

CSC 345
Adam Cunningham

5.13)

a,b)

data: n

Parent pointer: n

children pointers: $2n$

overhead fraction: pointers / (data + pointers)

a):

data: $4n$

pointer (parents and children): $4(n+2n)$

total bytes: $16n$

overhead fraction: $(n+2n)/4n = 3/4$

b)

data: $16n$

children pointers: $4 \cdot 2n$

total bytes: $24n$

overhead fraction: $8n/24n = 1/3$

c)

data: $n \cdot 8$

parent pointer: $n \cdot 4$

child pointers (internal nodes $\cdot 2$): $(n-1) \cdot 4$

total bytes: $8n + 4n + 4n - 4$

$= 4(2n+n-1) = 4(4n-1)$

overhead fraction $= (n + (n-1))/(4n-1) = 2n+1 / 4n-1$

about $1/2$

d)

data (leaves): $((n-1)/2 - 1) \cdot 8$

child pointers (internal $\cdot 2$): $(n-1) \cdot 4$

total: $8(n-1)/2 - 8 + 4(n-1)$

$= 4(n-1) - 8 + 4(n-1) = 4((n-1) - 2 + (n-1)) = 4(2n-4) = 8(n-2)$

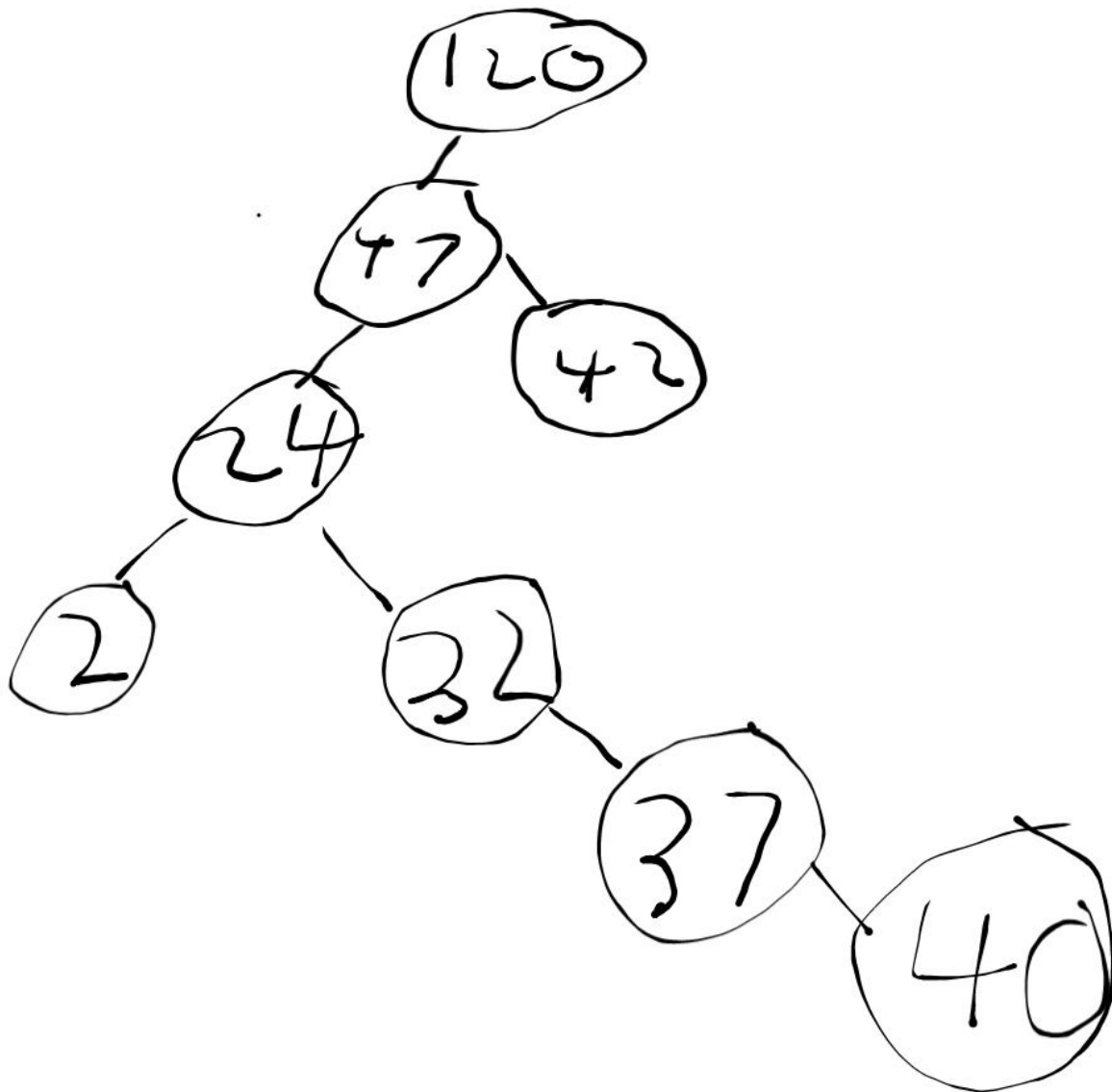
overhead fraction: $4(n-1) / 8(n-2) = 4((n-1)/(2n-4)) = (n-1) / 2(n-2)$

about $1/2$

5.17)NOTE:

child of 120 is supposed to be 42!

