1. The terminology $f(n) = O(n^2)$ is equivalent to saying $f(n)$ is of order $n^2$ or saying $f(n)$ is of complexity $n^2$. For each problem, (1) Find the complexity (2) select an appropriate picture from the list below (A-F) (or draw one of your own) to justify your answer.

a. for (int k = 0; k<n;k++)
   for (int j = 0; j < k; j++)  // Notice j ends at k (not n)
   x = x++; 

b. void doit(int n)
   { if (n <=0) return;
   doit(n/2);
   doit(n/2);
   for (int i=0;i<n;i++)
   cout << i;
   }

c. void doit(int n)
   { if (n <=0) return;
d. int doit(int look, int beg, int end)
   { if (beg > end) return -1;
     mid = (beg + end)/2;
     if (A[mid] == look)
       return mid;
     if (look < A[mid])
       return doit(look, beg, mid-1);
     return doit(look, mid+1, end);
   }

2. Consider the following sample data. For each set of timing information, indicate the complexity of the algorithm.

   a.
   \[
   \begin{array}{|c|c|}
   \hline
   n & T(n) \\
   \hline
   2 & 5 \\
   4 & 5 \\
   8 & 5 \\
   16 & 5 \\
   32 & 5 \\
   \hline
   \end{array}
   \]

   b.
   \[
   \begin{array}{|c|c|}
   \hline
   n & T(n) \\
   \hline
   2 & 5 \\
   4 & 10 \\
   8 & 20 \\
   16 & 40 \\
   32 & 80 \\
   \hline
   \end{array}
   \]

   c.
   \[
   \begin{array}{|c|c|}
   \hline
   n & T(n) \\
   \hline
   2 & 5 \\
   4 & 10 \\
   8 & 15 \\
   16 & 20 \\
   32 & 25 \\
   \hline
   \end{array}
   \]

   d.
   \[
   \begin{array}{|c|c|}
   \hline
   n & T(n) \\
   \hline
   2 & 4 \\
   4 & 16 \\
   8 & 256 \\
   16 & 65,536 \\
   32 & 4,294,967,296 \\
   \hline
   \end{array}
   \]
3. Suppose that \( T_1(N) = O(F(N)) \) and \( T_2(N) = O(F(N)) \). Note that if something is \( n^2 \), the run time may actually be \( 10n^2 + 47n + 15 \) as you throw out constants and ignore lesser order terms.

Which of the following are true:

a. \( T_1(N) + T_2(N) = O(F(N)) \)

b. \( T_1(N) - T_2(N) = O(F(N)) \)

c. \( T_1(N)/T_2(N) = O(1) \)

d. \( T_1(N) = O(T_2(N)) \)

4. Program A and B are analyzed and are found to have worst-case running times no greater than \( O(n \log n) \) and \( O(n^2) \), respectively. Answer the following questions if possible.

a. Which program has the better guarantee on the running time for large values of \( N \) (\( N > 10,000 \))?

b. Which program has the better guarantee on the running time for small values of \( N \) (\( N < 100 \))?

c. Which program will run faster on average for \( N = 1000 \)?

d. Can program B run faster than program A on all possible inputs?

5. What is the order of each of the following tasks in the worst case (using the best algorithm)?

a. Computing the sum of the first \( n \) even integers by using a for loop.

b. Displaying all \( n \) integers in a sorted linked list.

c. Displaying the last element in a linked list.

d. Searching an array of \( n \) integers for a particular value by using sequential search.

e. Searching an array of \( n \) integers for a particular value by using a binary search.

f. Adding an item to a stack of \( n \) items.

g. Adding an item to a queue of \( n \) items.

---

e. 

\[
\begin{array}{|c|c|}
\hline
n & T(n) \\
\hline
2 & 10 \\
4 & 16 \\
8 & 48 \\
16 & 128 \\
32 & 320 \\
\hline
\end{array}
\]

f. 

\[
\begin{array}{|c|c|}
\hline
n & T(n) \\
\hline
2 & 4 \\
4 & 15 \\
8 & 68 \\
16 & 270 \\
32 & 1024 \\
\hline
\end{array}
\]
6. An implementation of an algorithm has the following loop structure. What is the complexity of the algorithm?
   for ( int pass = 1; pass <= n; ++pass )
       for ( int index = 0; index < n; ++index )
           for ( int count = 1; count < 10; ++count )
               x++;

**Formula Approach**
The following theorem gives the mathematical computation for what we have been analyzing visually. This formula approach for computing complexity is only appropriate for recursion. Let:
- \( T(n) \) – amount of time to solve a problem of size \( n \).
- \( a \) – Number of recursive calls made.
- \( b \) – Number \( n \) is divided by in the recursive calls.
- \( k \) – The exponent on \( n \) which represents the amount of work in a single call.

Theorem:
- \( T(n) = aT(n/b) + O(n^k) \).
- If \( a > b^k \) then \( T(n) \) is \( O(n^{\log_b a}) \).
- If \( a = b^k \) then \( T(n) \) is \( O(n^k \log n) \).
- If \( a < b^k \) then \( T(n) \) is \( O(n^k) \).

7. According to the previous theorem, what is the complexity of the following piece of code?

```c
void doit (n){
    if (n <=1) return;
    doit(n/2);
    doit(n/2);
    doit(n/2);
    for(i=0;i<n;i++) x++;
}
```

8. According to the previous theorem, what is the complexity of the following piece of code?

```c
void doit (n){
    if (n <=1) return;
    for(i=0;i<n;i++)
        for (j=0; j < n; j++) x++;
    doit(n/2);
}
```

9. A problem with complexity \( O(n) \) doubles in size, what would you expect would happen to the execution time?
   a. Remains unchanged
b. Increases by a constant
c. Doubles
d. More than doubles
e. Quadruples (increases by four times)
f. Is squared

10. A problem with complexity \(O(1)\) doubles in size, what would you expect would happen to the execution time?
   a. Remains unchanged
   b. Increases by a constant
c. Doubles
d. More than doubles
e. Quadruples (increases by four times)
f. Is squared

11. A problem with complexity \(O(n^2)\) doubles in size, what would you expect would happen to the execution time?
   a. Remains unchanged
   b. Increases by a constant
c. Doubles
d. More than doubles
e. Quadruples (increases by four times)
f. Is squared

12. For the AVL tree below, which of the following are correct? Which are incorrect?
   a. If a key with value 42, 29, or 27 is inserted in the tree, it does not lose the AVL property.
   b. After the insertion of 20, a single rotation is necessary to restore the AVL property of the tree.
   c. A double rotation is necessary to restore the AVL property after the insertion of 20. This makes 16 the new root of the tree.
   d. A single rotation is needed to restore the AVL property if any node with value less than 14 is added to the present tree.

   14
   /   
  12   16
   /   
  14  40

13. An AVL tree of height 4 must have at least _____ nodes
   e. 21
   f. 7
   g. 14
   h. 12
   i. None of the above.

14. Consider the following AVL tree. Show the tree after insertion of 17 followed by deletion of 28.
15. Show the results of inserting items 1 through 8 in order in an initially empty AVL tree.
16. In the following AVL tree, show the tree after deletion of 15.

17. Show an AVL tree for which an addition will cause multiple rotations. We count a double rotation as one rotation not two (it’s just fancier).