```
// Rental charge calculation

#include <iostream>
#include <string>
using namespace std;

int main()
{
      // Get and validate starting odometer reading

    cout << "Odometer at start: ";
    int startReading;
    cin >> startReading;
    cin.ignore(10000, '\n');
    if (startReading < 0)
    {
        cout << "---" << endl
             << "The starting odometer reading must be nonnegative." << endl;
        return 1;
    }

      // Get and validate ending odometer reading

    cout << "Odometer at end: ";
    int endReading;
    cin >> endReading;
    cin.ignore(10000, '\n');
    if (endReading < startReading)
    {
        cout << "---" << endl
             << "The final odometer reading must be at least as large as the starting reading."
             << endl;
        return 1;
    }

      // Get and validate number of rental days

    cout << "Rental days: ";
    int rentalDays;
    cin >> rentalDays;
    cin.ignore(10000, '\n');
    if (rentalDays <= 0)
    {
        cout << "---" << endl
             << "The number of rental days must be positive." << endl;
        return 1;
    }

      // Get and validate customer name

    cout << "Customer name: ";
    string customer;
    getline(cin, customer);
    if (customer == "")
    {
        cout << "---" << endl
             << "You must enter a customer name." << endl;
        return 1;
    }

      // Get and validate luxury status

    cout << "Luxury car? (y/n): ";
    string luxury;
    getline(cin, luxury);
    if (luxury != "y"  &&  luxury != "n")
    {
        cout << "---" << endl
             << "You must enter y or n." << endl;
        return 1;
    }

      // Get and validate month

    cout << "Month (1=Jan, 2=Feb, etc.): ";
    int month;
    cin >> month;
    cin.ignore(10000, '\n');
    if (month < 1  ||  month > 12)
    {
        cout << "---" << endl
             << "The month number must be in the range 1 through 12." << endl;
        return 1;
    }

      // Mileage tier caps and rental rates

    const double DAILY_RATE_BASIC  = 33.00;
    const double DAILY_RATE_LUXURY = 61.00;
    const double TIER_1_CAP = 100;
    const double TIER_2_CAP = 400;
    const double TIER_1_RATE           = 0.27;
    const double TIER_2_RATE_NONWINTER = 0.21;
    const double TIER_2_RATE_WINTER    = 0.27;
    const double TIER_3_RATE           = 0.19;

      // Compute rental charge

    double daily_rate = DAILY_RATE_BASIC;
    if (luxury == "y")
```

```
    daily_rate = DAILY_RATE_LUXURY;

double charge = rentalDays * daily_rate;

int milesDriven = endReading - startReading;

if (milesDriven <= TIER_1_CAP)
    charge += milesDriven * TIER_1_RATE;
else
{
      // Compute charge for the portion of miles driven in tier 1

    charge += TIER_1_CAP * TIER_1_RATE;

      // Determine rate for tier 2

    double tier_2_rate = TIER_2_RATE_NONWINTER;
    if (month >= 12  ||  month <= 3)  //  December through March
        tier_2_rate = TIER_2_RATE_WINTER;

    if (milesDriven <= TIER_2_CAP)
    {
          // Add charge for remainder of miles driven (in tier 2)

        charge += (milesDriven - TIER_1_CAP) * tier_2_rate;
    }
    else
    {
          // Add charge for the portion of miles driven in tier 2
          // and the remainder of miles driven (in tier 3)

        charge += (TIER_2_CAP - TIER_1_CAP) * tier_2_rate +
                  (milesDriven - TIER_2_CAP) * TIER_3_RATE;
        }
    }

      // Print rental charge

    cout.setf(ios::fixed);
    cout.precision(2);
    cout << "---" << endl
         << "The rental charge for " << customer << " is $" << charge << endl;
}
```