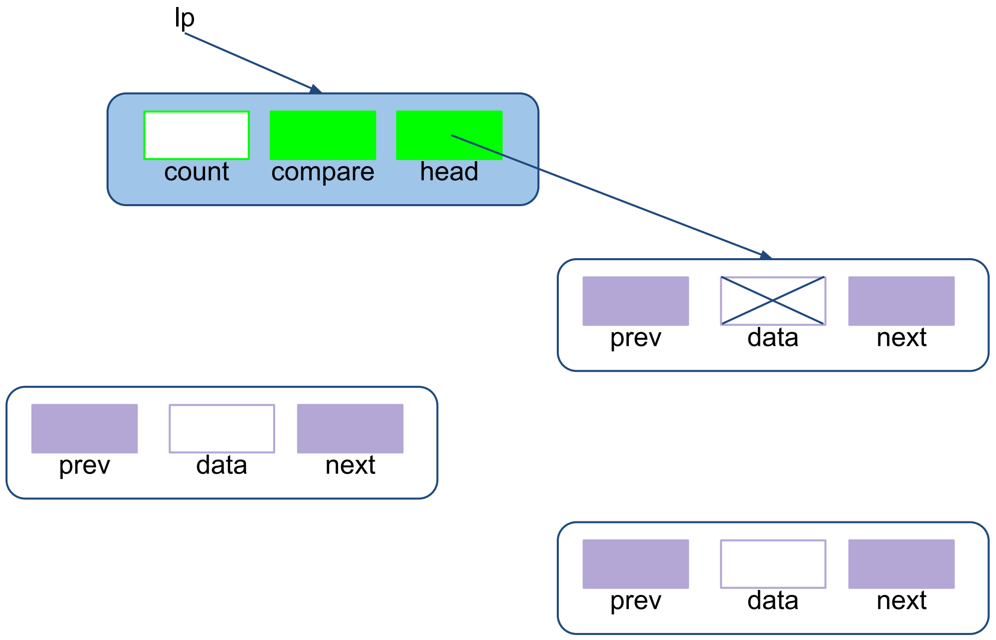
Part 1: Deque

Project 4 - week 1

List Interface

- list.h file
- Only implement those functions in your list.c for the first week.
- Complete the list.c before next week's lab

Structures

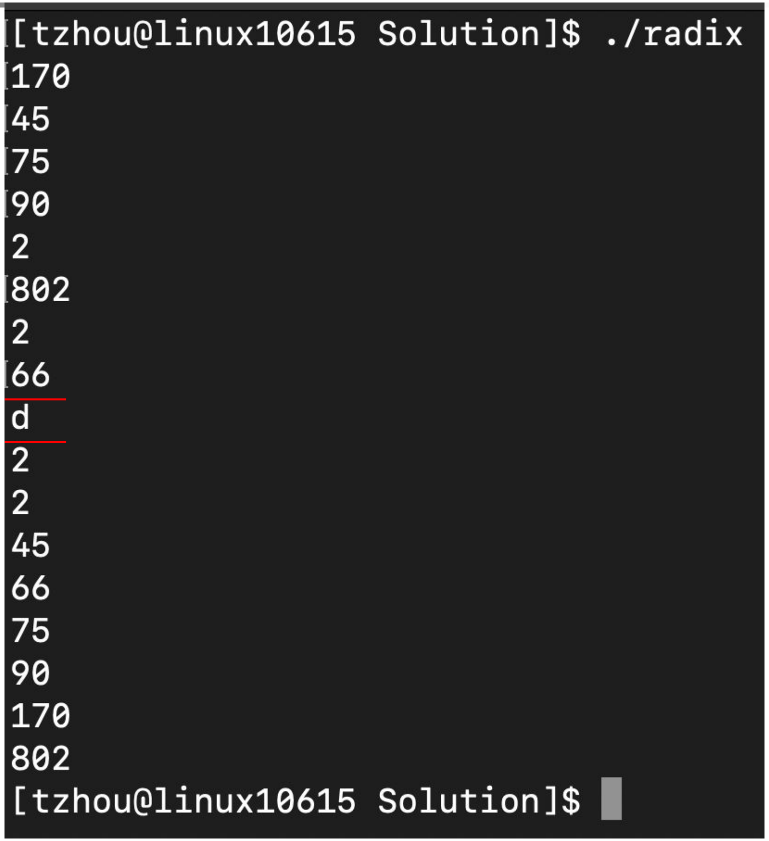


```
struct list{
    int count;
    struct node *head;
    int (*compare)();
};

struct node{
    void *data;
    struct node *next;
    struct node *prev;
};
```

Test Cases - “radix”

- Enter positive integers and any letter to start sorting.
- Only testing `addLast`, `removeFirst`, and `numItems`



```
[tzhou@linux10615 Solution]$ ./radix
170
45
75
90
2
802
2
66
d
2
2
45
66
75
90
170
802
[tzhou@linux10615 Solution]$
```

