EECS 560
93
67
Spring 2018
Exam 1

Scores

- Once you start the exam, you will not be excused from the room for any reason unless you turn in your exam. Once it is turned in, you cannot come back and continue working on it.
- Answer all questions.
- This exam is closed book and closed notes.
- No calculators, cell phones, head phones, or electronic devices of any sort will be allowed. No such devices should be out in the open.
- Sit as far apart in the room as possible. Preferably there should be at least one seat between everyone.
- You must write legibly and show all your work clearly for credit. Partial credit will only be given to meaningful answers. You will be graded according to your approach to the problems, mathematical rigor, and quality of your solutions.
- *Suggestion*: Quickly scan the entire test, do the problems you know best first. Then spend your time on the remaining problems.

1	6 /14
2	14 /14
3	8 /8
4	6 /6
5	10/10
6	3 /10
7	7 /13
8	10 /10
9	14/15
TOTAL	78 **/100**

Name: Jian Shen

KUID: 2861432

3/1/2018

6

1. If a set of 3,668 = (2^{12} – 428) records is being stored using a binary search tree (BST) T with 3,668 nodes (one record per node), answer the following questions with *integer solution* if possible.

1 (a) (2) What is the minimum height of T? What kind of tree structure of T will have a minimum height?

11 $\lfloor \log_2 3668 \rfloor = \boxed{12}$ ×

complete binary tree will have minimum height

2 (b) (2) What is the maximum height of T? What kind of tree structure of T will have a maximum height?

$3668 - 1 = \boxed{3667}$ ✓

skew tree ✓

2 (c) (2) What is the minimum number of leaf node(s) in T? What kind of tree structure of T will have a minimum number of leaf nodes?

$\boxed{1}$ ✓

skew tree ✓

0 (d) (5) What is the maximum number of leaf node(s) in T? What kind of tree structure of T will have a maximum number of leaf nodes?

$\lceil n/2 \rceil$ $\lceil \log_2 n \rceil = 13$ × $2048 - 427 + 213$

$= 2\overline{)3668}$ 1834 $= 1834$

complete binary tree

external binary tree ×

1 (e) (3) Recall that a BST should not be implemented using an array. But if T is to be implemented using an array A[0..m-1], compute the minimum size m of the array A in order to guarantee that any given BST with 3,668 nodes can be stored using the array A. What kind of tree structure of T will require a maximum number of array locations to store it?

minimum size of m is $\boxed{3668}$ ×

$1 + 2^1 + \dots 2^{12} =$ $2^{3668} - 1$

skew tree ✓ will require maximum # of array locations

(14)

2. Answer the following questions.

4 (a) (4) What are the two most important characteristics of a "good" hash function h(x)?

① complexity of insertion is constant. $T_w(n) = O(1)$ ✓

② values can be distributed evenly in the table. ✓

2 (b) (2) Given a hash table H with m buckets and n objects stored in H. Define the load factor λ of H.

$\lambda = \frac{n}{m}$ ✓

4 (c) (4) Recall that in monitoring the performance of a hash table, whenever λ approaches a constant value k, rehashing should be performed.

(i) What is k in an open hash table?

1 ✓

(ii) What is k in a closed hash table?

$\frac{1}{2}$ ✓

4 (d) (4) If closed hashing with quadratic probing is used, how should we choose m and λ in order to guarantee that insertion is always possible?

choose a primer number for m ✓

and $\lambda \leq \frac{1}{2}$ ✓

(8)

3. (8) Using the hash function *x mod m* and quadratic probing to construct a (closed) hash table H with m = 11 buckets by inserting a set of 8 records with keys {18, 35, 25, 4, 22, 13, 36, 24}, in the given order, into H. *Remark:* You must show your computations for locations and illustrate the final hash table H clearly for credit.

$(18+0^2)\%11=7$

$(35+0^2)\%11=2$

$(25+0^2)\%11=3$

$(4+0^2)\%11=4$

$(22+0^2)\%11=0$ ✓ +2

$(13+0^2)\%11=2$

$(13+1^2)\%11=3$ ✓

$(13+2^2)\%11=6$

$(36+0^2)\%11=3$

$(36+1^2)\%11=4$

$(36+2^2)\%11=7$ ✓

$(36+3^2)\%11=1$

$(24+0^2)\%11=2$

$(24+1^2)\%11=3$

$(24+2^2)\%11=6$ ✓

$(24+3^2)\%11=0$

$(24+4^2)\%11=7$

$(24+5^2)\%11=5$

H:

0	22
1	36
2	35
3	25
4	4
5	24
6	13
7	18
8	
9	
10	

✓

2 2 2 2
2 2 2
2 2 2

4. (6) When implementing an ADT for a set of records S, $|S| = 2^{10}$, it is determined that a find operation, find(x,S), will requires 0.005 second to execute. If the complexity of the find operation is given by $T(n) = 560n\lg_2 n$, compute the time required to execute this operation when $|S| = 2^{20}$. *Remark:* You must first set up your equation for T(n) and then evaluate it. Simplify your answer for T(n).

$$T(n) = \frac{0.005}{560\times 2^{10}\times \log_2 2^{10}} \times 560\times 2^{20}\lg_2 2^{20}$$

$$= 0.005 \times 2^{10}\times 2$$

$$= 0.01\times 2^{10}$$

$= 10.24$ seconds

5. (10) Using the definition of big-O to prove that $\dfrac{2n^4 - n^3 - 2n^2 + 4}{2n^2 - 2n - 27} = O(n^2)$.

$$\frac{2n^4 - n^3 - 2n^2 + 4}{2n^2 - 2n - 27} \le \frac{2n^4 + 4n^4}{2n^2 - (2n+27)}, \quad n > 0$$

$$\le \frac{6n^4}{2n^2 - (2n+48)}, \quad n > 0$$

$$\le \frac{6n^4}{2n^2 - (n^2)}, \quad 2n+48 < n^2,\ n > 8$$

$$\le 6n^2$$

Choose $k = 6$, $n_0 = 8$, $\dfrac{2n^4 - n^3 - 2n^2 + 4}{2n^2 - 2n - 27} \le 6(n^2)$, when $n > 8$

$\therefore$ we can prove assertion.

6. (10) By concentrating on the dominating step and by assuming that all basic operations require the same constant cost C, compute $T_W(n)$ in closed-form for the following program segment as discussed in class.
Remark: You must first set up the equation for $T_W(n)$ and then evaluate its sums for credits. Do not simplify the final expression.

```
x = 2;
y = 10;
k = 1;
while k ≤ n do
      x = x + x*y +210;
      y = y – x + 560;
      k=k+1;
endwhile;
for i = 1 to n do
   for  j = i to n do
         y = x * y / 2;
         for k = j to n do
               x = x + y – 10;
         endfor;
   endfor;
endfor;
```

$\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$

$i^2 = \frac{}{6}$

$\sum_{j=i}^{n} = \sum_{j=1}^{n} - \sum_{j=1}^{i-1} j$

$T_W(n) = Cn + \sum_{i=1}^{n}\sum_{j=i}^{n}(C + \sum_{k=j}^{n} C)$ ✓

$= \sum_{i=1}^{n}\sum_{j=i}^{n}(C + \sum_{k=j}^{n} C)$ ✓

$= C \cdot \sum_{i=1}^{n}\sum_{j=i}^{n}\sum_{k=j}^{n}$

$= C\sum_{i=1}^{n}\left(\sum_{j=1}^{n} - \sum_{j=1}^{i-1}\right)(n-j+1) \quad \left(\frac{(i+n)(n-i+1)}{2} \right)$

$= C\sum_{i=1}^{n}\left((n-i+1)(n+1) - \frac{(i+n)(n-i+1)}{2}\right)$

$= C$

$= C\sum_{i=1}^{n}\sum_{j=i}^{n}(n-j+1)$

$= C\sum_{i=1}^{n}$

$n-i+2$

$\frac{(n+\cancel{1})(n-i+1)}{2}$

$\sum_{i=1}^{n}\sum_{j=i}^{n}\sum_{k=j}^{n} C$

$= C\sum_{i=1}^{n}\sum_{j=i}^{n}(n-j+1)$

$= \sum_{i=1}^{n}\left(\sum_{j=1}^{n} - \sum_{j=1}^{i-1}\right)(n-j+1)$

$= \sum_{i=1}^{n}(n-i+1)(n+1) - \frac{(1+(n-i+1)) \times (n-i+1)}{2}$

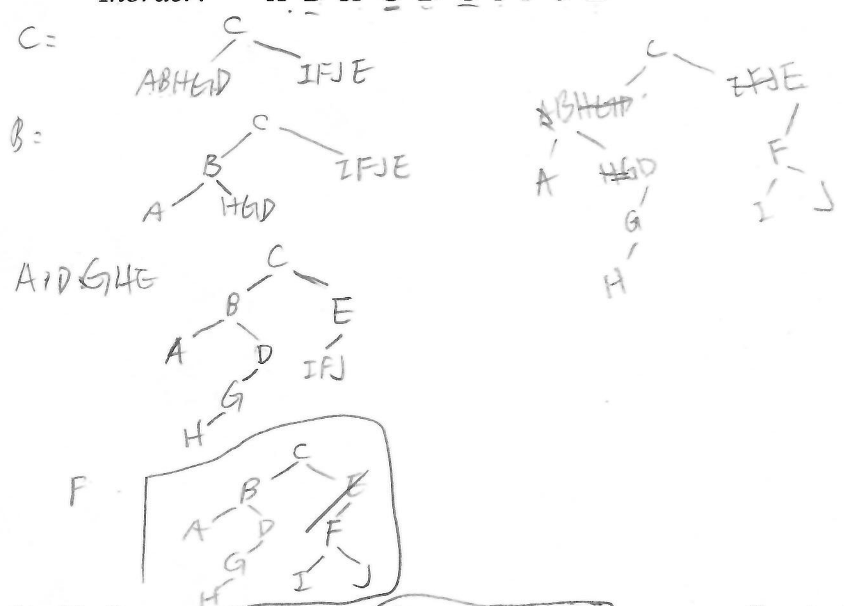
$= \sum_{i=1}^{n} n^2 + (2-i)n + (1-i) - \frac{}{2}$

7. Answer the following questions:

(a) (6) Construct the unique binary tree corresponding to the given pair of tree traversals if possible. If no such a tree is possible, you must justify your answer clearly for credits.

Preorder: C B A D G H E F I J
Inorder: A B H G D C I F J E

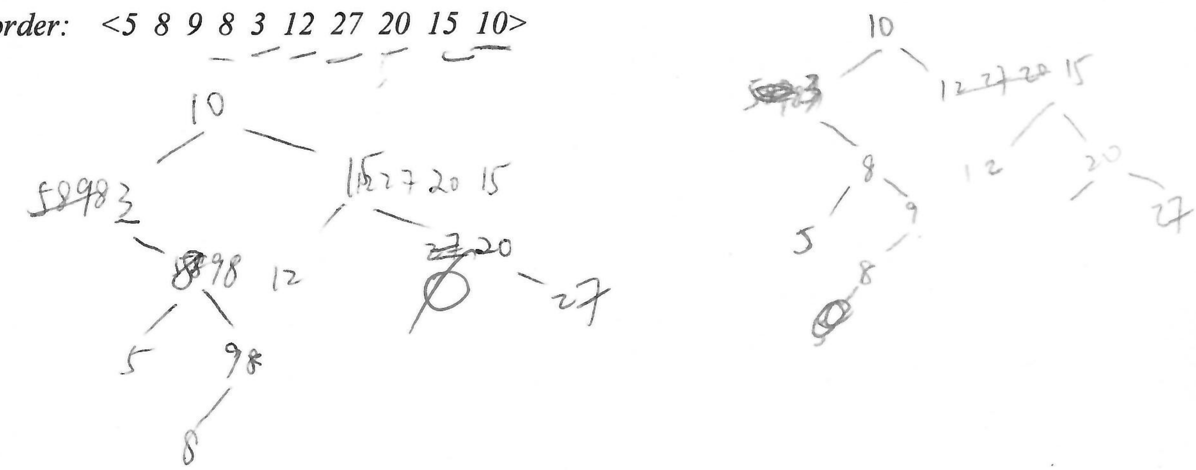
6



(b) (7) Construct the unique binary search tree corresponding to the given postorder tree traversal if possible. If no such a tree is possible, you must justify your answer clearly for credits.