47 = 42!!!



5.32)

a)

4) unsorted array

no checks are needed to maintain sorted order, as the data will naturally remain sorted

b)

4) an unsorted array

an unsorted array has $O(1)$ insertion time,

given that there 100,000 more insertions than searches, it makes sense to insert at the end and just do 10 sequential searches

c)

BST

$n \log(n)$ time to construct

$\log(n)$ to insert

$\log(n)$ time to search (given that the tree will likely be balanced)

this is more ideal than the other options since we have the same (and significant) number of searches and insertions

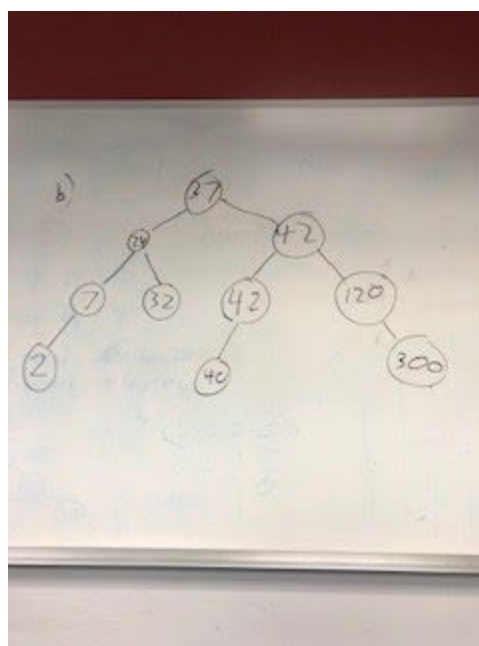
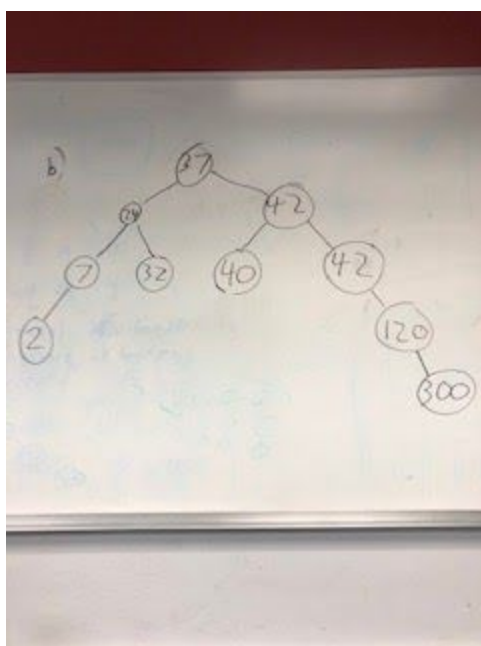
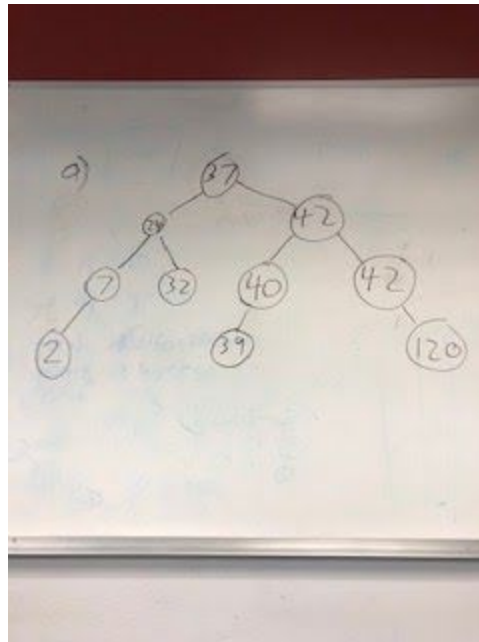
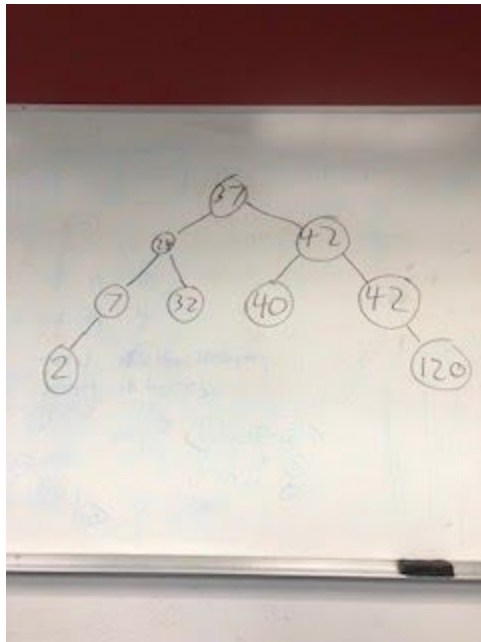
d)

