#include "stdafx.h"

#include<iostream>

#include<conio.h>

#include<stdlib.h>

#include<string>

#include<ctime>

#include<sstream>

using namespace std;

class phoneBill {

private:

double basicCost=0.0;

double callCharge=0.0;

double VAT = 0.14;

double offPeak=0.0;

double sharedDiscount=0.0;

double netCost=0.0;

double totalCost=0.0;

public:

phoneBill(char distanceCall,char sharedCall) {

if (distanceCall=='y') {

this->basicCost = 89.001;

this->callCharge = 1.761;

if (sharedCall == 'y') {

this->sharedDiscount = 0.5;

}

else

{

this->sharedDiscount = 0.0;

}

}

else

{

this->basicCost = 59.4;

this->callCharge = 0.759;

}

}

double getBasicCost() {

return this->basicCost;

}

double getCallCharge() {

return this->callCharge;

}

double getOffPeak() {

return this->offPeak;

}

double getSharedDiscount() {

return this->sharedDiscount;

}

double getNetCost() {

return this->netCost;

}

double getTotalCost() {

return this->totalCost;

}

double getVAT() {

return this->VAT;

}

int getHours(string time) {

int hours;

stringstream greek(time.substr(0, 2));

greek >> hours;

if (hours < 24) {

return hours;

}

else

{

throw string("Invalid Hour Exception");

}

}

int getMin(string time) {

int min;

stringstream greek(time.substr(3, 2));

greek >> min;

if (min < 60) {

return min;

}

else {

throw string("Invalid Minute Exception");

}

}

int getSec(string time) {

int sec;

stringstream greek(time.substr(6, 2));

greek >> sec;

if (sec < 60) {

return sec;

}

else

{

throw string("Invalid Second Exception");

}

}

void calculateOffPeakDiscount(string startingTime, char distanceCall) {

int hours,min,sec;

hours = getHours(startingTime);

min = getMin(startingTime);

sec = getSec(startingTime);

if ((hours >= 19 && hours <= 24) || (hours >= 0 && hours<7)) {

if (distanceCall == 'y') {

this->offPeak = 0.5;

}

else

{

this->offPeak = 0.4;

}

}

}

void calculateNetCost(string durationTime) {

int hours,min,sec;

hours = getHours(durationTime)\*3600;

min = getMin(durationTime)\*60;

sec = getSec(durationTime);

this->netCost = (hours + min + sec)\*this->callCharge;

this->netCost = (this->netCost\*(1 - this->offPeak));

this->netCost = (this->netCost\*(1 - this->sharedDiscount));

if (this->netCost < this->basicCost) {

this->netCost = this->basicCost;

}

}

void calculateTotalCost() {

this->totalCost = this->netCost + (this->netCost\*this->VAT);

}

};

int main()

{

string startingTime, durationTime;

char distanceCall, sharedCall;

cout << "Enter the strating time of the call(HH:MM:SS):\t";

cin >> startingTime;

cout << "Enter the duration of the call(HH:MM:SS):\t";

cin >> durationTime;

cout << "Was the duration was more than 50km(y/n):\t";

cin >> distanceCall;

cout << "Is the call a shared call(y/n):\t\t\t";

cin >> sharedCall;

try {

phoneBill\* phone = new phoneBill(distanceCall, sharedCall);

phone->calculateOffPeakDiscount(startingTime, distanceCall);

phone->calculateNetCost(durationTime);

phone->calculateTotalCost();

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Telephone Bill\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\tBasic Cost\t\t=" << phone->getBasicCost() << endl;

cout << "\tOff-Peak Discount\t=" << (phone->getOffPeak())\*100 << "%" << endl;

cout << "\tShared Call Discount\t=" << (phone->getSharedDiscount())\*100 << "%" << endl;

cout << "\tNet Cost\t\t=" << phone->getNetCost() << endl;

cout << "\tVAT\t\t\t=" << (phone->getVAT())\*100 << "%" << endl;

cout << "\tTotalCost\t\t=" << phone->getTotalCost() << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";

}

catch (string e) {

cout << e;

}

cin.clear();

cin.ignore(1000,'\n');

cin.get();

return 0;

}