“Radix” workflow

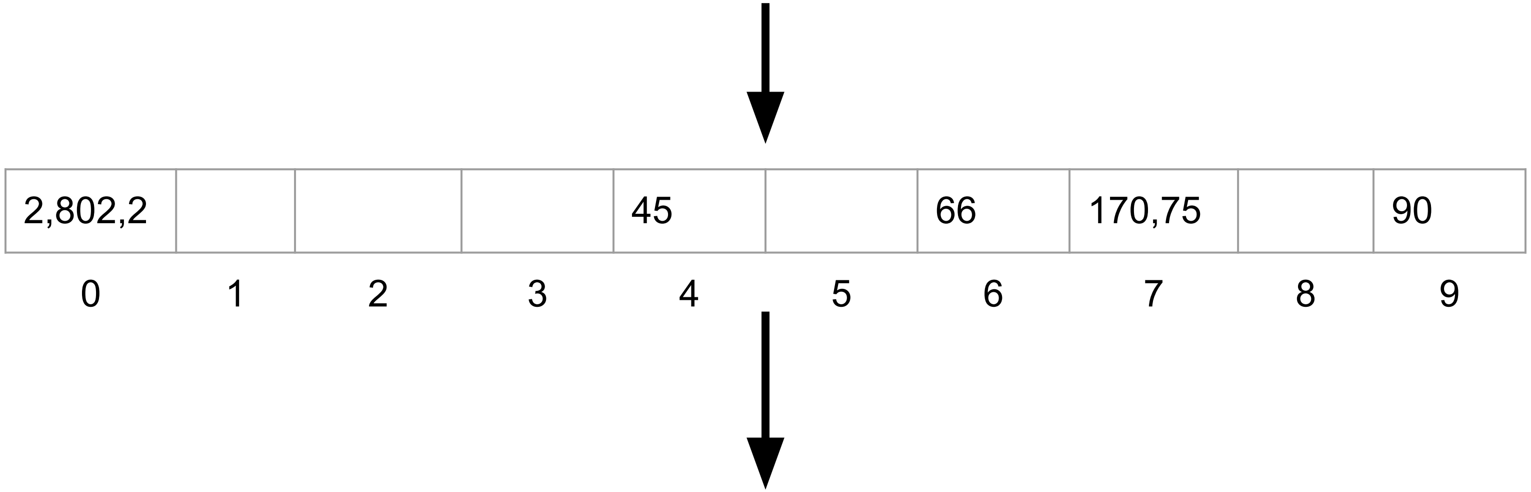
Buffer List: [170, 75, 45, 90, 2, 802, 2, 66] --------------- round 0 (inputs)

170,90		2,802,2			75,45	66			
0	1	2	3	4	5	6	7	8	9

Buffer List: [170, 90, 2, 802, 2, 75, 45, 66] --------------- round 1 (ones)

“Radix” workflow

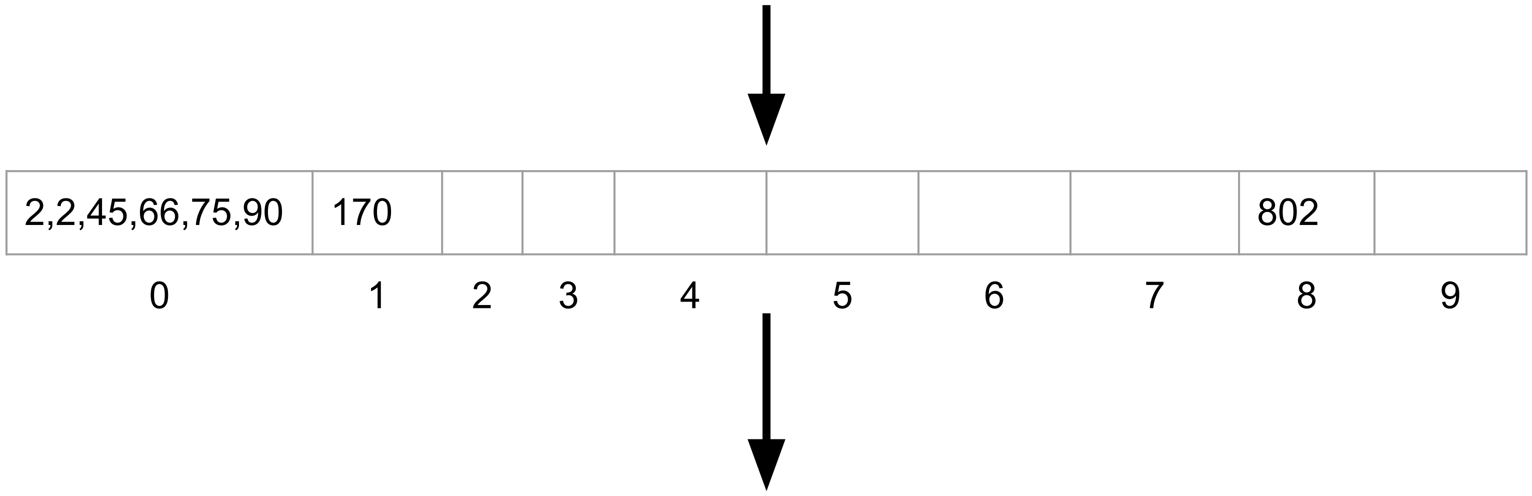
Buffer List: [170, 90, 2, 802, 2, 75, 45, 66] --------------- round 1 (ones)



