Name: Josh Clemens CS 113 Final Exam

Date: 5-17-23

Grade: _____ /100

Instructions: This is a take-home test. You are only allowed to use our class notes, material, and book. YOU MAY NOT USE ANYONE ELSE. If I suspect of cheating, I have the right to challenge your work and be able to show you know the material.

Make sure you show all your work and be as detailed as possible.

1. (10 points) Determine the big-O for each of the following functions. For this problem no need to show work, just the answer is ok.
   a. $f(n) = 10$ $O(1)$
   b. $f(n) = 3n + 7$ $O(n)$
   c. $f(n) = n^2 \log(n) + n^3 + n!$ $O(n!)$
   d. $f(n) = n^4 + 3^n$ $O(3^n)$
   e. $f(n) = 5n^3 + n^3 \log(n)$ $O(n^3 \log n)$
   f. $f(n) = n^4 + 9n^3 3^n + 4n^3 n! + 7$ $O(n^3 n!)$
   g. $f(n) = (n\, 3^n \log(n) + n^2 2^n)(n^3 + 2)$ $O(n^5 2^n)$

2. (10 points) **Determine T(n)** relationship between processing time and n. Then **determine a big-O** estimate for the number of operations (a Simple Statement takes one unit of time) used in this segment of an algorithm. **SHOW YOUR WORK.**

```
for(int i = 0; i<n; i++){
        for(int j = 10; j<n; j++){
                Simple Statement
                Simple Statement
                Simple Statement
                Simple Statement
                Simple Statement
        }
}
```

$5n^2$

i=0
J=10 J=11 J=12 ... J=n-1
+5 +5 +5
n times

5

i=1
J=10 J=11 J=12 ... J=n-1
n times

```
for(int i = 0; i<n; i++){
        for(int j = 0; j<n; j++){
                for(int k = 0; j<n; k++){
                        Simple Statement
                        Simple Statement
                        Simple Statement
                        Simple Statement
                        Simple Statement
                }
        }
}
```

$5n^3$

i=0
J=0
K=0 K=1 K=2 ... K=n-1
+5 +5 +5 +5
n times

5

i=1
J=1
K=1 K=2 K=3 ... K=n-1
+5 +5 +5 +5
n times

$T(n) = 5n^2 + 5n^3 = O(n^3)$

3. (10 points) **Determine T(n)** relationship between processing time and n. Then **determine a big-O** estimate for the number of operations (a Simple Statement takes one unit of time) used in this segment of an algorithm. **SHOW YOUR WORK.**

```
for(int i = 0; i<n; i++){
        for(int j = n-1; j>=i; j--){
                Simple Statement
                Simple Statement
        }
}
for(int i = 0; i<n; i=i + 2){
                Simple Statement
                Simple Statement
                Simple Statement
                Simple Statement
                Simple Statement
}
```

$2n^2+n$

$i=0, i=1, i=2 \cdots i=n-1$

$j=n, j=n-1, j=n-2 \cdots j=i$

2 2 2 2

$5\frac{n}{2}$

kth

$i=0 \quad i=2 \quad i=4 \cdots 5k<n\leq 5(k+1)$

5 5 5

$5k<n\leq 5(k+1)$

$\frac{5k}{5} < \frac{n}{5} \leq \frac{5(k+1)}{5}$

$T(n)=2n^2+n+5\frac{n}{2}=O(n^2)$

4. (10 points) Find $C, n_0, f(n)$ such that $|T(n)| \leq C|f(n)|$ whenever $n > n_0$.
   a. $T(n) = n^4 + 9n^3 + 4n + 7$

n^4

$n\geq 1 \qquad n\geq 1$

$n^4+9n^3+4n+7 \leq n^4+9n^4+4n^4+7n^4$

for all n $\qquad n\geq 1$

$n^4+9n^3+4n+7 \leq 21n^4$ for $n\geq 1$

$f(n)=n^4$

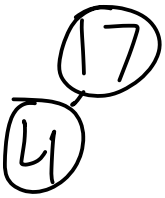
$C=21$

$n_0=1$

5. (10 points) Complete a Binary Search Tree 17, 4, 1, 5, 13, 22, 8, 27, 11, 3, 9, 10, 14, 6. **Show all your drawings, step by step to get full credit. Showing just the answer will give you 2 points. Just like in the lecture notes.**

