0) a) (4 pts) Determine the following sum in terms of n: $\sum_{i=n}^{2n}(2i-1)$ for positive integers n.

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

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

$$= 4n^2+2n-n^2+n-n-1$$

$$= \boxed{3n^2+2n-1}$$

b) (5 pts) An algorithm runs in O($\sqrt{n}$) time. When the algorithm is run with an input size of 14900, it takes 7 seconds to complete. How long will it take to complete on an input size of 59600?

$n=14,900$, $O(\sqrt{n})$ means there is some constant c such that

$$c\sqrt{14,900} = 7s \Rightarrow c = \frac{7s}{\sqrt{14,900}}$$

$$n = 59,600 \Rightarrow \frac{7s}{\sqrt{14,900}} \cdot \sqrt{59,600} = \frac{7s}{\sqrt{14,900}} \cdot 2\sqrt{14900} = \boxed{14s}$$

c) (5 pts) What is the (worst-case) run-time of the following segment of code in terms of n? Assume all variables are declared and initialized as integers. **Please give a Big-Oh bound and a proper justification for your answer.**

```
for (i=0; i<n; i++) {

  temp = n;
  while (temp > 0)
    temp = temp/2;
}
```

$\log_2 n$

n times

$\Rightarrow n\log_2 n$ times total

Note:
We know the inner loop will run K times where $\frac{n}{2^K} = 1 \Rightarrow$

$K = \log_2 n$

$O(n \log n)$

1) (5 pts) Consider a binary search in the array below for the value 28, which values are checked in the array before it is found? (Please list these in the order in which they are checked.)

	Low						Mid	Low		Mid	Low	Mid	High
index	0	1	2	3	4	5	6	7	8	9	10	11	12
value	2	3	7	10	11	15	18	20	22	26	27	28	30

$\text{Mid} = \dfrac{\text{Low} + \text{High}}{2}$

$\text{Mid} = \dfrac{0+12}{2} = 6 \Rightarrow \text{Low} = \text{Mid}+1$

$\text{Mid} = \dfrac{7+12}{2} = 9 \Rightarrow \text{Low} = \text{Mid}+1$

$\text{Mid} = \dfrac{10+12}{2} = 11$

18, 26, 28, ____, ____, ____, ____, ____

Note: More blanks than necessary are provided. Fill them in from the left. The last blank you fill in should have a 28 in it.

2) (10 pts) The permutation function is defined as follows: P(n, k) = n(n-1)(n-2)…(n-k+1), for positive integers n and k, with 1 ≤ k ≤ n. Note that $\boxed{\text{P(n, 1) = n.}}$ Write a recursive function that takes in integer parameters n and k and returns P(n, k) below.

Example: Perm(n,4) = n(n-1)(n-2)(n-3)

```
int perm(int n, int k) {

    if(K==1)
        return n;

    return (n-K+1)* perm(n, k-1);

}
```

4

Check our answer: perm(n, 4) → (n-4+1) * perm(n,3)

perm(n, 3) → ((n-3)+1) * perm(n,2)

perm(n,2) → (n-2+1) * perm(n,1)

perm(n,1) → n

n(n-1)(n-2)(n-3) ✓

3) (8 pts) Convert 137_8 to base 5. (Remember you need to use a two step process.)

