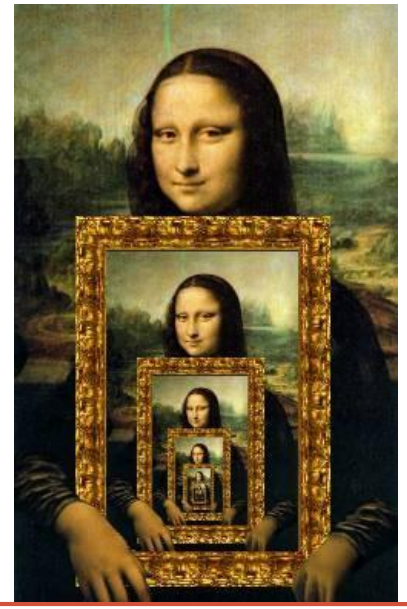
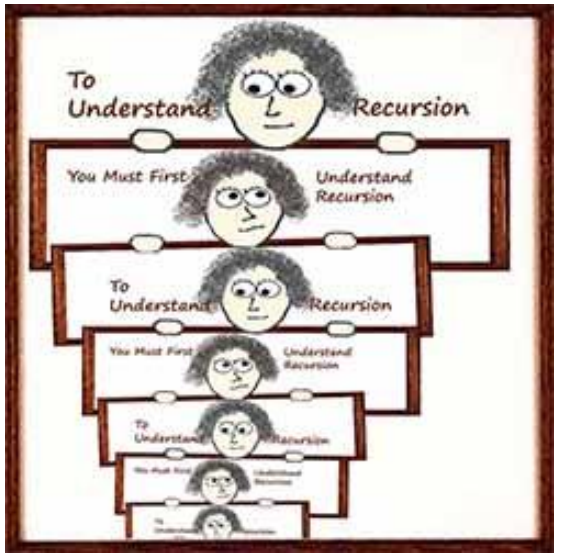
RECURSION AND FINAL REVIEW



Problem Solving with Computers-I

https://ucsb-cs16-wi17.github.io/



GitHub

Final Exam!

- Thursday (03/23) 4pm to 7pm NH 1006 and a overflow room
- Assigned seating will be posted on Piazza
- Everything we have covered so far is on the exam
- Duration: **3 hours**
- Closed book: no calculators, no phones, no computers
- Only 1 sheet (***double***-sided is ok) of written notes
  - Must be no bigger than 8.5” x 11”
  - You have to turn it in with the exam

How do you feel after the last two labs ?

A

B

C

D

E

Lab 08: When should you use a helper function?

```
bool isPalindrome(const char *s1) //recursive
```

deTartraTED

WasItACarOrACatISaw

Do we need a helper function for a recursive implementation of the above function?

A. Yes
B. No

Steps towards a recursive solution:

1. Identify the recursive structure in your input and or problem
2. Write the recursive step in plain English
3. Do you need a helper function?

Lab 08: Thinking recursively!

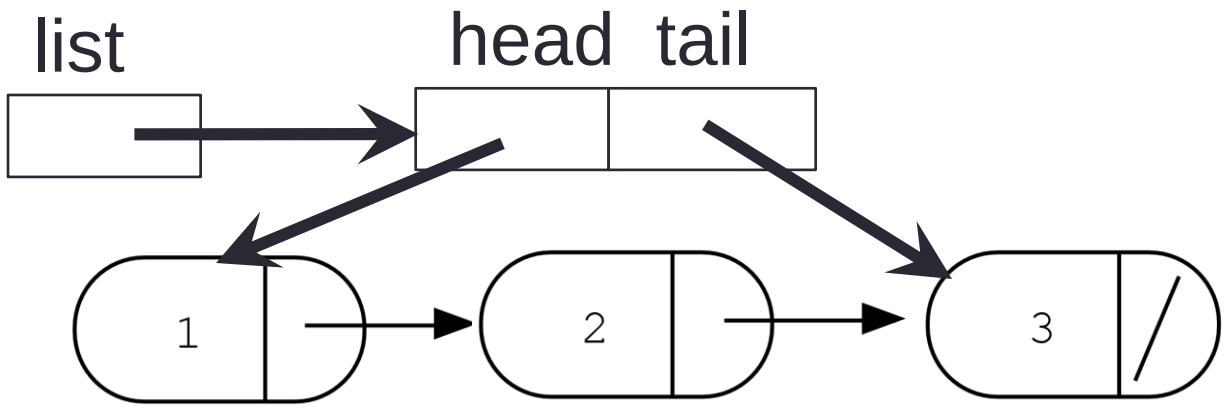
deTartraTED
WasItACarOrACatISaw

```
bool isPalindromeHelper(const char *s1, int len) //recursive
```

Steps towards a recursive solution:

1. Identify the recursive structure in your input and or problem
2. Write the recursive step in plain English
3. Do you need a helper function?
4. Implement (and test) the base case
5. Believe that you already have a correct implementation of the function that works on all smaller size inputs
6. Implement (and test) the recursive case

