2018 Sol

Stable Means it keeps equal elements in the same order they appears before Sorting

Stable	not Stable
- insertion	- Selection
- bubble	- quick
- Count	- Heap
- merge	

Question (1) Circle True or False [10 points] one for each.

1. Merge sort is not stable sort. T----------F
2. Minimum time for Comparison sorting algorithm nlogn. T----------F
3. Quick sort (n^2) is not better than Merge sort ($n\log n$) in worst case. T----------F
4. Heap sort ($n\log n$) is better than Quick sort (n^2) in worst case. T----------F
5. Count sort is not in place algorithm. T----------F
   ↳ means sort array without using extra memory
6. Insertion sort required extra memory equal to array size. T----------F
7. Selection sort is not in place algorithm T----------F
8. Stack is LILO → lifo ≠ fifo T----------F
9. Queue is FILO → fifo = lifo T----------F
10. In Max Heap root is minimum number. T----------F
    Max

Check proof in it's file

Algo	best	Avg	worst	in Place	Stable
Selection	$O(n^2)$	$O(n^2)$	$O(n^2)$	✓	✗
insertion	$O(n)$	$O(n^2)$	$O(n^2)$	✓	✓
bubble	$O(n)$	$O(n^2)$	$O(n^2)$	✓	✓
Merge	$O(n\log n)$	$O(n\log n)$	$O(n\log n)$	✗	✓
quick	$O(n\log n)$	$O(n\log n)$	$O(n^2)$	✓	✗
Counting	$O(n+k)$	$O(n+k)$	$O(n+k)$	✗	✓
Heap	$O(n\log n)$	$O(n\log n)$	$O(n\log n)$	✓	✗

Question (2)- Convert the following infix expression 4+5*6/3-2*7 to postfix expression using stack after that evaluate the postfix expression. Show the stack values after each step[10 points] 5 for each one.

Token	Action	Stack	Postfix	Explanation
4	append to postfix	[ ]	4	
+	stack empty. push '+'	[+]	4	4 pushed
5	append to postfix	[+]	4 5	
*	top '+' has lower presendance. push '*'	[+ , *]	4 5	* has higher precendance than +.
6	append to postfix	[+ , *]	4 5 6	
/	top '*' has equal precendance. pop '*' to postfix, push '/'.	[+ , /]	4 5 6 *	equal precedance -> pop then push
3	append to postfix	[+ , /]	4 5 6 * 3	
-	top '/' has higher presendance. pop '/' to postfix . top '+' has equal presendance. pop '+' to postfix . push '*' .	[-]	4 5 6 * 3 / +	pop all operators with higher or equal precendance than '-' , then push '-'
2	append to postfix	[-]	4 5 6 * 3 / + 2	
*	top '-' has lower precendance . push '*' .	[- , *]	4 5 6 * 3 / + 2	* has higher precendance than - .
7	append to postfix	[- , *]	4 5 6 * 3 / + 2 7	
end	pop all from stack to postfix.	[ ]	4 5 6 * 3 / + 2 7 * -	pop remaining operators in stack

