Part I: Foundations (28 Points)

Question 1 (3 Points): Conte di Savoia on Taylor Street (between Racine and Ashland) sells 6 different kinds of pastries. How many different pastry trays can they make that have 20 pastries, with at least 3 of those 20 being cannoli?

Note that the order of the pastries on the tray does not make a tray different, but having different numbers of pastries does. (I.e., “10 apple strudels and 10 cannoli” is the same tray as “10 cannoli and 10 apple strudels”, but “11 apple strudels and 9 cannoli” is different.

You may use exponents and factorial symbols in your answer; you do not need to simplify to a single integer.

Question 2 (5 Points): Suppose there are 10 students in a class and that each will receive one of the following grades: A, B, C, D, F. For each of the following you need not give a final numerical answer; an equivalent expression is sufficient.

Part I: How many distinct ways are there for the instructor to assign the grades?

Part II: Now suppose the Dean has stated that at least two of the students must receive F’s. With this added constraint on the instructor, how many distinct grade assignments are there?

Question 3 (4 Points): Suppose you have an alphabet with \( k \) characters. How many strings of length \( n \) from this alphabet are palindromes?

Question 4 (5 Points):

Part I: Define what it means for a binary relation \( \rho \) to be symmetric.

Part II: Now recall that a binary relation \( \rho \) is antisymmetric if the following holds:

\[
(\forall x)(\forall y)(x \in S \land y \in S \land (x, y) \in \rho \land (y, x) \in \rho \rightarrow x = y)
\]

Give an example of a nontrivial relation that is neither symmetric nor antisymmetric. (Think about the definitions carefully).

Question 5 (3 Points): Write a truth table for the following logical expression.

\[
A \lor B' \rightarrow (A \lor B)'\]
Question 6 (2 Points): Draw a Venn Diagram (shaded circle diagram) for the following set.

\[ \{x | ((x \in A) \text{ or } (x \in B)) \text{ and } x \notin A \cap B \} \]

Question 7 (5 Points): A complete ternary tree is defined as a tree in which every non-leaf node has exactly 3 children and every leaf is the same distance from the root. The height of a complete ternary tree is defined as the distance (number of links) from the root to a leaf. A ternary tree with just one node has height 0.

Prove the following by mathematical induction: “a complete ternary tree of height \( h \) has a total of \( \frac{3^{h+1}-1}{2} \) nodes.”
Part II: Data Structures and Algorithms (52 pts)

Question 1 (8 Points): Which of the following problems is known to be solvable in running time \(O(n^3)\)? (Give a list of the problem numbers.)

1. Finding the longest simple path from a given start vertex to a given end vertex in a directed acyclic graph on \(n\) vertices with nonnegative integer weights.
2. Finding the shortest path from a given start vertex to a given end vertex in a directed graph on \(n\) vertices with arbitrary integer weights.
3. Finding the longest simple path from a given start vertex to a given end vertex in a directed graph on \(n\) vertices with nonnegative integer weights.

Question 2 (5 Points): True or False: Circle your final answers.

- T  F  If an algorithm runs in \(O(n)\) time on the worst case input then it runs in \(O(n)\) time on every input.
- T  F  Running BFS on a directed unweighted graph with cycles will produce shortest paths from the source vertex to all other reachable vertices (where path length is simply the number of edges).
- T  F  \(\log(n!) = o(n^2)\)
- T  F  Running DFS on an undirected graph may produce cross edges.
- T  F  The lowest weight edge in a graph with all unique edge-weights is always included in any minimum spanning tree.
**Question 3 (6 Points):** Code for the Floyd-Warshall all-pairs shortest paths algorithm (the dynamic programming based $O(V^3)$ algorithm) appears below.

Write a loop invariant for the end of the outer loop (on variable $k$) – i.e., at the point in the code indicated, what can be claimed about $\text{adj}[i][j]$ with respect to $k$.

