**Lab Experience ~~5~~ 6**

**Objectives:**

1. Understanding the conditional ternary operator.
2. Replacing if/else statements with assignment statements
3. Writing a program using the decision structure.

**The Conditional Ternary Operator**

The Ternary operator in C++ is a shorthand if/else statement and can be used anywhere an if/else statement can be used. The syntax of the ternary operator is:

(expression ?if\_true : if\_false)

which is the same thing as

if(expression)

if\_true

else

if\_false

The conditional Ternary operator can replace an if/else or an if/else if statement. For example consider the following if/else if statement:

if( a > b)

largest = a;

else if(b > a)

largest = b;

else

cout<< “The two values are equal\n\n”;

The above can be rewritten using the conditional ternary operator as follows:

**a >b ?largest = a : (b > a ? largest = b : cout<< “The two values are equal\n\n”);**

Nesting ternary operators beyond one level is not recommended because the code becomes confusing.

**Replacing the if/else statement with an assignment statement**

This is done only when the if/else statement is relatively simple. A common usage of if statements is to set the value of the Boolean variable as follows:

**if(shieldStrength<minShieldSafety)**

**shieldWarningFlag = true;**

**else**

**shieldWarningFlag = false;**

A flag variable called shieldWarningFlag is being either set or disabled dependent on the relational expression

**shieldStrength<minShieldSafety**

Since the result of the relational expression is either a one or a zero, the if statement can be eliminated by using an assignment statement as follows:

**shieldWarningFlag = shieldStrength<minShieldSafety;**

The above statement will assign a value of true to shieldWarningFlag if the relational expression is true or it will assign the value of false to shieldWarningFlag if the relational expression is false.

The compiler might generate a warning message concerning the evaluation of the assignment statement. To eliminate the warning message a cast to type Boolean of the integer value is necessary.

**shieldWarningFlag = static\_cast<bool>(shieldStrength<minShieldSafety);**

What is the benefit of eliminating the if statement? The machine language code generated by an if statement is generally 5 to 10 statements, whereas with an assignment statement, the number of statements reduces to 1 to 3 internally. What this means is the size of the executable (i.e. the program that runs on your computer) will be reduced which in turns conserves memory.

**Lab Exercises**

**Directions:**

Start Microsoft word and record the questions and answers to all of the exercises in the lab 5 word document   
Answer the following questions based on material presented in lecture and found in chapters 1-4 of our textbook.

**Short Answer**

1. Convert each if/else if statementinto a ternary expression.

a) if(score > 59)

passFlag = true;

else

passFlag = false;

**score> 59 ? passFlag = true : passFlag = false**

b) if(x == 2)

y = 1;

else

y = 0;

**x == 2 ? y = 1 : ( y = 0)**

1. Convert the following if statements into an assignment statement.

a) if(score > 59)

passFlag = true;

else

passFlag = false;

**passFlag = score > 59** //it may be necessary to comment

b) if(x == 2)

y = 1;

else

y = 0;

**y = x == 2**

**Programming Exercise:**

Write a C++ program to calculate the parking fare for customers who park their cars in a parking lot given the following information:

1. A character showing the type of vehicle: ‘c’ or ‘C’ for car, ‘b’ or ‘B’ for bus, ‘t’ or ‘T’ for truck
2. An integer between 0 and 24 showing the hour the vehicle entered the lot.
3. An integer between 0 and 60 showing the minute the vehicle entered the lot.
4. An integer between 0 and 24 showing the hour the vehicle left the lot.
5. An integer between 0 and 60 showing the minute the vehicle left the lot.

This is a public lot and to encourage people to park for a short period of time, the management uses two different rates for each type of vehicle as shown in the table below:

|  |  |  |
| --- | --- | --- |
| Type | First Rate | Second Rate |
| Car | $0.00/hr first 3 hours | $1.50/hr after the first 3 hours |
| Truck | $1.00/hr for first 2 hours | $2.30/hr after the first two hours |
| Bus | $2.00/hr for first hour | $3.70/hr after the first hour |

All time input is in military time, but the time displayed to the user should use AM/PM. **You may assume no one parks for more than 24 hours in the lot.**

As an example of user input is shown below:

Type of vehicle? C

Hour vehicle entered the lot (0-24)? 14

Minute vehicle entered the lot (0-60)? 23

Hour vehicle left the lot (0-24)? 18

Minute vehicle left the lot (0-60)? 8

The output format is shown below:

Parking Lot Charge

Type of vehicle: Car

TIME-IN 2:23 PM

TIME-OUT 6:08 PM

--------

PARKING TIME 03:45

ROUNDED TOTAL 04

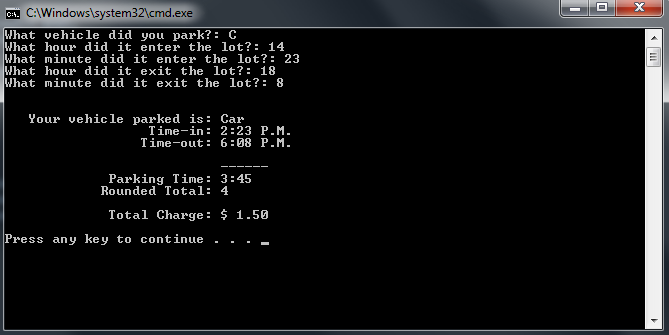
TOTAL CHARGE $ 1.50

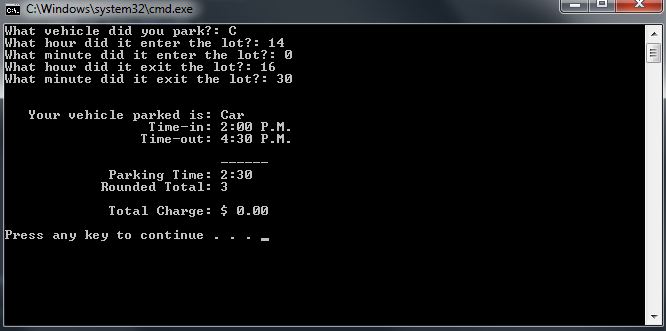
Notice you want to subtract the time-in from the time-out. In this case since 08 is smaller than 23 you would borrow 1 hour which is 60 minutes from the hour time, thus 60 + 8 = 68 minutes and 68 – 23 = 45.

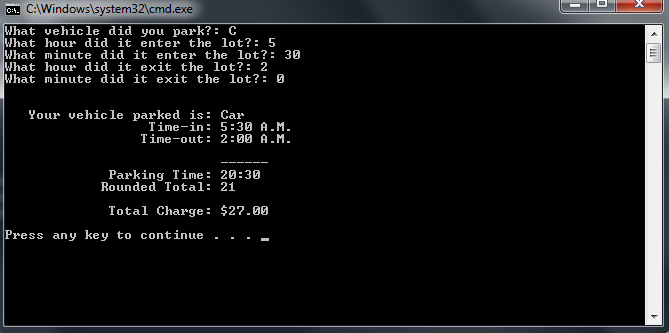
The other calculations are similar.

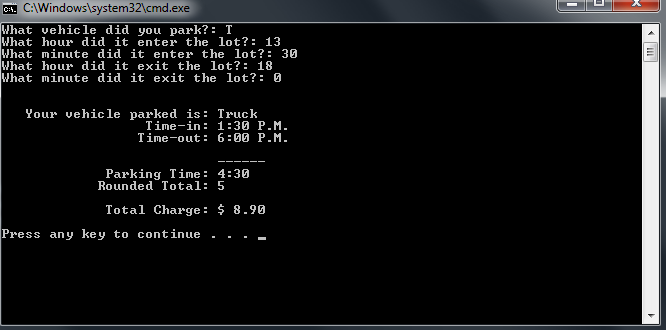
Note: You must also round up the minutes to the nearest hour. There is no fractional charge for minutes.

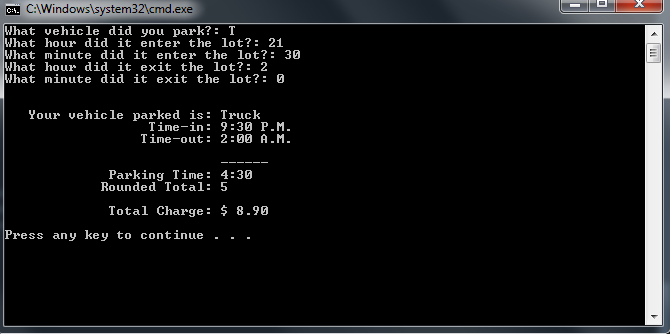
**Copy and paste your program into your word document. Below the program, paste a snapshots of the output window of the test runs of your programs. Your test plan should test every possible scenario of the program. I.e. Charges for all types of vehicles with multiple hours for each.**

