1. Each of these ten questions has one of the following five answers:

A: \( \Theta(1) \)    B: \( \Theta(\log n) \)    C: \( \Theta(n) \)    D: \( \Theta(n \log n) \)    E: \( \Theta(n^2) \)

Choose the correct answer for each question. Each correct answer is worth +1 point; each incorrect answer is worth \(-1/2\) point; and each “I don’t know” is worth +1/4 point. Negative scores will be recorded as 0.

(a) What is \( \frac{3}{n} + \frac{n}{3} \) ?

(b) What is \( \sum_{i=1}^{n} \frac{i}{n} \) ?

(c) What is \( \sqrt{\sum_{i=1}^{n} i} \) ?

(d) How many bits are required to write the number \( n! \) (the factorial of \( n \)) in binary?

(e) What is the solution to the recurrence \( E(n) = E(n-3) + 17n \) ?

(f) What is the solution to the recurrence \( F(n) = 2F(n/4) + 6n \) ?

(g) What is the solution to the recurrence \( G(n) = 9G(n/9) + 9n \) ?

(h) What is the worst-case running time of quicksort?

(i) Let \( X[1..n,1..n] \) be a fixed array of numbers. Consider the following recursive function:

\[
WTF(i, j) = \begin{cases} 
0 & \text{if } \min\{i, j\} \leq 0 \\
-\infty & \text{if } \max\{i, j\} > n \\
X[i, j] + \max \left\{ \begin{array}{ll}
WTF(i-2, j+1) \\
WTF(i-2, j-1) \\
WTF(i-1, j-2) \\
WTF(i+1, j-2) \\
\end{array} \right. & \text{otherwise}
\end{cases}
\]

How long does it take to compute \( WTF(n, n) \) using dynamic programming?

(j) The Rubik’s Cube is a mechanical puzzle invented in 1974 by Ernő Rubik, a Hungarian professor of architecture. The puzzle consists of a \( 3 \times 3 \times 3 \) grid of ‘cubelets’, whose faces are covered with stickers in six different colors. In the puzzle’s solved state, each face of the puzzle is one solid color. A mechanism inside the puzzle allows any face of the cube to be freely turned (as shown on the right). The puzzle can be scrambled by repeated turns. Given a scrambled Rubik’s Cube, how long does it take to find the shortest sequence of turns that returns the cube to its solved state?
2. Let $T$ be a rooted tree with integer weights on its edges, which could be positive, negative, or zero. The weight of a path in $T$ is the sum of the weights of its edges. Describe and analyze an algorithm to compute the minimum weight of any path from a node in $T$ down to one of its descendants. It is not necessary to compute the actual minimum-weight path; just its weight. For example, given the tree shown below, your algorithm should return the number $-12$.

![Tree Diagram]

The minimum-weight downward path in this tree has weight $-12$.

3. Describe and analyze efficient algorithms to solve the following problems:

(a) Given a set of $n$ integers, does it contain two elements $a, b$ such that $a + b = 0$?
(b) Given a set of $n$ integers, does it contain three elements $a, b, c$ such that $a + b = c$?

4. A common supersequence of two strings $A$ and $B$ is another string that includes both the characters of $A$ in order and the characters of $B$ in order. Describe and analyze and algorithm to compute the length of the shortest common supersequence of two strings $A[1..m]$ and $B[1..n]$. You do not need to compute an actual supersequence, just its length.

For example, if the input strings are ANTHROPOBIOLOGICAL and PRETERDIPLOMATICALLY, your algorithm should output 31, because a shortest common supersequence of those two strings is **PREANANTHEROHOLDP**OBIOP**LOMAT**ICALLY.

5. [Taken directly from HBSO.] Recall that the Fibonacci numbers $F_n$ are recursively defined as follows: $F_0 = 0$, $F_1 = 1$, and $F_n = F_{n-1} + F_{n-2}$ for every integer $n \geq 2$. The first few Fibonacci numbers are $0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, \ldots$.

**Prove** that any non-negative integer can be written as the sum of distinct non-consecutive Fibonacci numbers. That is, if any Fibonacci number $F_n$ appears in the sum, then its neighbors $F_{n-1}$ and $F_{n+1}$ do not. For example:

\[
\begin{align*}
88 &= 55 + 21 + 8 + 3 + 1 &= F_{10} + F_{8} + F_{6} + F_{4} + F_{2} \\
42 &= 34 + 8 &= F_{9} + F_{6} \\
17 &= 13 + 3 + 1 &= F_{7} + F_{4} + F_{2}
\end{align*}
\]