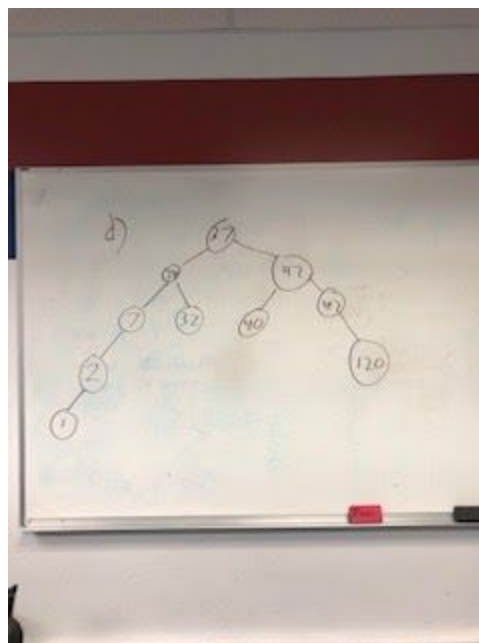
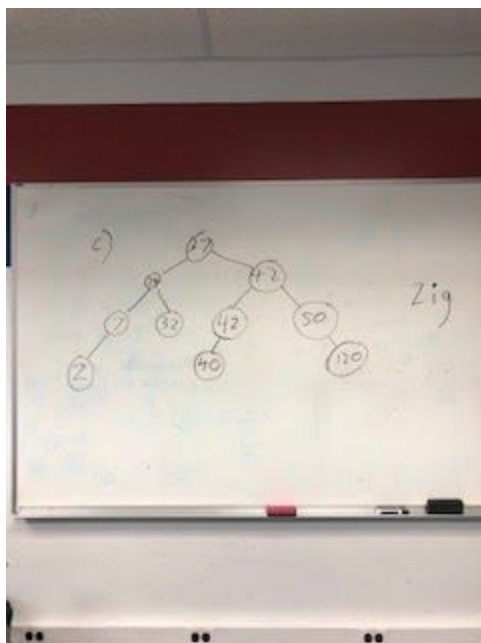
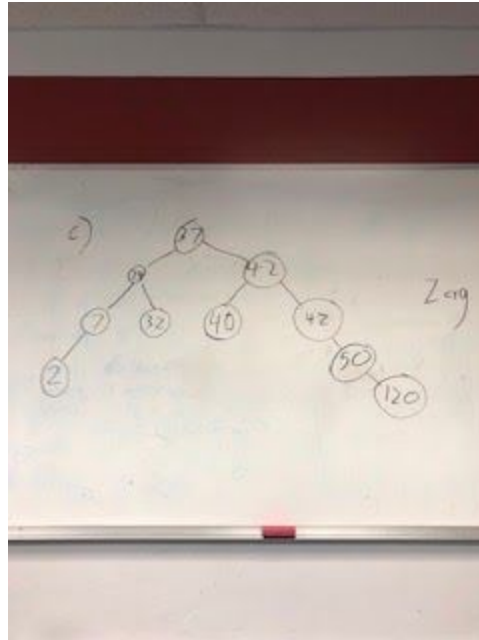
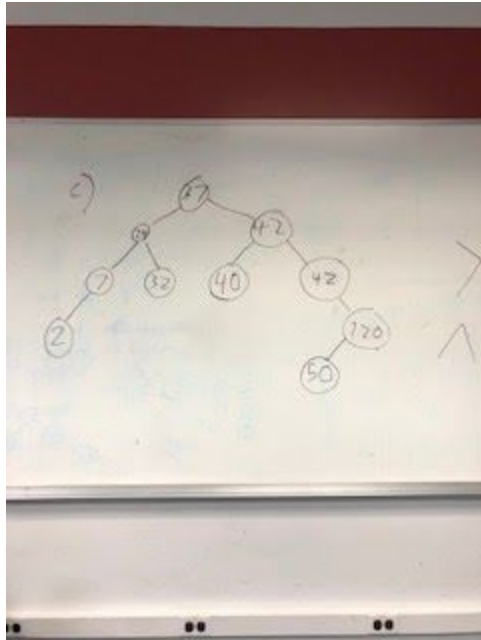
BST