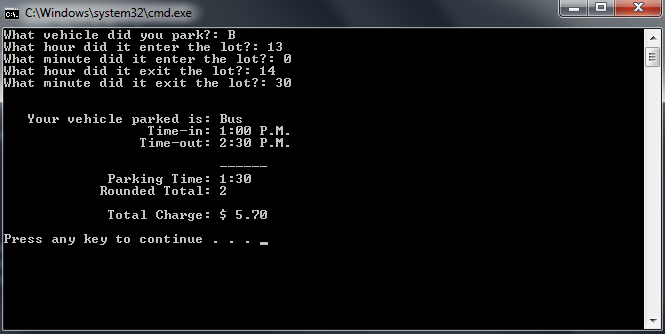
****

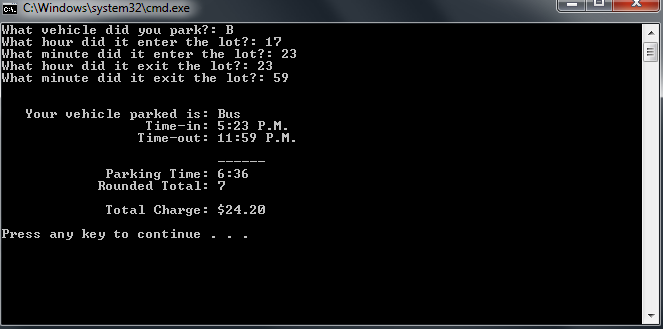
****

****

****

****

****

****

**//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**// Programmer: Chris Dang Class: CSCI 1106 Fall 2014**

**//**

**// Description: This program will calculate the amount charged**

**// after inputting the time the vehicle entered the**

**// parking lot, and exit time based on what kind of**

**// vehicle the user was driving.**

**//**

**//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**#include <iostream>**

**#include <iomanip>**

**#include <string>**

**using namespace std;**

**double CAR\_RATE\_1 = 0.00; //Car is $1.50 per hour before 3 hours (hours < 3)**

**double CAR\_RATE\_2 = 1.50; //Car is $1.50 per hour after 3 hours (hours > 3)**

**double TRUCK\_RATE\_1 = 1.00; //Truck is $1.00 per hour before 2 hours (hours < 2)**

**double TRUCK\_RATE\_2 = 2.30; //Truck is $2.30 per hour after 2 hours (hours > 2)**

**double BUS\_RATE\_1 = 2.00; //Bus is $2.00 per hour before 1 hour (hours < 1)**

**double BUS\_RATE\_2 = 3.70; //Bus is $3.70 per hour after 1 hour (hours > 1)**

**int CAR\_FIRST\_RATE\_HOURS = 3; //These are the times for how long the first rate**

**int TRUCK\_FIRST\_RATE\_HOURS = 2; //will apply for each kind of vehicle**

**int BUS\_FIRST\_RATE\_HOURS = 1;**

**int main () {**

**char inputVehicle;**

**int timeInHour, timeInMinute, timeOutHour, timeOutMinute;**

**int outputHour, outputMinute, roundedTotal, recordHourIn;**

**int recordMinuteIn, recordHourOut, recordMinuteOut, inTimeOfDay, outTimeOfDay;**

**double charge;**

**string outputVehicle;**

**cout << "What vehicle did you park?: ";**

**cin >> inputVehicle;**

**cout << "What hour did it enter the lot?: ";**

**cin >> timeInHour;**

**recordHourIn = timeInHour;**

**inTimeOfDay = timeInHour;**

**cout << "What minute did it enter the lot?: ";**

**cin >> timeInMinute;**

**recordMinuteIn = timeInMinute;**

**cout << "What hour did it exit the lot?: ";**

**cin >> timeOutHour;**

**recordHourOut = timeOutHour;**

**outTimeOfDay = timeOutHour;**

**cout << "What minute did it exit the lot?: ";**

**cin >> timeOutMinute;**

**recordMinuteOut = timeOutMinute;**

**// calculates parking time///////////////////**

**if (timeOutMinute < timeInMinute)**

**{**

**timeOutHour -= 1; //subtract 1 hour to**

**timeOutMinute += 60; //add 60 minutes so that minute subtraction results in positive number**

**}**

**if (timeOutHour < timeInHour)**

**timeOutHour += 24; //add 24 hours so that hour subtraction results in positive number**

**outputMinute = timeOutMinute - timeInMinute;**

**outputHour = timeOutHour - timeInHour;**

**// end calculations ////////////////////////**

**// rounding \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**if (outputMinute > 0) // rounds parking time up to full hour**

