Seneca College  
  
ECT  
  
  
  
  
  
PRG355C  
  
  
  
  
  
  
Final Project  
Cars and Services Database

December 1st, 2014

Professor Goran Svenk

Name: Jayme Ferreira Bandeira Cunha

Student Number: 027691146

Problem description:

The purpose of this program is to create, edit and handle a database of cars and the services done on them. The code is implemented in object oriented programming C++ language. It consists of two classes named Car and Service for creating objects to be handled by the file handling functions. The Car data is consisted of the car model, plate number and year. The Service data is consisted of its description, time between two services in months, the last date when the service was done, and the next service date calculated inside the class using the last date as reference. Service class incorporates a static function that displays the current system date and time. The program offers the option to create or edit a file named “CarsAndServices.txt” in which all the data for cars and services will be added line by line sequentially. There is a interactive menu in the program that shows 5 options for the user to choose from. Option 1 is to store a car’s model followed by its service, if the plate number already exists the car is not added. Option 2 is to store a service to be done on a car, if the plate number is not in the file the car is not added. Option 3 is to perform a search on the file for a specific car by its plate number and list all its service records to be done, if car is not found the program returns a error message. Option 4 is to display all cars to be serviced on a specified date, if date is not found the program will return a error message. Option 5 exits the program. The code contains file error handling and overloads the operator “<<” in the Car and Service classes.

Source Code:

/\*

\* File: main.cpp

\* Author: Jayme-PC

\*

\* Created on November 15, 2014, 3:57 PM

\*/

#ifdef \_MSC\_VER

#define \_CRT\_SECURE\_NO\_WARNINGS

#endif

#include<iostream>

#include<fstream>

#include<sstream>

#include<string>

#include<ctime>

using namespace std;

struct Date

{

int day;

int month;

int year;

};

class Car

{

private:

string model;

int year;

protected:

string plate\_number;

public:

Car()

{

cout<<"\n\tType the car model, plate number and year: "<<endl;

cout<<"\t";

cin>>model>>plate\_number>>year;

}

Car(string modl, string plate\_n, int yer)

{

model = modl;

plate\_number = plate\_n;

year = yer;

}

string getPlate() {return plate\_number;}

//Operator<< overloaded to insert Car data into the file

friend ostream& operator<<(ostream &output, const Car& c)

{

return output<<c.model<<" "<<c.plate\_number<<" "<<c.year;

}

};

class Service

{

string description;

int time\_btwn\_services; //represented in months

Date lastDate;

Date nextDate;

public:

Service()

{

cout<<"\n\tType the description of the service: "<<endl;

cout<<"\t";

cin.ignore(2000, '\n');

getline(cin,description);

cout<<"\n\tDetermine the time between services in months: "<<endl;

cout<<"\t";

cin>>time\_btwn\_services;

cout<<"\n\tDate of the service in the format dd mm yyyy: "<<endl;

cout<<"\t";

cin>>lastDate.day>>lastDate.month>>lastDate.year;

nextServiceDate(lastDate,time\_btwn\_services);

}

Service(string desc, int tbs, Date lD)

{

description = desc;

time\_btwn\_services = tbs;

lastDate = lD;

nextServiceDate(lD, tbs);

}

void nextServiceDate(Date lastDate, int tbs)

{

int tm, ty;

ty = tbs/12;

tm = tbs%12;

nextDate.year = lastDate.year + ty;

nextDate.month = lastDate.month + tm;

nextDate.day = lastDate.day;

if(nextDate.month>12)

{

nextDate.month = nextDate.month-12;

nextDate.year++;

}

}

static void display\_date() // Display date and time

{

time\_t now = time(0); // current date/time based on current system

char\* dt = ctime(&now); // convert to string form

cout<<"\nDate and time: "<<dt<<endl; //display date/time

}

//Operator<< overloaded to insert Service data into the file

friend ostream& operator<<(ostream &output, const Service& s)

{

return output <<" "<<s.description<<" "<<s.time\_btwn\_services<<" "

<<s.lastDate.day<<" "<<s.lastDate.month<<" "<<s.lastDate.year<<" "

<<s.nextDate.day<<" "<<s.nextDate.month<<" "<<s.nextDate.year<<endl;

}

};

//===================================================================================

//File I/O functions

void insertData(Car c, Service s)

{

ofstream myfile("F:\\PRG\\CarsAndServices.txt", ios::app);

if(myfile.is\_open())

{

myfile << c << s;

myfile.close();

}

else cout << "\tUnable to open file";

}

//Open and write to a file

void insertCarAndService(Car c, Service s)

{

string line, plate, mod;

int found = 0;

ifstream myfile1("F:\\PRG\\CarsAndServices.txt");

if(myfile1.is\_open())

{

while ( getline (myfile1,line) )