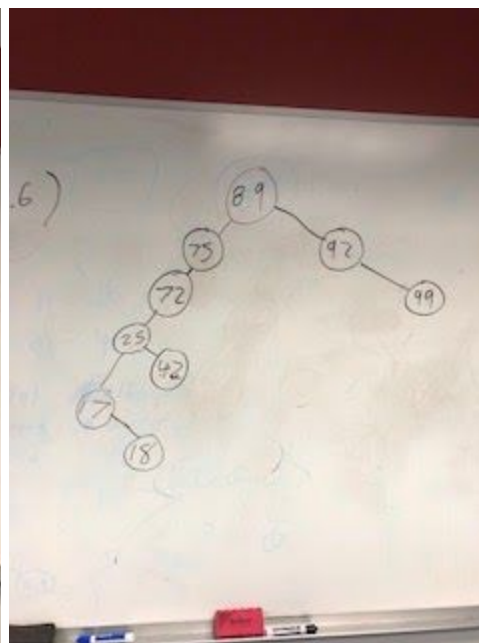
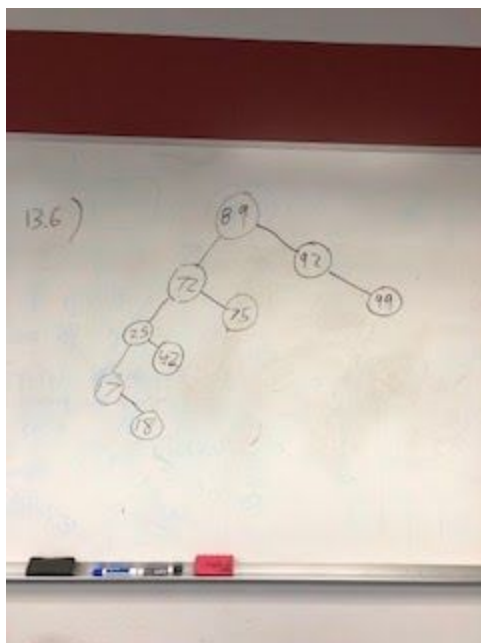
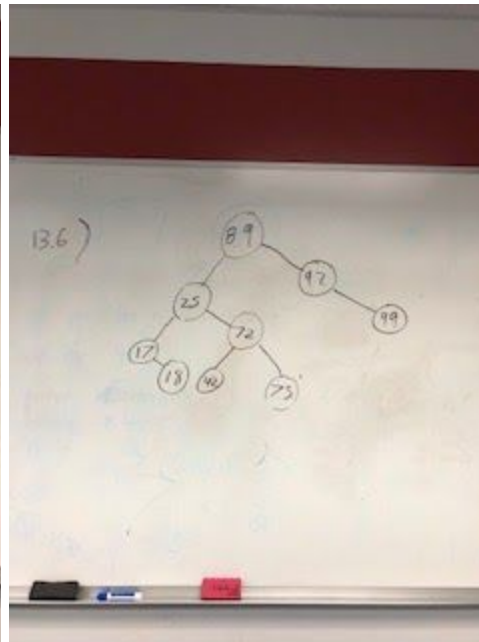
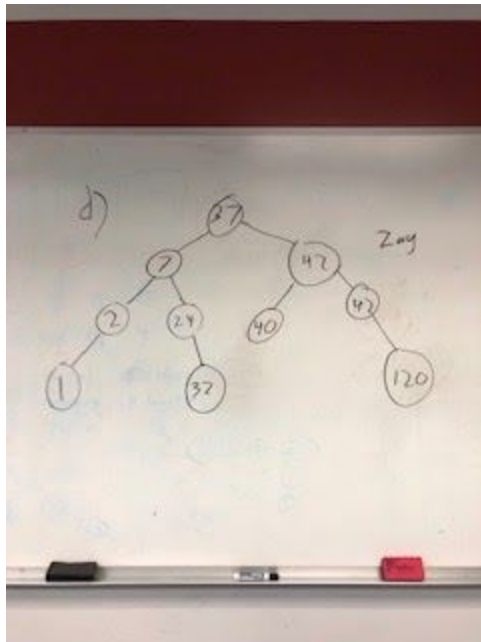
search time is competitive with sorted

