Proiect sfarsit de an

Algoritmi si structuri de date

Program care tine evidenta pieselor auto, pe baza unui cod intern cu datele: piesa, sistemul mecanic de care apartine piesa, pret fara tva, pret cu tva, cota tva si importatorul piesei.

Datele sunt salvate in fisier si pot fi recitite oricand.

Algoritmii programului:

typedef struct piese{

int cod;

string tip;

string mecanism;

string importator;

float pret;

float pret\_tva;

float tva;

struct piese \*urm;

}o\_piesa;

fstream inf;

Algoritm evidenta este:

Cheama creaza();

Cheama afisare\_meniu();

Sfarsitalgoritm.

Subalgoritm creaza() este:

varf = NULL;

cursor=varf;

Sfarsitsubalgoritm

Subalgoritm afisare\_meniu este:

Date: int x, string m;

x=-1;

Cattimp x!=0 executa:

Cheama meniu();

Citeste x;

Selecteaza x dintre:

Daca x=0 atunci:

Tipareste "Iesire...";

break;

Sfarsitdaca;

Daca x=1 atunci:

Cheama get\_from\_file();

break;

Sfarsitdaca;

Daca x=2 atunci:

Cheama afisare\_lista\_intreaga();

break;

Sfarsitdaca;

Daca x=3 atunci:

Cheama citire\_piese();

break;

Sfarsitdaca;

Daca x=4 atunci:

Date: c;

Tipareste:"Introduceti codul: ";

Citeste c;

Cheama cautare\_dupa\_cod(c);

break;

Sfarsitdaca;

Daca x=5 atunci:

Tipareste: "Introduceti mecanismul: ";

Citeste m;

Cheama cautare\_dupa\_mec(m);

break;

Sfarsitdaca;

Daca x=6 atunci:

Tipareste:"Introduceti importatorul: ";

Citeste m;

Cheama cautare\_dupa\_imp(m);

Break;

Sfarsitdaca;

Daca x=7 atunci:

Tipareste:"Introduceti tipul: ";

Citeste m;

Cheama cautare\_dupa\_tip(m);

Break;

Sfarsitdaca;

Altfel:

Tipareste: "Optiune gresita";

Break;

Sfarsitselecteaza;

Sfarsitcattimp;

Sfarsitsubalgoritm;

Subalgoritm get\_from\_file este:

Date: int cod,c; string t,m,imp;float p,p\_t, tva;

inf.open("data.txt",ios::in);

Daca (!inf.is\_open() ) atunci:

Tipareste: "Fisierul nu a putut fi deschis!";

get\_from\_file=0;

Altfel:

Tipareste:"Fisierul este deschis";

Sfarsitdaca.

Cattimp( preia din fisier: c,t,m,imp,p,p\_t,tva, !inf.eof() ) executa:

Daca cod==0 atunci:

Cheama adauga\_dupa(c,t,m,imp,p,p\_t,tva);

Sfarsitdaca.

preia din fisier: c,t,m,imp,p,p\_t,tva;

cod=1;

Cheama adauga\_dupa(c,t,m,imp,p,p\_t,tva);

Sfarsitcattimp.

Daca cod==0 atunci:

Tipareste:"Lista este goala!";

Sfarsitdaca.

inf.close();

Sfarsitsubaglrotim.

Subalgoritm afisare\_lista\_intreaga este:

piese \*p;

p=varf;

Daca p==NULL atunci:

Tipareste:"Lista este goala"

Altfel:

Cattimp p!=NULL executa:

Tipareste: p->cod, p->tip, p->mecanism, p->importator, p->pret, p->tva, p->pret\_tva;

p=p->urm;

Sfarsitcattimp;

Sfarsitdaca.

Sfarsitsubalgoritm.