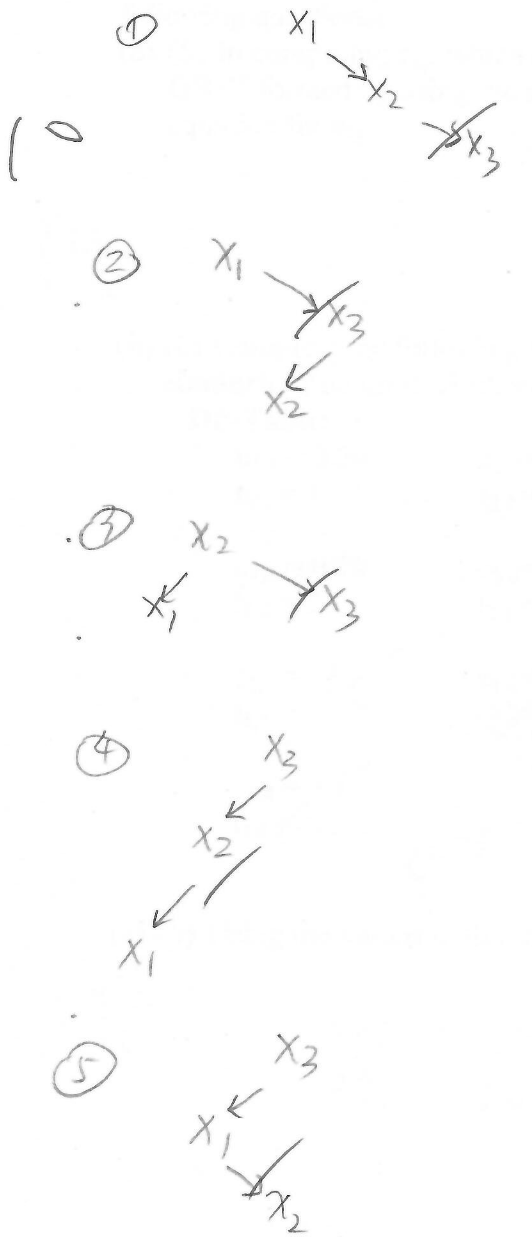
Postorder: <5 8 9 8 3 12 27 20 15 10>

1



No such tree is possible, cause we have to know Inorder to distinguish Left and Right subtree.

8. (10) Recall that for a given set S with n elements, the binary search trees (BST) representation of S is not unique. If you are given a set S of 3 records with keys $\{x_1, x_2, x_3\}$, $x_1 < x_2 < x_3$, construct all possible BST T that can be used to store S. *Remark:* You must illustrate all your trees clearly for credit.



9. Given a set S of 4 records with keys x_i, $1 \le i \le 4$, such that $x_1 < x_2 < \ldots < x_4$, a search key x, and the probability $Pr(x = x_i) = p_i$, $1 \le i \le n$, with $p_1 = 0.20$, $p_2 = 0.30$, $p_3 = 0.35$, $p_4 = 0.15$. If dynamic programming technique is used to construct an optimal binary search tree (OBST) T for S such that the average number of comparisons in finding x in T is minimized, answer the following questions:

(a) (5) In computing $c_{i,j}$, which is the average number of comparisons for finding x in an OBST formed by using the records the set $\{x_i, x_{i+1}, \ldots, x_j\}$, construct the forward equation for $c_{i,j}$.

$$C_{i,j} = \min_{i \le k \le j} \{ C_{i,k-1} + C_{k+1,j} \} + \sum_{l=i}^{j} P_l$$

(4

-1

(b) (6) Complete the following DP-table using the values of $c_{i,j}$ and $t_{i,j}$ as discussed in class.
Remark: You must illustrate the computation of each $c_{i,j}$ and $t_{i,j}$ clearly for credit.

DP-Table:

$c_{1,1} = 0.20$ $t_{1,1} = 1$	$c_{2,2} = 0.30$ $t_{2,2} = 2$	$c_{3,3} = 0.35$ $t_{3,3} = 3$	$c_{4,4} = 0.15$ $t_{4,4} = 4$
$c_{1,2} = 0.70$ $t_{1,2} = 2$	$c_{2,3} = 0.95$ $t_{2,3} = 3$	$c_{3,4} = 0.65$ $t_{3,4} = 3$	
$c_{1,3} = 1.4$ $t_{1,3} = 2$	$c_{2,4} = 1.25$ $t_{2,4} = 3$		
$c_{1,4} = 1.85$ $t_{1,4} = 2$			

$$C_{13} = \min\{ \overset{k=1}{C_{10} + C_{23}}, \overset{k=2}{C_{11} + C_{33}}, C_{12} + C_{43} \} + .2 + .3 + .35 = 1.4$$

$$C_{24} = \min\{ \overset{2}{C_{21} + C_{34}}, \overset{3}{C_{22} + C_{44}}, \overset{4}{C_{23} + C_{54}} \} + .3 + .35 + .15 = 1.25$$

$$C_{14} = \min\{ \overset{1}{C_{10} + C_{24}}, \overset{2}{C_{11} + C_{34}}, \overset{3}{C_{12} + C_{44}}, \overset{4}{C_{13} + C_{54}} \} + .2 + .3 + .35 + .15 = 1.85$$

(c) (4) Using the values in the DP-table, illustrate how you can construct the OBST for S.

$t_{1,4} = 2 \rightarrow$ ① $x_2 (.3)$

$\Downarrow$

② $x_2 (.3)$, $(.2)\ x_1$

$\Downarrow$

$n_{t=3} = 4$, $n_{t=4} = 1$ $\rightarrow$ ③ x_2; x_1, x_3; x_4

99

Jian Shen

2861432

EECS 560

HW#2

1.(a) $\lfloor \log_2 4090 \rfloor = 11$

the min height is $\boxed{11}$

(b) the max height is $\boxed{4089}$

(c) min # leaves is $\boxed{1}$

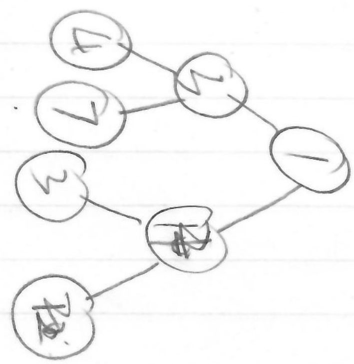
(d) $\lceil 4090/2 \rceil = 2045$

max # leaves is $\boxed{2045}$

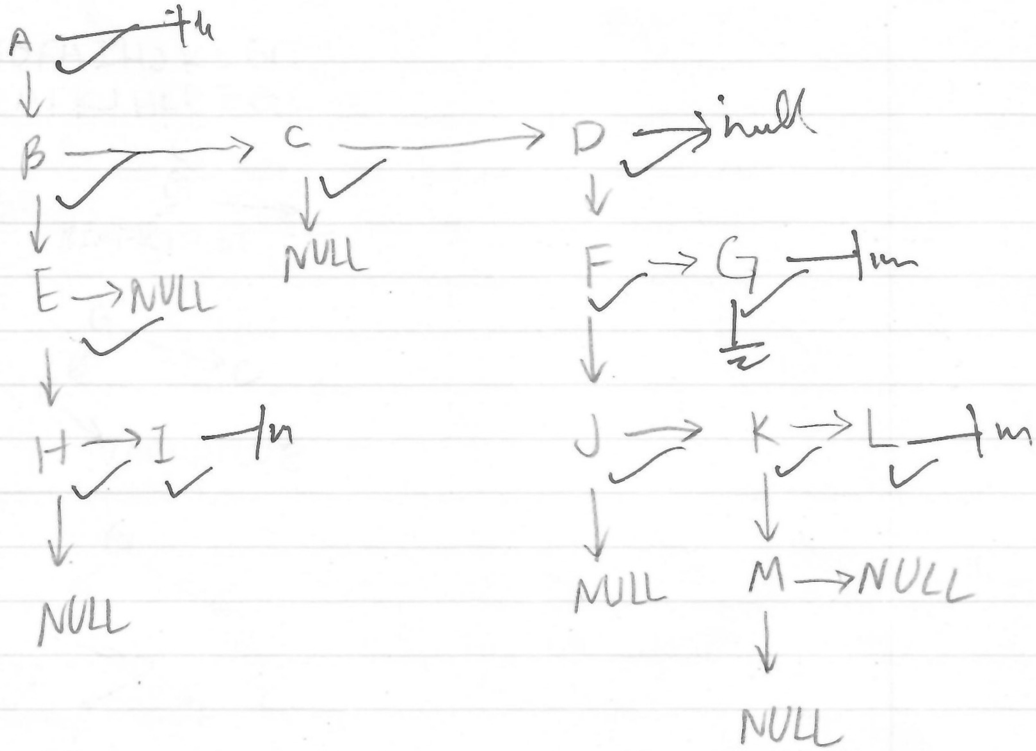
(e) the size of array is $\boxed{2^{h+1}-1}$, depending on height

In this case, with min height, size of array is $2^{12}-1 = \boxed{4095}$

with max height, size of array is $\boxed{2^{4090}-1}$

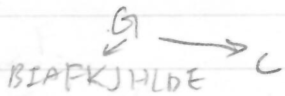


2. T=

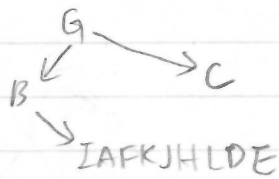


3(a) Pre GBDFAIHJKLEC
In BIAFKJHLDEGC

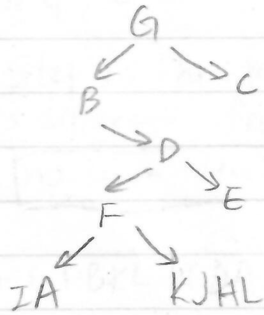
consider G:



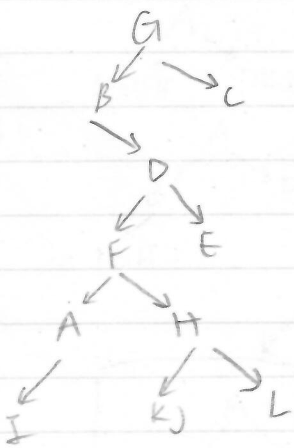
B:



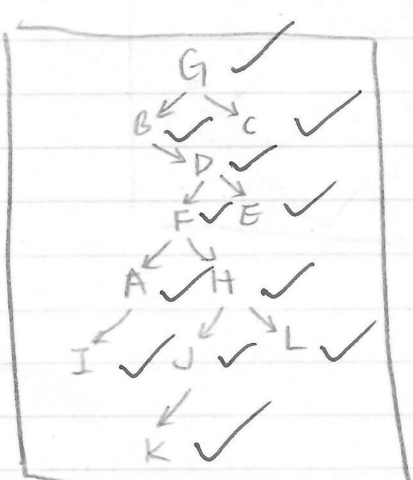
D.F:



AIH:



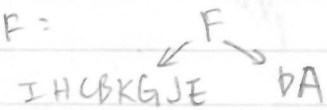
J:



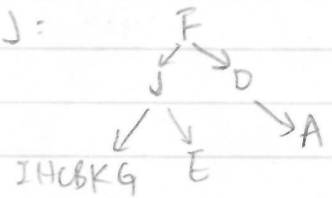
b. Post : HIBCAKGEJDF

In : IHCBKGJEFDA

Step 1 : F :



Step 2 : D.J :



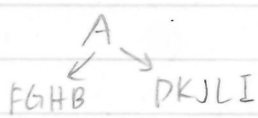
From Step 2, the post order of implementation we got is ...EJADF, which is different from given post order. There is a contradiction

Hence, no such tree exists.

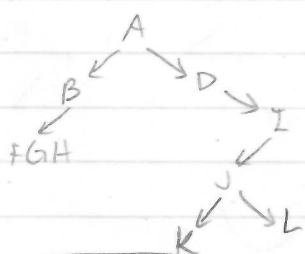
(c) Post : HGFBKLJIDA

In : FGHBADKJLI

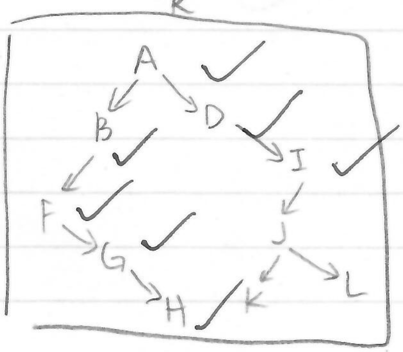
A :