Buffer List: [2,802,2,45,66,170,75,90] --------------- round 2 (tens)

“Radix” workflow

Buffer List: [2,802,2,45,66,170,75,90] --------------- round 2 (tens)

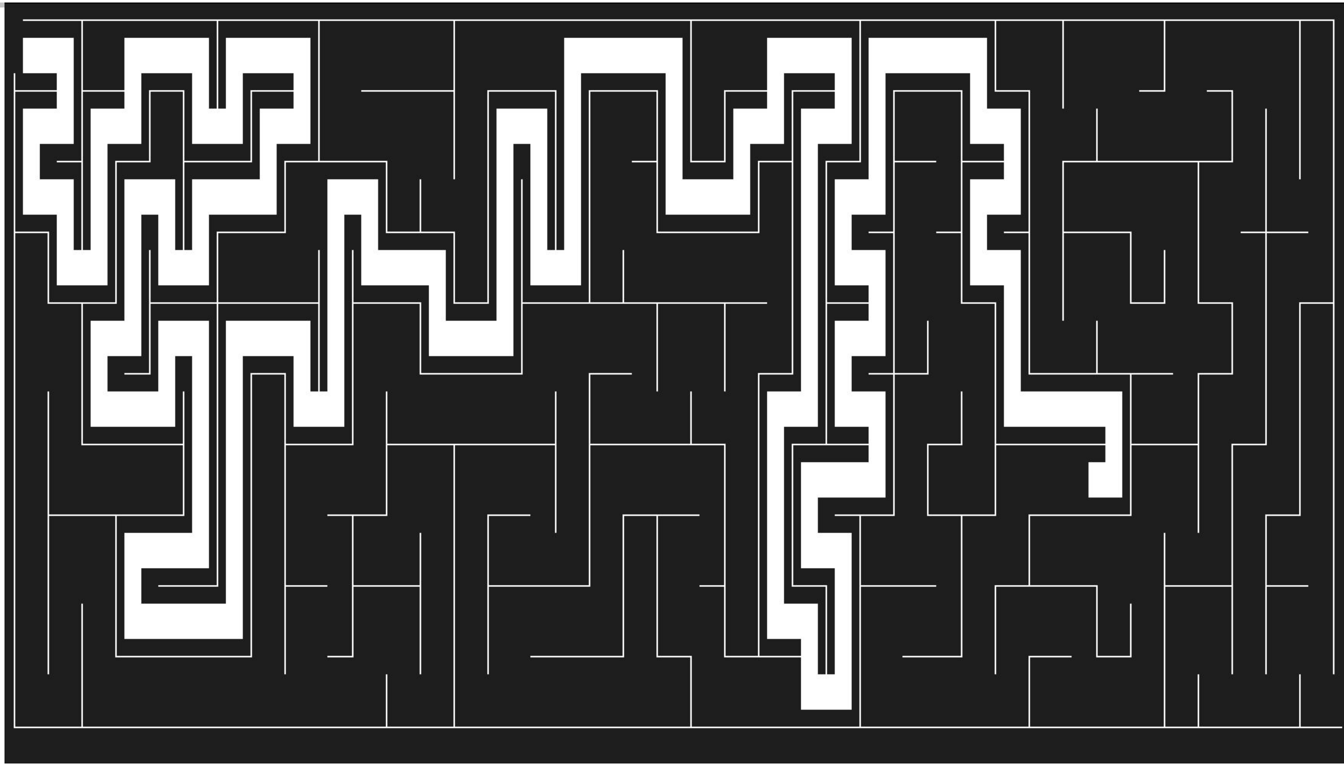


2,2,45,66,75,90	170							802	
0	1	2	3	4	5	6	7	8	9

Buffer List: [2,2,45,66,75,90,170,802] --------------- round 3 (hundreds)
done!

Test Cases - “maze”

- Shrink window size first
- Crtl + c to terminate the program



Debugging "maze"

- Building "maze":
  - `addFirst`, `removeFirst`, and `numItems`
- Solving "maze":
  - `addLast`, `removeLast`, and `getLast`

Notes

- Draw the graph to help you debug.
- Finish the rest of the functions before next week's lab:
  - `getFirst`, `removeItem`, `findItem`, `getItems`