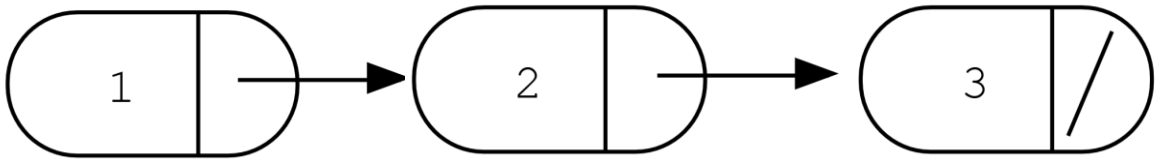
Recursion on lists: compute the sum of all elements




```
int sum(LinkedList *list)
```

Recall the steps towards a recursive solution

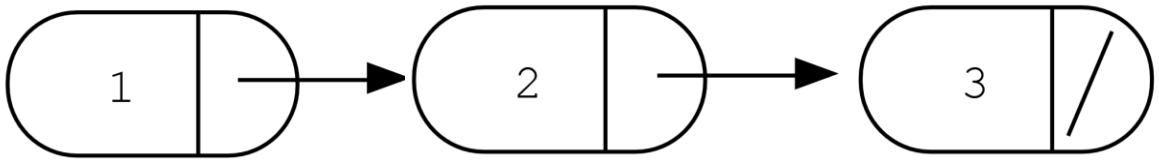
Recursion on lists: compute the sum of all elements




```
int sumHelper(Node *head)
```

1. Identify the recursive structure in your input and or problem
2. Write the recursive step in plain English
3. Do you need a helper function?
4. Implement (and test) the base case
5. **Believe that you already have a correct implementation of the function that works on all smaller size inputs**
6. Implement (and test) the recursive case

Recursion on lists: compute the sum of all elements

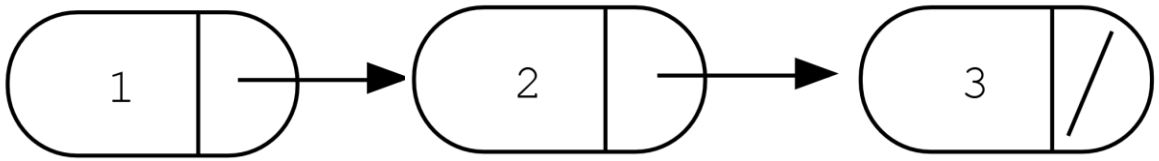



```
int sumHelper(Node *head){
    if(head!=NULL)
        head->data+sum(head);
}
```

Which of the following is true about the given implementation?

A. It is correct
B. It will not return the correct sum
C. It will result in a segfault

Under the hood of recursive calls (review)

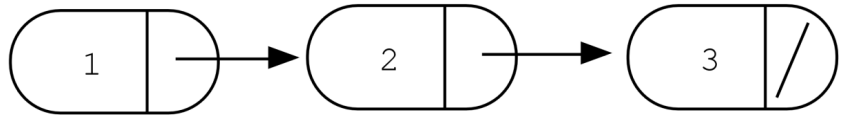



```
int sumHelper(Node *head){
    if(head==NULL)
        return 0;
    return head->data+sum(head);
 }
```

```
int sum(LinkedList *list){
        sumHelper(list->head);
}
```

Recursion on lists: delete a value recursively

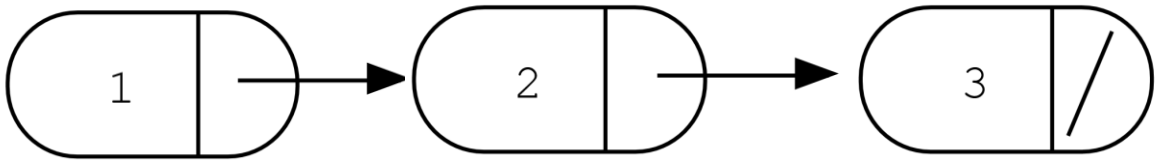
void deleteNodeRecursive(LinkedList *list, int value)



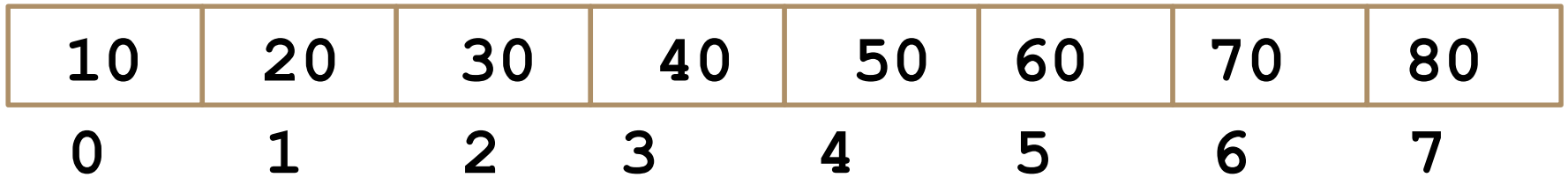
Node* deleteNodeRecursiveHelper(Node *head, int value)