Subalgoritm citire\_piese este:

Date:int c; string t,m,imp; float pr, p\_tva,tva;

Citeste c,t,m,imp,pr,tva;

p\_tva=pr+((pr\*tva)/100);

Cheama adauga\_dupa(c,t,m,imp,pr,p\_tva,tva);

Cheama print\_to\_file(c,t,m,imp,pr,p\_tva,tva);

Sfarsitsubalgoritm.

Subalgoritm cautare\_dupa\_cod(int c) este:

piese \*p;

p=varf;

int n=0;

Cattimp p!=NULL executa:

Daca(c==p->cod) atunci:

Cheama afisare\_inf(p);

n=1;

Sfarsitdaca.

p=p->urm;

Sfarsitcattimp;

Daca n==0 atunci:

Tipareste:"Codul nu a fost gasit";

Sfarsitdaca.

Sfarsitsubalgoritm.

Subalgoritm cautare\_dupa\_mec(string mec) este:

piese \*p;

p=varf;

int n=0;

Cattimp p!=NULL executa:

Daca mec==p->mecanism atunci:

Cheama afisare\_inf(p);

n=1;

Sfarsitdaca.

p=p->urm;

Sfarsitcattim;

Daca n==0 atunci:

Tipareste:"Informatia dorita nu a fost gasita";

Sfarsitdaca.

Sfarsitsubalgoritm.

Subalgoritm cautare\_dupa\_imp(string imp) este:

piese \*p;

p=varf;

int n=0;

Cattimp p!=NULL executa:

Daca imp==p->importator atunci:

Cheama afisare\_inf(p);

n=1;

Sfarsitdaca.

p=p->urm;

Sfarsitcattimp;

Daca n==0 atunci:

Tipareste:"Informatia dorita nu a fost gasita";

Sfarsitdaca.

Sfarsitsubalgoritm.

Subalgoritm cautare\_dupa\_tip(string t) este:

piese \*p;

p=varf;

int n=0;

Cattimp p!=NULL executa:

Daca t==p->tip atunci:

Cheama afisare\_inf(p);

n=1;

Sfarsitdaca.

p=p->urm;

Sfarsitcattimp;

Daca n==0 atunci:

Tipareste:"Informatia dorita nu a fost gasita";

Sfarsitdaca.

Sfarsitsubalgoritm.

Subalgoritm adauga\_dupa(int cod, string tip, string mec, string imp, float pret, float pret\_tva, float tva) este:

piese \*p=new piese;

p->cod = cod;

p->tip= tip;

p->mecanism = mec;

p->importator = imp;

p->pret = pret;

p->pret\_tva = pret\_tva;

p->tva = tva;

Daca varf==NULL atunci:

p->urm=varf;

varf=p;

cursor=p;

Altfel:

p->urm=cursor->urm;

cursor->urm=p;

cursor=p;

Sfarsitdaca.

Sfarsitsubalgoritm.

Subalgoritm meniu este:

Tipareste:"1.Citeste lista pieselor\n";

Tipareste:"2.Afiseaza lista completa\n";

Tipareste:"3.Adauga piesa\n";

Tipareste"4.Cauta dupa cod\n";

Tipareste"5.Cauta dupa mecanism\n";

Tipareste"6.Cauta dupa importator\n";

Tipareste"7.Cauta dupa tip\n";

Tipareste"0.Iesire aplicatie\n";

Sfarsitsubalgoritm.

Subalgoritm print\_to\_file(int cod, string tip, string mec, string imp, float pret, float pret\_tva, float tva) este:

inf.open("data.txt",ios::app);

Daca !inf.is\_open() atunci:

Tipareste:"Fisierul nu a putut fi deschis";

print\_to\_file=0;

Altfel:

Tipareste:"Fisierul este deschis!";

Sfarsitdaca.

Scrie in fisier cod, tip, mec, imp, pret, tva, pret\_tva;

inf.close();

Sfarsitsubalgoritm.

Subalgoritm afisare\_inf(piese \*p) este:

Tipareste: p->cod, p->tip,p->mecanism, p->importator, p->pret, p->tva;

Sfarsitsubalgoritm.

