1344. Angle Between Hands of a Clock

Medium Math Leetcode Link

Problem Description

In this problem, we are asked to find the smaller angle between the hour and minute hands of a clock based on the provided `hour` and `minutes` values. A clock is a circle, and a full circle is 360 degrees. As time progresses, both the hour and minute hands move around the clock at different speeds. The hour hand makes a complete rotation every 12 hours, while the minute hand does so every hour (60 minutes). The goal is to calculate the angle in degrees between the two hands, which could be thought of as two vectors originating from the center of the clock and pointing towards the current positions of the hands. Importantly, we want to find the smallest of the two possible angles created by these hands. The precision of the answer should be within `10^-5` degrees for it to be considered correct.

Intuition

To determine the angle between the hour and minute hands, we first calculate the angle of each hand relative to 12 o'clock, or `0` degrees. For the hour hand (`h`), it moves 30 degrees each hour (`1/12` of a full circle) and an additional 0.5 degrees each minute (since it progresses `1/12` of an hour per 60 minutes). For the minute hand (`m`), it moves at a consistent pace of 6 degrees per minute (`1/60` of a full circle).

Once we have the angles for each hand, we get the difference between them using `abs(h - m)`. This gives us one of the angles created by the hour and minute hands—the larger one if it's over 180 degrees. To find the smaller angle, we simply subtract the larger angle from 360 degrees if necessary, since the sum of both angles is always 360 degrees (the full circle).

The use of `min(diff, 360 - diff)` gives us the smaller angle, as required by the problem statement. This is because the angle between the two hands must always be less than or equal to 180 degrees. If `diff` is greater than 180, `360 - diff` gives us the smaller angle. If `diff` is less than or equal to 180, then it is already the smaller angle.

Solution Approach

The approach used in the provided solution is straightforward and does not involve complex data structures or patterns. It focuses on mathematics and geometry concepts applied to the context of a clock.

Here is the breakdown of the implementation steps:

1. Calculate the angle of the hour hand from 12 o'clock. This is done using the formula `30 * hour + 0.5 * minutes`. Since there are 12 hours on a clock, the hour hand moves `360 degrees / 12 = 30 degrees` every hour. Additionally, for every minute that passes, the hour hand will move an additional `0.5 degrees` (`30 degrees / 60 minutes`), because it is gradually moving towards the next hour mark.

2. Calculate the angle of the minute hand from 12 o'clock using the formula `6 * minutes`. The minute hand moves `360 degrees / 60 = 6 degrees` every minute.

3. Find the absolute difference between these two angles with `abs(h - m)`. This difference is one of the two angles formed by the hour and minute hands.

4. The last step is to determine the smaller of the two possible angles. If the difference calculated is greater than half a rotation (`180 degrees`), the smaller angle would be the supplement of that angle, which is `360 degrees - diff`. If it is less than `180 degrees`, it is already the smaller angle. The `min(diff, 360 - diff)` function returns the smaller angle.

No further algorithms or complex data structures are used in this solution; the primary focus is on understanding the movement of the clock hands and applying basic arithmetic to find the desired angle.

Example Walkthrough

Let's walk through a small example. Assume that we have `hour = 3` and `minutes = 30`. We want to find the smaller angle between the hour and minute hands of a clock at 3:30.

1. **Calculate the angle of the hour hand from 12 o'clock**:
   - Using the formula `30 * hour + 0.5 * minutes`:
   - `30 * 3 + 0.5 * 30 = 90 + 15 = 105 degrees`.
   - So, the angle for the hour hand from the 12 o'clock position is 105 degrees.
2. **Calculate the angle of the minute hand from 12 o'clock**:
   - Using the formula `6 * minutes`:
   - `6 * 30 = 180 degrees`.
   - So, the angle for the minute hand from the 12 o'clock position is 180 degrees.
3. **Find the absolute difference between these two angles**:
   - `abs(h - m) = abs(105 - 180) = abs(-75) = 75 degrees`.
   - The difference between the two angles is 75 degrees.
4. **Determine the smaller angle**:
   - Since the difference is less than 180 degrees, the smaller angle is the same as the difference.
   - We use `min(diff, 360 - diff)` to confirm this:
   - `min(75, 360 - 75) = min(75, 285) = 75 degrees`.

