1. **Graphs** [5 points]

   Suppose a simple graph \( G \) with \( n \) nodes (\( n > 1 \)) has a node of degree \( n - 1 \). Then what can you say about the diameter of \( G \)? Choose one.

- □ A. equals 1 [2 points]
- □ B. equals 2 [2 points]
- ✔ C. equals 1 or 2 [2 points]
- □ D. equals 1 if \( G \) is connected, but it is possible that \( G \) is not connected
- □ E. none of the above is necessarily true

2. **Functions** [5 points]

   Let \( f : \mathbb{R} \to \mathbb{R} \) be a monotonically increasing function and \( g : \mathbb{R} \to \mathbb{R} \) be a monotonically decreasing function. That is, \( \forall x, y \in \mathbb{R}, x \geq y \rightarrow (f(x) \geq f(y) \land g(x) \leq g(y)) \). Then which of the functions below are monotonically decreasing? Choose all the correct options.

   [Hint: If the answer is not clear to you, you may want to try working with examples.]

- ✔ A. \( f \circ g \) [2 points]
- ✔ B. \( g \circ f \) [2 points]
- □ C. \( f \circ f \) [0.5 points for not marking (0 if no markings)]
- □ D. \( g \circ g \) [0.5 points for not marking (0 if no markings)]
- □ E. None of the above [total 1 point (if none above marked)]

3. **Recurrence Relation** [5 points]

   Which of the following recurrences describe(s) a polynomial function of \( n \)? (A polynomial function has the form \( \Theta(n^c) \) for some constant \( c \).) Choose all the correct answers.

- □ A. \( A(1) = A(2) = 1; \text{ for } n > 2, A(n) = A([n/2]) + n\log n \) [1 point if only this selected]
- □ B. \( B(1) = B(2) = 1; \text{ for } n > 2, B(n) = 2B(n - 1) + 1 \) [1 point if only this selected]
- □ C. \( C(1) = C(2) = 1; \text{ for } n > 2, C(n) = C(n - 1) + C(n - 2) \) [1 point if only this selected]
- ✔ D. \( D(1) = D(2) = 1; \text{ for } n > 2, D(n) = D(n - 1) + n \) [5 - \( x \) points, if \( x \) other options selected]
- □ E. None of the above [2 points (0 if no markings)]
4. **Induction**

Let us define a function $P : \mathbb{N} \rightarrow \mathbb{N}$ as follows:

$$
P(0) = 2 \\
P(1) = 1 \\
P(n) = P(n - 1) + 6P(n - 2) \quad \text{for } n \geq 2.
$$

Use induction to prove that $P(n) = 3^n + (-2)^n$ for every integer $n \geq 0$.

**Base case or cases:**

**Solution:** We verify that $3^0 + (-2)^0 = 1 + 1 = 2 = P(0)$ and $3^1 + (-2)^1 = 3 - 2 = 1 = P(1)$.

**Inductive hypothesis:**

**Solution:** Suppose there exists an integer $k > 1$ such that for all $n \in \mathbb{N}$, $n \leq k$, $P(n) = 3^n + (-2)^n$.

**The inductive step:**

**Solution:** Then, we shall prove that $P(k + 1) = 3^{k+1} + (-2)^{k+1}$.

By the recursive definition, $P(k + 1) = P(k) + 6P(k - 1)$. Since $k > 1$, we have $k, k - 1 \geq 0$. Also $k, k - 1 \leq k$. Hence, by the induction hypothesis, $P(k) = 3^k + (-2)^k$ and $P(k - 1) = 3^{k-1} + (-2)^{k-1}$. Hence

$$
P(k + 1) = 3^k + (-2)^k + 6(3^{k-1} + (-2)^{k-1}) \\
= 3^k + (-2)^k + 2 \cdot 3^k - 3 \cdot (-2)^k \\
= (1 + 2) \cdot 3^k + (1 - 3)(-2)^k = 3^{k+1} + (-2)^{k+1}
$$

Hence, by induction, it follows that for all $n \in \mathbb{N}$, $P(n) = 3^n + (-2)^n$. 