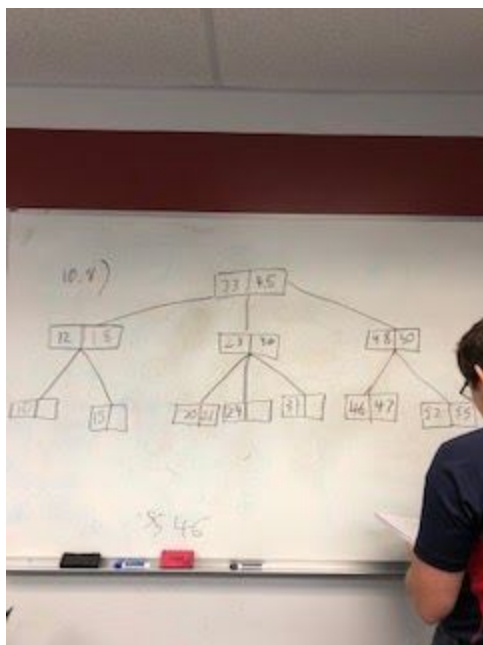
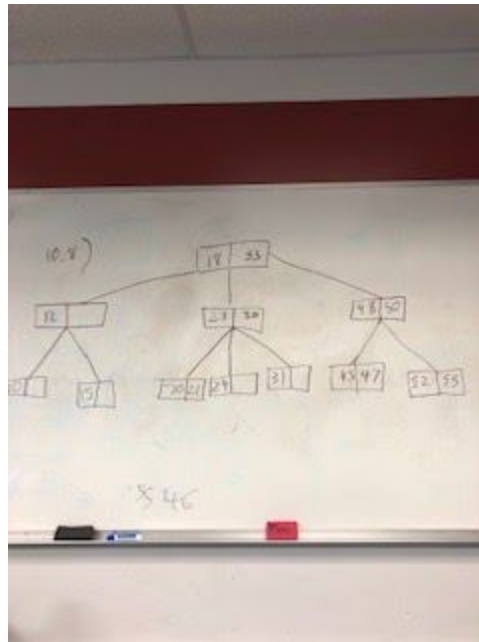
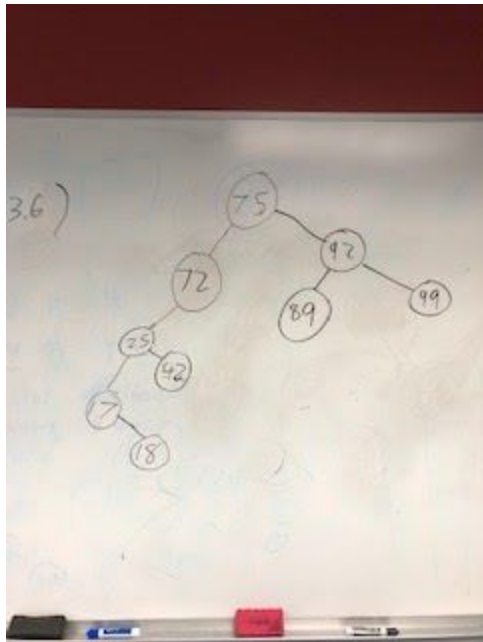
insertion time is $\log(n)$ vs n times faster than sorted

Following are pictures of all trees Steven and I assembled on the board yesterday
We had trouble with 10.8 but the others should be good









20, 39, 59, 79, 45, 9


8 > T
AVL
SPLAY

2-3

26

20, 39, 59, 79, 45, 53, 13

BST
AVL
SPLAY
2-3




```

graph TD
    20((20)) --> 39((39))
  
```

20, 39, 59, 79, 45, 53, 13

BST
AVL
SPLAY
2-3




```

graph TD
    20((20)) --> 39((39))
    20 --> 59((59))
  
```

20, 39, 59, 79, 45, 53, 13

BST
AVL
SPLAY
2-3




```

graph TD
    20((20)) --> 39((39))
    20 --> 59((59))
    59 --> 75((75))
  