The smaller angle between the hour and minute hands when the time is 3:30 is 75 degrees.

Python Solution

```python
class Solution:
    def angleClock(self, hour: int, minutes: int) -> float:
        # Calculate the angle of the hour hand relative to 12:00.
        # The hour hand moves 30 degrees every hour, plus an additional 0.5 degrees every minute.
        hour_angle = 30 * hour + 0.5 * minutes

        # Calculate the angle of the minute hand relative to 12:00.
        # The minute hand moves 6 degrees every minute.
        minute_angle = 6 * minutes

        # Calculate the difference between the two angles.
        angle_diff = abs(hour_angle - minute_angle)

        # The angle between the two hands may be the smaller of the two possibilities:
        # either angle_diff directly, or the complementary angle, which is 360 - angle_diff.
        return min(angle_diff, 360 - angle_diff)

```

Java Solution

```java
class Solution {

    /**
     * Calculates the smaller angle between the hour and minute hands of a clock.
     *
     * @param hour The current hour shown by the clock.
     * @param minutes The current minutes shown by the clock.
     * @return The smaller angle between the hour and minute hands.
     */
    public double angleClock(int hour, int minutes) {
        // Calculate the angle of the hour hand from 12 o'clock.
        // The hour hand moves 0.5 degrees per minute (30 degrees per hour and 1/60 of that per minute).
        double hourHandAngle = 30 * hour + 0.5 * minutes;

        // Calculate the angle of the minute hand from 12 o'clock.
        // The minute hand moves 6 degrees per minute.
        double minuteHandAngle = 6 * minutes;

        // Find the absolute difference between the two angles.
        double angleDifference = Math.abs(hourHandAngle - minuteHandAngle);

        // Calculate the smaller angle between the two clock hands.
        // It should be less than or equal to 180 degrees.
        return Math.min(angleDifference, 360 - angleDifference);
    }
}

```

C++ Solution

```cpp
class Solution {
public:
    double angleClock(int hour, int minutes) {
        // Calculate the angle of the hour hand
        // The hour hand moves 30 degrees per hour plus another 0.5 degrees per minute
        double hourAngle = 30 * hour + 0.5 * minutes;

        // Calculate the angle of the minute hand
        // The minute hand moves 6 degrees per minute
        double minuteAngle = 6 * minutes;

        // Calculate the absolute difference between the two angles
        double angleDifference = abs(hourAngle - minuteAngle);

        // The angle should be the smaller one between angleDifference and 360 - angleDifference
        // because the hands could be on either sides of each other
        return min(angleDifference, 360 - angleDifference);
    }
};

```

Typescript Solution

```typescript
// Calculate the angle between the hour hand and the minute hand on a clock
// @param {number} hour - The hour on the clock
// @param {number} minutes - The minutes on the clock
// @returns {number} - The smallest angle between the hour and minute hands
function angleClock(hour: number, minutes: number): number {
    // Calculate the position of the hour hand
    // Each hour represents 30 degrees (360 degrees / 12 hours)
    // Each minute adds 0.5 degrees to the hour hand (30 degrees / 60 minutes)
    const hourHandPosition: number = 30 * hour + 0.5 * minutes;

    // Calculate the position of the minute hand
    // Each minute represents 6 degrees (360 degrees / 60 minutes)
    const minuteHandPosition: number = 6 * minutes;

    // Calculate the absolute difference between the two hands
    const difference: number = Math.abs(hourHandPosition - minuteHandPosition);

    // The angle between the hands could be the difference or 360 - difference
    // We need the smallest angle
    return Math.min(difference, 360 - difference);
}

```

Time and Space Complexity

The code consists of simple arithmetic operations and a minimum of two values. Here is the analysis of its complexities:

- **Time Complexity**: The algorithm computes the angle by performing a constant number of arithmetic operations, irrespective of the input size. Thus, the time complexity is `O(1)`.

- **Space Complexity**: It uses a fixed amount of space for the variables `h`, `m`, and `diff` and does not allocate any additional space that scales with the input size. Therefore, the space complexity is also `O(1)`.