#include<iostream>

#include<fstream>

#include<iomanip>

#include<string>

using namespace std;

s t r u c t automobile

{

short i n t year;

s t r i n g category, make, models color;

};

const short i n t NUM\_OF\_VEHICLES = 3;

s t r u c t garage

{

automobile vehicles[NUM\_OF\_VEHICLES];

double lengthj width;

};

void initialize\_garage(garage &g);

garage populategarage();

void displaygarage(const garage &g);

void swap\_car\_with\_neighbor(garage &mg, garage &ng);

ifstream i n p u t f i l e;

ofstream outputfile;

void main()

garage mygarage, neighborgarage;

inputfile.open("C:/path/garage-input-information.txt");

i f ( ! i n p u t f i l e)

{

cout << " i n p u t f i l e not found!\n";

system("pause");

e x i t ( e ) ;

}

outputfile.open("C:/Path/garage-output - i n f o r m a t i o n . t x t " ) ;

i f ( l o u t p u t f i l e)

{

cout << "Outputfile not created!\n";

system("pause");

e x i t ( 0 ) ;

}

// outputfile << "Populate My Garage:";

initialize\_garage(mygarage);

mygarage = populategarage();

o u t p u t f i l e << setw(50) << right << "My Populated Garage";

displaygarage(mygarage);

o u t p u t f i l e << "\n\n\n";

// outputfile << "Populate Neighbor's Garage:";

initialize\_garage(neighborgarage);

neighborgarage = populategarage();

o u t p u t f i l e << setw(55) << right << "Neighbor's Populated Garage";

displaygarage(neighborgarage);

o u t p u t f i l e << "\n\n";

swap\_car\_with\_neighbor(mygarage, neighborgarage);

o u t p u t f i l e << "\nAfter swapping car with neighbor my garage i s as follow:";

displaygarage(mygarage);

o u t p u t f i l e << "\n\nAfter swapping car with neighbor, neighbor's garage i s as follow:";

C: \Users\Path\\/lsual S t u d i o 2ei2\ . . Xgarage-swapperNgarage-swapper-with-f i l e i o . cpp 2

displaygarage(neighborgarage);

cout << "\n\n\n";

systemC'pause");

v o i d i n i t i a l i z e \_ g a r a g e ( g a r a g e &g)

{

f o r ( i n t i = 0 ; i < NUM\_OF\_VEHICLES; i++)

{

g . v e h i c l e s [ i ] . c a t e g o r y = "\0";

g . v e h i c l e s [ i ] . m a k e = " \ 0 " j

g . v e h i c l e s [ i ] . m o d e l = "\0";

g . v e h i c l e s [ i ] . c o l o r = "\0";

g . v e h i c l e s [ i ] . y e a r = 0;

}

g . l e n g t h = 0.0; g.width = 0.0;

}

garage p o p u l a t e g a r a g e ()

{

garage t g ; //temporary garage

s t r i n g i n s t r u c t i o n \_ p r o m p t;

i n i t i a l i z e \_ g a r a g e ( t g ) ;

//

f o r ( i n t i = 0; i < NUM\_OF\_VEHICLES; i++)

{

o u t p u t f i l e << "\n\nEnter VEHICLE " << i + 1 << '

g e t l i n e ( i n p u t f i l e , t g . v e h i c ; l e s [ i ] . c a t e g o r y );

i f ( t g . v e h i c l e s [ i ] . c a t e g o r y == " \ 0 " )

/ / i = NUM\_OF\_VEHICLES;

break;

e l se

{

g e t l i n e ( i n p u t f i l e ^ t g . v e h i c l e s [ i ] . m a k e );

g e t l i n e ( i n p u t f i l e , t g . v e h i c l e s [ i ] . m o d e l );

g e t l i n e ( i n p u t f i l C j t g . v e h i c l e s [ i ] . c o l o r );

i n p u t f i l e >> t g . v e h i c l e s [ i ] . y e a r;

i n p u t f i l e . i g n o r e ( ) ;

}

}

}

i n p u t f i l e >> t g . l e n g t h;

i n p u t f i l e >> t g . w i d t h;

i n p u t f i l e . i g n o r e ( ) ;

r e t u r n t g ;

v o i d d i s p l a y g a r a g e ( c o n s t garage &g)

{

f o r ( i n t i = 0; i < NUM\_OF\_VEHICLES; i++)

{

o u t p u t f i l e << "\n\nVEHICLE " << i + 1 << ":\n";

o u t p u t f i l e << s e t w ( 1 2 ) << r i g h t << "category: "; o u t p u t f i l e << s e t w ( 1 5 ) << l e f t << g . v e h i c l e s [ i ].

c a t e g o r y;

o u t p u t f i l e << s e t w ( 1 2 ) << r i g h t << "Make: "; o u t p u t f i l e << s e t w ( 2 0 ) << l e f t << g . v e h i c l e s [ i ] . m a k e;

o u t p u t f i l e << endl;

o u t p u t f i l e << s e t w ( 1 2 ) << r i g h t << "Model: "; o u t p u t f i l e << s e t w ( 1 5 ) << l e f t << g . v e h i c l e s [ i ] . m o d el

o u t p u t f i l e << s e t w ( 1 2 ) << r i g h t << "Color: "; o u t p u t f i l e << s e t w ( 2 0 ) << l e f t << g . v e h i c l e s [ i ] . c o l or

o u t p u t f i l e << s e t w ( 1 2 ) << r i g h t << "Year: "; o u t p u t f i l e << s e t w ( 4 ) << l e f t << g . v e h i c l e s [ i ] . y e a r;

C:\Users\Path\Visual Studio 2012\..Agarage-swapperVgarage-swapper-with-fileio.cpp 3

o u t p u t f i l e << "\n\nGarage Size: " << g.length \* g.width << " square f t . ";

}

void swap\_car\_with\_neighbor(garage &mg, garage &ng)

{

i n t choice = 0; garage tg; //temporary garage

while (choice < 1 || choice > NUM\_OF\_VEHICLES)

{

cout << "\n\nEnter the car from your garage that you /n";

cout << "wish to swap with neighbor (1/2/3) or 0 to quit: ";

c i n >> choice;

i f (choice == 0) \*

break;

}

i f (choice != 0)

{

i n t i = --choice;

t g . v e h i c l e s [ i ] . c a t e g o r y = ng.vehicles[i].category;

tg.vehicles[i].make = ng.vehicles[i].make;

tg.vehicles[i].model = ng.vehicles[i].model;

t g . v e h i c l e s [ i ] . c o l o r = n g . v e h i c l e s [ i ] . c o l o r;

t g . v e h i c l e s [ i ] . y e a r = n g . v e h i c l e s [ i ] . y e a r;

ng.vehicles[i].category = mg.vehicles[i].category;

ng.vehicles[i].make = mg.vehicles[i].make;

ng.vehicles[i].model = mg.vehicles[i].model;

n g . v e h i c l e s [ i ] . c o l o r = mg.vehicles[i].color;

n g . v e h i c l e s [ i ] . y e a r = mg.vehicles[i].year;

mg.vehicles[i].category = t g . v e h i c l e s [ i ] . c a t e g o r y;

mg.vehicles[i].make = tg.vehicles[i].make;

mg.vehicles[i].model = tg.vehicles[i].model;

mg.vehicles[i].color = t g . v e h i c l e s [ i ] . c o l o r;

mg.vehicles[i].year = t g . v e h i c l e s [ i ] . y e a r ;

}

}