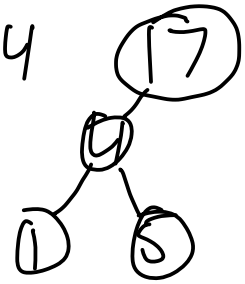
5.1

5.2

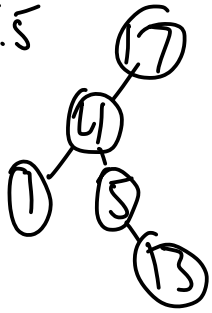


5.3

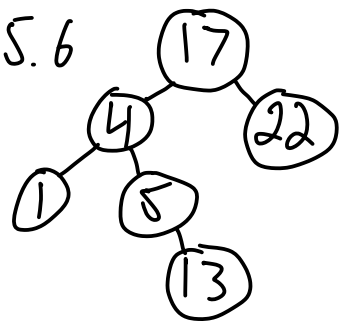
5.4



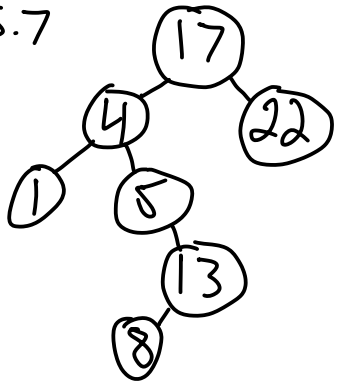
5.5



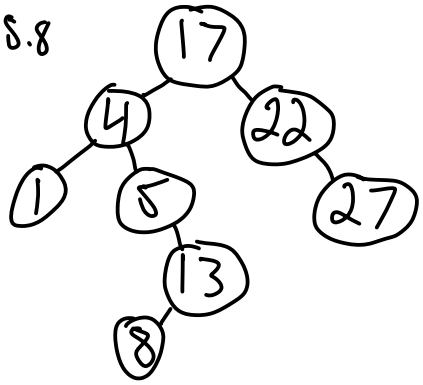
5.6



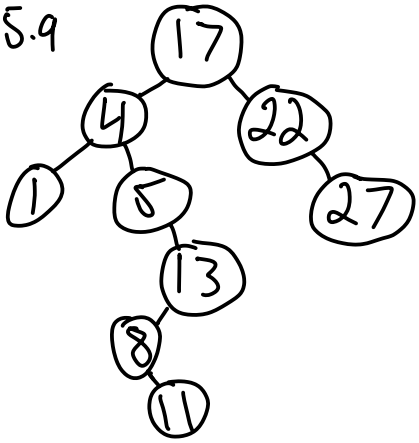
5.7



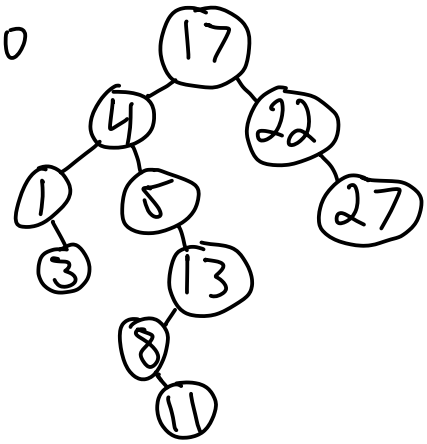
5.8



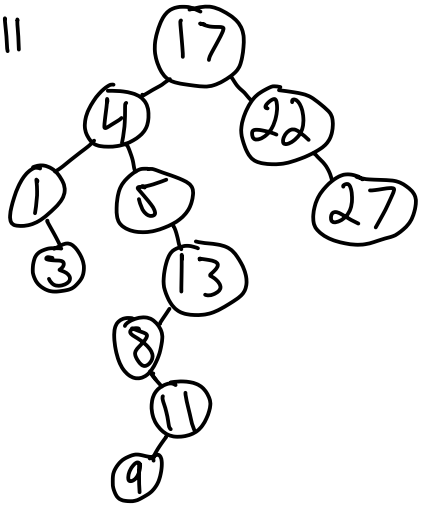
5.9



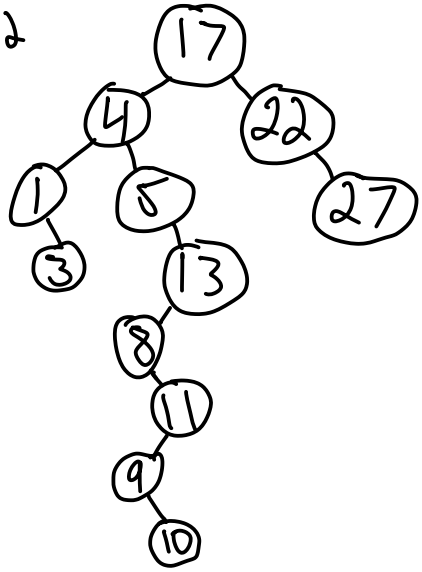
5.10



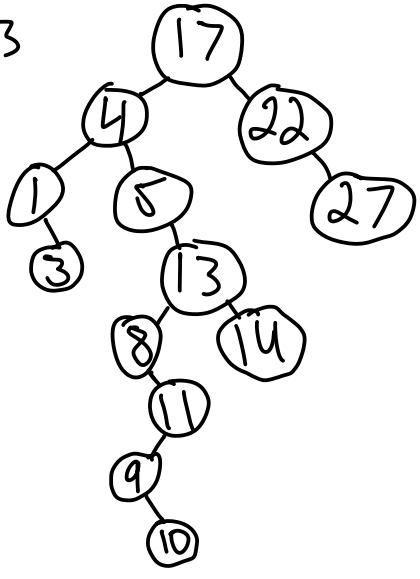