Final Postfix Expression: 4 5 6 * 3 / + 2 7 * -

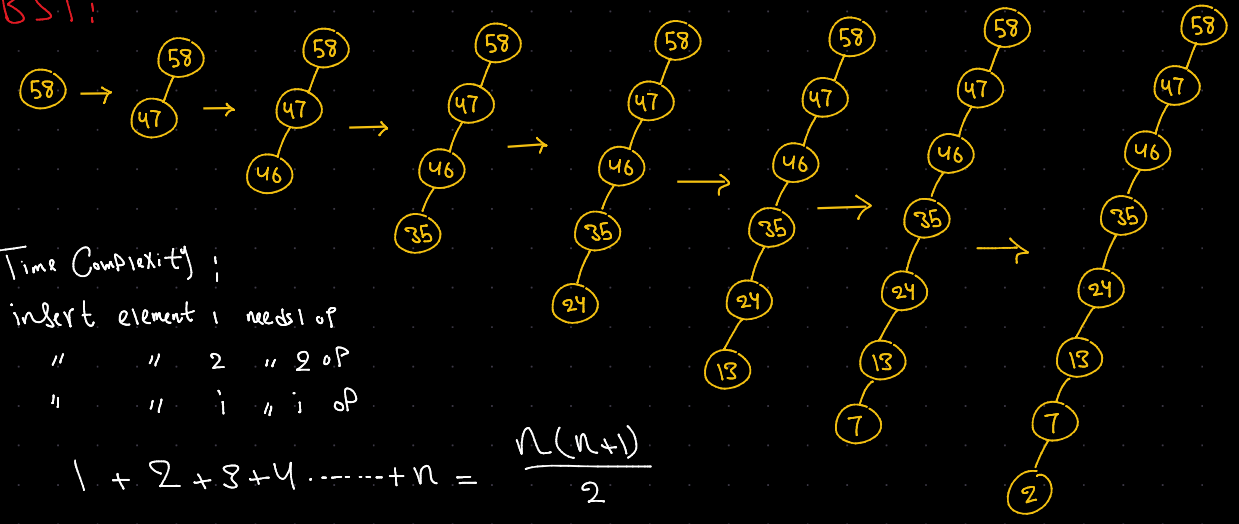
evaluating Postfix

Token	ACtion	Stack	
4	push 4.	[4]	
5	push 5.	[4 , 5]	
6	push 6.	[4 , 5 , 6]	
*	pop 6 and 5 -> 5*6 = 30, push 30 .	[4 , 30]	(5 * 6 = 30)
3	push 3.	[4 , 30 , 3]	
/	pop 3 and 30 -> 30/3 = 10, push 10 .	[4 , 10]	(30 / 3 = 10)
+	pop 10 and 4 -> 4+10 = 14, push 14 .	[14]	(4 + 10 = 14)
2	push 2	[14 , 2]	
7	push 7	[14 , 2 , 7]	
*	pop 7 and 2 -> 2*7 = 14, push 14 .	[14 , 14]	(2 * 7 = 14)
-	pop 14 and 14 -> 14-14 = 0, push 0 .	[0]	(14 - 14 = 0)

Final Res = 0

Question (3) insert the following number in a BST tree (58,47,46,35,24,13,7,2) , show each step and show the exact time complexity for inserting all nodes. after that add the values in AVL tree and show its average time complexity to insert all nodes [10 points]. 5 for each.

BST:



Time Complexity :

insert element 1 needs 1 op

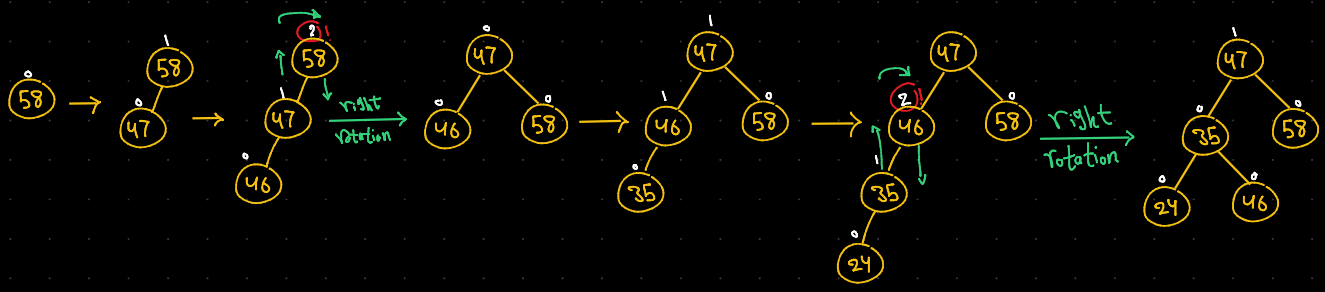
" " 2 " 2 op

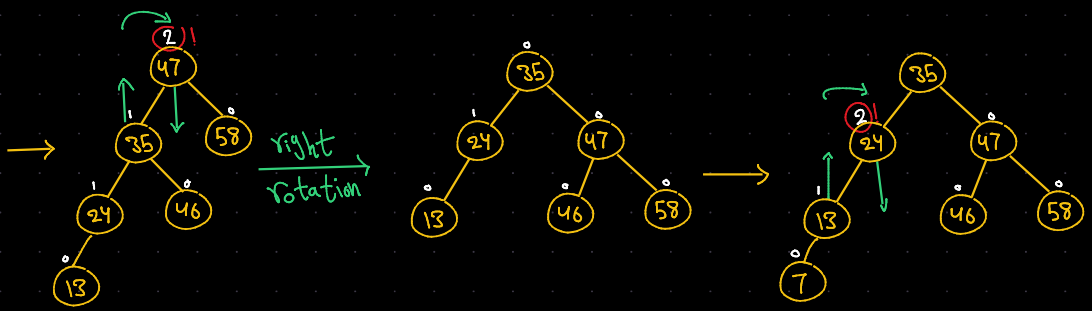
" " i " i op

$$1 + 2 + 3 + 4 \cdots\cdots + n = \frac{n(n+1)}{2}$$

$$\simeq O(n^2)$$

AVL:





Avg time is $O(n \log n)$

→ insert n nodes each insert is $O(\log n)$

So $O(n \log n)$

Question (4) Prove that BUILD-MAX-HEAP function order of O(N) by mathematical steps and drawing [10 points]

logn
$\sum_{H=0} n*h/2^h *h = n/2*(2) = 2$

*let n_h be The number of nodes at height h.

$$n_h = \left\lceil \frac{n}{2^{h+1}} \right\rceil$$

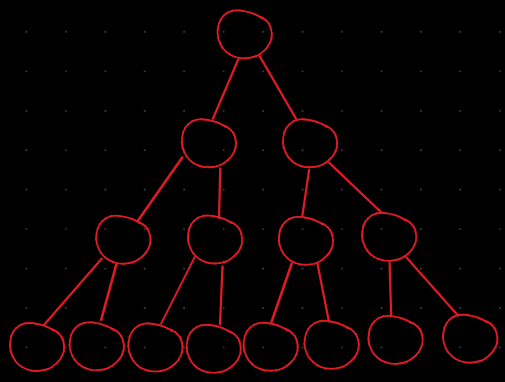
- Total Cost T(n) is:

$$T(n) = \sum_{h=0}^{H=\log n} n_h \cdot h$$

$$= \sum_{h=0}^{H=\log n} \frac{n}{2^{h+1}} \cdot h$$

$$= \frac{n}{2} \sum_{h=0}^{\log n} \frac{h}{2^h}$$ We know = 2

So $T(n) = \frac{n}{2} \cdot 2 = n \rightarrow O(n)$



h	n_h	Cost Per node	total Cost at h
0	$\leq \frac{n}{2^1}$	O(0)	O(0)
1	$\leq \frac{n}{2^2}$	O(1)	$O(\frac{n}{4})$
2	$\leq \frac{n}{2^3}$	O(2)	$O(\frac{n}{4})$
3	$\leq \frac{n}{2^4}$	O(3)	$O(\frac{3n}{16})$

Question (5) write a pseudocode for HEAP-EXTRACT-MAX, HEAP-INCREASE-KEY, MAX-HEAP-INSERT [10 points] 3-4-3 points

```
HEAP-EXTRACT-MAX(A)
  if A.heap-size < 1
     error "heap underflow"
  max = A[1]
  A[1] = A[A.heap-size]
  A.heap-size = A.heap-size - 1
  MAX-HEAPIFY(A, 1)
  return max
```

```
HEAP-INCREASE-KEY(A, i, key)
  if key < A[i]
     error "new key is smaller than current key"
  A[i] = key
  while i > 1 and A[PARENT(i)] < A[i]
     exchange A[i] with A[PARENT(i)]
     i = PARENT(i)
```

```
MAX-HEAP-INSERT(A, key)
  A.heap-size = A.heap-size + 1
  A[A.heap-size] = -∞
  HEAP-INCREASE-KEY(A, A.heap-size, key)
```

Question (6) [10 points]

- **what is the avareg time complexity of the follwing algorithms? [5 points] 0.5 for each one. Assume A=1, B=lgn , C=n , D=nlgn E=n^2 put the correct letter beside each one of the following algorithms.**

1. Heap-maximum ..... O(1) .....
2. Push and pop in periority queue as heap .... O(lgn) .....
3. Push in periority queue as linked list ..... O(n) .....
4. Sequentail search ..... O(n) ...
5. Count sort ..... O(n) ....
6. radix sort ..... O(n) ...
7. Merge sort ..... O(nlgn)
8. HEAP SORT .. O(nlgn)
9. shell sort ..... $O(n^2)$ ....
10. Selection sort ..... $O(n^2)$ ....

- **[5 poins] one for each If the following array is a heap**

index	1	2	3	4	5	6	7	8	9	10
value	20	19	18	15	14	13	12	8	4	2

A. What is the root of heap ----- ~~20~~

B. Left and right childs of 14 ~~left~~ : 2 , no right

C. Parent of 4 --~~15~~--

D. List the leaves nodes from left to right ~~13, 12, 8, 4, 2~~

E. List non termenal nodes from root 20, ~~19, 18~~, 15, 14