Midterm Topics

- Stack
- Queue
- LLL, CLL, DLL
  - Ordered List
    - Absolute (holes)
    - Relative (no holes)
  - Efficiency Discussion

CLOSED BOOK
CLOSED NOTES
1 hour 50 min
- Modulo 2 bit
- The only thing you need to know before you use - all have to be initialized - all contiguous
- all arrays same size
- all objects have same size

↑ Memory = at compile time "guess"

↓ Direct Access

Array [Index]

↑

Absolute List (House)
pos (current) pos

\begin{align*}
\text{previos} & \rightarrow \text{next} = \text{current} \\
\text{previos} & \rightarrow \text{next} = \text{previos} \\
\text{previos} & \rightarrow \text{next} = \text{new node}
\end{align*}

\frac{1}{2} \left\lfloor \text{current} \rightarrow \text{next} \right\rfloor

\text{while (current. pos > current pos)}

\text{node current = head;}

\text{node previos = NULL}
Queues

Figure 1: Schematic diagram of a queue.

- Queue operations: enqueue (add), dequeue (remove)
- Front and rear indicators
- Double ended queue
(a) Compare the efficiency of using an array versus a linear linked list for implementing a stack ADT.

1. (25 points) Short Answer. Limit your answers to 1-2 sentences.
List that does not allow holes (and why)

(a) Why wouldn't you pick an array for a "relative" ordered list?

(b) Which data structure is best for implementing an "absolute" ordered list (a list that allows holes) and why?
What are the problems with arrays in general (List 2)?

Memory usage:

Run time efficiency:

Select an array as a data structure. Now that we have talked about arrays, explain why we might
Questions

What data members would you need for the allocated array?

The data in each node is a pointer to a dynamically allocated array. Implementing using a linear linked list, using the following prototype.

Write the code for the dequeue operation when
d. Write a function that will copy a linear linked list of

\[
\text{head} \quad j
\]

\(? should the top be? (The head or the tail)?

b. When implementing a stack using a linear linked list, where

\{ queue int dequeue () { returns false if there are no items in the

You may have a tail pointer if you choose.

Write the code to remove the last node from a linear linked list.
How would this change if it were a doubly linked list. Just show the code changes here... don't re-write it all!
struct node
{
    node *next;
    char *name;
}

3. Write a function that will copy every other node (2nd, 4th, 6th, etc.) in a linear linked list (excluding the names in the list) to a new NUL list. Assume that the new list has a head pointer that is NULL to begin with. Make no assumptions about the list that you are copying from. Make sure to completely copy the data.
2. (30 points) Write a function that will copy a doubly linked list (including the names in the list) to a new doubly linked list: Perform a complete copy of all data.

```c
struct node {
    char * name;    // a doubly linked list of dynamic names
    node * next;
    node * prev;
};
```

- Copy a LLL
- Find the longest item in a LLL
- Copy an array into a new LLL
- Copy a CLL into a LLL
- Convert a LLL into a CLL

- Remove the last node in a LLL
  - with a tail
  - without a tail

- Implement dequeue
  - using a LLL
  - using a CLL

- Implement Push, Pop LLL
Write class interface for a stack using a dynamically allocated array of videos.

class Stack
{
public:
    stack();
    ~stack();

private:
    video * array;
    int top;
};