5.11



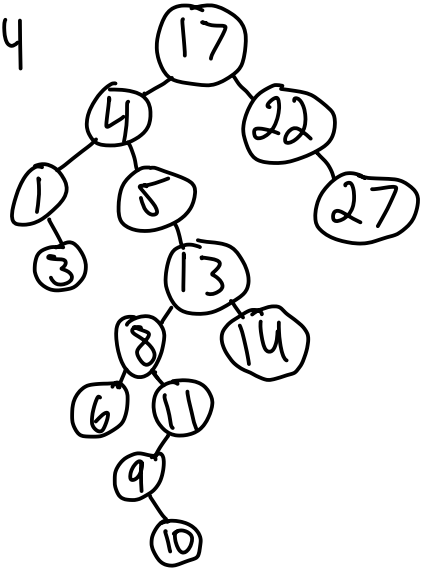
5.12



5.13

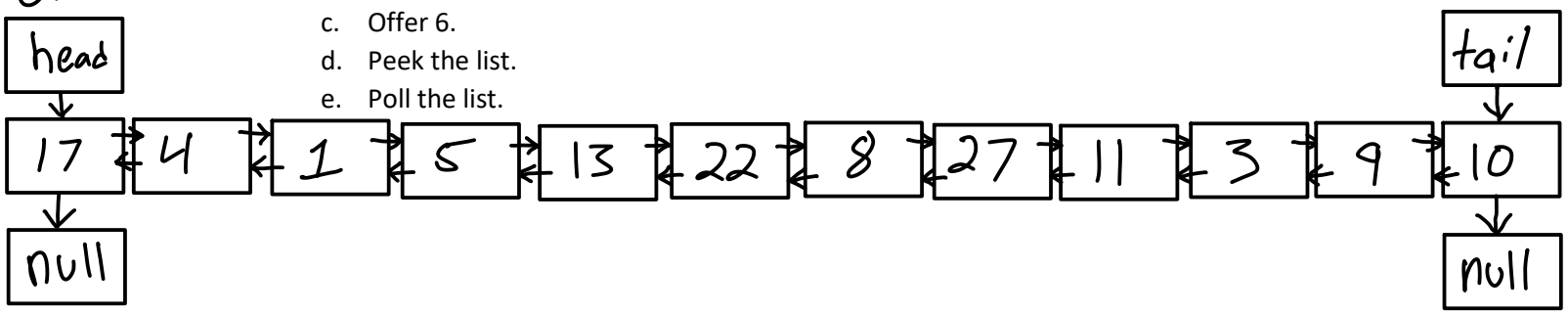


5.14

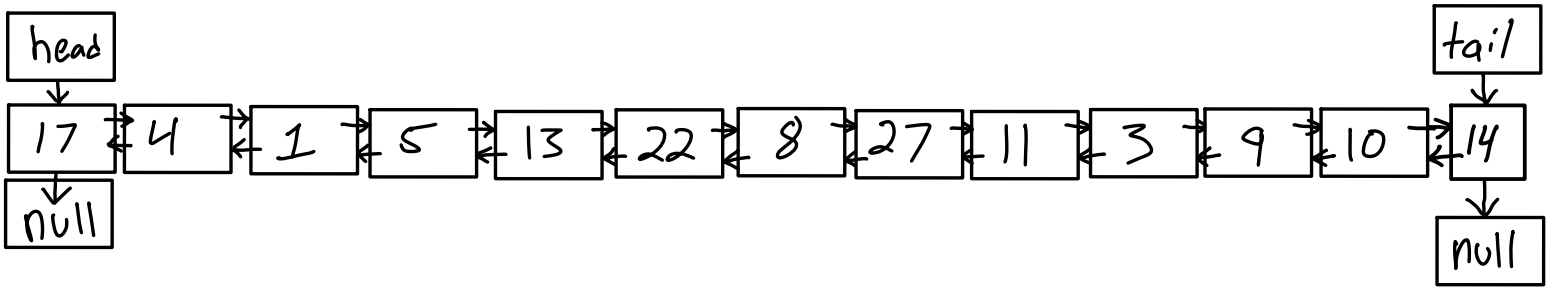


6. (10 points) Update the **Queue** List with the given commands. **Draw a new picture for each part.**
   a. Add the following numbers to the **Double-LinkedList**: 17, 4, 1, 5, 13, 22, 8, 27, 11, 3, 9, 10 in that order.
   b. Add 14.
   c. Offer 6.
   d. Peek the list.
   e. Poll the list.

