FIT3155: Week 6 Tutorial - Answer Sheet

(Scribe: Dinithi Sumanaweera)

Question 1

Using mathematical induction, for a binomial tree B_k of order k, prove that:

(a) B_k contains 2^k nodes.

(b) B_k has a height k.

(c) B_k has exactly k-choose-d nodes at each depth $0 \leq d \leq k$.

(1). $\underline{B_k \text{ contains } 2^k \text{ nodes.}}$

Base case : $k=0$

$B_0 \rightarrow o$

$2^k = 2^0 = 1$ node.

Assuming : B_{k-1} contains 2^{k-1} nodes

$B_k = B_{k-1} + B_{k-1}$

$\therefore$ B_k should have $(2^{k-1} + 2^{k-1})$ nodes.

$= 2 \times 2^{k-1} = \underline{\underline{2^k}}$ nodes.

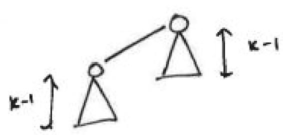
(2). $\underline{B_k \text{ has a height } k.}$

Base Case : $k=0$

$B_0 \rightarrow o$ has height 0

Assuming : B_{k-1} has a height $(k-1)$

$B_k = B_{k-1} + B_{k-1}$



height becomes $(k-1)+1 = \underline{\underline{k}}$

(3). B_k has exactly ${}^kC_d$ nodes at dept $0 \le d$

Base case: $k=0$, $d=0$

B_0 has ${}^0C_0 = 1$ nodes at depth 0

$\underline{k=1,\ d=0}$

B_1 has ${}^1C_0 = 1$

${}^1C_1 = 1$

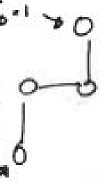
$\underline{k=2}$

B_2 has

${}^2C_0 = 1$

${}^2C_1$ nodes.

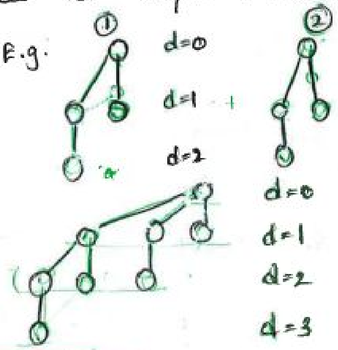
${}^2C_2$



Assuming, B_{k-1} has exactly ${}^{k-1}C_d$ nodes at depth $0 \le d \le k$

$$B_k = B_{k-1} + B_{k-1}$$

E.g.



$\# \text{ nodes}(k-1, d) = {}^{k-1}C_d$

$\# \text{ nodes}(k-1, d-1) = {}^{k-1}C_{d-1}$

$$\boxed{n_d^{new} = n_d^1 + n_{d-1}^2}$$

$$n_d^{new} = {}^{k-1}C_d + {}^{k-1}C_{d-1}$$

$$= \frac{(k-1)!}{(k-1-d)!\,d!} + \frac{(k-1)!}{(k-1-d+1)!\,(d-1)!}$$

$$= \frac{(k-1)!}{(k-1-d)!\,d!} + \frac{(k-1)!}{(k-d)!\,(d-1)!} = \frac{(k-1)!}{(k-d-1)!\,(d-1)!}\left[\frac{1}{d} + \frac{1}{(k-d)}\right]$$

$$= \frac{(k-1)!\ (k-\cancel{d}+\cancel{d})}{(k-d-1)!\,(d-1)!\ (k-d)\cdot d} = \frac{k!}{(k-d)!\,d!}$$