```

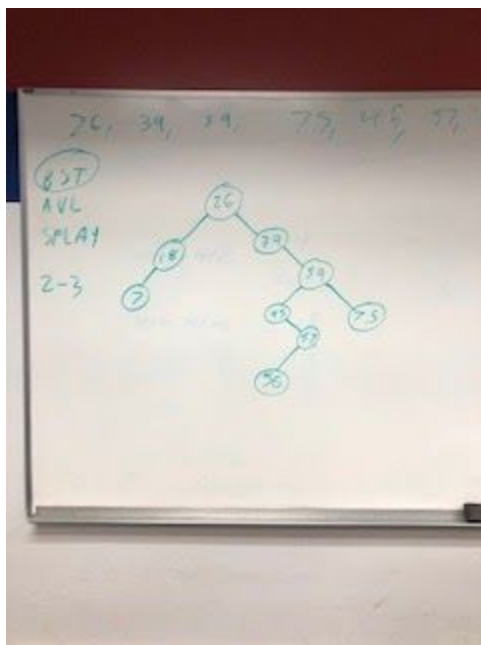
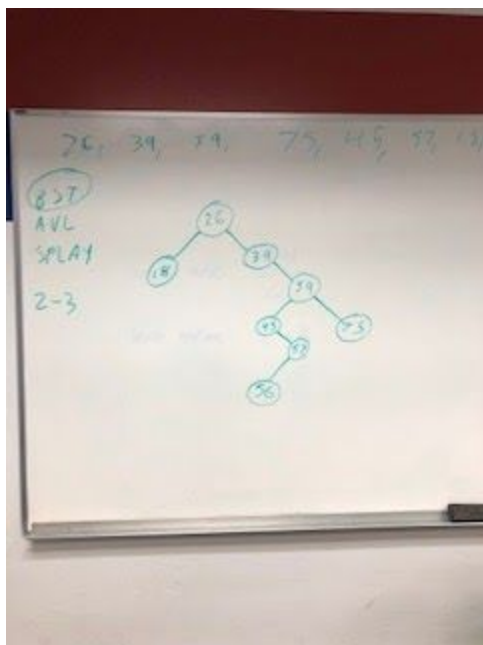
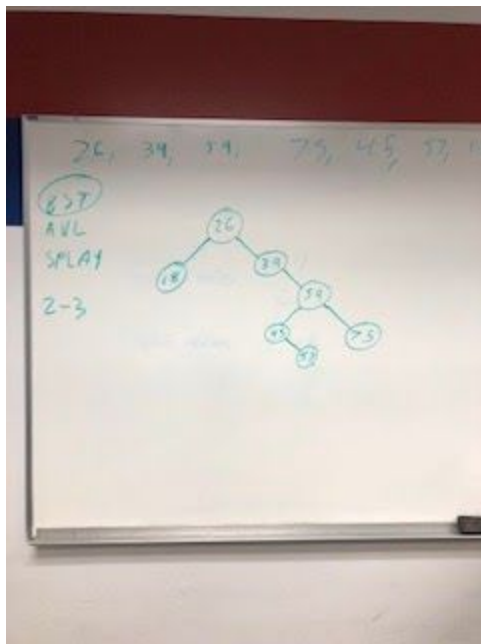
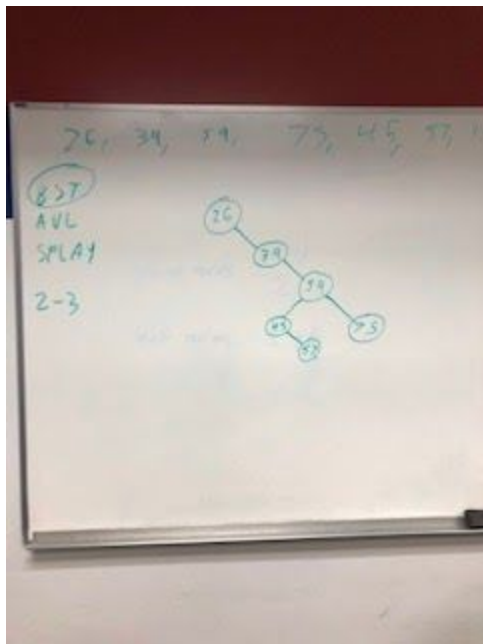
20, 39, 59, 79, 45, 53, 13

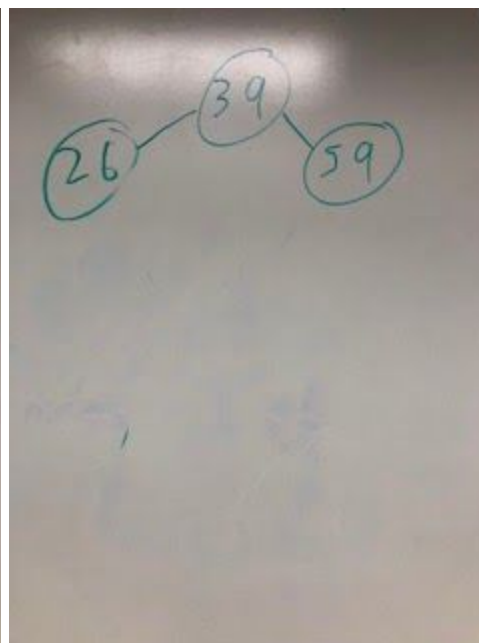
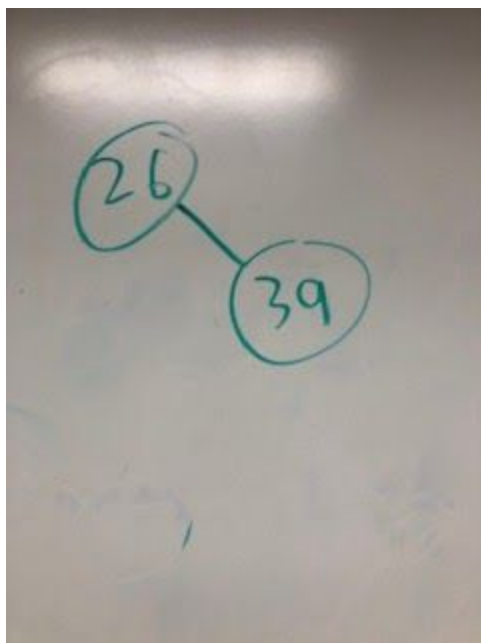
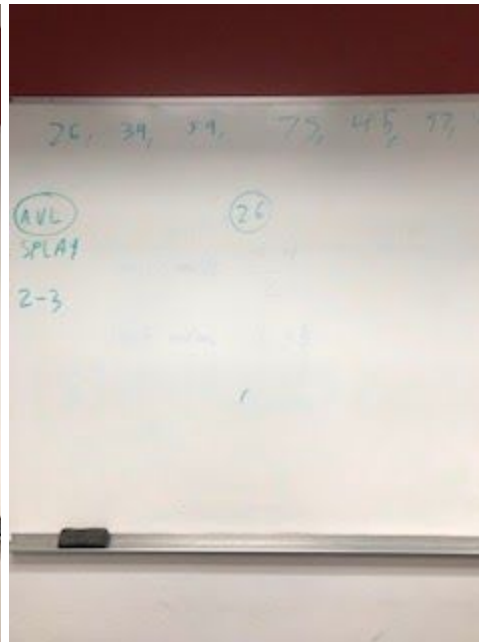
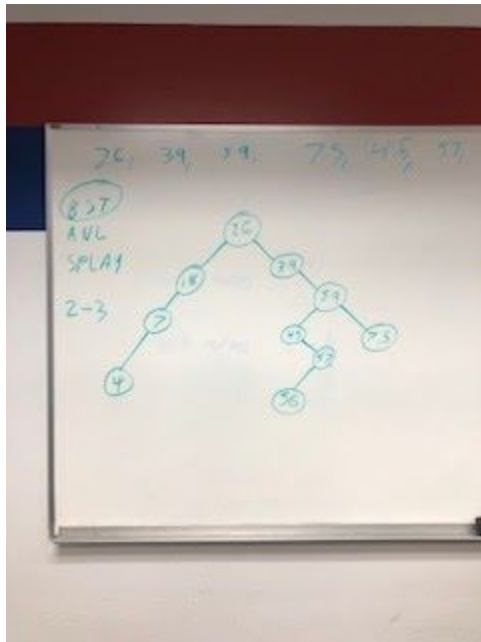
BST
AVL
SPLAY
2-3