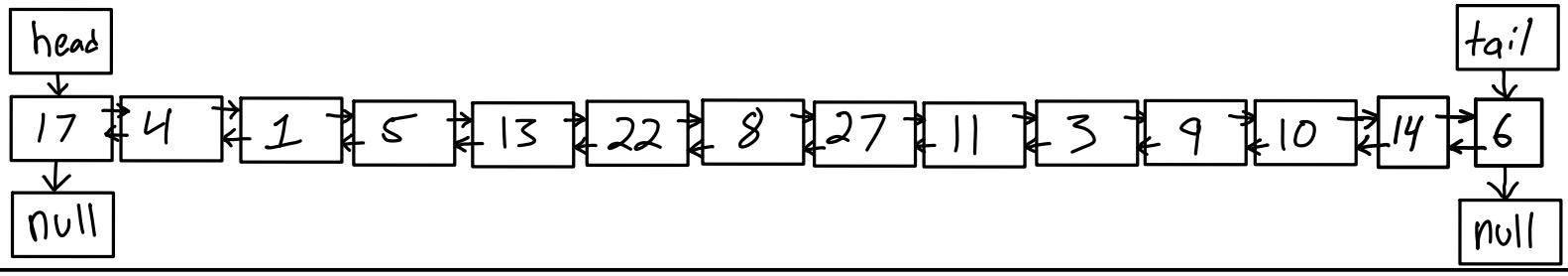
6.a



6.b Add 14



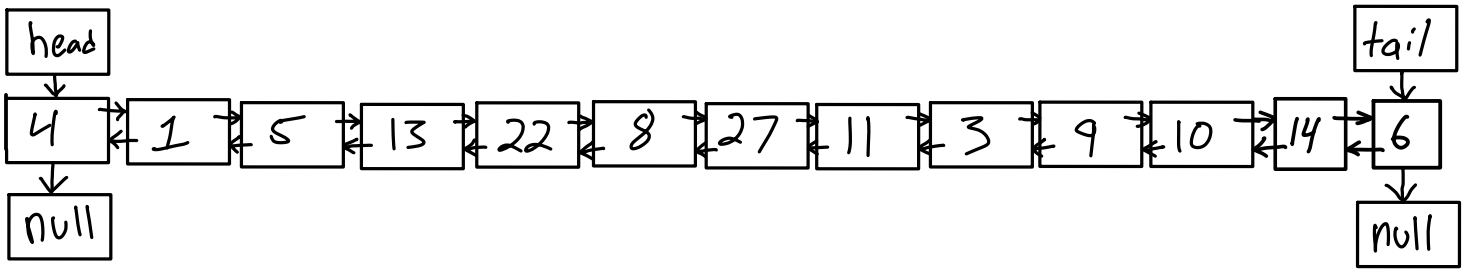
6.c Offer 6. Return true if Successful



6.d Peek() return Element at front of Queue

17

6.e Poll() returns 17 and removes front element of Queue

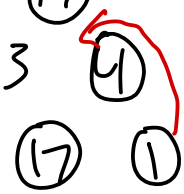


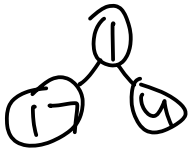
7. (10 points) Complete a min Heap priority queue ~~array~~ tree 17, 4, 1, 5, 13, 22, 8, 27, 11, 3, 9, 10, 14, 6.
Show all your drawings, step by step to get full credit. Showing just the answer will give you 2 points. Just like in the lecture notes.

