**DESCRIPTION**

**We have created a code that allows you to automate some of the sales of a company's vehicles, such as trucks, cars and motorbikes. In itself, the company is just starting its activity, hence the conclusion that the program code is not perfect and not complete. At a minimum, it would be nice to add exception handling and save the result to a file. The code itself contains 1 main class and 3 subclasses. They are all abstract and their functions are virtual. Polyphorism is also present. Containers like vector are also used. It has meaningful inheritance.**

#include <string>

#include <iostream>

#include <vector>

using namespace std;

class pTS

{

public:

pTS() {};

~pTS() {};

virtual void print\_info() = 0;

virtual double calc\_tax() = 0;

protected: string marka;

string reg;

double cena;

};

class kamioni : public pTS

{

public:

int tovar\_sp;

int br\_osi;

kamioni(

string marka,

string reg,

double cena,

int tovar\_sp,

int br\_osi

) {

this->marka = marka;

this->reg = reg;

this->cena = cena;

this->tovar\_sp = tovar\_sp;

this->br\_osi = br\_osi;

}

virtual void kamion()

{

marka = "";

reg = "";

cena = 0;

tovar\_sp = 0;

br\_osi = 0;

};

virtual void kamion(string m1, string r, double c, int ts, int br)

{

marka = m1;

reg = r;

cena = c;

tovar\_sp = ts;

br\_osi = br;

};

~kamioni() {};

virtual void print\_info()

{

cout << "Марка: " << marka << endl;

cout << "Модел: " << reg << endl;

cout << "Цена: " << cena << " лв. " << endl;

cout << "Товароспособност: " << tovar\_sp << " т. " << endl;

cout << "Брой оси: " << br\_osi << endl;

};

virtual double calc\_tax()

{

if (tovar\_sp > 3)

return cena = cena + 300;

};

};

class koli : public pTS

{

public:

int janti;

int gumi;

koli(

string marka,

string reg,

double cena,

int janti,

int gumi

)

{

this->marka = marka;

this->reg = reg;

this->cena = cena;

this->janti = janti;

this->gumi = gumi;

}

virtual void kola()

{

marka = "";

reg = "";

cena = 0;

janti = 0;

gumi = 0;

};

virtual void kola(string m1, string r, double c, int j, int g)

{

marka = m1;

reg = r;

cena = c;

janti = j;

gumi = g;

};

~koli() {};

virtual void print\_info()

{

cout << "Марка: " << marka << endl;

cout << "Модел: " << reg << endl;

cout << "Цена: " << cena << " лв. " << endl;

cout << "Джанти: " << janti << endl;

cout << "Гуми: " << gumi << endl;

};

virtual double calc\_tax()

{

if (janti == 1)

cena = cena + 100;

if (janti == 2)

cena = cena + 200;

if (gumi == 0)

cena = cena + 200;

if (gumi == 2)

cena = cena - 100;

return cena;

};

};

class motori : public pTS

{

public:

int kaski;

int bag\_kuf;

motori(

string marka,

string reg,

double cena,

int kaski,

int bag\_kuf

)

{

this->marka = marka;

this->reg = reg;

this->cena = cena;

this->kaski = kaski;

this->bag\_kuf = bag\_kuf;

}

virtual void motor()

{

marka = "";

reg = "";

cena = 0;

kaski = 0;

bag\_kuf = 0;

};

virtual void motor(string m1, string r, double c, int k, int bk)

{

marka = m1;

reg = r;

cena = c;

kaski = k;

bag\_kuf = bk;

};

~motori() {};

virtual void print\_info()

{

cout << "Марка: " << marka << endl;

cout << "Модел: " << reg << endl;

cout << "Цена: " << cena << " лв. " << endl;

cout << "Каски: " << kaski << endl;

cout << "Багажен куфар: " << bag\_kuf << endl;

};

virtual double calc\_tax()

{

if (kaski == 1)

cena = cena + 50;

if (kaski == 2)

cena = cena + 100;

if (kaski == 3)

cena = cena + 150;

if (bag\_kuf == 1)

cena = cena + 100;

if (bag\_kuf == 2)

cena = cena + 200;

if (bag\_kuf == 3)

cena = cena + 300;

if (bag\_kuf == 4)

cena = cena + 200;

return cena;

};

};

class TS

{

public:

TS() {};

~TS() {};

void Add(pTS\* obj)

{

vec.push\_back(obj);

};

void List()

{

cout << "Transportni sredstva:\n ";

for (int i = 0; i < vec.size(); i++)

vec[i]->print\_info();

cout << "Kraina cena: " << vec[0]->calc\_tax();

vec.erase(vec.begin());

};

private:

vector<pTS\*>vec;

};

int main()

{

setlocale(LC\_ALL, "Bulgarian");

TS ts;

string m1;

string r;

double c;

char ch;

int tov;

int br;

int g;

int j;

int k, b;

do

{

cout << "\n\nMain Menu\n=============================\n";

cout << "1. Kamioni\n2. Koli\n3. Motori\n4. Spisak\n";

cout << "0. Izhod\nIzberete: ";

cin >> ch;

cin.ignore(255, '\n');

switch (ch)

{

case '1':cout << "Marka: "; getline(cin, m1);

cout << "Reg. nom: "; cin >> r;

cout << "Cena: "; cin >> c;

cout << "Tovarosposobnost: "; cin >> tov;

cout << "Broi osi: "; cin >> br;

ts.Add(new kamioni(m1, r, c, tov, br));

break;

case '2':cout << "Marka: "; getline(cin, m1);

cout << "Reg. nom: "; cin >> r;

cout << "Cena: "; cin >> c;

cout << "0. нормални, 1.алуминиеви, 2.спортни" << endl;

do

{

cout << "Janti: "; cin >> j;

} while (j < 0 || j>2);

cout << "0.Нови, 1.10% износване, 2.15% износване" << endl;

do

{

cout << "Gumi: "; cin >> g;

} while (g < 0 || g>2);

ts.Add(new koli(m1, r, c, j, g));

break;

case '3':cout << "Marka: "; getline(cin, m1);

cout << "Reg. nom: "; cin >> r;

cout << "Cena: "; cin >> c;

cout << "0.0, 1.1, 2.2, 3.3" << endl;

do {

cout << "Kaski: "; cin >> k;

} while (k < 0 || k>3);

cout << "0.0, 1.1, 2.2, 3.3, 4.4" << endl;

do {

cout << "Bagajni kufari: "; cin >> b;

} while (b < 0 || b>4);

ts.Add(new motori(m1, r, c, k, b));

break;

case '4':ts.List();

break;

default:

break;

}

} while (ch != '0');

};