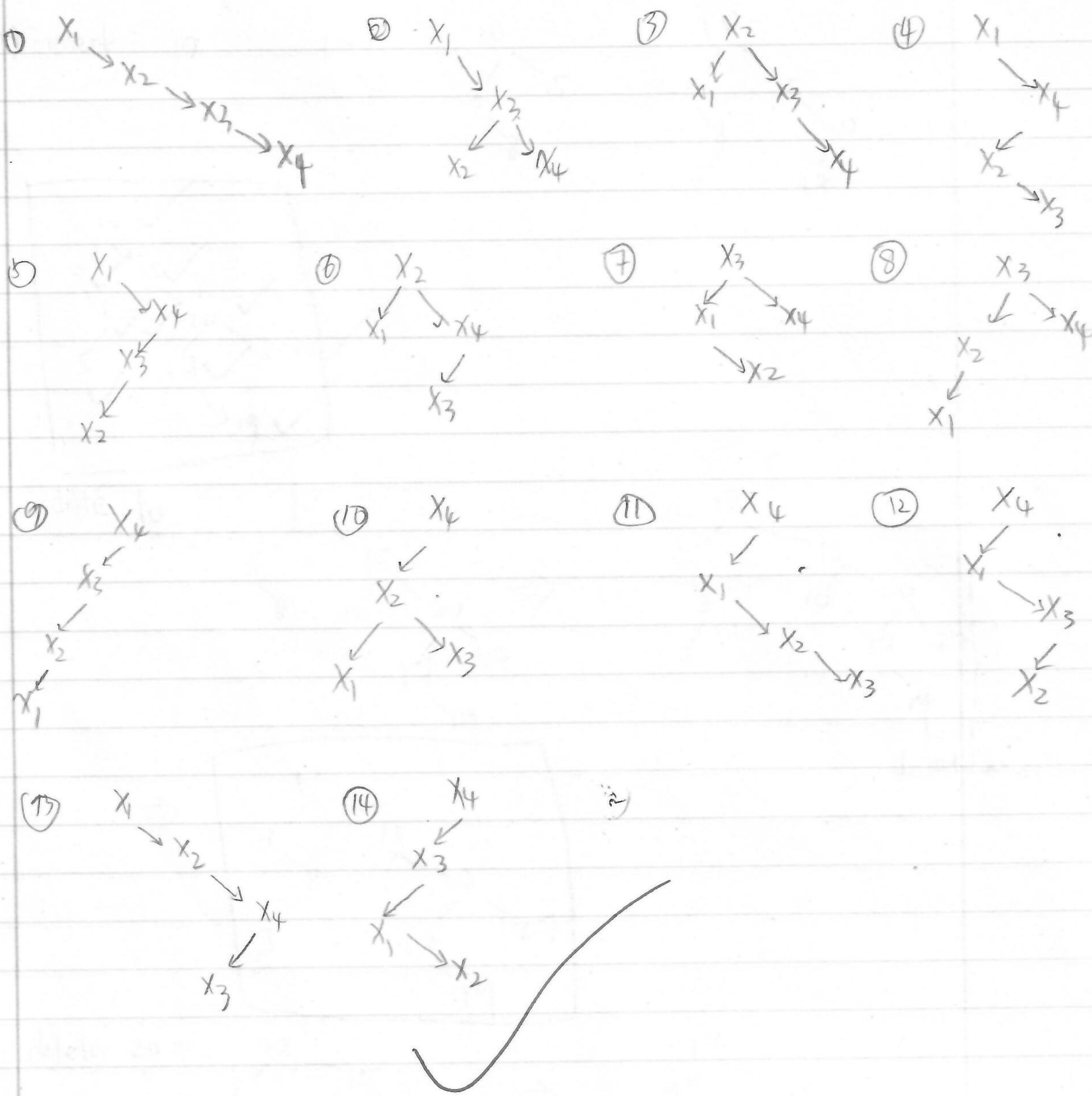
DIJLKB :



F.G :

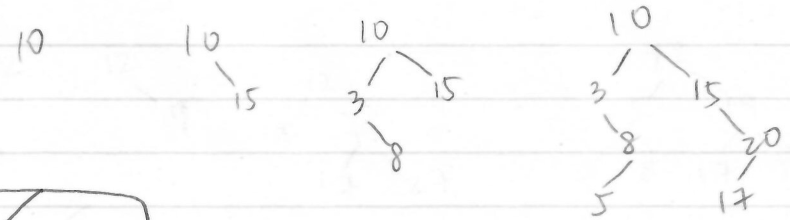


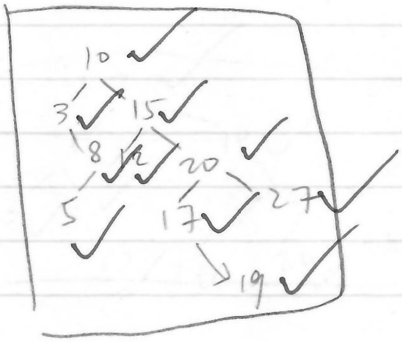
4



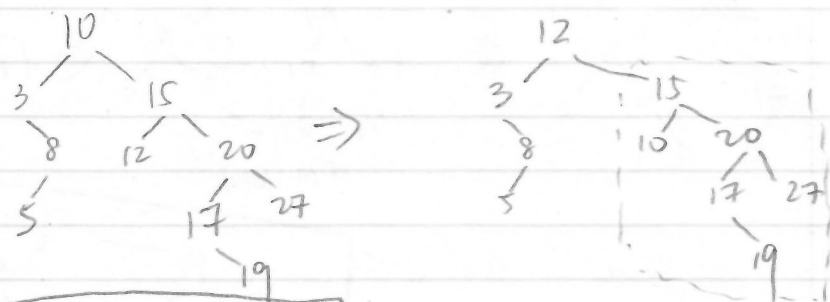
1 $X_1 \to X_2 \to X_3 \to X_4$
2 X_1 X_3 X_2 X_4
3 X_2 X_1 X_3 X_4
4 X_1 X_4 X_2 X_3
5 X_1 X_4 X_3 X_2
6 X_2 X_1 X_4 X_3
7 X_3 X_1 X_4 X_2
8 X_3 X_4 X_2 X_1
9 X_4 X_3 X_2 X_1
10 X_4 X_2 X_1 X_3
11 X_4 X_1 X_2 X_3
12 X_4 X_1 X_3 X_2
13 X_1 X_2 X_4 X_3
14 X_4 X_3 X_1 X_2

5(a) insert =

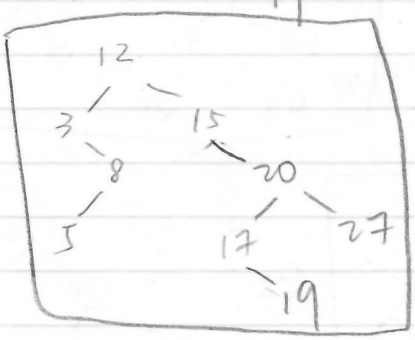




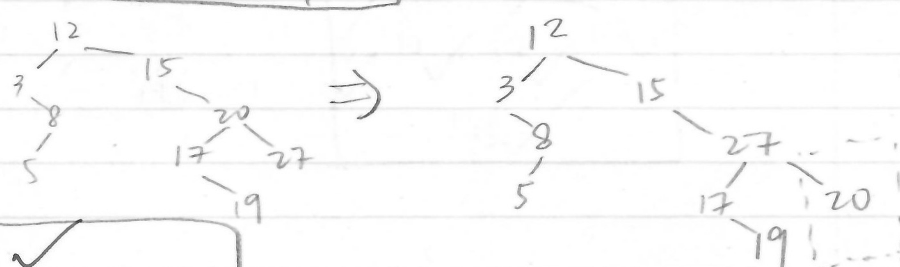
delete 10:



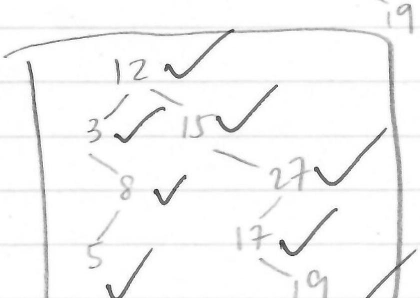
delete min



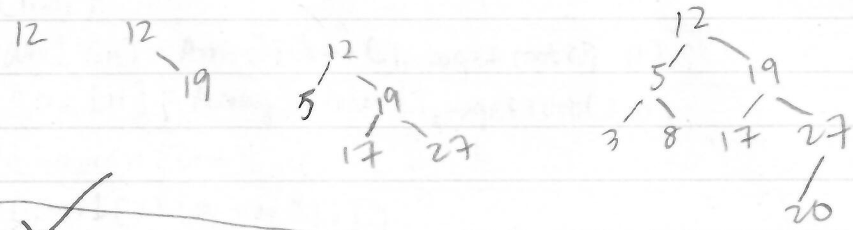
delete 20:

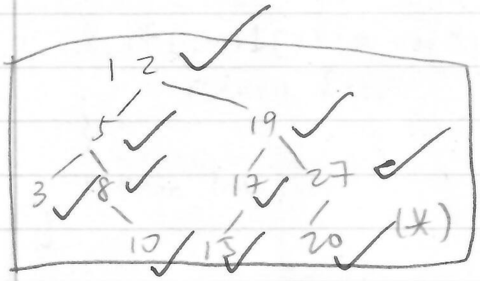


delete min

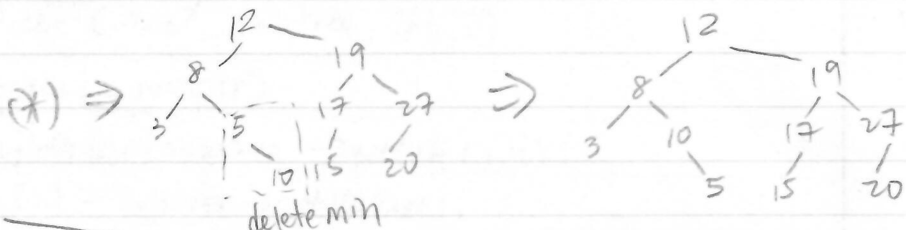


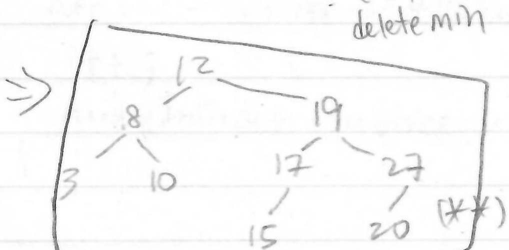
(b) insert:



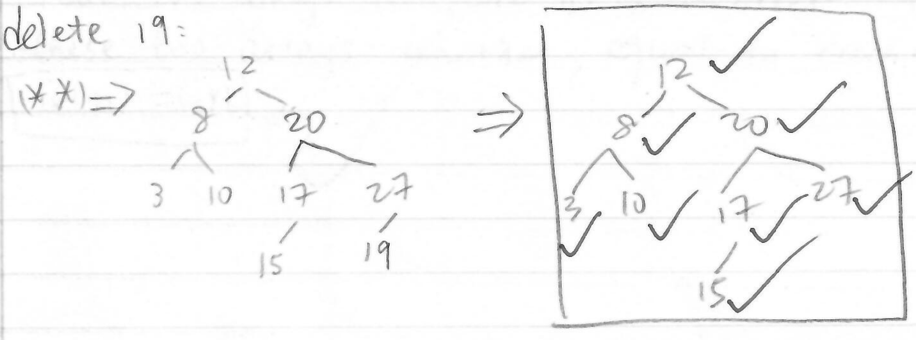


delete 5:





delete 19:



6 Equal(T_1,T_2)

```
{ int arr1[n] = ArrayInOrder(T1->getroot(), 0);
  int arr2[n] = ArrayInOrder(T2->getroot(), 0);
  for(int i=0; i<n; i++)
  { if(arr1[i] != arr2[i])
        return false;
  }
  return true;
}

ArrayInOrder(Node* subtree, int i)
{   if(subtree != nullptr)
    { ArrayInOrder(subtree->getLeft(), i);
      arr[i] = subtree->getValue();
      i++;
      ArrayInOrder(subtree->getRight(), i);
    }
}
```

Create two arrays to store T_1, T_2 in order. then compare these two arrays whether equal to eachother.

$\boxed{T_W(n) = Cn}$;

7 Given $\{x_1, x_2, x_3, x_4\}$ with $p_1 = .35$, $p_2 = .2$, $p_3 = .15$, $p_4 = .3$

$j-i=0$	$C_{11} = .35$	$C_{22} = .2$	$C_{33} = .15$	$C_{44} = .3$
$j-i=1$	$C_{12} = .75$	$C_{23} = .5$	$C_{34} = .6$	
$j-i=2$	$C_{13} = 1.2$	$C_{24} = 1.15$		
$j-i=3$	$C_{14} = 1.95$			

$C_{1,2} = \min\{\overset{k=1}{C_{10} + C_{22}}, \overset{k=2}{C_{11} + C_{32}}\} + \sum_{i=1}^{2} p_i = \min\{.2, .35\} + .55 = .75$

$t_{1,2} = 1$

$C_{23} = \min\{\overset{k=2}{\underline{C_{21}} + C_{33}}, \overset{k=3}{C_{22} + \underline{C_{43}}}\} + \sum_{i=2}^{3} p_i = \min\{.15, .2\} + .35 = .5$

$t_{23} = 2$

$C_{34} = \min\{\overset{k=3}{\underline{C_{32}} + C_{44}}, \overset{k=4}{C_{33} + \underline{C_{54}}}\} + \sum_{i=3}^{4} p_i = \min\{.3, .15\} + .45 = .6$

$t_{34} = 4$

$C_{13} = \min\{\overset{k=1}{\underline{C_{10}} + C_{23}}, C_{11} + C_{33}, C_{12} + \underline{C_{43}}\} + \sum_{i=1}^{3} p_i = \min\{.5, .5, .75\} + .7 = 1.2$

$t_{13} = 1$

$C_{24} = \min\{\underline{C_{21}} + C_{34}, C_{22} + C_{44}, C_{23} + \underline{C_{54}}\} + \sum_{i=2}^{4} p_i = \min\{.6, .5, .5\} + .65 = 1.15$

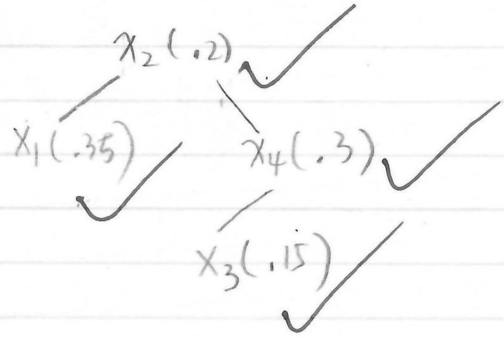
$t_{24} = 3$

$C_{14} = \min\{\underline{C_{10}} + C_{24}, C_{11} + C_{34}, C_{12} + C_{44}, C_{13} + \underline{C_{54}}\} + \sum_{i=1}^{4} p_i = \min\{1.15, .95, 1.05, 1.2\} + 1$

$= 1.95$

$t_{14} = 2$

	1	2	3	4
1	.35	.75 $^{(1)}$	1.2 $^{(1)}$	1.95 $^{(2)}$
2		.2	.5 (2)	1.15 $^{(3)}$
3			.15	.6 $^{(4)}$
4				.3



EECS 560
Spring 2018
Exam 2

Scores

<table>
<tr><td rowspan="13">

- Once you start the exam, you will not be excused from the room for any reason unless you turn in your exam. Once it is turned in, you cannot come back and continue working on it.
- Answer all questions.
- This exam is closed book and closed notes.
- No calculators, cell phones, head phones, or electronic devices of any sort will be allowed. No such devices should be out in the open.
- You must write legibly and show all your work clearly for credit. Partial credit will only be given to meaningful answers. You will be graded according to your approach to the problems, mathematical rigor, and quality of your solutions.
- ***All data structures and their implementations are based on our lectures and lecture notes.***
- ***You must show your tree clearly after each insertion/deletion/modification for credit. No credit will be given if you do not show all your trees/work.***
- *Suggestion*: Quickly scan the entire test, do the problems you know best first. Then spend your time on the remaining problems.