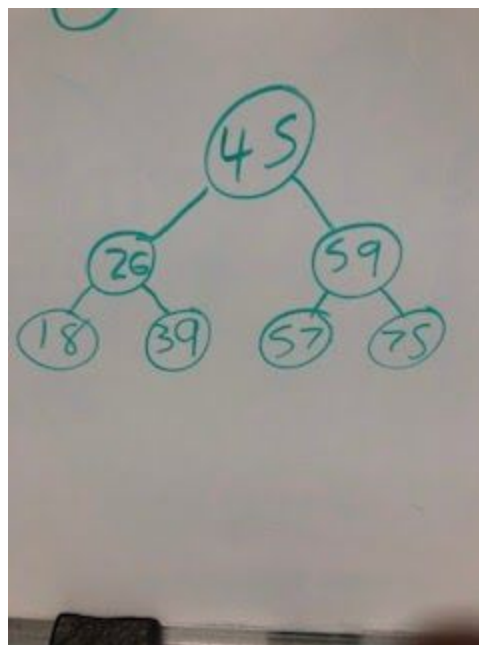
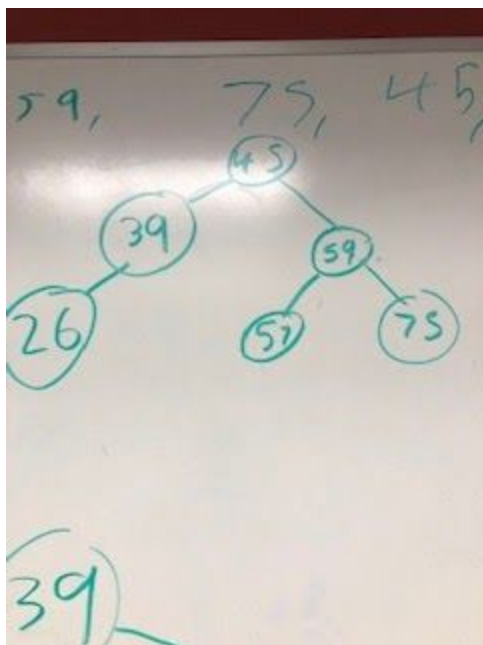
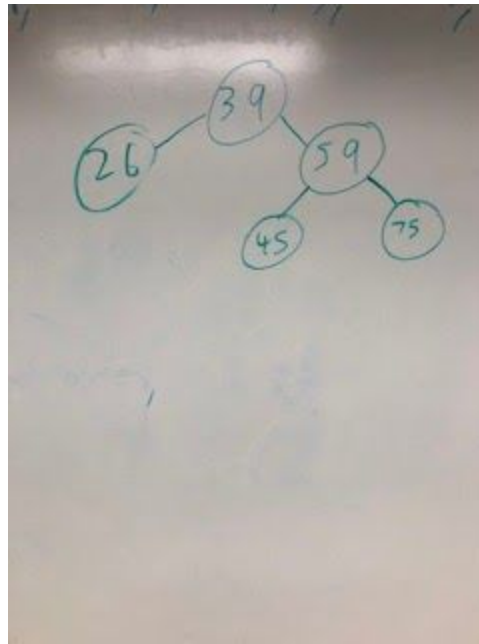
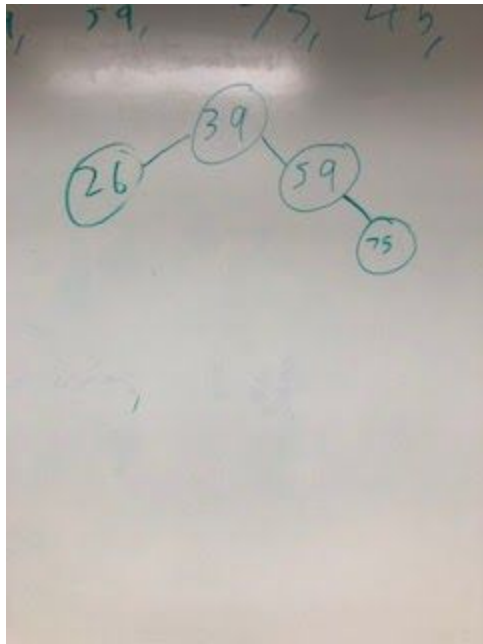