7.1

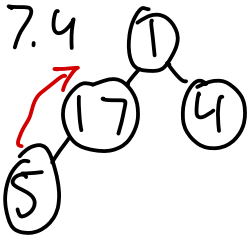
7.2

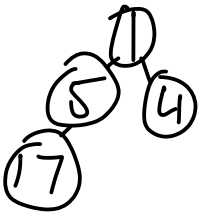
7.3



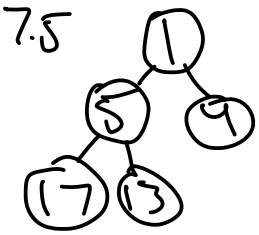


7.4

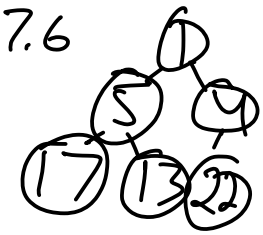




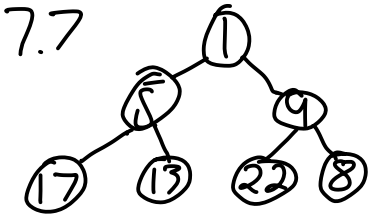
7.5



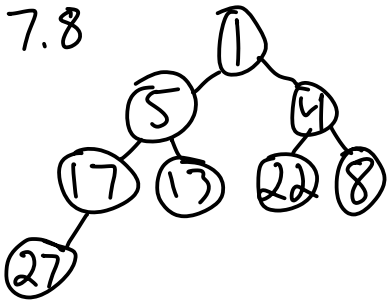
7.6



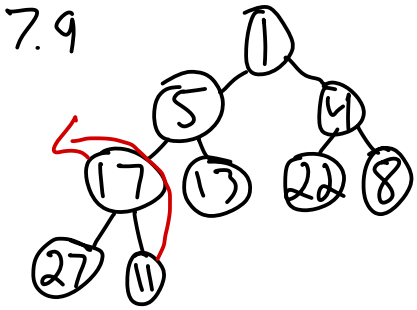
7.7

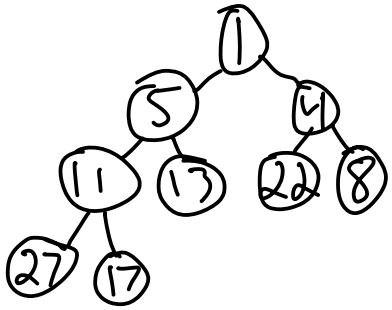


7.8

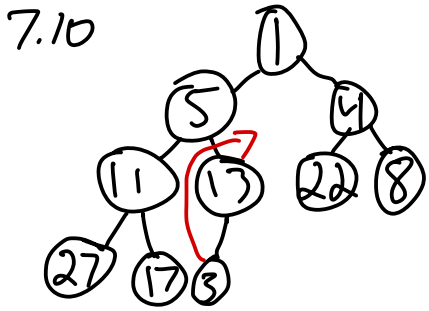


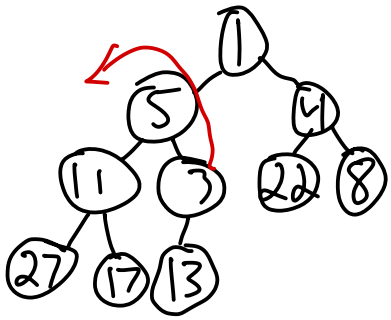
7.9

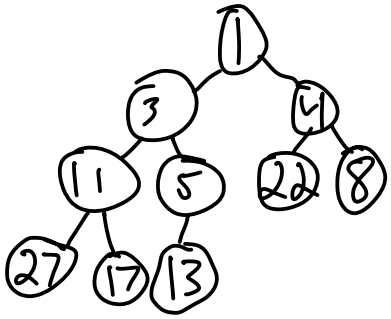




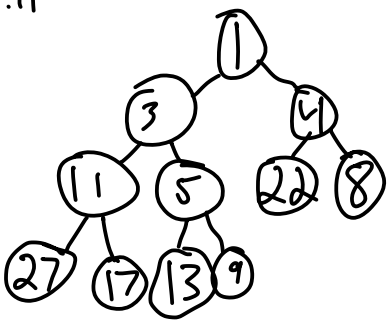
7.10



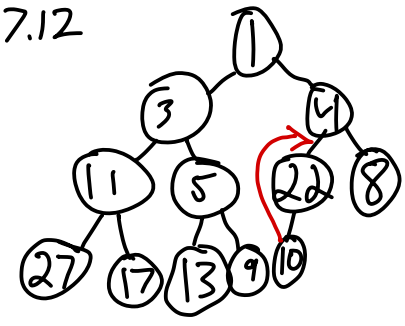


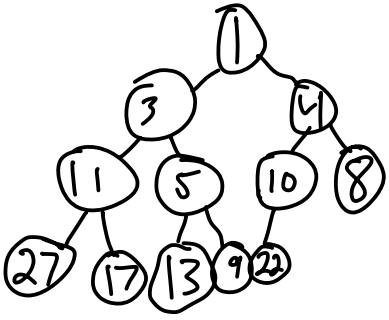


7.11

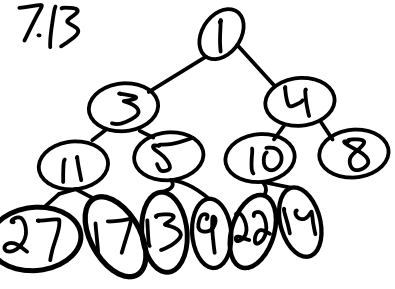


7.12

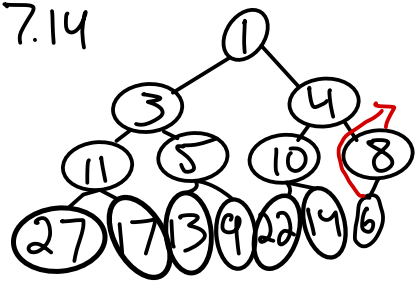


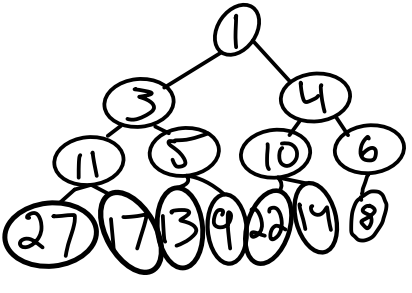

1
3
4
11
5
10
8
27
17
13
9
22

7.13


1
3
4
11
5
10
8
27
17
13
9
22
14

7.14


1
3
4
11
5
10
8
27
17
13
9
22
14
6


1
3
4
11
5
10
6
27
17
13
9
22
14
8

8. (10 points) Fill in the HashCode Table values for size 7 and 15 and draw array for both

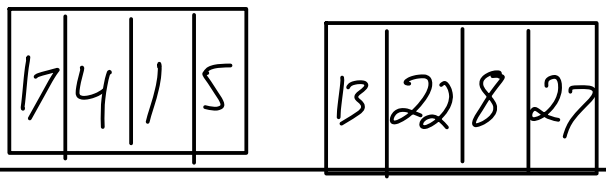
Animal	hashCode()	hashCode()%7	hashCode()%15
Dog	68892	5	12
Tiger	80806047	0	12
Cat	67510	2	10
Bear	2066388	2	3
Porcupine	-1642033963	-3+7=4	-13+15=2
Rabbit	-1854778310	0	-5+15=10
Cheetah	-1887932010	-6+7=1	0

index	Animal
[0]	Tiger
[1]	Rabbit
[2]	Cat
[3]	Bear
[4]	Porcupine
[5]	Dog
[6]	Cheetah

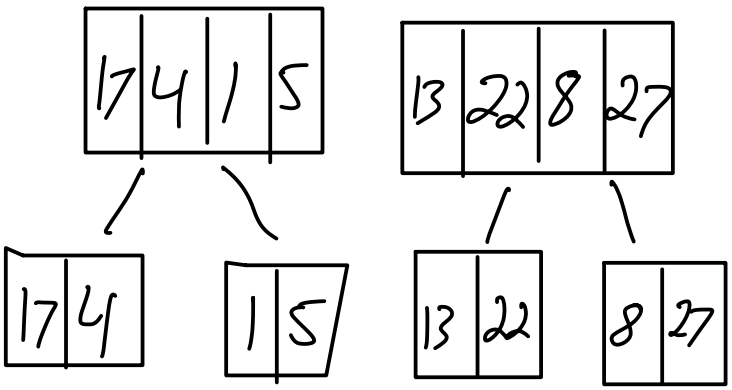
index	Animal
[0]	Cheetah
[1]	null
[2]	Porcupine
[3]	Bear
[4]	null
[5]	null
[6]	null
[7]	null
[8]	null
[9]	null
[10]	Cat
[11]	Rabbit
[12]	Dog
[13]	Tiger
[14]	null

9. Use the merge method to sort the data values 17, 4, 1, 5, 13, 22, 8, and 27 **show every step, and drawing. Showing just the answer will give you 2 points.**