</td><td>1</td><td>17 /20</td></tr>
<tr><td>2</td><td>10 /10</td></tr>
<tr><td>3</td><td>15 /15</td></tr>
<tr><td>4</td><td>10 /15</td></tr>
<tr><td>5</td><td>10 /10</td></tr>
<tr><td>6</td><td>10 /10</td></tr>
<tr><td>7</td><td>10 /10</td></tr>
<tr><td>8</td><td>10 /10</td></tr>
<tr><td></td><td></td></tr>
<tr><td></td><td></td></tr>
<tr><td></td><td></td></tr>
<tr><td></td><td></td></tr>
<tr><td>**TOTAL**</td><td>92 **/100**</td></tr>
</table>

Name: Jian Shen

KUID: 2861432

4/10/2018

1. (20) Given a set of n records with distinct keys to be implemented using the given data structures as shown. Fill in the following table with the best possible worst-case complexity $T_w(n)$ in big-O for the corresponding operations. If more than one algorithm exists for a given operation, you must give the complexity of the best possible algorithm.

17

	build	insert	find	deleteMin
2-3 Tree	O(nlgn)	O(lgn)	O(lgn) .	O(lgn)
Minmax Heap	~~O(nlgn)~~ O(n)	O(lgn)	~~O(lgn)~~ O(n)	O(lgn)
Min Pairing Heap	O(n)	O(1)	O(n)	O(n)
Min Leftist Heap	O(nlgn)	O(lgn)	* O(n)	O(lgn)
Min Skew Heap	O(n²)	O(n)	O(n)	~~O(lgn)~~ O(n)

left most leave node

2. (10) By concentrating on the dominating step and by assuming that all basic operations require the same constant cost C, compute $T_W(n)$ in *closed-form* for the following program segment as discussed in class. *Remark:* You must first set up the equation for $T_W(n)$ and then evaluate its sums for credits. Do not simplify the final expression.

10

```
x = 560; y = 660; k = 1;
for i = 1 to n do
    for j =1 to i do
        for k =j to n² do
            x = 3x + 2y + 210;
        endfor;
    endfor;
endfor;
while k ≤n² do
    for i = 1 to n do
        k = 2*k + i;
    endfor;
    x = x + y;
endwhile;
```

$$T_W(n) = \sum_{i=1}^{n}\sum_{j=1}^{i}\sum_{k=j}^{n^2} C$$

$$= C\sum_{i=1}^{n}\sum_{j=1}^{i}(n^2-j+1)$$

$$= C\sum_{i=1}^{n}\left[(n^2+1)i - \frac{(1+i)i}{2}\right]$$

$$= \frac{C}{2}\sum_{i=1}^{n}\left[(2n^2+1)i - i^2\right]$$

$$= \frac{C}{2}\left[(2n^2+1)\frac{(1+n)n}{2} - \frac{n(n+1)(2n+1)}{6}\right]$$

$$= \frac{C}{12}(6n^4+4n^3+2n)$$

$$= \frac{C}{6}(3n^4+2n^3+n)$$

$$= O(n^4)$$

$(n^2+n)(2n+1)$

$(2n^2+1)i - i - i^2$

$2n^3+3n^2+n$

$\frac{(4n^2+2)(n^2+n)}{2} - \frac{\cancel{2n^3+n^2+2n+n}}{6}$

$4n^4+4n^3+2n^2+2n$

$12n^4+12n^3+6n^2+6n-2n^3-3n^2-n$

$12n^4+10n^3+3n^2+5n$

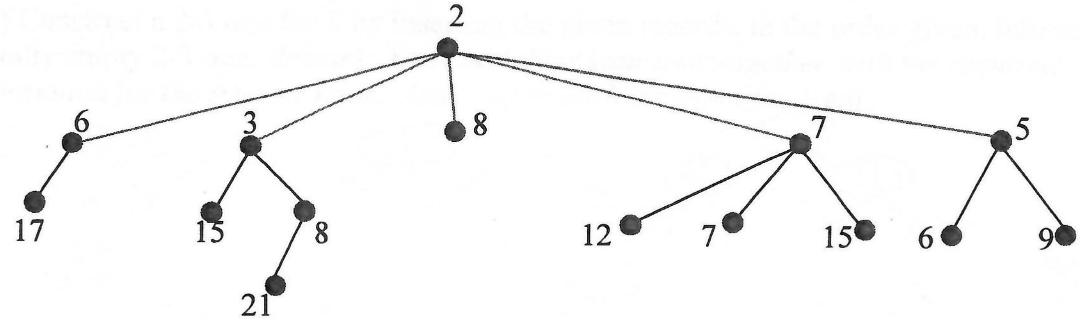
$(2n^2+1)(n^2+n)$ $\quad n^2+n$

$2n^4+2n^3+n^2+n$

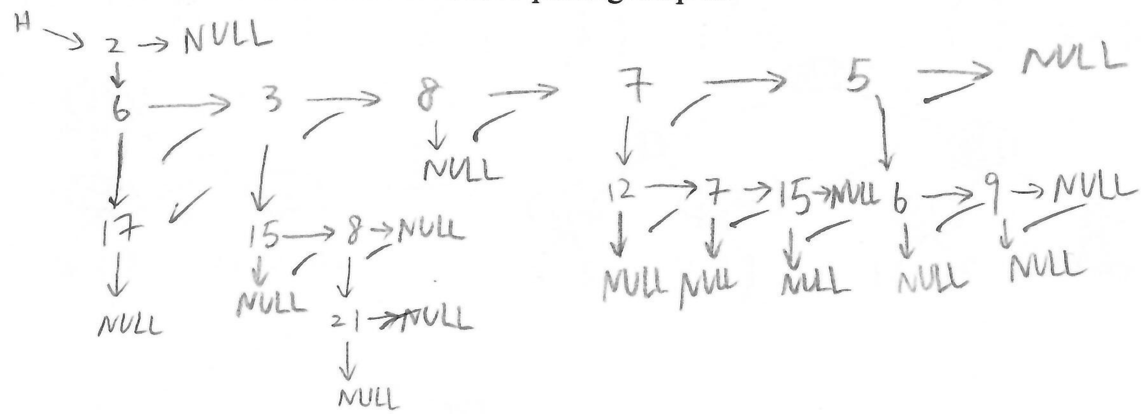
$6n^4+6n^3+3n^2+3n-2n^3-n^2-2n-n$

$3 \quad \frac{6n^4+4n^3+3n^2+5n}{6}$

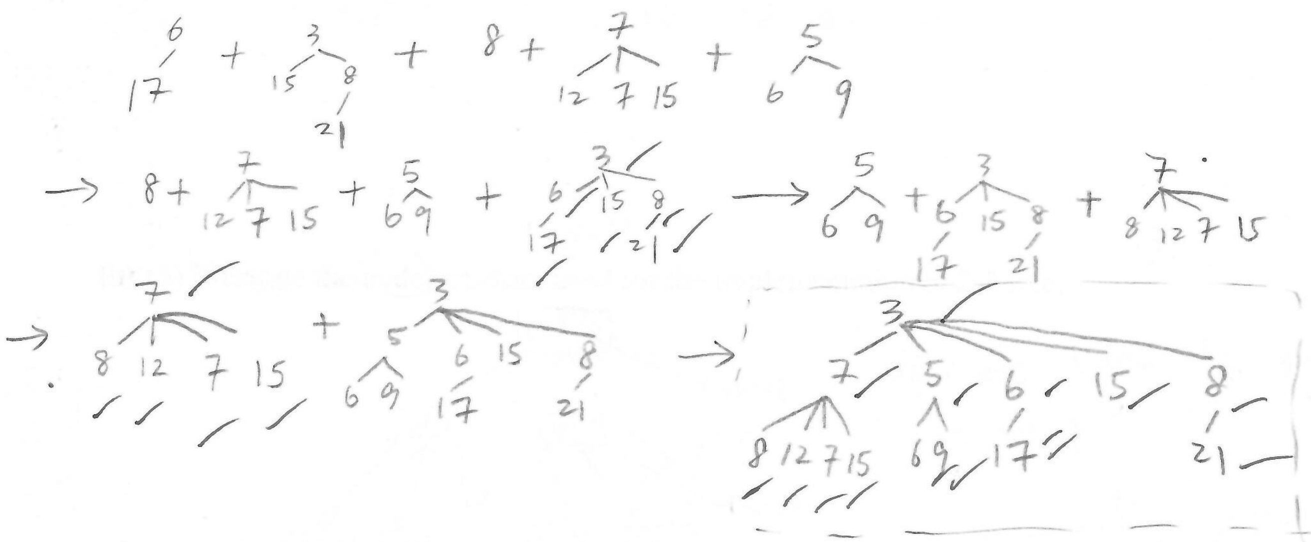
3. Consider the following pairing heap H.



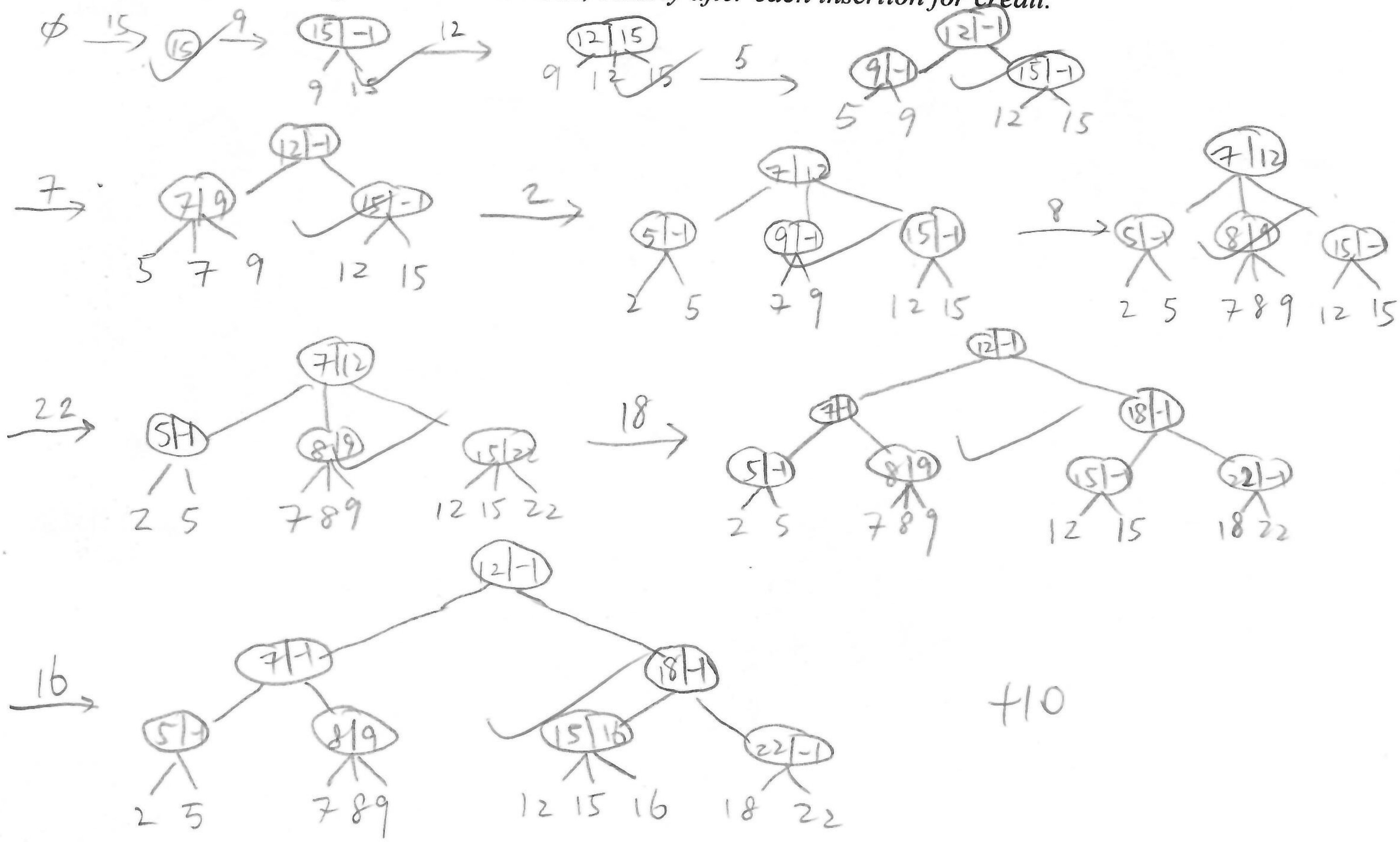
(a) (5) Illustrate the data structure for the above pairing heap H.



