FINAL PRACTICE!

Problem Solving with Computers-II

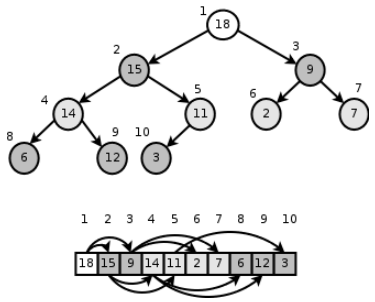
Link to class handout:

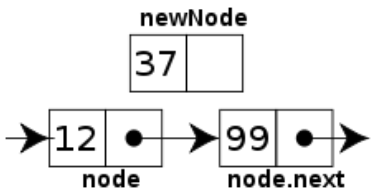
https://bit.ly/one-problem-to-rule-them-all

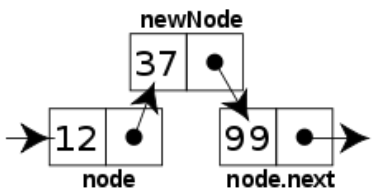
I can deal with pressure, and deadlines.

Resources for the Final Exam

- Leet code problems categorized to help you focus on the most relevant ones.
  - Those marked as “Optional challenge” won’t be on the final
- Code from lectures: https://github.com/ucsb-cs24-w26/cs24-w26-lectures
- Last year’s final exam posted on Canvas for Practice (under module 10)
- Tool to visualize data structures: https://visualgo.net/
- If you missed your Mock interview, complete during section tomorrow.







INSERTION-SORT(A)	*cost*	*times*
1 **for** $j = 2$ **to** $A.length$	c_1	n
2 $key = A[j]$	c_2	$n-1$
3 // Insert $A[j]$ into the sorted sequence $A[1..j-1]$.	0	$n-1$
4 $i = j-1$	c_4	$n-1$
5 **while** $i > 0$ and $A[i] > key$	c_5	$\sum_{j=2}^{n} t_j$
6 $A[i+1] = A[i]$	c_6	$\sum_{j=2}^{n} (t_j - 1)$
7 $i = i-1$	c_7	$\sum_{j=2}^{n} (t_j - 1)$
8 $A[i+1] = key$	c_8	$n-1$

Data Structures and C++

Complexity Analysis

One problem to rule them all!

Problem (LP04): Longest Consecutive Sequence

Description: Given an unsorted array of integers `nums`, return the length of the longest consecutive elements sequence.

Try solving it in as many different ways as you can, then pick the best!

Ex: `nums = [100, 4, 200, 1, 3, 2]`

Output = 4

`nums = [1, 0, 1, 2]`

output = 3

Approach 1 → Sort + ?

```
nums = [100, 4, 200, 1, 3, 2]
```

= [1, 2, 3, 4, 100, 200]

nums[i] + 1 == nums[i+1] ?

i			seqLen = 1	maxLen = 1
0	1 + 1 == 2	✓ (T)	2	
1	2 + 1 == 3	✓ (T)	3	
2	3 + 1 == 4	✓ (T)	4	
3	4 + 1 == 100	X (F)	1	
4	100 + 1 == 200	X (F)	1	

4 , 1

① max(maxLen, seqLen) = 4
② seqLen = 1

↑ Repeat

After

maxLen = 4

nums = [1, 0, 1, 2]

= [0, 1, 1, 2]

Think through case with duplicates.

i	seqlen = 1
0	2
1	2
2	3

If there is a duplicate do nothing

```
if ( nums[i] == nums[i+1])
    continue //do nothing
```

Approach 2 – Use a Data Structure that maintains order (std::set) or partial order (priority-queue)

```
nums = [100, 4, 200, 1, 3, 2]
```

```
nums = [1, 0, 1, 2]
```

num : nums

1

2

3

4

100

200

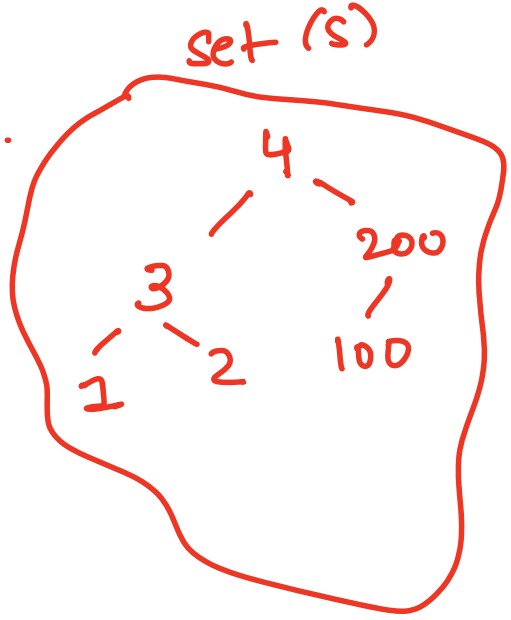
Check if next element is consecutive

① s.count(num+1) == 1

② s.contains(num+1)

③ Use an iterator

} ways to find a key in a set.



