Friday, 09 March 2018

LAB DEMO 05

PS2 Debrief (1)

- Basically, we assume that the “unknown data structure” is a `stack`, a `queue`, and a `priority_queue`, then keep simulating the insertion to and the removal from that “unknown data structure” until proven false for a certain type of data structure, a.k.a. “innocent until proven guilty”
- Stack only for A
- Stack and Queue for B
- Stack, Queue, and Priority Queue for C
  - Just whack with <stack>, <queue>, <priority_queue>, easy PS2 ☺️

PS2 Debrief (2)

- For Subtask D, you are asked to implement own DS :O
  - You can literally follow D03 example about Stack for this part
  - Then extend InsertAfterTail to deal with the Queue part…
- But but… we know that there won’t be more than 100 000 items in that unknown data structure
- So we do NOT have to force our self to use Linked List solution to implement a Stack or a Queue
  - We can use array/vector (of size 100 000) with Insertion/Removal at the back to emulate a Stack
  - We can use array/vector (of size 100 000) with two pointers front and back to emulate a Queue (we will not cause index out of bound :O)
    - We do NOT even need to deal with circular array for queue :O
  - For PQ… see BinaryHeapDemo

PS3 Status (as of today)

Name	A	B	C	D (ignored)
?	AC	AC	AC	Ignored
?	AC	AC		
?	AC			
The rest of you…				

~8 more days to complete PS3

Hacking Solution for PS3 Subtask I

UpdateDilation() and **GiveBirth()** can make things difficult

- But in Subtask I, 1 ≤ **n** ≤ 15
- You can just use an array of size 15 and keep **re-sorting** the positions of up to 15 women for every **ArriveAtHospital()**, **UpdateDilation()**, and **GiveBirth()** operations that can change the ordering
- This way, if done correctly, can give you “free” 1 point
- This is not a “proper PQ” solution though and only uses sorting knowledge that we have learned in the first half of CS2040C :O…
- But this is a solution that you should write if you have nothing else for the harder subtasks, e.g. during individual tests… 🙁
  - At least non zero

Easiest Solution for PS3 Subtask II

It is a classic PQ example! Read the wording carefully!

Easiest solution: Just use C++ STL priority_queue!

- Implement a “woman object”
  - Important note: Real life woman is NOT an object!
    - PS: Some senior students name this variable to “mommy” ☺️
- Or, we can just use pair or tuple from earlier
  - pair<int, int> woman, first field is dilation, second field is arrival index
    - We can negate the second field :O
- DONE, ArriveAtHospital==push, GiveBirth==pop, Query==top

PS: Other solutions exist, like the one in Tut07 later!

Why PS3 Subtask III is Harder?

Why it is not easy for C++ priority_queue to handle **UpdateDilation()** operation efficiently, i.e. **faster than O(n)**?

- This requires ability to modify a key inside the Priority Queue (likely Binary Heap) where this key can be anywhere in the Binary Heap (not necessarily in the root – the easiest place)
- This operation is sometimes called as **heapUpdateKey(i, v)**
- To do this efficiently, we need something that is hidden in VisuAlgo

Note, the **GiveBirth()** operation is also more complex now

- It may involve deletion of a key that is not necessarily the current maximum of the Binary Heap :O

heapUpdateKey(i, v)

To update the value of a key **i** to a new value **v** (where **i** is not necessarily the root---index **1**), we need:

1. A way to fix (Max) Heap property as changing the previous value to a new value **v** may cause violation of (Max) Heap property

   Hint: Anything to consider?

2. A way to quickly identify this index **i**

   Hint: Something that you learn yesterday? (Thu of Week 07),
   see next slide

C++ STL unordered_map **SHORT** demo

unordered_set is similar

- constructor
- insert, operator [], find, count
- range-based for loop to access the keys (in unordered fashion)
- erase
- empty, size
- http://en.cppreference.com/w/cpp/container/unordered_map

GiveBirth(i)

To delete key **i** (where **i** is not necessarily the root---index **1**),
we just need:

```
heapUpdateKey(i, INF) // i will be at the root now
ExtractMax() // then ☺️
```

Of course you still need a fast way to map a woman name to her index **i** quickly, the same thing that we discussed earlier

PS: Other ways exists

PS3…

All the best in clearing PS3, if you have not done so

- Subtask IV requires you to avoid using STL :O…
  - AC-ing it shows Steven that you really understand Binary Heap and Hash Table concepts (or some other concept :O)…
  - 0 point, totally optional this time

Remember: If you keep delaying **your first attempt**
for PS3, you may run out of time even though
you have ~8 days working time for PS3

VisuAlgo Training Mode

PS3 is clearly about Binary Heap + Hash Table (or?)

Make sure that you understand the explanation in:
https://visualgo.net/en/heap?slide=1 (until the last slide)

You can use VisuAlgo Online Quiz training mode to check your basic understanding about Binary Heap on “infinite” number of random questions:
https://visualgo.net/training?diff=Hard&n=5&tl=5&module=heap
(direct URL problematic now, will debug later; just use main page)

Mock PE 1

Solve https://open.kattis.com/problems/cd

Before this Lab session runs out (xx.45)!!

Start from this template code (share your repl link) that already have

Gradual hints will be added in few minutes interval