Step 1: Convert to Decimal

$137_8 = 1 * 8^2 + 3 * 8^1 + 7 * 8^0$

$= 64 + 24 + 7 = 88 + 7 = \boxed{95_{10}}$

Step 2: Convert Decimal to Base 5

$5 \overline{)95} = 19$ Remainder 0

$5 \overline{)19} = 3$ Remainder 4

$5 \overline{)3} = 0$ Remainder 3

STOP

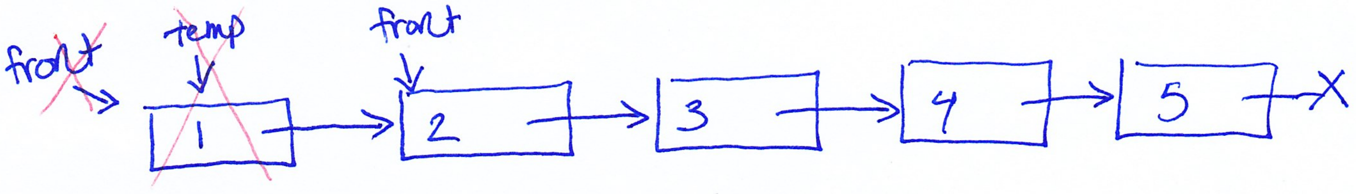
$\boxed{\boxed{340_5}}$ ← Final Answer

Step 3: Check our answer

$3 \cdot 5^2 + 4 \cdot 5^1 + 0 \cdot 5^0 = 75 + 20 + 0 = \boxed{95_{10}}$ ✓

4) (10 pts) Write a **recursive function** that deletes every other node in a linked list pointed to by the pointer front, which is passed in as a parameter. In particular, make sure you delete the first, third, fifth, etc. nodes and return a pointer to the new list. If the list has zero or one item in it, NULL should be returned.

```
struct ll* {
    int data;
    struct ll* next;
};

struct ll* delEveryOther(struct ll* front) {

    if( front == NULL || front -> next == NULL)
            return NULL;

    struct ll * temp = front;
    front = temp -> next;
    free( temp);
    front -> next = del Every Other( front -> next);

    return front;

}
```



5) (10 pts) What is the return value of running the function call question5(T), where T is a pointer to the tree below. The function question5 is also defined below. Please show your work so you can earn partial credit.

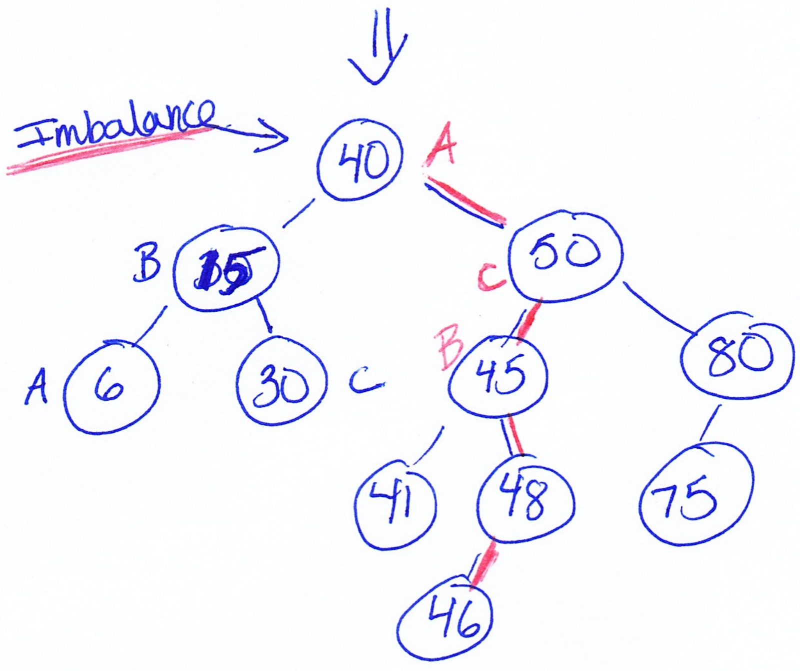
```
                    30 +1
                  /       \
             1 15           43 +3
              /  \          /
         1 6    22 3      36 +1            =  44
          / \      \      / \
      5 3   8 5   1 25  +5 32  41 +3
                   / \           \
               5 23   28 5        42 +5
```