Approach 3

no sorting or sorted DS.

```
nums = [100, 4, 200, 1, 3, 2]
```

num: H

Avg. all find, contains, count

(3)

H. contains (4) ✓

H. contains (5) X

seqlen = 2

(1)

H. contains (2) ✓

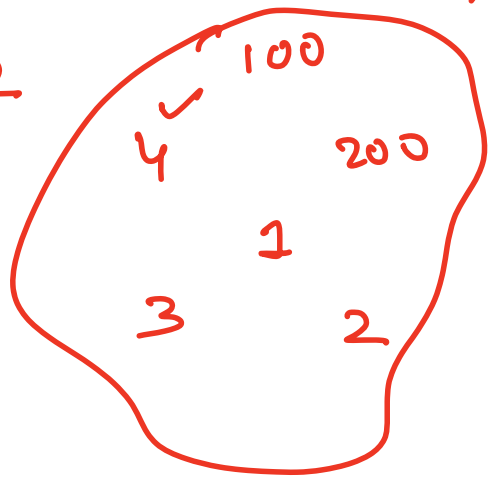
H. " (3)

H. " (4)

" (5) X

seq len = 4

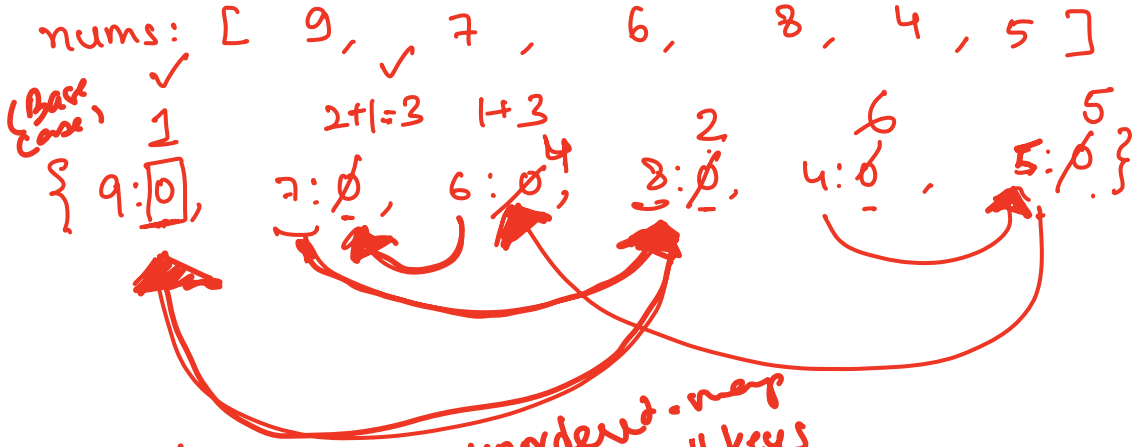
hashtable (H)



You can also visualize as

{ 100, 4, 200, 1, 3, 2 }

Approach 4 ?



Pseudocode

Initialize M[key]:0 for all keys ← unordered_map

max_len = 0

for each key (k) in M:

maxlen = max (maxlen, lcs(M,k))

Pseudo code.

lcs (M: unordered_map, key : int)

if key ∉ M: // key does not exist

return 0

if M[key] > 0

return M[key]

return 1+lcs(M, key+1)

Approach: Recursively compute the lcs for each key.

Use an unordered-map to store the lcs starting at each key. ← Also called memoization

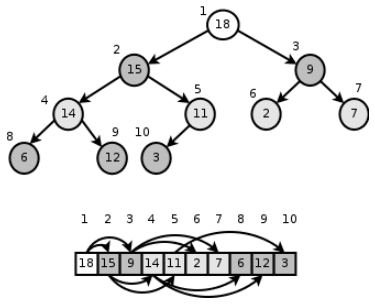
and reuse computed values.

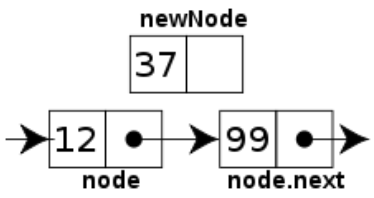
CS24: Learning goals

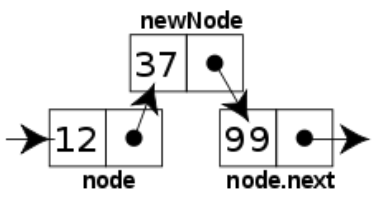
Correct coding, clear thinking, no AI shortcuts

- Design and implement **larger programs** that **run fast**
- Organize **data** in programs using **data structures**
- **Analyze** the **complexity** of your programs
- Prep for **technical interviews**

Grow confident in your problem solving skills!







INSERTION-SORT(A)	*cost*	*times*
1 **for** $j = 2$ **to** $A.length$	c_1	n
2 $key = A[j]$	c_2	$n-1$
3 // Insert $A[j]$ into the sorted sequence $A[1..j-1]$.	0	$n-1$
4 $i = j-1$	c_4	$n-1$
5 **while** $i > 0$ and $A[i] > key$	c_5	$\sum_{j=2}^{n} t_j$
6 $A[i+1] = A[i]$	c_6	$\sum_{j=2}^{n} (t_j - 1)$
7 $i = i-1$	c_7	$\sum_{j=2}^{n} (t_j - 1)$
8 $A[i+1] = key$	c_8	$n-1$

Data Structures and C++

Complexity Analysis

Break: Please take a moment to fill the course evaluations!

PROBLEM SOLVING II

Student-FO

https://go.blueja.io/2Py4AgwuLkGbGaMq5b8sKg

To access the evaluation, scan this QR code with your mobile phone.

Thank you and all the best !