(b) (10) Perform deleteMin on H by using the multi-pass method for merging. *Remark: You must show your trees clearly after each merging operation for credit.*



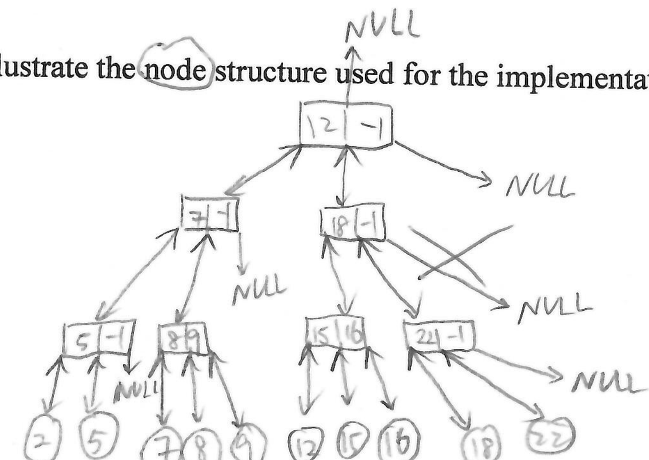
4. Given a set S of 10 records with priorities {15, 9, 12, 5, 7, 2, 8, 22, 18, 16}.
   (a) (10) Construct a 2-3 tree for S by inserting the given records, in the order given, into an initially empty 2-3 tree. *Remark: You must show your tree, together with the required information for the interior node, clearly after each insertion for credit.*



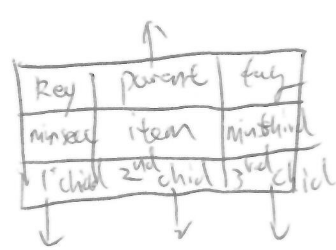
+10

(b) (5) Illustrate the node structure used for the implementation of 2-3 tree.

0

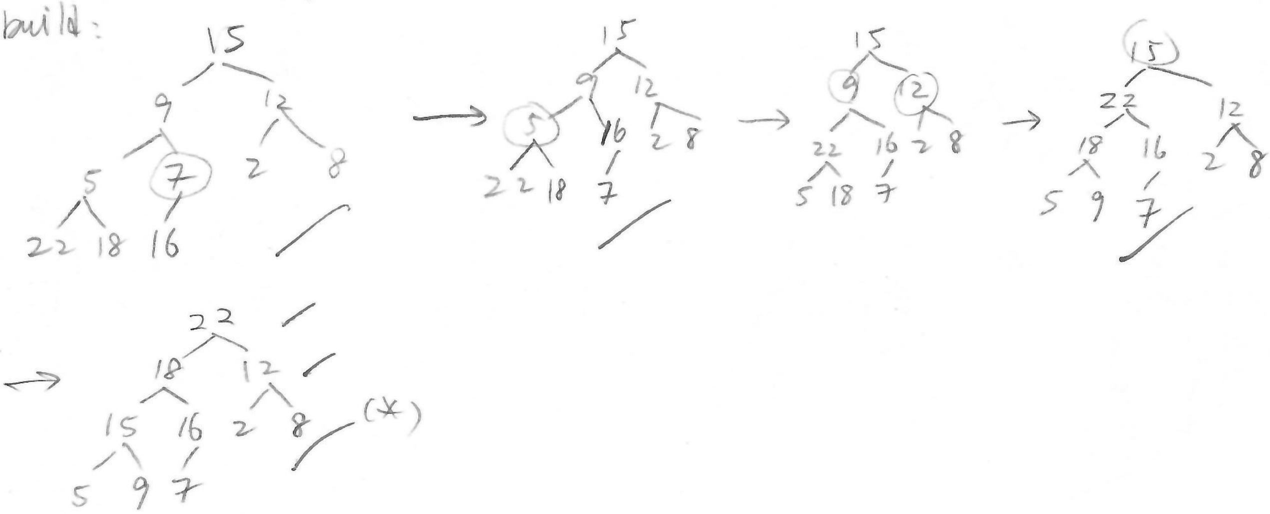


This's not what the question is asking.

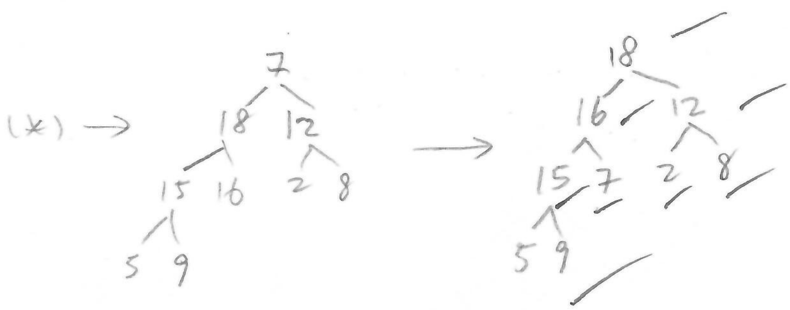


5. (10) Given a set S of 10 records with priorities {15, 9, 12, 5, 7, 2, 8, 22, 18, 16}. Construct a max 2-heap for S by using the bottom-up buildHeap operation. When done, perform deleteMax once. *Remark: You must show your tree clearly after each insertion/deletion for credit.*

build:

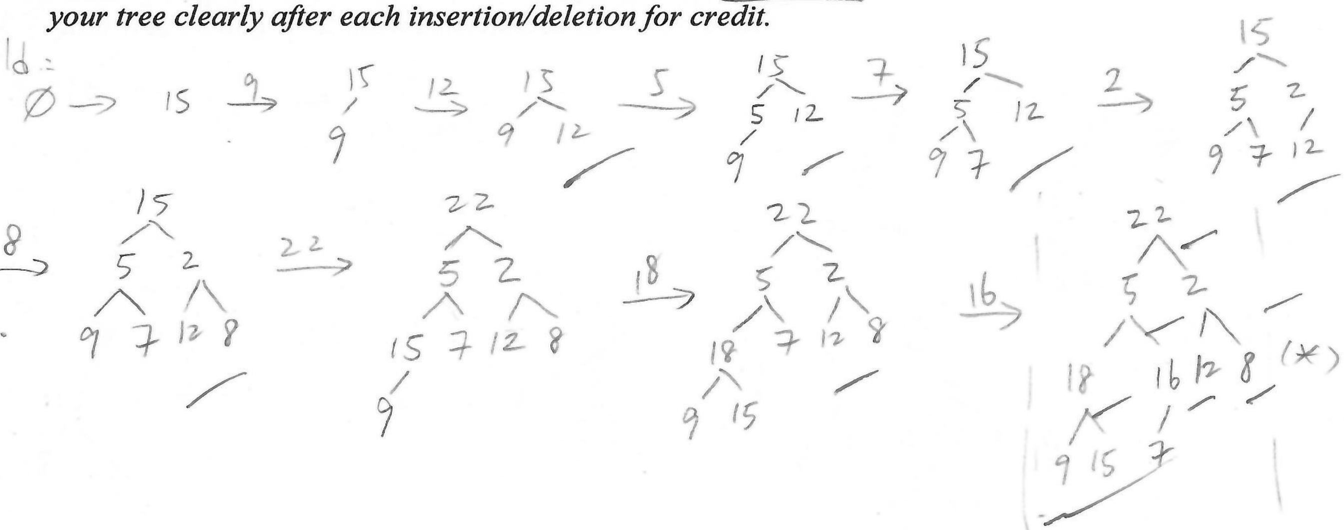


deleteMax:

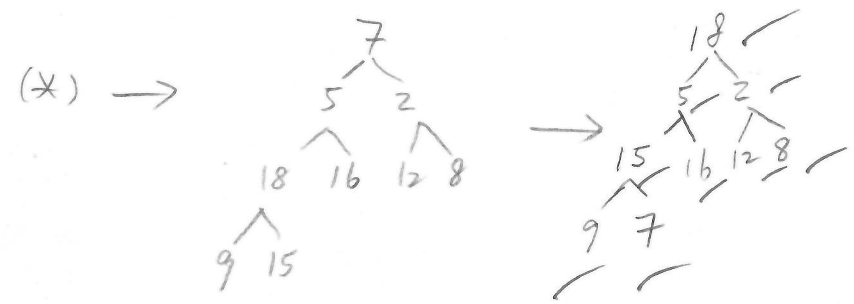


6. (10) Given a set S of 10 records with priorities {15, 9, 12, 5, 7, 2, 8, 22, 18, 16}. Construct a maxMin heap for S by inserting the given records, in the order given, into an initially empty maxMin heap. When done, perform deleteMax once. *Remark: You must show your tree clearly after each insertion/deletion for credit.*

build:

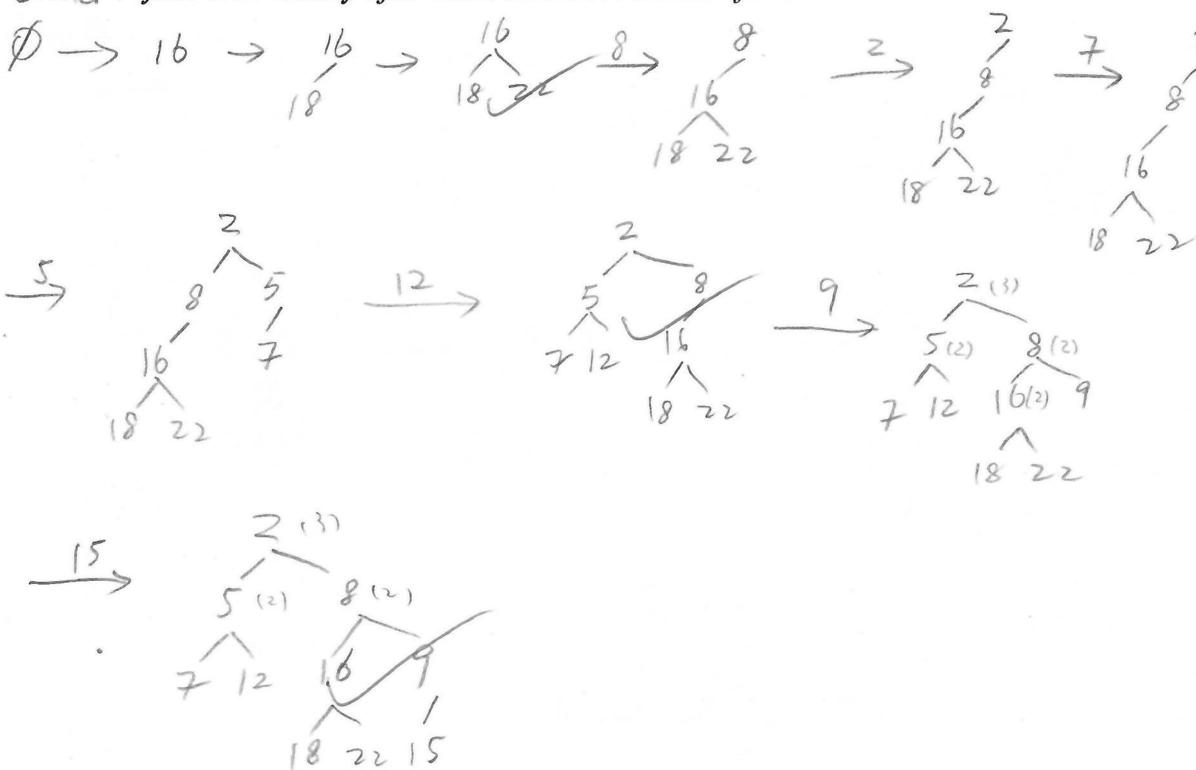


deleteMax:

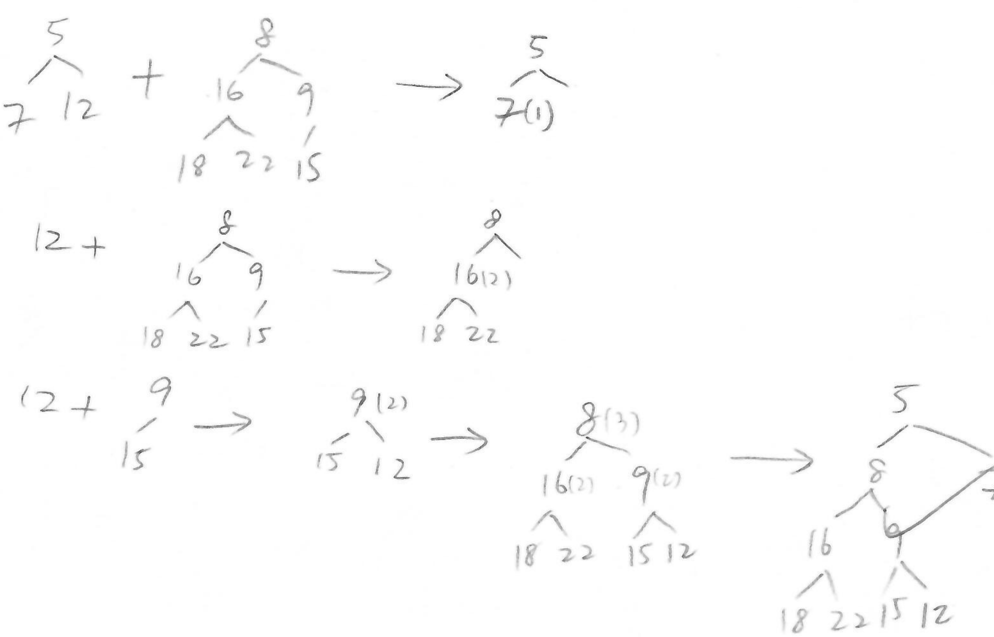


7. (10) Given a set H of 10 records with priorities {16, 18, 22, 8, 2, 7, 5, 12, 9, 15}. Construct a min leftist heap for H by inserting the given records, in the order given, into an initially empty leftist heap. When done, perform deleteMin once. *Remark: You must show your tree clearly after each insertion/deletion for credit.*

build :



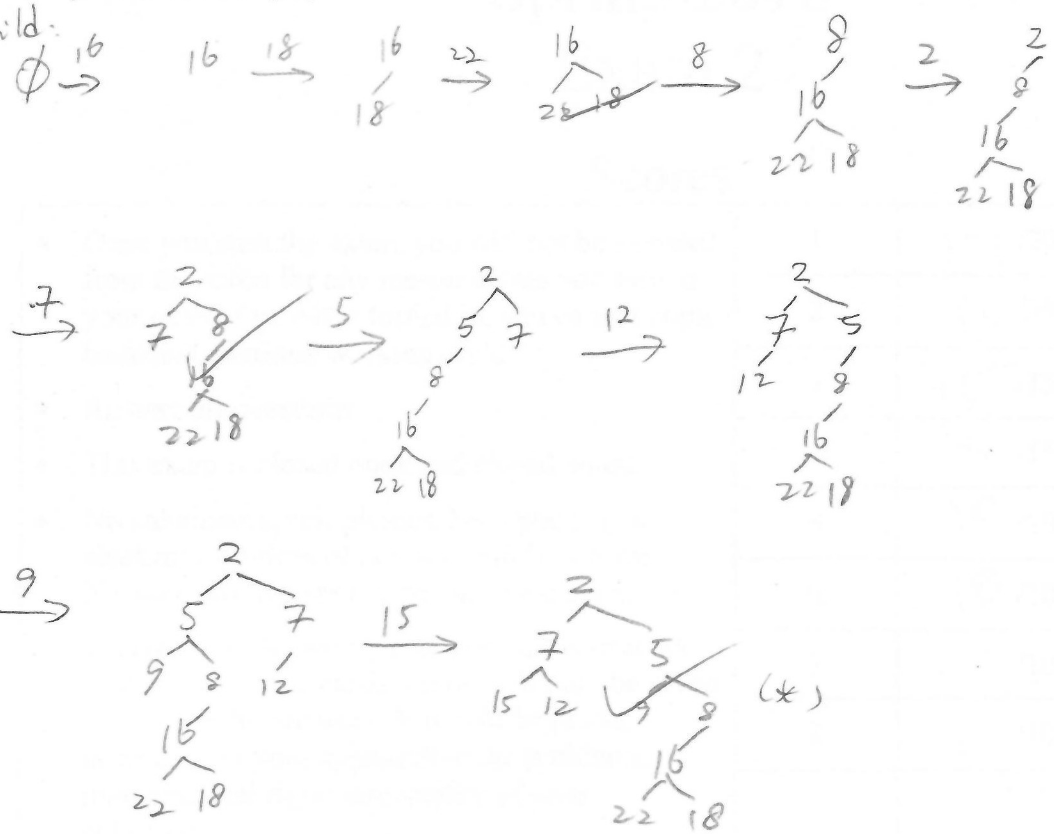
deleteMin :



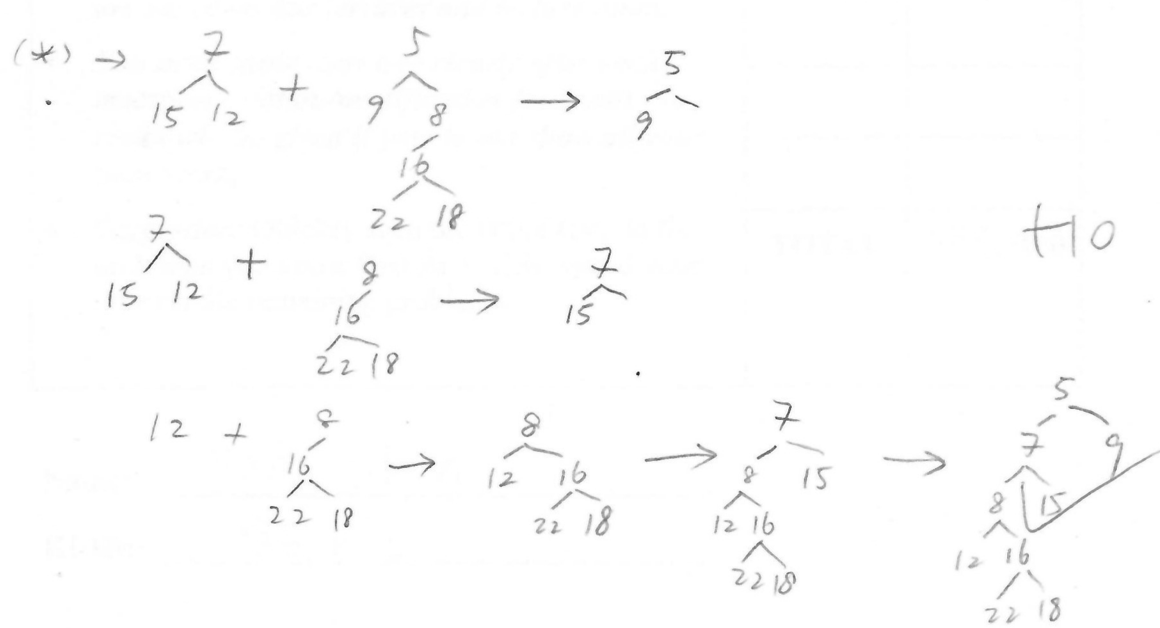
+10

8. (10) Given a set H of 10 records with priorities {16, 18, 22, 8, 2, 7, 5, 12, 9, 15}. Construct a <u>min</u> <u>skew</u> heap for S by inserting the given records, in the order given, into an initially empty skew heap. When done, perform deleteMin once. *Remark: You must show your tree clearly after each insertion/deletion for credit.*

build:



deleteMin:



+10

93

Jian Shen

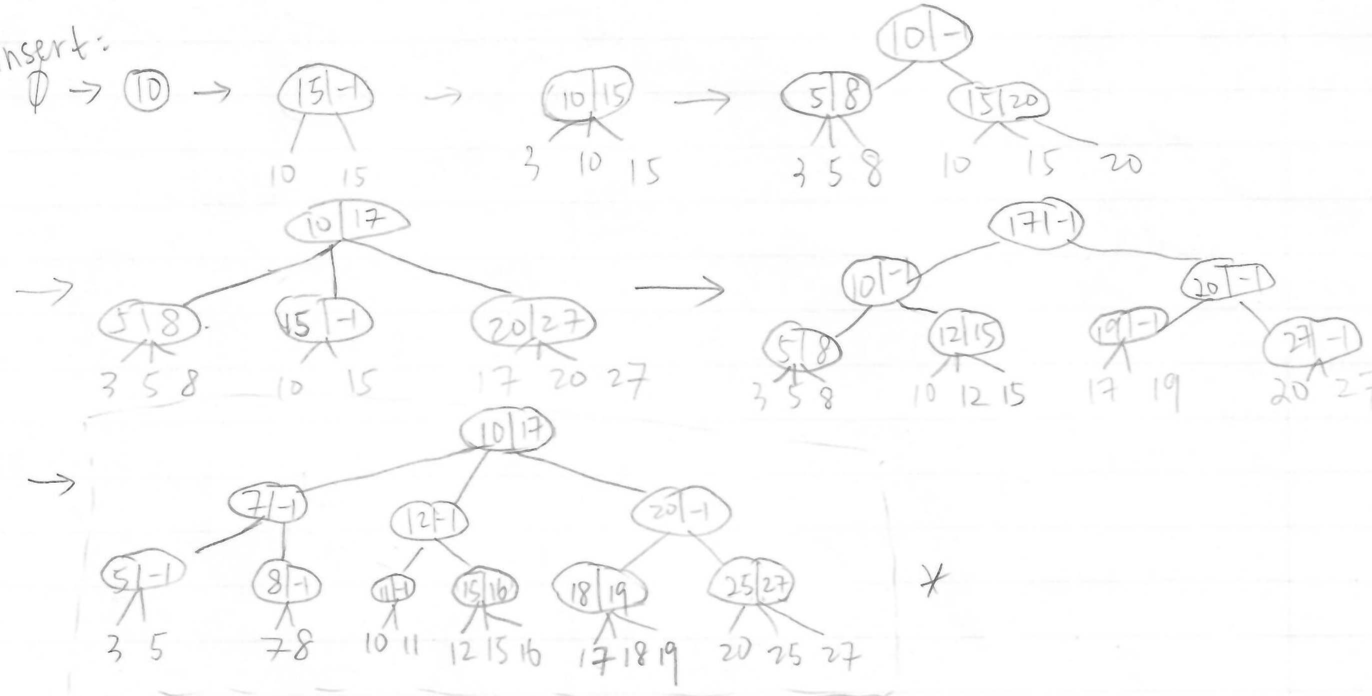
2861432

EECS 560

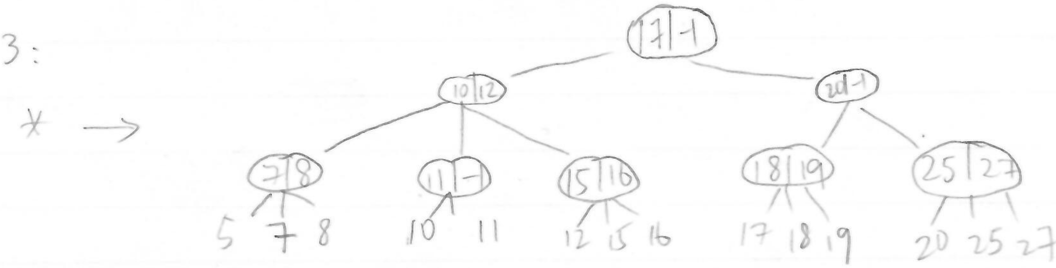
HW#3

Jian Shen
2861432
Mar 29 2018
HW#3

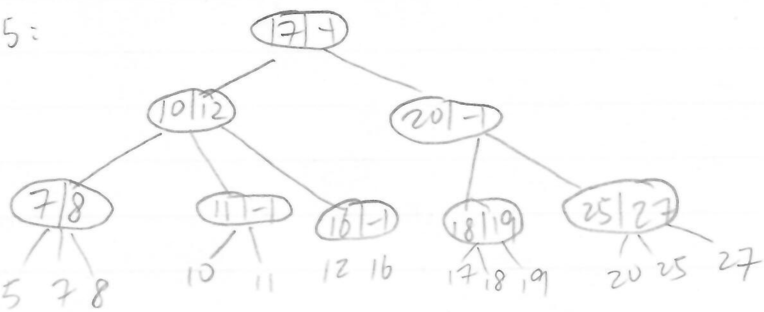
1(a) insert:



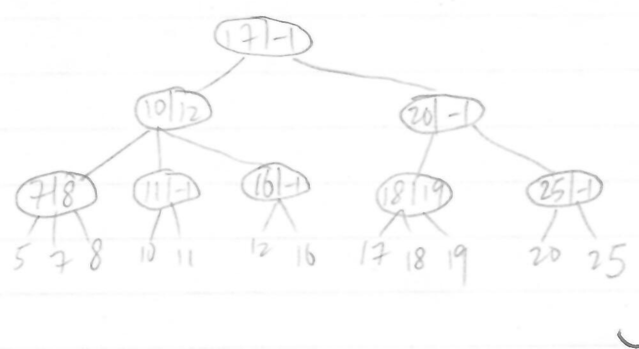
delete 3:



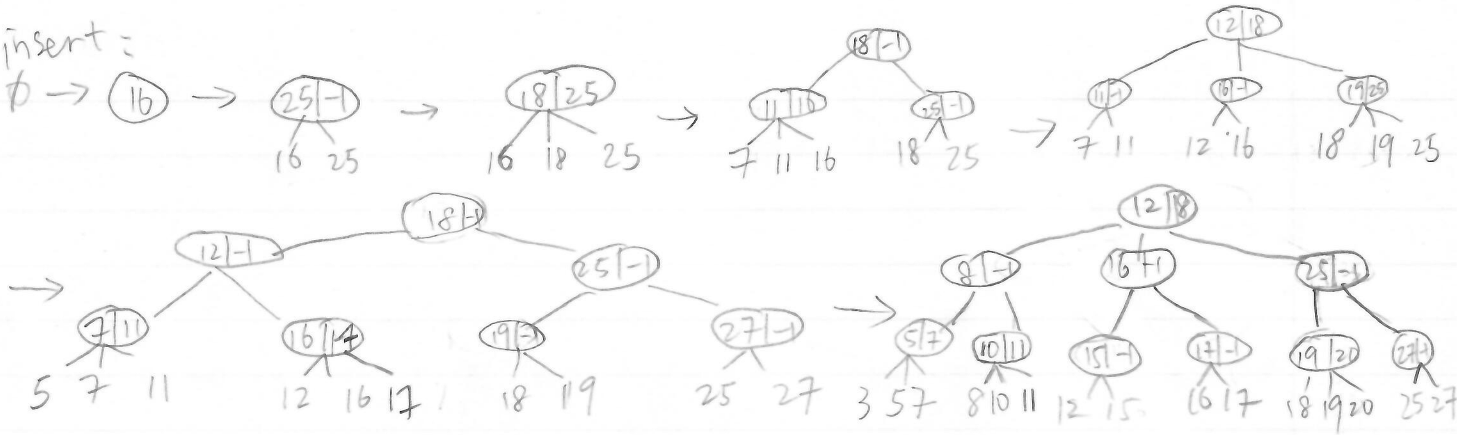
delete 15:



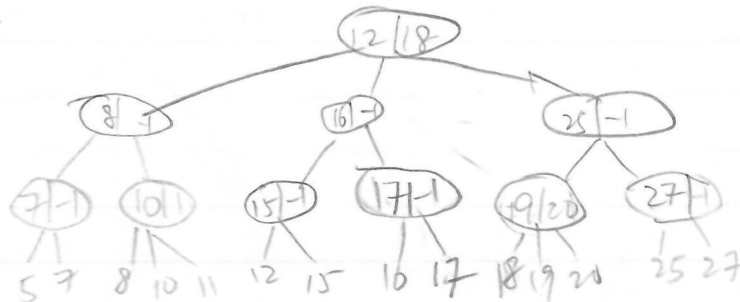
delete 27:



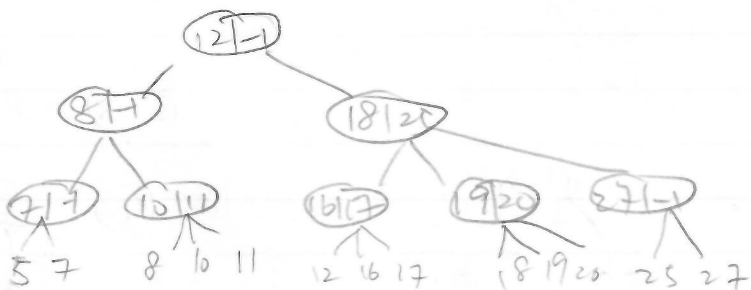
1(b) insert:



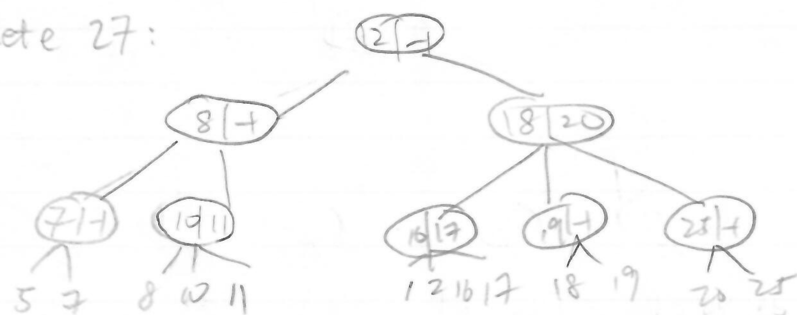
delete 3:



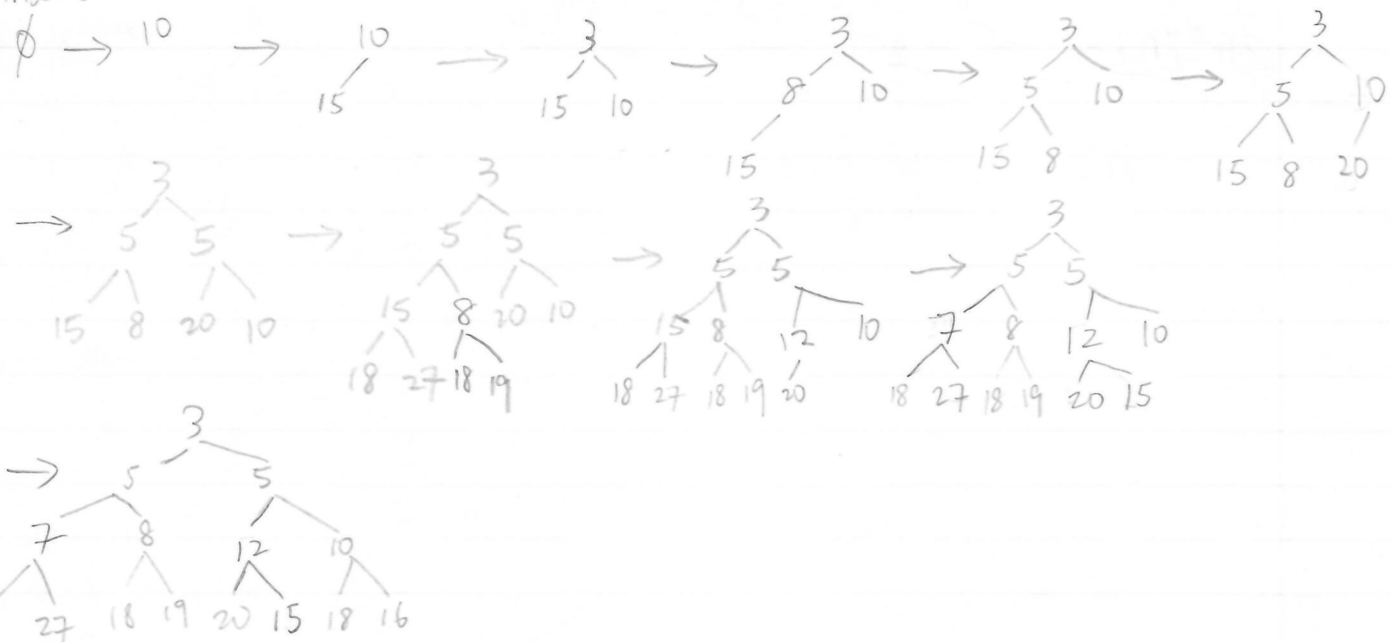
delete 15:



delete 27:

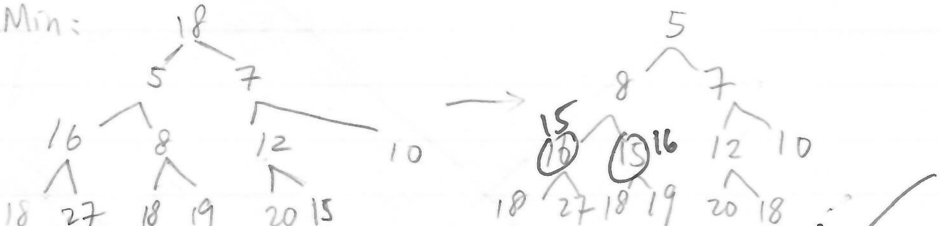


2(a) insert:



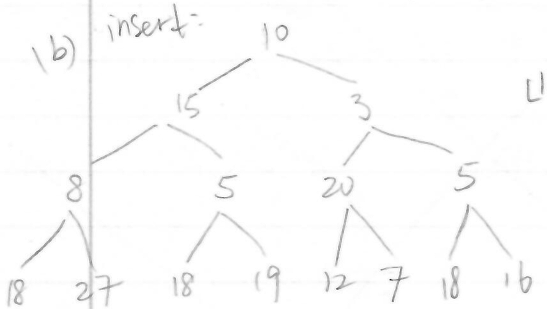
deleteMin:

deleteMin:

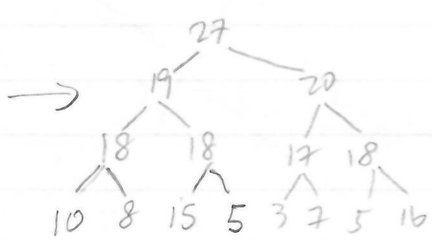


−1

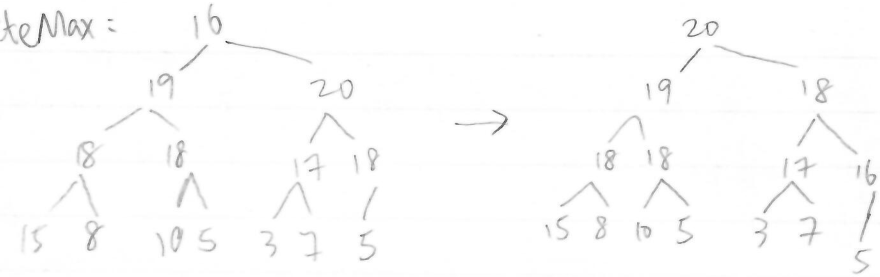
(b) insert:



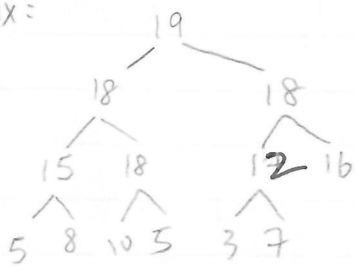
$\lfloor 15/2 \rfloor - 1 = 6$



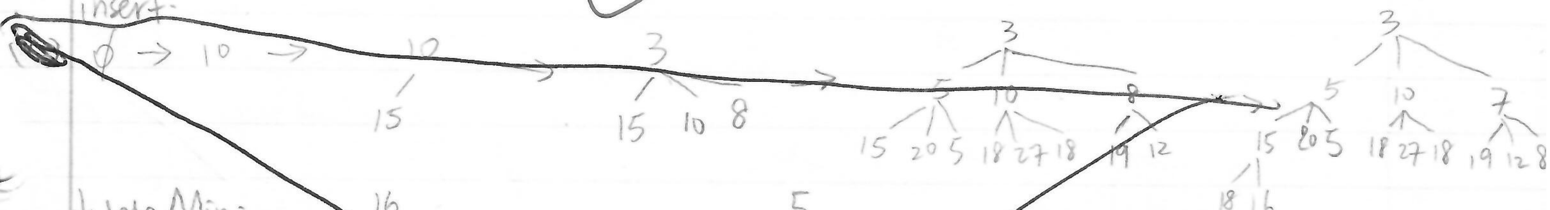
deleteMax:



deleteMax:



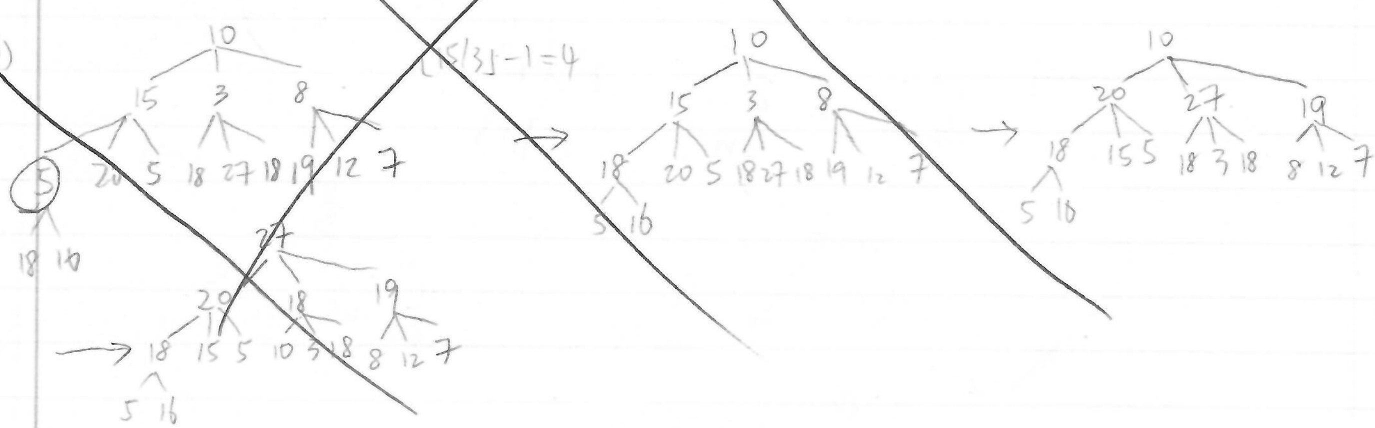
insert:



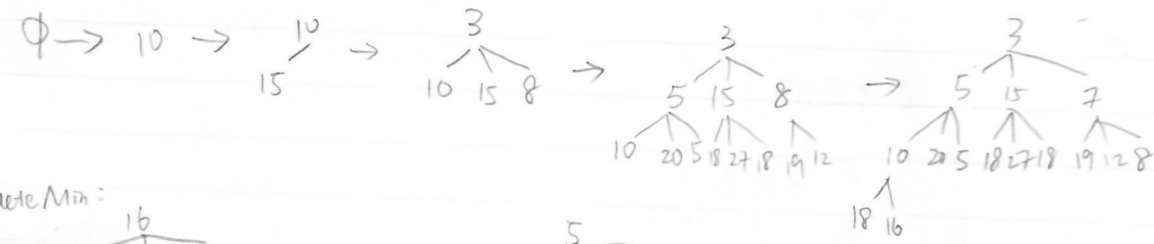
deleteMin:

deleteMin:

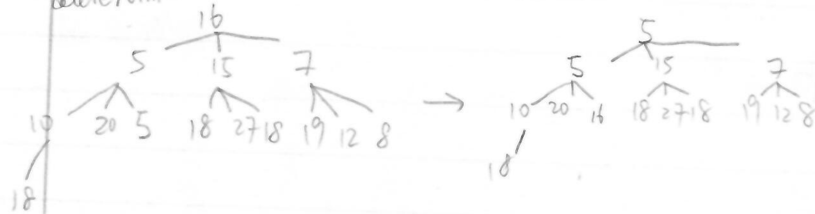
(d)



2(c) insert:



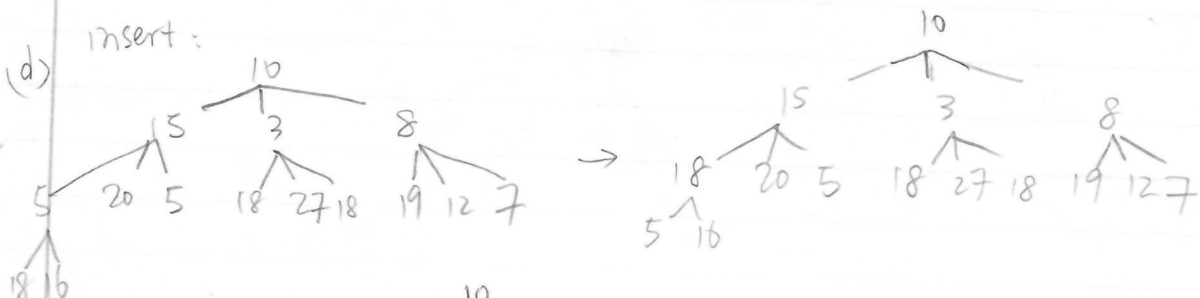
delete Min:

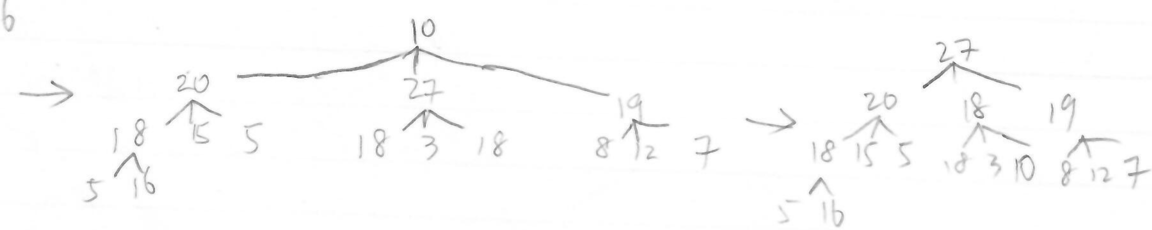


delete Min:

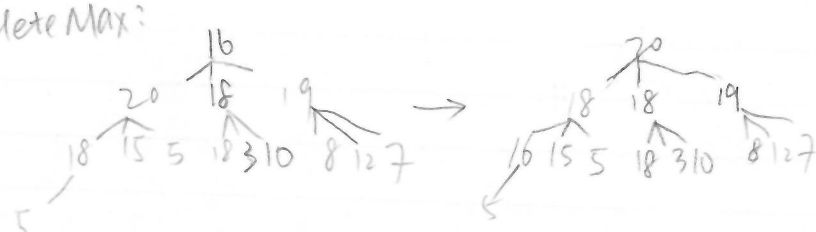
-1

(d) insert:





delete Max:

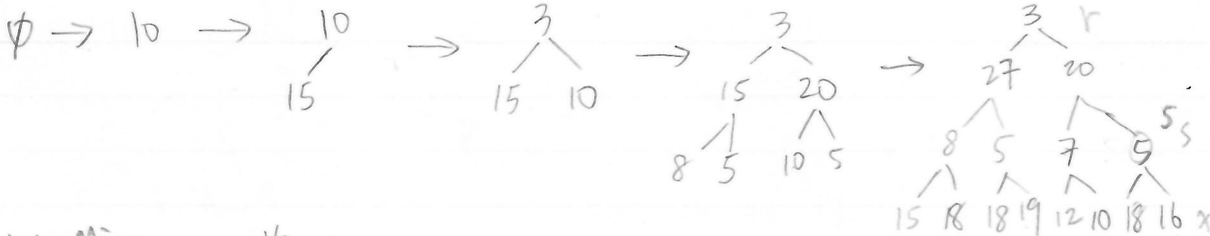


delete Max:

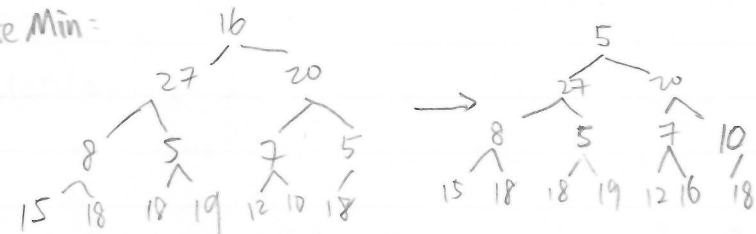
10 15 3 8 5 20 5 18 27 18 19 12 7 8 16

insert:

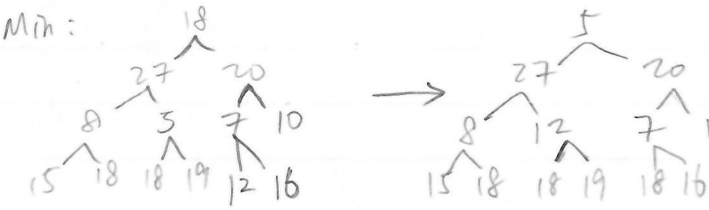
2(e)



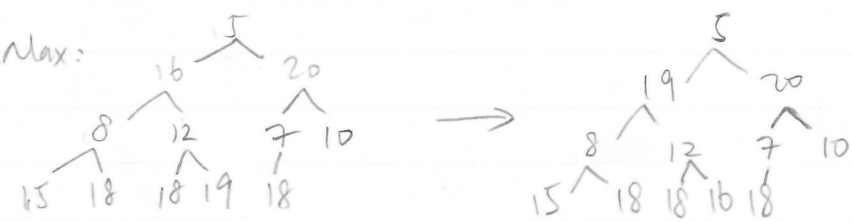
deleteMin:



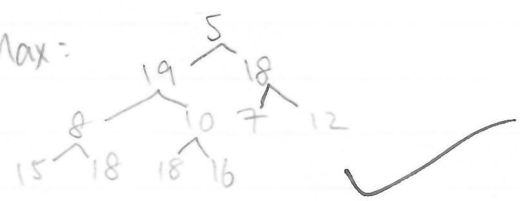
deleteMin:



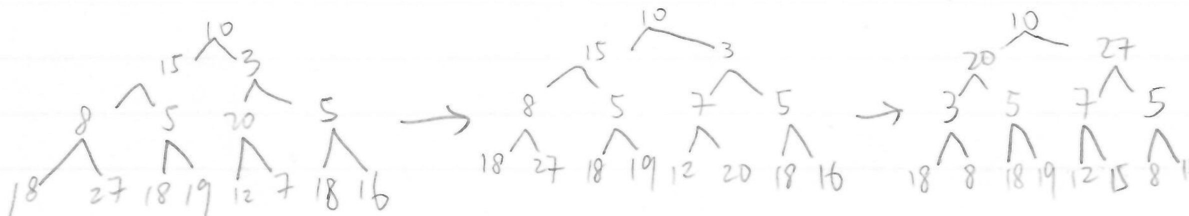
deleteMax:

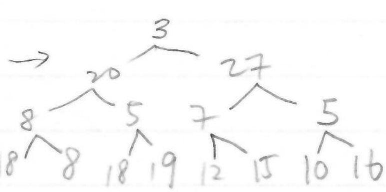


deleteMax:

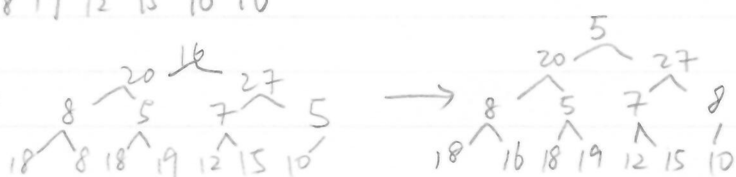


(f)

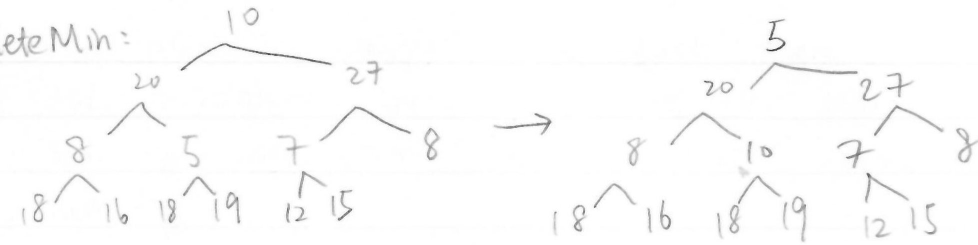




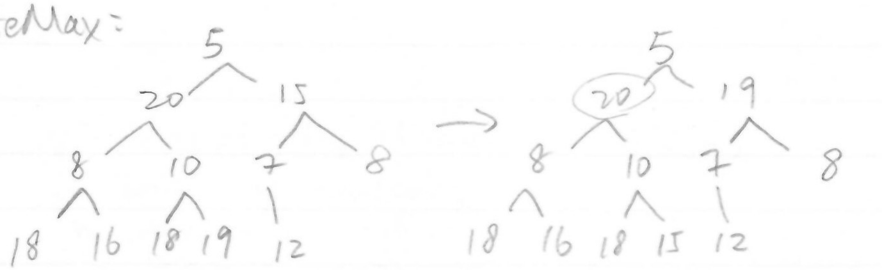
deleteMin:



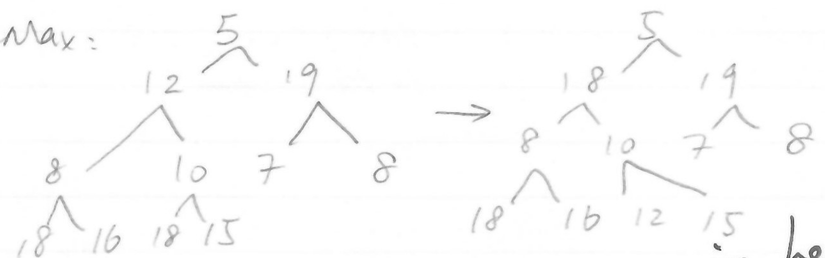
deleteMin:



deleteMax:



deleteMax:



Build a max min heap.

-5

3\. // find minimum element from first leave to last used index.
// set minimum element to last used index
// set last used element to 0, indicates which is empty
// heapify

```
int firstLeaf = floor((heapSize-1 -1)/5)+1;
int min = arr[firstLeaf]
int index = firstLeaf;
while (index > heapSize-1)
{    if ((index+1) ≤ min)
        { min = index+1;
        }
}
arr[min] = arr[lastUsedIndex];
arr[lastUsedIndex] = 0;
heapSize--;
while (arr[min] > arr[floor((min-1)/2)])
{   arr[floor(min-1)/2)] = arr[min]);
    min = floor((min-1)/2);
}
```

\# of leaves

level of heap

$T_w(n) = C \times \max(x, y)$ (where x is maximam # of leaves, y is the level of heap

$= C\lceil n/2 \rceil$

$= \frac{c}{2} n$

$= cn$

4. // go to A[i],
// change it to newkey
// heapify

```
A[i] = newKey;
while ( A[i] < A[parent of i])
  {  A[i] = A[parent of i];
     i = parent of i;
  }
 A[i] = newkey;
```

$$T_W(n) = C \cdot \lfloor \log_2 n \rfloor$$
$$= C \log_2 n$$