**roundedTotal = outputHour + 1; // if the parking time is not a whole hour**

**else**

**roundedTotal = outputHour;**

**// end round \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**// Military to Convential Time Conversion**

**if (recordHourIn > 12)**

**recordHourIn -= 12;**

**if (recordHourOut > 12)**

**recordHourOut -= 12;**

**// end conversation**

**//%%%%%%%%%%%%%%%%%% calculates the Charge for parking time %%%%%%%%%%%%%%%%%%%%%%%%%%%**

**if (inputVehicle == 'C')**

**{**

**outputVehicle = "Car";**

**if (roundedTotal > CAR\_FIRST\_RATE\_HOURS)**

**{**

**charge = (roundedTotal - CAR\_FIRST\_RATE\_HOURS) \* CAR\_RATE\_2;**

**}**

**else**

**charge = CAR\_RATE\_1;**

**}**

**if (inputVehicle == 'T')**

**{**

**outputVehicle = "Truck";**

**if (roundedTotal > TRUCK\_FIRST\_RATE\_HOURS)**

**charge = (TRUCK\_RATE\_1 \* TRUCK\_FIRST\_RATE\_HOURS) + ((roundedTotal - TRUCK\_FIRST\_RATE\_HOURS) \* TRUCK\_RATE\_2);**

**else**

**charge = TRUCK\_RATE\_1 \* roundedTotal;**

**}**

**if (inputVehicle == 'B')**

**{**

**outputVehicle = "Bus";**

**if (roundedTotal > BUS\_FIRST\_RATE\_HOURS)**

**charge = (BUS\_RATE\_1 \* BUS\_FIRST\_RATE\_HOURS) + ((roundedTotal - BUS\_FIRST\_RATE\_HOURS) \* BUS\_RATE\_2);**

**else**

**charge = BUS\_RATE\_1 \* roundedTotal;**

**}**

**//%%%%%%%%%%%%%%%%%%%%%%%%% end charge calculations %%%%%%%%%%%%%%%%%%%%%%%%%%%%%**

**cout << fixed << setprecision(2) << showpoint << endl << endl;**

**cout << " Your vehicle parked is: " << outputVehicle << endl;**

**cout << " Time-in: " << recordHourIn << ":" ;**

**if(recordMinuteIn < 10) // adds 0 before single digit minutes (from 4:4 to 4:04)**

**cout << '0' << recordMinuteIn ;**

**else**

**cout << recordMinuteIn;**

**//used to determine and display the time of day (A.M. or P.M.)**

**if (inTimeOfDay > 0 && inTimeOfDay < 12)**

**cout << " A.M." ;**

**else**

**cout << " P.M." ;**

**cout << endl;**

**cout << " Time-out: " << recordHourOut << ":" ;**

**if(recordMinuteOut < 10)**

**cout << '0' << recordMinuteOut ;**

**else**

**cout << recordMinuteOut ;**

**if (outTimeOfDay > 0 && outTimeOfDay < 12)**

**cout << " A.M." ;**

**else**

**cout << " P.M." ;**

**cout << endl;**

**cout<< endl;**

**cout << " ------" << endl;**

**cout << " Parking Time: " << outputHour << ":" ;**

**if(outputMinute < 10)**

**cout << '0' << outputMinute;**

**else**

**cout << outputMinute ;**

**cout << endl;**

**cout << " Rounded Total: " << roundedTotal << endl << endl;**

**cout << " Total Charge: $" << setw(5) << charge << endl << endl;**

**return 0;**

**}**

**//end main**

**Due Dates:**According to the Lab Six drop box closing date/time.

**What to hand in:**

1. Compress the .cpp file from the programming exercise and the word document created into a single file called **{yourname}Lab6.zip** e.g. timwrennlab6.zip Note:**If your name is not part of the zip filename, I will not open the zipped file.*Note: Only compress the .cpp files for the programming exercise.***
2. Hand in a print out of your program and the word document.
3. Place the zipped file into the D2L DropBox folder titled Lab 6.