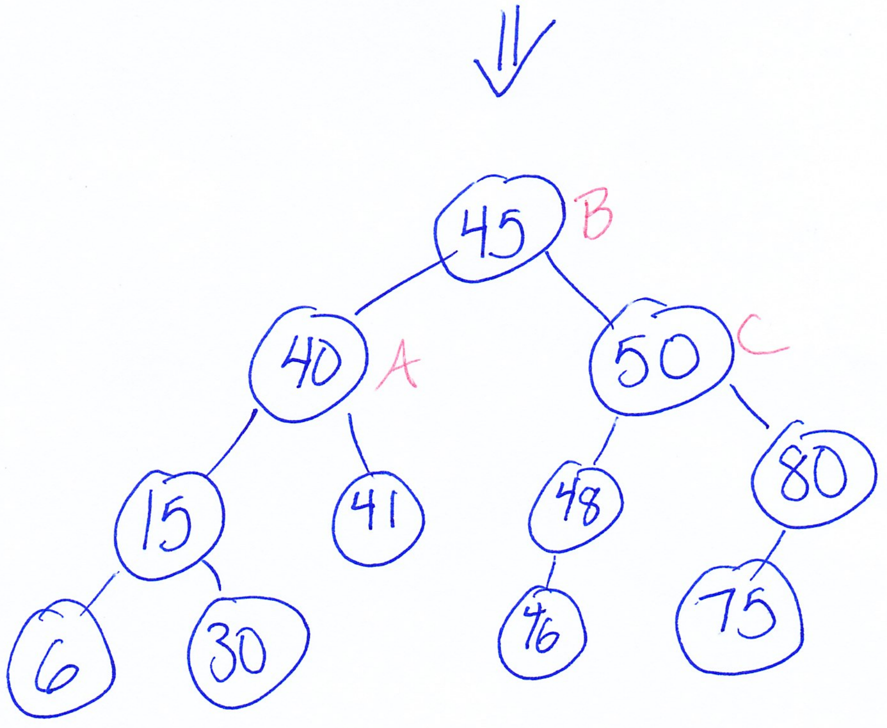
```
struct tree_node {
    int data;
    struct tree_node* left;
    struct tree_node* right;
};

int question5(struct tree_node *root) {

    int sum = 0;
    if (root != NULL) {
        if (root->left == NULL && root->right == NULL)
          sum += 5;
        else if (root->left == NULL || root->right == NULL)
          sum += 3;
        else
          sum++;
        sum += question5(root->right);
        sum += question5(root->left);
    }
    return sum;
}
```

Return Value 44

6) (5 pts) Show the result of deleting 35 from the AVL Tree below:

```
                                40
imbalance -->        /                    \
             C 30                         50
               /    \                  /      \
       B 15         ~~35~~          45          80
          /                         / \         /
   A  6                         41  48      75
                                     /
                                   46
```





Final is Balanced

7) (5 pts) Evaluate the following postfix expression showing the contents of the stack in each of the positions (A and B) indicated.

6 8 2 + * A 7 3 – / B 10 + 3 –

A
7
60

B
10
15

Value of Postfix expression: 22

$25 - 3 = 22$
~~3~~
~~25~~
~~10~~
~~$60/4 = 15$~~
~~$7 - 3 = 4$~~
~~3~~
~~7~~
~~$6 * 10 = 60$~~
~~$8 + 2 = 10$~~
~~2~~
~~8~~
~~6~~

Working stack

Master Theorem: Given $T(n) = A\,T(n/B) + O(n^k)$

Then $T(n) =$ $O(n^{\log_B A})$ if $A > B^k$
$O(n^k \log n)$ if $A = B^k$
$O(n^k)$ if $A < B^k$

8) (6 pts) Solve the following three recurrences using the Master Theorem:

a) $T(n) = 8T(n/3) + O(n^2)$ $\boxed{O(n^2)}$

$A = 8,\ B = 3,\ k = 2$ $8 < 3^2$ Case 3

b) $T(n) = 8T(n/2) + O(n^2)$ $\boxed{O(n^3)}$

$A = 8,\ B = 2,\ k = 2$ $8 > 2^2$ case 1 $O(n^{\log_2 8})$

c) $T(n) = 25T(n/5) + O(n^2)$ $\boxed{O(n^2 \log n)}$

$A = 25,\ B = 5,\ k = 2$ $25 = 5^2$ case 2

9) (5 pts) Show the contents of the following array after each pass of an insertion sort.

Index	0	1	2	3	4	5	6
Start	7	2	6	1	5	3	4
	2	7					
	2	6	7				
	1	2	6	7			
	1	2	5	6	7		
	1	2	3	5	6	7	
End	1	2	3	4	5	6	7

10) (5 pts) Consider running the partition function on the array below (using index 0 as the partition element) in the manner shown in class. Show the final result of partitioning this array.

						High	Low		
	Pivot	Low →	Low ⟶			Low	High ←	High ←	High
Index	0	1	2	3	4	5	6	7	8
Original	6	3	~~9~~ 4	5	1	~~12~~ 2	~~2~~ 12	~~4~~ 9	15
Answer	2	3	4	5	1	6	12	9	15

11) (5 pts) Consider a hash table that uses the quadratic probing technique with the following hash function f(x) = (3x+4)%12. (The hash table is of size 12.) If we insert the values 5, 8, 3, 2 and 6, in that order, show where these values would end up in the table?