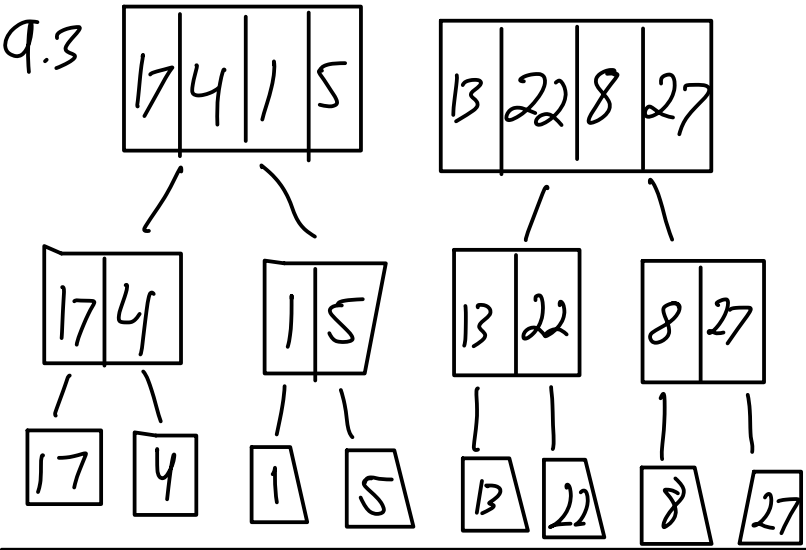
9.1



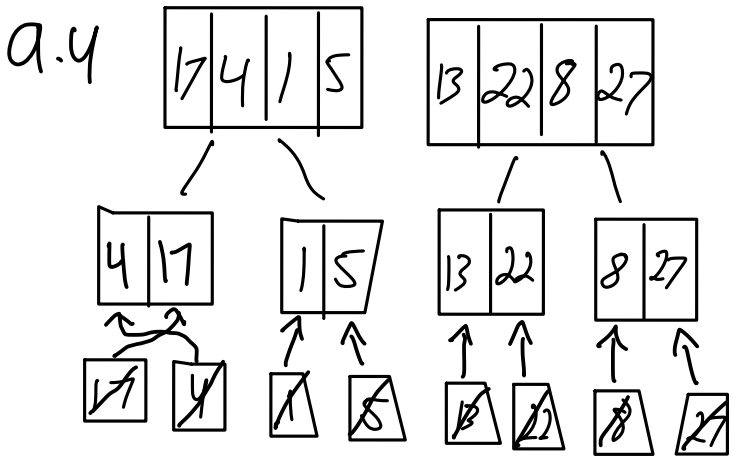
9.2



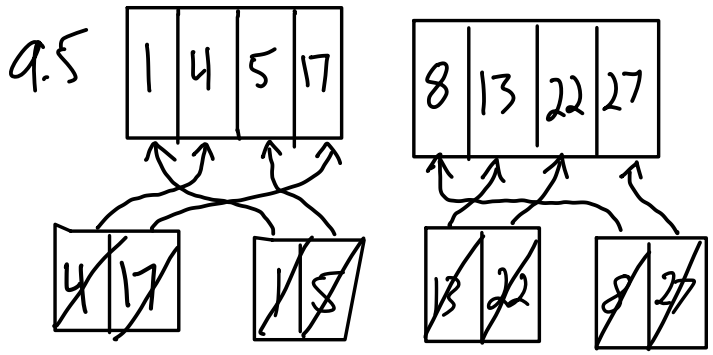
9.3



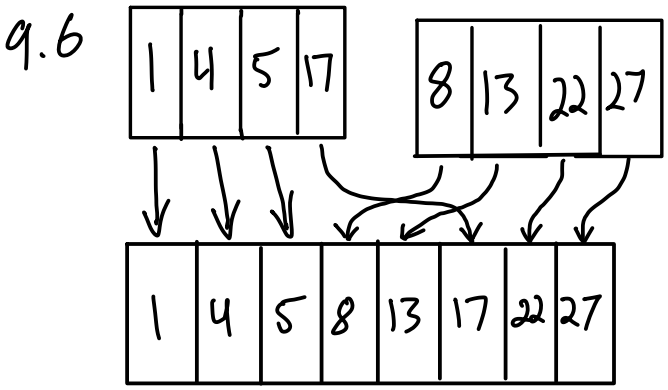
9.4



9.5



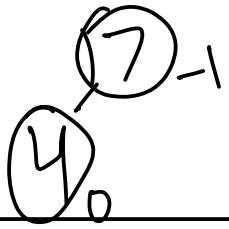
9.6



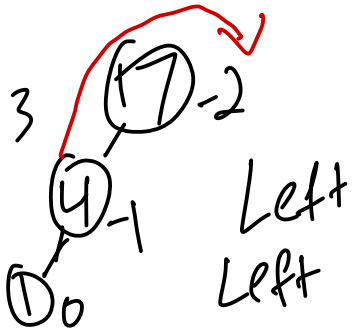
10. (10 points) Complete the AVL tree for 17, 4, 1, 5, 13, 22, 8, 27, 11, 3, 9, 10, 14, 6 make sure you show every step, drawing, and balance of each node using the $h_R - h_L$. **Show all your drawings, step by step to get full credit. Showing just the answer will give you 2 points. Just like in the lecture notes.**

