[LZ78 Encoding example in slides #15] [decoding too]