Hint: this loop invariant essentially why the algorithm works!

```plaintext
// adj[][] is initially the weight matrix.
// If there is no edge (i,j) then adj[i][j] = Infinity
// where Infinity is a sufficiently large number (e.g., n*MAX_EDGE_WEIGHT).
for (k = 1 to n){
    for (i = 1 to n){
        for (j = 1 to n){
            adj[i][j] = min(adj[i][j], adj[i][k] + adj[k][j]);
        }
    }
    // LOOP INVARIANT HERE
}
```

**Question 4 (6 Points):**

**Part (A)** Give example functions $f(n)$ and $g(n)$ where $f(n) = o(g(n))$.

**Part (B)** Is the following statement TRUE or FALSE (if it is true, give a clear argument; if it is false, give a counter-example):

“If $f(n) = o(g(n))$ then $\log(f(n)) = o(\log(g(n)))$.”
Question 5 (9 Points): Below are three recursive Java methods. In each case, the “problem size” $n$ is given by $r - l + 1$ – the number of elements in the subarray being processed. For each method do the following:

(1) Derive and explain a recurrence relation for the runtime $T(n)$ of the recursive method.

(2) Give and explain a tight Big-$\Theta$ bound on the $T(N)$.

```java
public static int foo(int a[], int l, int r) {
    if(r < l) return 0;
    if(r == l) return a[r];
    int i = l+1;
    int x = 0;
    while(i <= r) {
        x += a[i];
        i *= 2;
    }
    int m = (l+r)/2;
    return x + foo(a, l, m) + 2*foo(a, m+1, r);
}

public static int bar(int a[], int l, int r) {
    if(r < l) return 0;
    if(r == l) return a[r];
    int m = (l+r)/2;
    int x = bar(a, l, m) * bar(a, m+1, r);
    int i = l;
    int j = r;
    while(i <= j) {
        x += a[i] + 2*a[j];
        i++;
        j--;
    }
    return x;
}

public static int fubar(int a[], int l, int r) {
    if(r < l) return 0;
    if(r == l) return a[r];
    int s1=0;
    int s2=0;
    int m = (l+r)/2;
    for(int i=1; i<=m; i++)
        s1 += a[i];
    for(int i=m+1; i<=r; i++)
```
s2 += a[i];
if(s1 < s2)
    return s2 + fubar(a, l, m);
else
    return s1 + fubar(a, m+1, r);
}
Question 6 (5 Points): Suppose the following values are inserted into a “generic” (i.e., not self-balancing) binary search tree:

10, 2, 8, 11, 14, 3, 42

(They will not necessarily be inserted in the given order).
Devise an insertion order which will result in the tree having minimum height.

Question 7 (6 Points): Draw a minimum sized AVL tree of height 4 (an AVL tree with 1 node has height 0). (In other words, among all AVL trees of height 4, draw one with a minimum number of nodes). How many nodes does your tree have?
Recall that a Binary Search Tree $T$ is an AVL tree if and only if the following holds: for all nodes $v$ in $T$, the height of $v$’s subtrees differ by at most 1. Sometimes this is called “height-balanced-1”.

Question 8 (7 Points): Below is (part of) a simple java class for a Binary Tree node. The method isBST is supposed to determine if the given tree is a valid Binary Search Tree. (E.g., the developer might use it as a sanity checker).
Is the algorithm correct?
If your answer is YES give a concise argument of correctness.
If your answer is NO give and explain a counterexample.

```java
public class BTNode {

    private int key;
    private BTNode left;
    private BTNode right;

    // Returns true iff binary tree rooted at
    // t is a valid Binary Search Tree with
    // respect to the stored keys.
    public static boolean isBST(BTNode t) {
        if(t == null) return true;
        if(t.left != null && t.key < t.left.key)
            return false;
        else if(t.right != null && t.key > t.right.key)
            return false;
        else
            return isBST(t.left) && isBST(t.right);
    }
}
```
Part III: Theory of Computation (40 pts)

Question 1 (3 Points): Consider the following language over the alphabet of the 10 decimal digits (i.e., \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}): 

\[ L = \{ n : n = \text{closing price of oil rounded to nearest U.S. dollar on Dec. 29, 2017} \} \]

What is the smallest class of languages of which \( L \) is a member?
(a) Languages decidable in linear deterministic time.
(b) \( \text{NP} \)
(c) \( \text{NP-complete languages} \)
(d) \( \text{P} \)
(e) The undecidable languages

Question 2 (3 Points): The FUBAR problem is a new decision problem that you have just begun to study. So far you have been able to show the following two things about the FUBAR problem:

- There is a polynomial-time reduction from the Satisfiability problem (“Is a given Boolean formula satisfiable?”) to the FUBAR problem.
- You have a nondeterministic method that solves the FUBAR problem and that never consumes more than a polynomial amount of space.