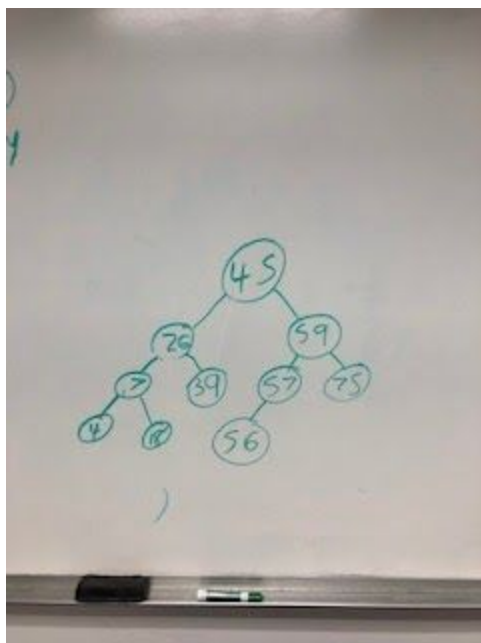
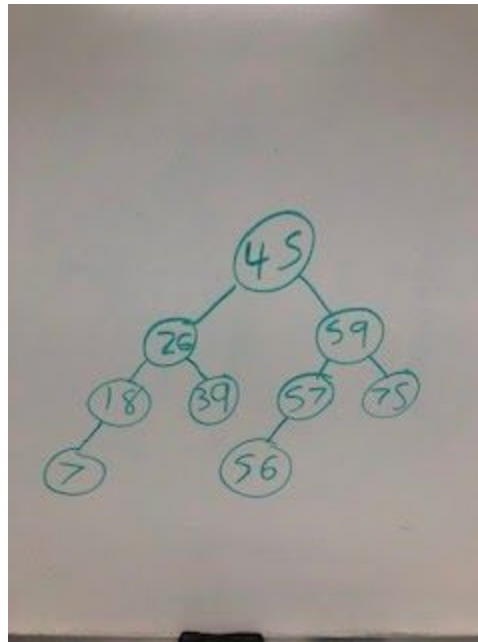
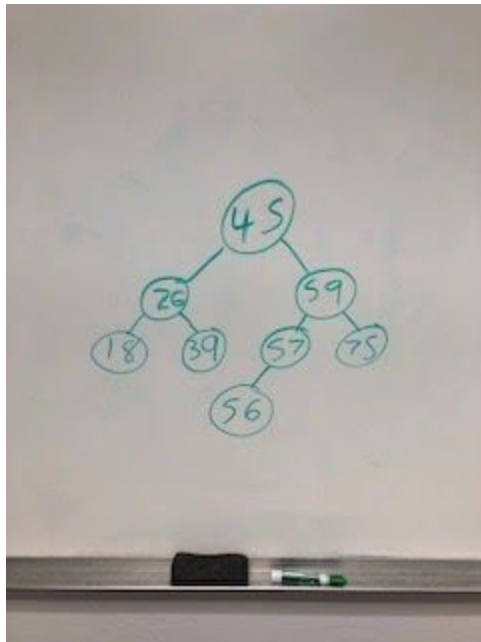
```

graph TD
    20((20)) --> 39((39))
    20 --> 59((59))
    39 --> 45((45))
    59 --> 75((75))
  
```







26, 39, 59,

SPLAY

2-3

26