Index	0	1	2	3	4	5	6	7	8	9	10	11
Value		3			8			5			2	6

$f(5) = 15 + 4 \% 12 = 7$

$f(8) = 24 + 4 \% 12 = 28 \% 12 = 4$

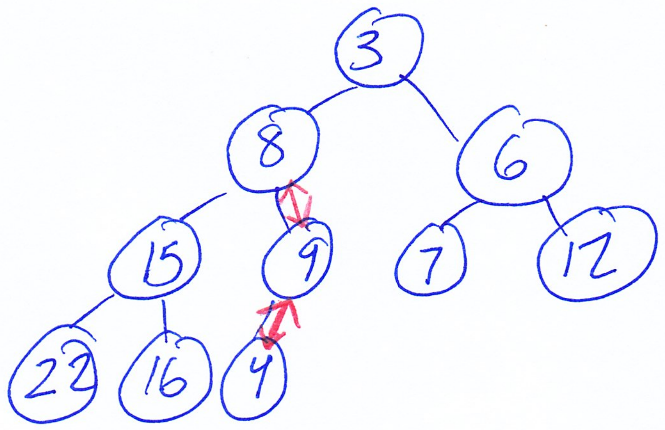
$f(3) = 9 + 4 = 13 \% 12 = 1$

$f(2) = 6 + 4 = 10$

$f(6) = 18 + 4 = 22 \% 12 = \underset{\substack{\uparrow \\ \text{taken}}}{10}$, try $10 + 1 = 11$

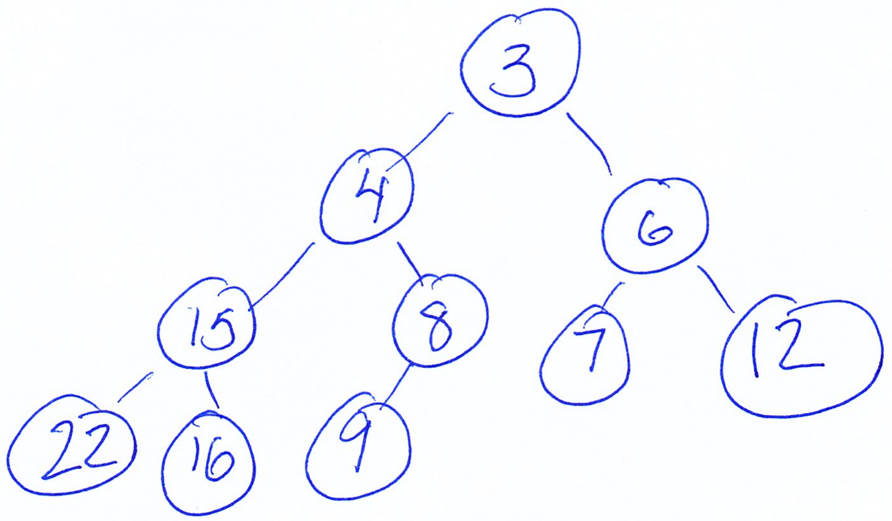
12) (5 pts) Draw the heap corresponding to the following array:

Index	1	2	3	4	5	6	7	8	9
Value	3	8	6	15	9	7	12	22	16



What you notice is that if you fill each row of the heap going from left to right with each element in the array, you meet the rule that the children of element at index i are at index 2i and 2i+1.

13) (5 pts) Show the result of inserting the value 4 into the heap from the previous question.



14) (10 pts) Use backtracking to get out of the following maze shown below. Your goal is to start at the S and end at the square marked F. You are not allowed to go to squares marked with X's. Show all paths attempted, and finally draw in the actual path taken a bit bolder. Any time you are given a choice of paths, take them in the following order: UP, RIGHT, DOWN, LEFT. (These are the only four directions you can move.) Also, remember not to go to squares you have previously traversed.

S				X					
	X	X				X	X	X	X
	X	X	X	X					
	X	X		X		X	X	X	X
	X	X		X		X	X	X	
			X			X	X	X	
	X	X	X	X	X	X	X	X	
					X	X		X	
	X		X		X	X		X	
	X		X						F