What do you know for sure about the FUBAR problem given current knowledge?
(a) It is \( \text{NP complete} \)
(b) It is in the complexity class \( \text{NP} \)
(c) It is in the complexity class \( \text{P} \)
(d) It is not in the complexity class \( \text{P} \)
(e) It is \( \text{NP hard} \).
Question 3 (4 Points): The minimum number of states of a Deterministic Finite State Automaton (DFSA) over the alphabet \( \{0, 1\} \) recognizing the language containing the single string of \( n \) 0s (\( n \) is fixed) is:

(a) \( 2^n \)  (b) \( n + 2 \)  (c) \( n^2 \)  (d) 10.

Question 4 (4 Points): Let \( L \) be the language over \( \{0, 1\} \) consisting of all strings that have a 1 in the third symbol from the end (e.g., the string 0100 is in \( L \)). The minimum number of states of a DFSA recognizing \( L \) is:

(a) 50  (b) 5  (c) 8  (d) 100.

Question 5 (4 Points): For a DFSA \( A \), recognizing language \( L \), interchanging final and non-final states gives an automaton that recognizes \( \bar{L} \), the complement of \( L \). Show, by means of an example, that the above is not true for Non-deterministic Finite State Automaton (NFSA).
Question 6 (6 Points): For each of the following languages over \( \{0, 1\} \), indicate whether the language is regular or not.
(a) The set of all strings \( s \) that contain the pattern 101.
(b) The set of strings \( \{0^n10^{3n} : n \geq 0\} \).
(c) The set of all strings \( s \) such that the number of 0s and number of 1s in \( s \) differ by no more than 3.
(d) The set of all strings \( s \) such that the number of 0s and number of 1s \textit{upto any point} in \( s \) differ by no more than 3.

Question 7 (6 Points): Consider the languages over a fixed alphabet \( \Sigma \) having at least two symbols. Answer true or false to the following questions.

1. If \( L_1 \) is non-regular then its complement \( \overline{L_1} \) is also non-regular.
2. If \( L_1 \subseteq L_2 \) and \( L_2 \) is regular then \( L_1 \) is also regular.
3. If \( L_1 \) and \( L_2 \) are regular then \( L_1 \cap L_2 \) is also regular.

Question 8 (2 Points): Let \( L \) be the set of all properly paranthesized arithmetic expressions using the symbols \((,\),\( a, +, * \) where \((,\) are the paranthesis symbols, \( a \) is a variable and \( +, * \) are addition and multiplication symbols. \( L \) is:
(a) Regular
(b) Context Free
(c) Decidable but not context free
(d) Undecidable.
Question 9 (8 Points): Let $RL, CFL, Decidable$ and $Recognizable$ denote the class of regular, context free, Turing decidable and Turing Recognizable languages, respectively. We know that $RL \subseteq CFL \subseteq Decidable \subseteq Recognizable$ forming a strict hierarchy. For each language $A$ given below do the following: if $A$ belongs to any of the above classes then write down the earliest class in the above hierarchy to which $A$ belongs (For example, if $A$ is the set of all strings of the form $a^n b^n$ then your answer should be CFL); if $A$ belongs to none of them then write “none”.

1. $A = \{u : u \in \{0,1\}^* \text{ and has equal number of 0s and 1s}\}$.  
   \textbf{Answer:}

2. $A$ is the set of all strings $s$ in $\{0,1\}^*$ such that the length of $s$ is a multiple of 5.  
   \textbf{Answer:}

3. $A = \{<M> : M \text{ is a Turing machine that halts on the empty input string}\}$.  
   \textbf{Answer:}

4. $A$ is the set of all strings $s$ in $\{0,1,2\}^*$ containing equal number of 0s, 1s and 2s.  
   \textbf{Answer:}