$$= \underline{\underline{{}^kC_d}}.$$

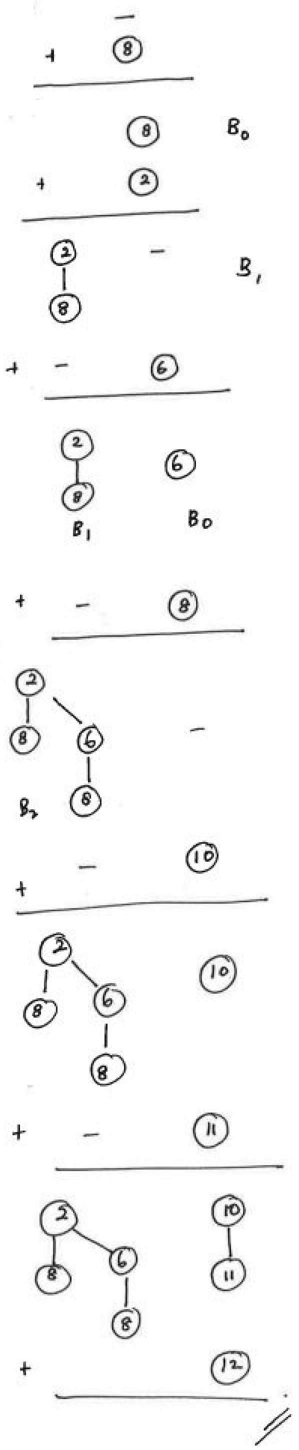
Question 2

Insert the following elements in a binomial heap:
8, 2, 6, 8, 10, 11, 12

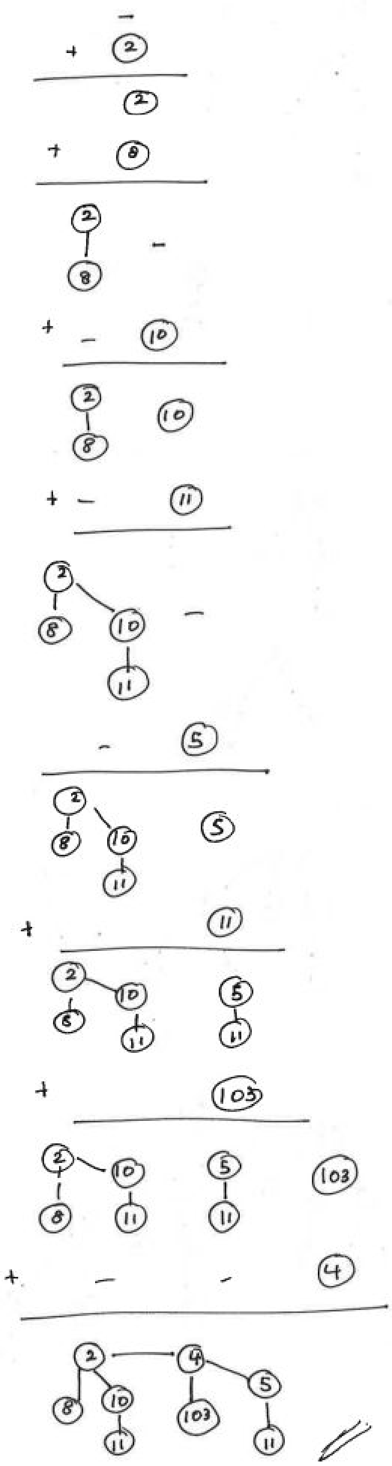
Question 3

2, 8, 10, 11, 5, 11, 103, 4

Q(2)

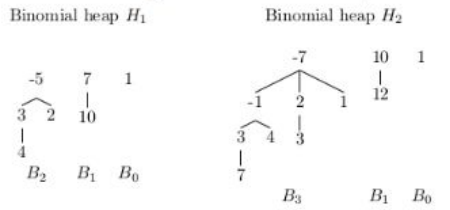


Q(3)