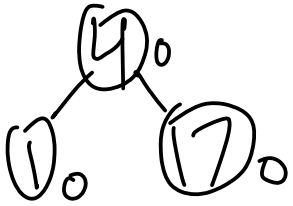
10.1

10.2

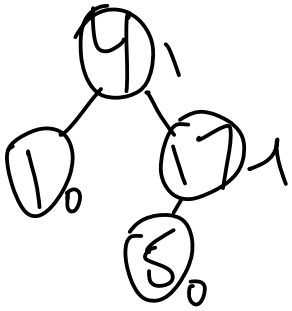


10.3

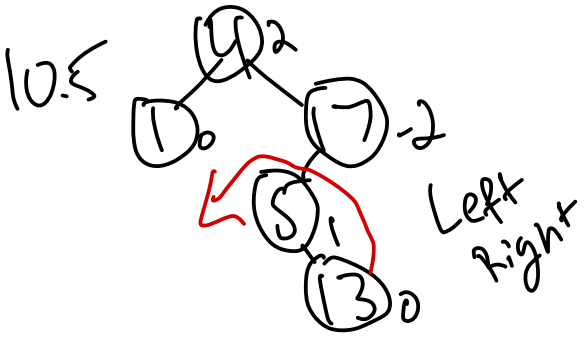


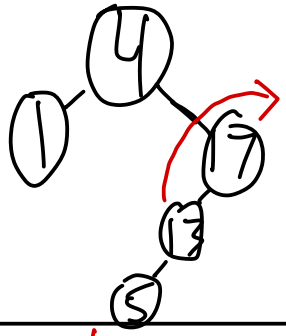


10.4

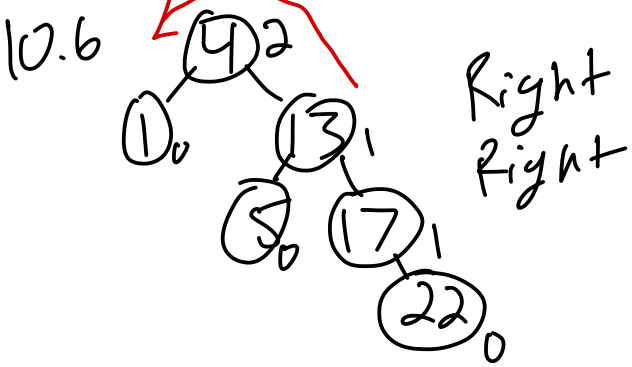


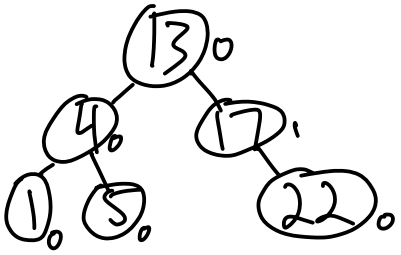
10.5



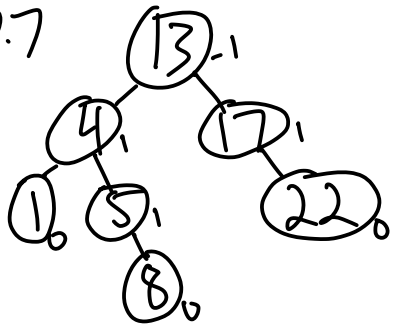


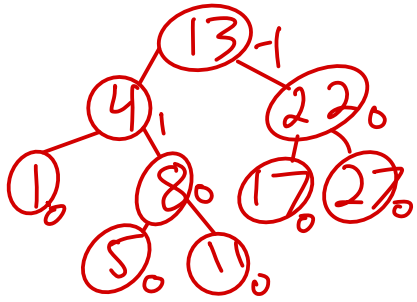
10.6



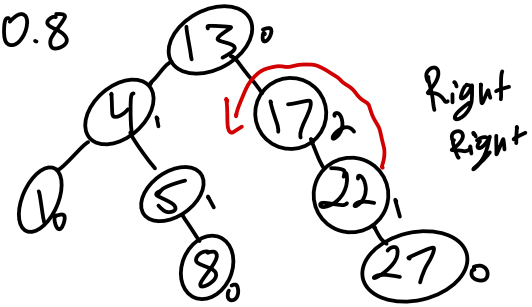


10.7

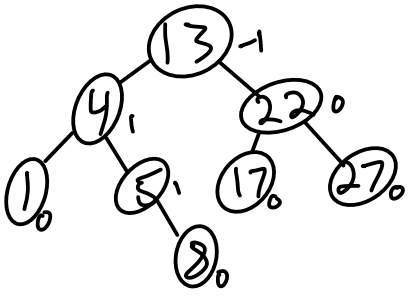




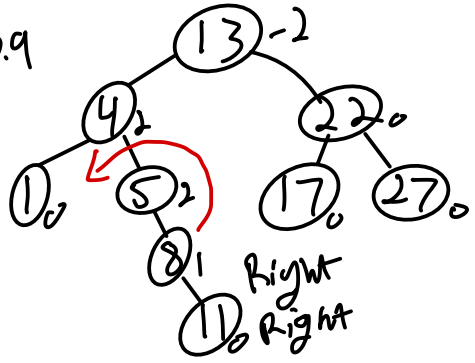
10.8



Right
Right

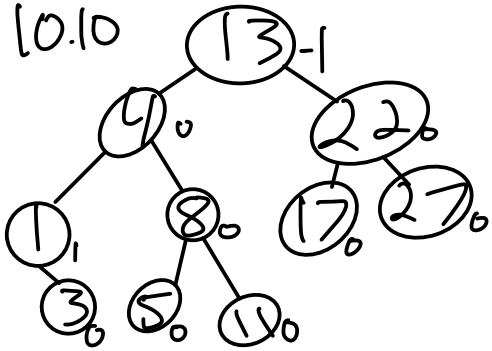


10.9

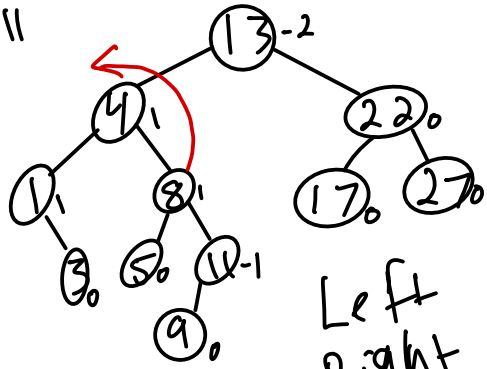


Right
Right

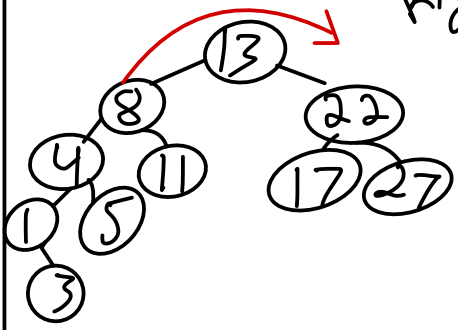
10.10



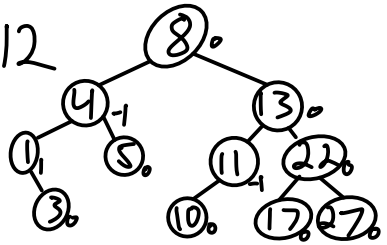
10.11



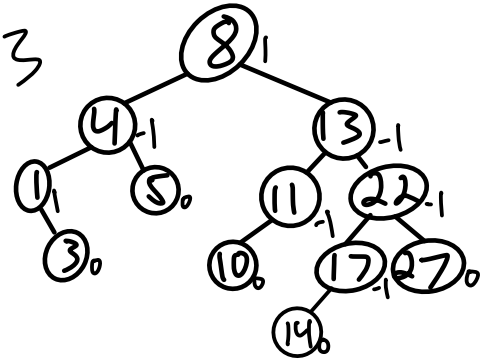
Left
Right



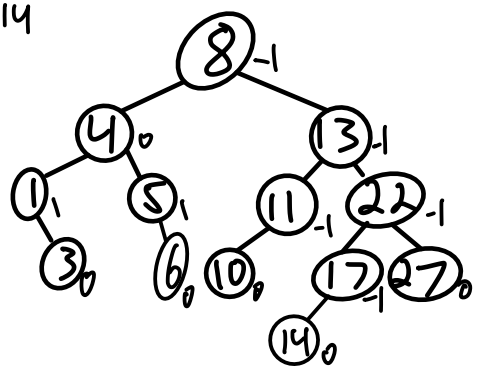
10.12



10.13



10.14



