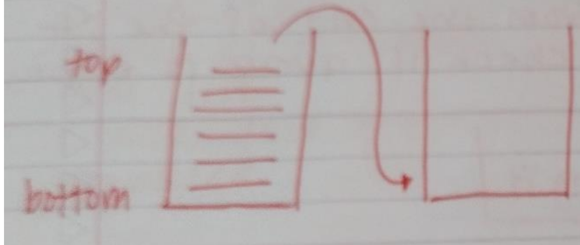
Queue Data Structure (Design One)
Functions:
- Enqueue
- Dequeue
- Peek (optional)



What we're looking to do:

Queue Data Structure (Design two)
Functions:
- Insert
- Delete
- Peek

Objective: allow insert or removal of data at a given location (rather than at the top or bottom.)

$$\begin{tikzcd} & \text{Queue (Linked list)} \arrow[dl] \arrow[dr] & \\ \text{Enqueue} & & \text{Dequeue} \end{tikzcd}$$

Enqueue

- check if queue is full
- If true, begin pushing values/elements from list.
- If false, task is completed.

Dequeue

- check if queue is empty (NULL)
- If true, task is completed.
- If false, begin popping values/elements from list.

Create struct:

- our values could reference real people waiting in line like their name/ID number

```
struct QueueNode {
    int id; // or name or both!
    QueueNode *next;
};
struct Queue {
    void Enqueue () {
        ~~void~~ bool checkQueue () {
            if true // if Queue is full
                start push // go to next element in list
            if false // if Queue is empty
                return // task finished.
        }
    void Dequeue(){
        if true // if top of list == NULL
            return //task finished
        if false // Queue still has values/elements
            start pop
```

$$\begin{tikzcd} & \boxed{\text{LIST}} \arrow[dl] \arrow[dr] & \\ \underline{\text{Insert( )}} & & \underline{\text{Remove( )}} \end{tikzcd}$$

inserts new value/
• element to desired position

Removes element/value
• from the desired position in the list

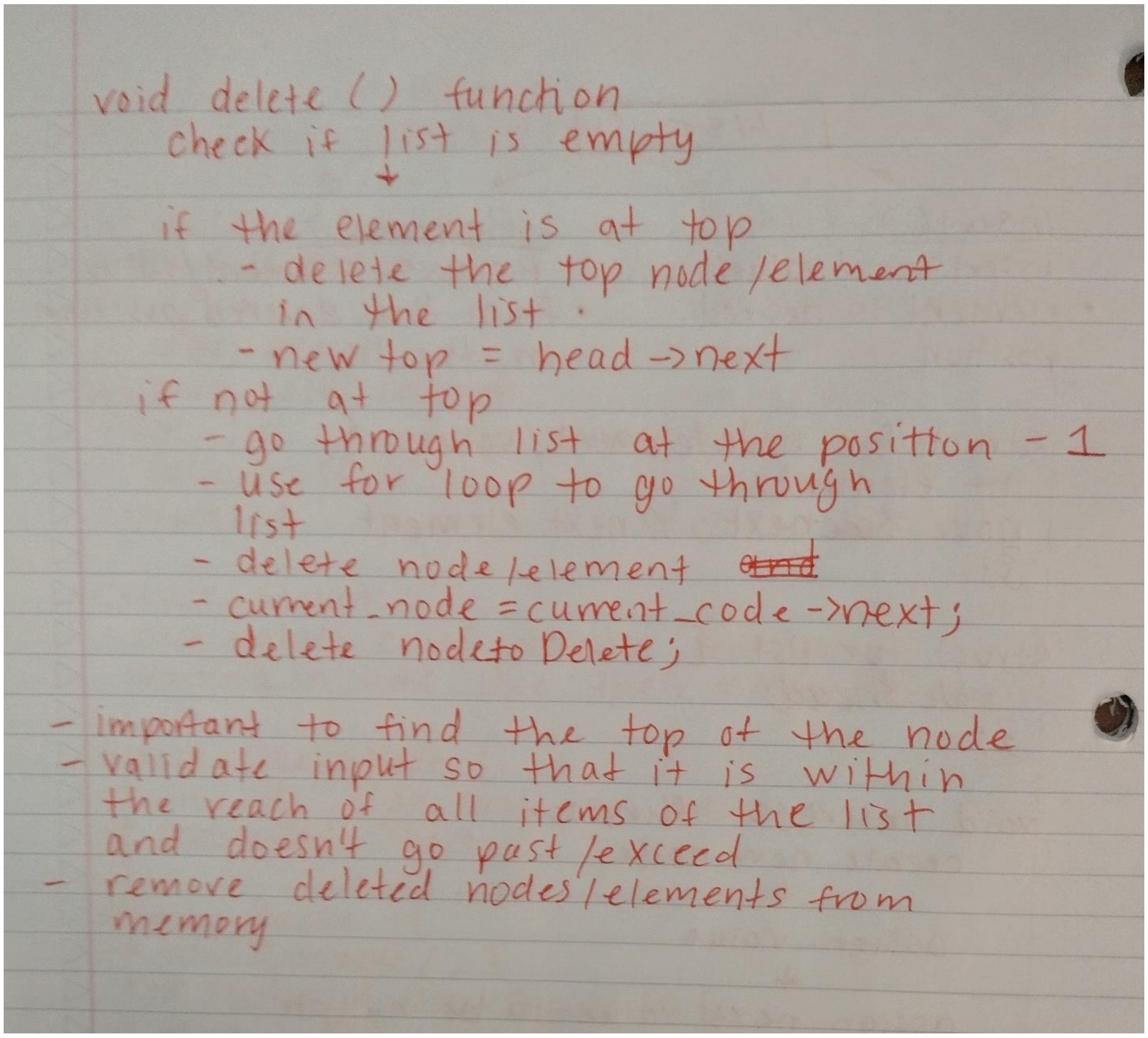
struct for node/elements/values {
int element;
node *~~ele~~next; //next element
};

struct for list {
node *head;
};

void insert() function
create new node
$\downarrow$
assign value
$\downarrow$
assign next value to be nullptr

if position in list is at the top, everything else must be pushed down the list

if anywhere else in list, find position – 1
using for loop to increment accordingly
if greater than length of list
return an error or do nothing



void delete () function

  check if list is empty

  ↓

  if the element is at top
  - delete the top node/element in the list.
  - new top = head->next

  if not at top
  - go through list at the position - 1
  - use for loop to go through list
  - delete node/element ~~and~~
  - current_node = current_code->next;
  - delete nodetoDelete;

- important to find the top of the node
- validate input so that it is within the reach of all items of the list and doesn't go past/exceed
- remove deleted nodes/elements from memory