Recall the steps towards a recursive solution

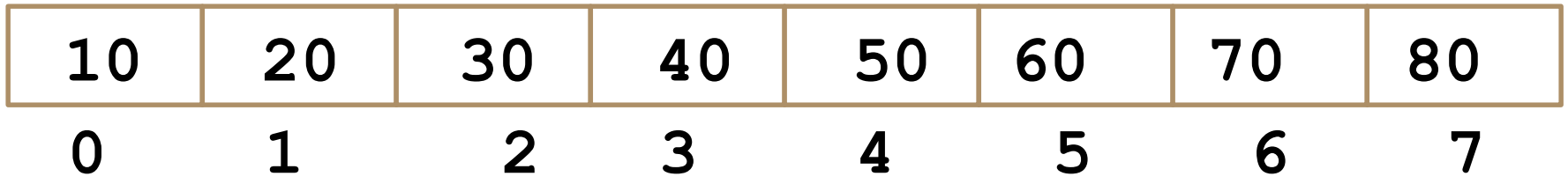
How do you decide which data structure to use?



Searching for a value in a sorted array



Searching for a value in a sorted array

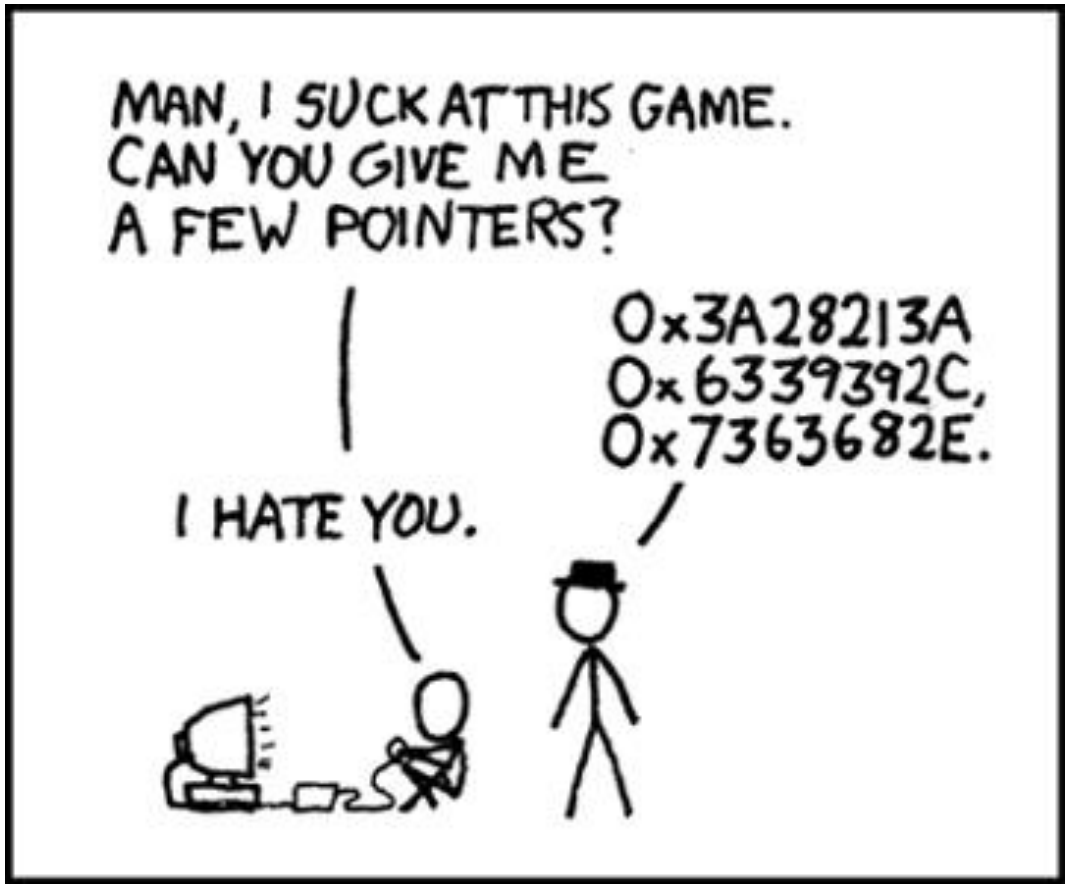


Recall the steps towards a recursive solution

Some comic relief…

HTTP://XKCD.COM/1296/

Some comic relief



HTTP://XKCD.COM/138/

Final words…

Which concepts do you need more help with?

Stay posted on Piazza

- Tutor lab hours tomorrow in CSIL
- Extended office hours next week
- Seating arrangement for exams (you may be in a different exam hall)

Final words

- You can debug your code!
- …….But you have to write it systematically!

Good luck with the final and PA8!