5. **Graph: Proof by Contradiction.**

In a simple graph $G$, a path $P$ is said to be *maximal* if there is no other path $P'$ in $G$ such that $P$ is contained in (i.e., a subgraph of) $P'$.

Prove that if $P$ is a maximal path which ends at a node $u$, then $\text{degree}(u) \leq \text{length}(P)$.

[Hint: Use proof by contradiction.]

**Solution:**

Suppose, for the sake of contradiction, that there is a simple graph $G$ with a maximal path $P$ such that it ends in a node $u$ with $\text{degree}(u) > \text{length}(P)$.

Let $P = (v_0, v_1, \ldots, v_\ell)$, so that $\ell = \text{length}(P)$ and $v_\ell = u$. Note that there are $\ell$ nodes in $P$ other than $u$. Since $\text{degree}(u) > \ell$, at least one of the neighbors of $u$ is a node $w \notin \{v_0, \ldots, v_\ell\}$. Then $P' = (v_0, v_1, \ldots, v_\ell, w)$ is a valid path and $P$ is strictly contained in $P$. This contradicts the assumption that $P$ is a maximal path!

Hence, we conclude that our hypothesis was wrong: a simple graph $G$ cannot have a maximal path $P$ which ends in a node $u$ with $\text{degree}(u) > \text{length}(P)$. That is, for every maximal path $P$ which ends at a node $u$, $\text{degree}(u) \leq \text{length}(P)$. 
6. **Bijection.** [15 points]

Let $A$ be the set of all infinitely long binary strings (with symbols from $\{0, 1\}$) and $B$ be the set of all infinitely long ternary strings (with symbols from $\{0, 1, 2\}$). Show that there is a bijection between $A$ and $B$.

[Hint: Give one-to-one functions in both directions.]

**Solution:**

Let $f : A \to B$ be the identity function. This is a well-defined function since $A \subseteq B$. It is also invertible and hence one-to-one.

Let $g : B \to A$ be defined as follows. Given a string $b_0b_1b_2\ldots$ in which each $b_i \in \{0, 1, 2\}$, let $g(b_0b_1b_2\ldots) = a_0a_1a_2\ldots$, where for each $i \in \mathbb{N}$

$$a_{2i}a_{2i+1} = \begin{cases} 00 & \text{if } b_i = 0 \\ 01 & \text{if } b_i = 1 \\ 10 & \text{if } b_i = 2 \end{cases}$$

$g$ is invertible because, given $a_0a_1a_2\cdots = g(b_0b_1b_2\cdots)$, each $b_i$ can be uniquely computed from the two bits $a_{2i}a_{2i+1}$; hence $b_0b_1b_2\ldots$ can be uniquely determined from $g(b_0b_1b_2\ldots)$.

Then by the Cantor-Schröder-Bernstein theorem, this implies that there is a bijection between $A$ and $B$. 
7. Design and Analysis of an Algorithm. [15 points]

The following recursive function finds the \( k \)-th smallest element in a given array \( A \). That is, if \( k = 1 \), it outputs the smallest element, and if \( k = |A| \), the largest element.

The algorithm calls a procedure \texttt{Split} which takes an array \( A \) and a number \( x \) and splits \( A \) into three arrays \( (W, X, Y) \) such that \( W \) has those elements of \( A \) which are smaller than \( x \), \( Y \) has those elements of \( A \) which are greater than \( x \) and \( X \) has those elements of \( A \) which are equal to \( x \). So \( |A| = |W| + |X| + |Y| \). It is possible that one or more of the arrays \( W, X, Y \) are empty.