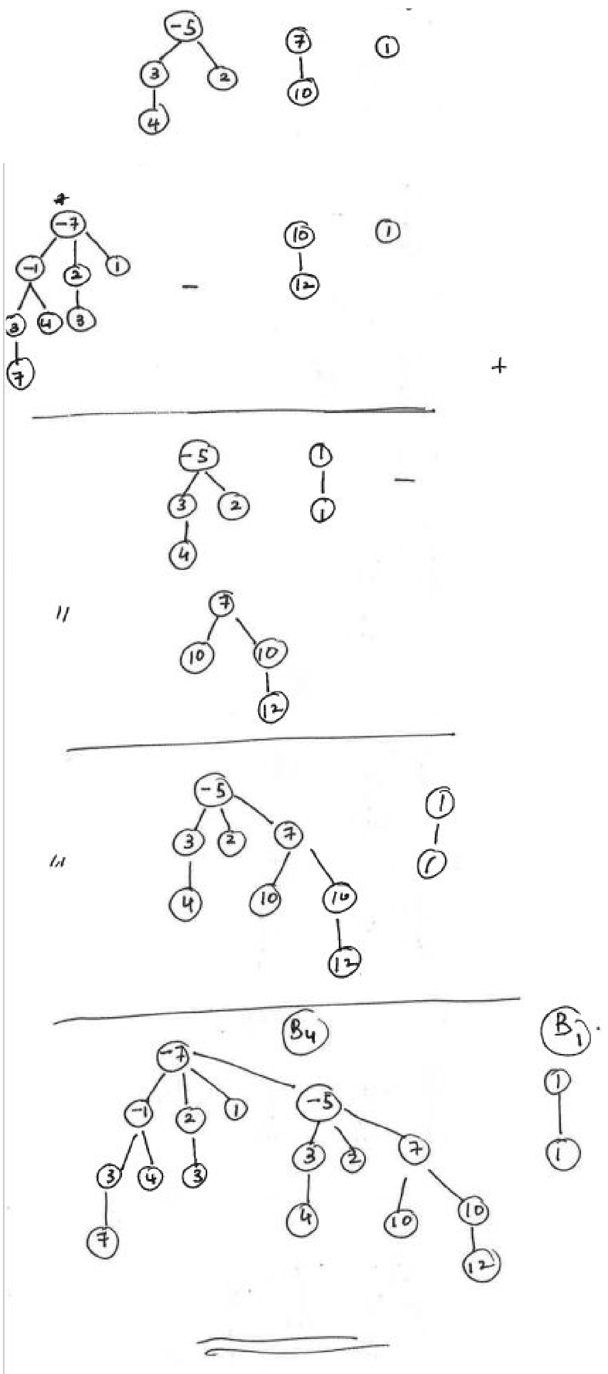


Question 4

Perform `merge` on the following two binomial heaps:

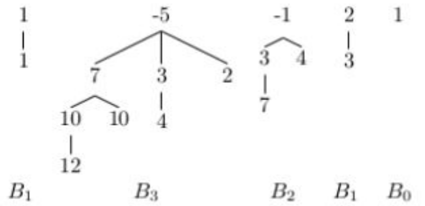


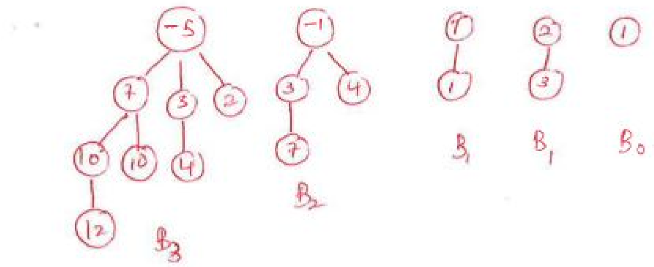
B3 **B2** **B1** **B0**

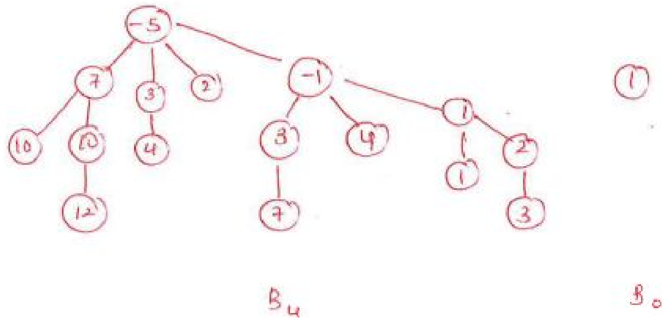


Question 5

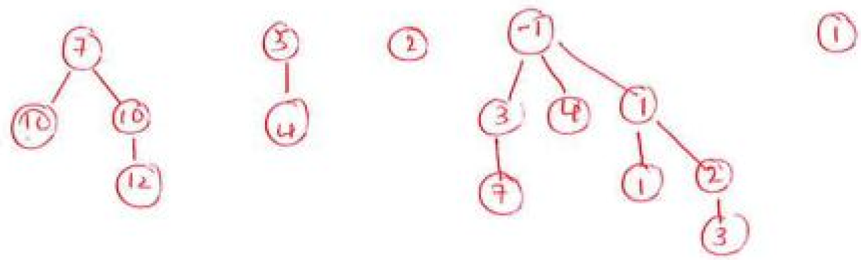
Perform `merge` followed by `extract-min` on the following (improper state of) binomial heap:

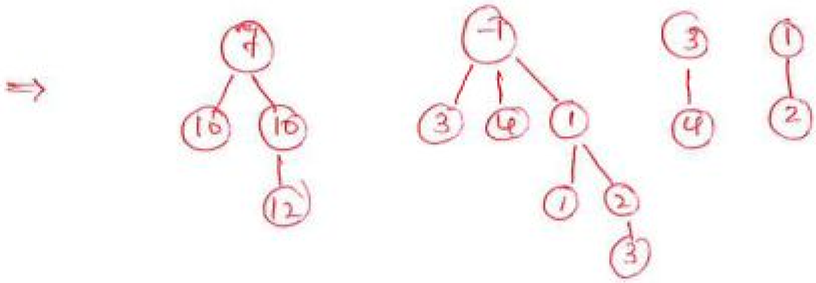


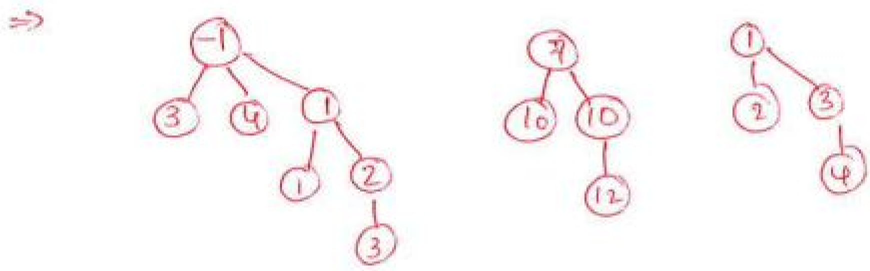


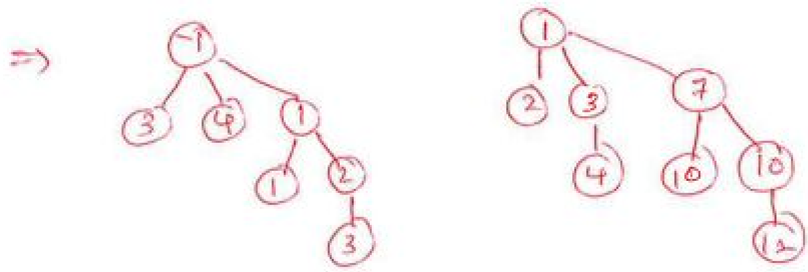


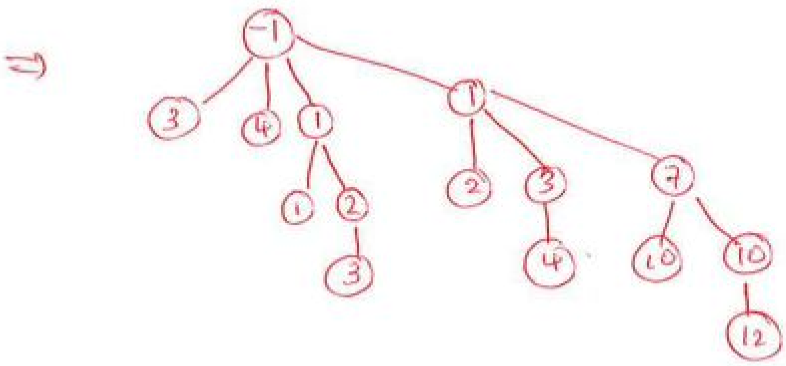
After extracting min, which is -5,











Question 6

Show that the amortized complexity to insert n elements into a binomial heap is O(n)

```
Consider the binary representation of a binomial heap.
Recall that 1 in index i of the binary representation indicates
that the binomial heap has order i binomial tree (B_i).
Note: indexing starts from least significant bit (rightmost bit) to the most
significant bit (leftmost)

Whenever we insert an element to the binomial heap, it is
equivalent to merging with another binomial heap which has only
order 0 binomial tree (B_0_new). Therefore the merge always start
from merging B_0_new with B_0 in the binomial heap if such B_0
exists.
```

If the current merge resulted in a binomial tree of order that already exists in the heap, the merge has to continue until all such future clashes get resolved.

Accordingly, the possible cases are as follow.

Binary representation	# of merges	# of insertions	Total operations
(1)................ 0	0	1	1
(2)................ 01	1	1	2
(3)............. 011	2	1	3
(4)........... 0111	3	1	4
(5)........ 01111	4	1	5

etc.

Now, if we have N elements to insert, the binary representation will have log2(N) positions at most. Here we need to get the total number of operations involved in inserting N elements. For that, we need to first get the number of instances each of the above case occur.

Recall that if a binary representation ends in 0, it represents an even number, else an odd number. Whenever we divide a number by 2, its binary representation shifts to the right by 1 bit. On the other hand, if we have N elements, there are N/2 even numbers and N/2 odd numbers <= N.

This means, we can conclude that there are N/2 number of case (1) binary representations that end in a 0.
If so, there N/2 binary representations ending in 1. But for case (2) we need to get the number of binary representations ending in 01. If we consider N/2 odd numbers, N/4 of them will be even and N/4 of them will be odd, indicating that there are N/4 cases ending in 01 (case (2)). Accordingly, we can have the number of times each binary representation case may occur (i.e. case (1) - N/2, case (2) - N/4, case (3) - N/8 and so on).

Therefore, the total number of operations over N insertions
= [(N/2)*1] + [(N/4)*2] + [(N/8)*3] + … <= 2N ← O(N).

Thus, a single insertion to a binomial heap is O(1) amortized.