Declaring some named constants with meaningful names would make

the code more clear. 2 points deducted.

//

// main.cpp

// Project2

//

// Created by Yunong on 10/17/14.

// Copyright (c) 2014 Yunong. All rights reserved.

//

#include <iostream>

#include <string>

using namespace std;

int main(){

int voice, sms, month;

string name;

cout << "Minutes used: "; cin >> voice;

cout << "Text messages: "; cin >> sms; cin.ignore(10000, '\n');

cout << "Customer name: "; getline(cin, name);

cout << "Month number (1=Jan, 2=Feb, etc.): "; cin >> month;

cout << "---\n"; // Read data

if (voice < 0) cout << "The number of minutes used must be nonnegative.\n";

else

if (sms < 0) cout << "The number of text messages must be nonnegative.\n";

else

if (name == "") cout << "You must enter a customer name.\n";

else

if ((month < 1) || (month > 12)) cout << "The month number must be in the range 1 through 12.\n";

// Detect any wrong data. If any mistake is detected, other sentences under "else" won't be executed. So there will be only one error message no matter how many mistakes the input data have.

else {

voice = (voice<=500) ? 0 : voice-500;

sms = (sms<=200) ? 0 : sms-200;

// Preprocess read data. If voice or sms is in the free range, set them to zero so that when multiplying, the product becomes zero. Otherwise, subtract the free amount from the usage.

double costSms, total;

costSms = ((month>=6)&&(month<=9)) ? 0.02 : 0.03;

total = costSms\*sms+voice\*0.45+40.00;

if (sms>200) total += (0.11-costSms)\*(sms-200);

// Calculate the bill. The reason why costSms should be subtracted from the fee of per message beyond 400th is that we have multiplied the amount of messages beyond 400 by costSms. If we don't subtract costSms from 0.11, the final result will exceed by that product.

cout.setf(ios::fixed);

cout.setf(ios::showpoint);

cout.precision(2);

cout << "The bill for " << name << " is $" << total << endl; // Print the result.

}

}