1: \textbf{function} \texttt{FindElement}(\texttt{A}: array of reals, \texttt{k}: int) \rightarrow \text{Find} \, k \text{th} \, \text{smallest} \, \text{element} \, \text{in} \, \text{array} \, \texttt{A}
2: \textbf{if} \, |\texttt{A}| < \texttt{k} \, \textbf{then} \rightarrow \text{\texttt{A} stands for the size of the array A}
3: \quad \textbf{return} \, \text{“error”}
4: \quad \textbf{x} := \texttt{A}[1] \rightarrow \text{x is the first element of A}
5: \quad (W, X, Y) := \texttt{Split}(\texttt{A}, \texttt{x}) \rightarrow \text{partitions A into 3 arrays as specified above}
6: \quad \textbf{if} \, |W| \geq \texttt{k} \, \textbf{then} \rightarrow \text{in this case the} \, k \text{th} \, \text{smallest element is in W}
7: \quad \quad \textbf{return} \, \texttt{FindElement}(\texttt{W, Expression}_1)
8: \quad \textbf{if} \, |W| + |X| < \texttt{k} \, \textbf{then} \rightarrow \text{in this case the} \, k \text{th} \, \text{smallest element is in Y}
9: \quad \quad \textbf{return} \, \texttt{FindElement}(\texttt{Y, Expression}_2)
10: \quad \textbf{return} \, \texttt{x} \rightarrow \text{if neither of the above two cases hold, the} \, k \text{th} \, \text{smallest element is in X}

(a) The algorithm is stated in terms of two expressions \texttt{Expression}_1 \, \text{and} \, \texttt{Expression}_2. \, \text{For the algorithm to be correct, what should they be, in terms of} \, |\texttt{A}|, |W|, |X|, |Y|, \, \texttt{k}.

i. \texttt{Expression}_1 = k. \quad \{4 \text{ points}\}

ii. \texttt{Expression}_2 = k - |W| - |X|. \quad \{4 \text{ points}\}

(b) If \texttt{Split}(\texttt{A}, \texttt{x}) \text{ takes time} \Theta(|\texttt{A}|), \text{ how much time does} \texttt{FindElement}(\texttt{A}, \texttt{k}) \text{ take in the worst case? Write your answer in the form} \Theta(f(|\texttt{A}|)). \text{ Briefly justify your answer.}

[Hint: To show a lower-bound, consider what happens if \texttt{A} \text{ is sorted in ascending order, and} \, k = |\texttt{A}|. \text{ To show this matches an upper-bound, note that} \max(\texttt{Expression}_1, \texttt{Expression}_2) \leq |\texttt{A}| - 1.]

\textbf{Solution:} \quad \{7 \text{ points}\}

Suppose \texttt{A} consists of \( n \) distinct elements, sorted in ascending order and \( k = n \). Then after the call to \texttt{Split}, \( |W| = 0 \) and \( |X| = 1 \), and the recursive call is to \texttt{FindElement}(\texttt{Y, k’t}), where \texttt{Y} could still be sorted in ascending order with \( |Y| = n - 1 \). Thus, the time taken, by the algorithm is given by the recurrence \( T(n) = \Theta(n) + T(n - 1) \) (and \( T(1) = \Theta(1) \)). Thus, \( T(n) = \Theta(n^2) \).

This is the worst possible, since in each level of recursion, the size of the array reduces by at least one.

\footnote{This is not the most efficient algorithm for this task. You will learn about more efficient ones in later courses.}
8. **State Diagram**

Design a deterministic finite state acceptor that accepts all binary strings which represent an even number, when interpreted as a number in base 3 as well as when interpreted as a number in base 2. (The empty string is interpreted as the number 0.)

For example, the string 110 is $2^2 + 2 + 0 = 6$ in base 2 and $3^2 + 3 + 0 = 12$ in base 3, and should be accepted. But the strings 10 (equals 3 in base 3) and 101 (equals 5 in base 2) should be rejected.

You machine will be given the input digit by digit, most-significant-digit first (i.e., left to right). The states of such a machine are shown below. Each state is labeled as $(a, b)$ where $a \equiv x \pmod{2}$ and $b \equiv y \pmod{2}$, with $x$ being the number seen so far in base 2, and $y$ being the number seen so far in base 3. Thus, for example, after seeing the input 110, $x = 6, y = 12$ and hence the machine will be in state $(0, 0)$.

Add all the edges and clearly mark the labels on the edges. Remember to mark the start state and final state(s) using the standard convention in state diagrams.

[Hint: A number in base 3 is even iff it has an even number of 1s. A number in base 2 is even iff it ends in 0.]