{

stringstream lineStream (line);

lineStream >> mod >> plate;

if(plate == c.getPlate())

{

found = 1;

myfile1.close();

cout<<"\n\tCar could not be inserted, plate already exists"<<endl;

}

}

}

else cout << "\tUnable to open file";

if(!found) insertData(c,s);

}

//Find a car in a file and add the service

void insertCarWithService(string inPlate)

{

string mod, plate, line;

int y;

ifstream myfile("F:\\PRG\\CarsAndServices.txt");

int found = 0;

if (myfile.is\_open())

{

while ( getline (myfile,line) )

{

stringstream lineStream (line);

lineStream >> mod >> plate;

if(plate == inPlate)

{

lineStream >> y;

Car c(mod, plate, y);

Service \*s1 = new Service;

insertData(c, \*s1);

found = 1;

myfile.close();

}

}

if(!found) cout<<"\n\tCar not found!!"<<endl;

}

else cout << "\tUnable to open file";

}

void findCarAndList(string inPlate)

{

string mod, plate, line;

string desc, colon;

Date lD;

int y, tbs;

ifstream myfile("F:\\PRG\\CarsAndServices.txt");

int found = 0;

if (myfile.is\_open())

{

while ( getline (myfile, line) )

{

stringstream lineStream (line);

lineStream >> mod >> plate;

if(plate == inPlate)

{

lineStream >> y >> desc >> tbs >> lD.day

>> lD.month >> lD.year;

Service s(desc, tbs, lD);

cout << "\t" << s;

found = 1;

}

}

myfile.close();

if(!found) cout<<"\n\tCar not found!!"<<endl;

}

else cout << "\tUnable to open file";

}

void findDateAndList(Date d1)

{

string mod, plate, line;

string desc, colon;

Date lD, nD;

int y, tbs;

ifstream myfile("F:\\PRG\\CarsAndServices.txt");

int found = 0;

if (myfile.is\_open())

{

while ( getline (myfile, line) )

{

stringstream lineStream (line);

lineStream >> mod >> plate >> y >> desc >> tbs

>> lD.day >> lD.month >> lD.year

>> nD.day >> nD.month >> nD.year;

if(d1.day == nD.day && d1.month == nD.month && d1.year == nD.year)

{

cout << "\t" << mod << " " << plate << " " << y << endl;

found = 1;

}

}

myfile.close();

if(!found) cout<<"\n\tDate not found!!"<<endl;

}

else cout << "\tUnable to open file";

}

int main()

{

Service::display\_date();

int exit = 0;

int menu;

bool create;

cout<<"Press 1 to create a new file or 0 to open the existing one"<<endl;

cin>>create;

if(create)

{

ofstream myfile("F:\\PRG\\CarsAndServices.txt");

if(!myfile)

{

cout<<"Error creating file!\n"<<endl;

system("pause");

return 0;

}

myfile.close();

}

else

{

ifstream myfile("F:\\PRG\\CarsAndServices.txt");

if(!myfile)

{

cout<<"Error opening file!\n"<<endl;

system("pause");

return 0;

}

myfile.close();

}

do

{

cout<<"\nChoose what you would like to do: \n"<<endl;

cout<<"\t1. Store a car data and the service to be done on it"<<endl;

cout<<"\t2. Store a service to be done on a car"<<endl;

cout<<"\t3. Find a car and list all its service records"<<endl;

cout<<"\t4. Find a date and display the cars to be serviced on that date"<<endl;

cout<<"\t5. Exit\n"<<endl;

cout<<"\t";

cin>>menu;

switch (menu)

{

case 1:

{

Car \*c1 = new Car;

Service \*s1 = new Service;

insertCarAndService(\*c1, \*s1);

delete c1;

delete s1;

break;

}

case 2:

{

string plate;

cout<<"\n\tType in the car plate: "<<endl;

cout<<"\t";

cin>>plate;

insertCarWithService(plate);

break;

}

case 3:

{

string plate;

cout<<"\n\tType in the car plate to be found: "<<endl;

cout<<"\t";

cin>>plate;

cout<<"\n\tServices done on car plate number "<<plate<<":"<<endl;

cout<<"\n\tService - Months Between - Last Service - Next Service"<<endl;

findCarAndList(plate);

break;

}

case 4:

{

Date d;

cout<<"\n\tType in the date you would like to look for services in the format dd mm yyyy: "<<endl;

cout<<"\t";

cin>>d.day>>d.month>>d.year;

cout<<"\n\tCars to be serviced on the date "<<d.day<<" "<<d.month<<" "<<d.year<<":"<<endl;

cout<<"\n\tModel - Plate - Year"<<endl;

findDateAndList(d);

break;

}

case 5:

exit = 1;

break;

default:

break;

}

} while (!exit);

system("pause");

return 0;

}



