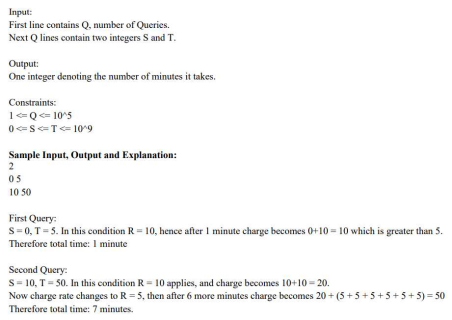
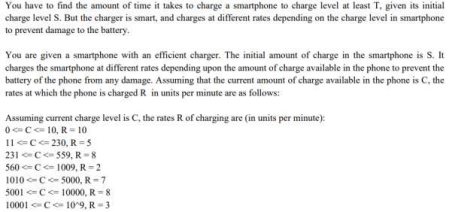
**CODING ASSESMENT-1**

****

**#include <iostream>**

**#include <vector>**

**using namespace std;**

**typedef long long ll;**

**// Struct to define charging intervals**

**struct Interval {**

**ll start, end, rate;**

**};**

**// Function to compute charging time**

**int compute\_charging\_time(ll S, ll T) {**

**if (S >= T) return 0;**

**vector<Interval> intervals = {**

**{0, 10, 10},**

**{11, 229, 5},**

**{230, 559, 8},**

**{560, 1009, 2},**

**{1010, 5000, 7},**

**{5001, 10000, 8},**

**{10001, 1000000000LL, 3}**

**};**

**ll time = 0;**

**ll current = S;**

**for (const auto& interval : intervals) {**

**if (current > interval.end) continue;**

**ll interval\_start = max(current, interval.start);**

**ll interval\_end = min(interval.end, T);**

**if (interval\_start > interval\_end) continue;**

**ll units\_needed = interval\_end - interval\_start + 1;**

**time += (units\_needed + interval.rate - 1) / interval.rate; // Ceil division**

**current = interval\_end + 1;**

**if (current > T) break;**

**}**

**return time;**

**}**

**int main() {**

**// Sample input**

**int Q = 2;**

**vector<pair<ll, ll>> queries = {**

**{0, 5},**

**{10, 50}**

**};**

**// Process and print results**

**for (int i = 0; i < Q; ++i) {**

**ll S = queries[i].first;**

**ll T = queries[i].second;**

**int result = compute\_charging\_time(S, T);**

**cout << result << endl;**

**}**

**return 0;**

**}**