11. (10 points) **EXTRA CREDIT** Use Huffman coding to encode these symbols with given frequencies: A: 10, B: 25, C: 5, D: 15, E: 30, F: 7, G: 8. Show the Huffman code tree. 1) **Show all your drawings, step by step to get full credit. 2) Show your final Huffman binary tree with 0 and 1 with character symbols and no weight values. 3) Show the table of character symbols with its corresponding binary string. Just like in the lecture notes.**

11.1

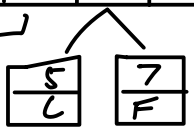
5	7	8	10	15	25	30
C	F	G	A	D	B	E

5+7=12

11.2

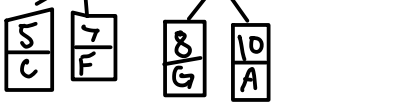
8	10	12	15	25	30
G	A	null	D	B	E

8+10=18



11.3

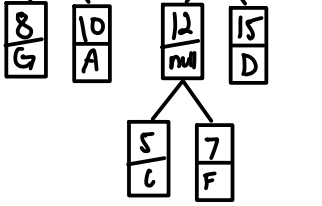
12	15	18	25	30
null	D	null	B	E



12+15=27

11.4

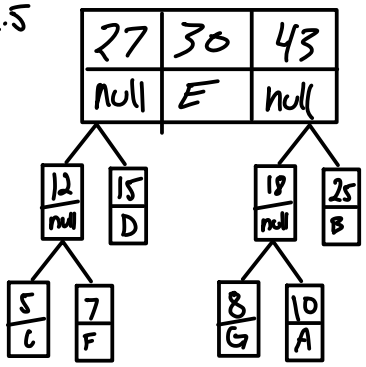
18	25	27	30
null	B	null	E



18+25=43

11.5

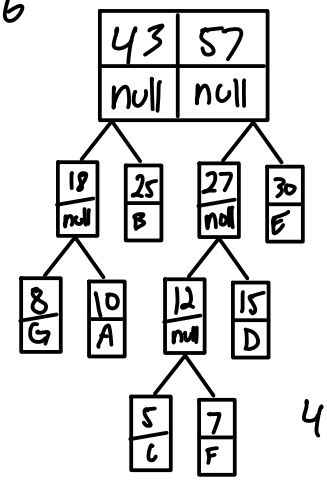
27	30	43
null	E	null



27+30=57

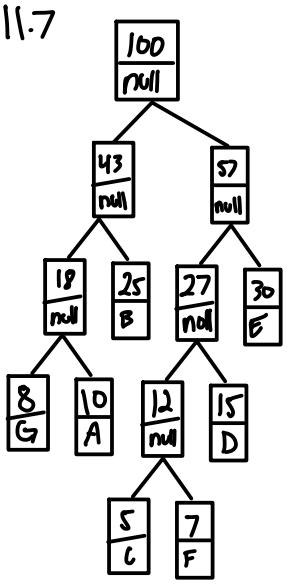
11.6

43	57
null	null

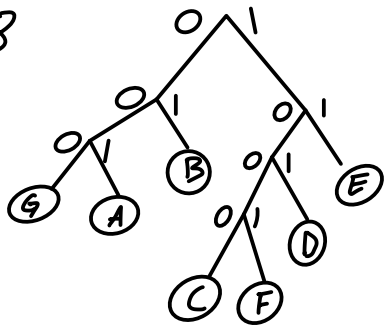


43+57=100

11.7



11.8



11.9

A=001
B=01
C=1000
D=101
E=11
F=1001
G=000