Codul sursa al programului:

//Nicorici Adrian, program care tine evidenta pieselor auto

#define \_CRT\_SECURE\_NO\_WARNINGS\_// la unele functii in Visual Studio apare eroare

//"Functia nu este sigura" si cu acest define se elimina acest mesaj de eroare.

#include <iostream>

#include <stdio.h>

#include <string>

#include <fstream>

//alte librarii de care am avut nevoie, string pt stringuri si fstream pt filestream.

using namespace std;

fstream inf;//variabila globala la fisier.

typedef struct piese{//structura cu datele din lista;

int cod;

string tip;

string mecanism;

string importator;

float pret;

float pret\_tva;

float tva;

struct piese \*urm;

}o\_piesa;

o\_piesa \*varf, \*cursor;

//prototipul functiilor

void creaza();

void aduaga\_dupa(int cod,string tip, string mec,string imp, float pret, float pret\_tva, float tva);

void get\_from\_file();

void meniu();

void print\_to\_file(int cod,string tip, string mec,string imp, float pret, float pret\_tva, float tva);

void citire\_piese();

void afisare\_meniu();

void afisare\_inf(piese \*p);

void cautare\_dupa\_cod(int c);

void cautare\_dupa\_mec(string mec);

void cautare\_dupa\_imp(string imp);

void cautare\_dupa\_tip(string t);

void afisare\_lista\_intreaga();

void creaza(){//creez lista

varf = NULL;//varful este null

cursor = varf;

}

void adauga\_dupa( int cod, string tip, string mec,string imp, float pret, float pret\_tva,float tva){

//functia de adaugare in lista;

piese \*p=new piese;//alocarea spatiului de memorie

p->cod = cod;

p->tip = tip;

p->mecanism = mec;

p->importator = imp;

p->pret = pret;

p->pret\_tva = pret\_tva;

p->tva = tva;

//memorarea datelor in variabile

if(varf == NULL){//daca varfu ii gol punem in varf

p->urm=varf;

varf=p;

cursor=p;

}

else {//varful nu este gol si punem informatiile dupa varf;

p->urm=cursor->urm;

cursor->urm=p;

cursor=p;

}

}

void get\_from\_file(){//preluarea datelor din fisier

int cod=0;//variabile in care memoram datele(temporar) apoi sunt adaugate in lista;

int c;

string t,m,imp;

float p,p\_t,tva;

inf.open("data.txt",ios::in);

if(!inf.is\_open()){//verificam daca s-a deschis sau nu fisierul(aici am fost ajutat de domnul si doamna Persa);

cout << "Fisierul nu a putut fi deschis!" << endl;

return;//daca nu sa putut deschide fisierul revenim la meniu

}

else{//fisierul a fost deschis

cout << "Fisierul este deschis!" << endl;

}

/\* incepem sa punem datele in lista si facem citirea din fisier

\*/

while(inf >> c >> t >> m >> imp >> p >> p\_t >> tva,!inf.eof()){

cod = 1;//codul devine 1 pt a sti daca lista este goala sau nu.

adauga\_dupa(c,t,m,imp,p,p\_t,tva);//adaugam in lista

}

if(cod == 0){//daca codul o ramas 0 inseamna ca nu este nimic in fisier si avem o lista goala

cout << "Lista goala!" << endl;

}

inf.close();//inchidem fisierul

}

void meniu(){//textul din meniu cele 8 optiuni posibile

cout << "1.Citeste lista pieselor " << endl;

cout << "2.Afiseaza lista completa " << endl;

cout << "3.Adauga piesa" << endl;

cout << "4.Cauta dupa cod" << endl;

cout << "5.Cauta dupa mecanism" << endl;

cout << "6.Cauta dupa importator" << endl;

cout << "7.Cauta dupa tip" << endl;

cout << "0.Iesire aplicatie" << endl;

}

void print\_to\_file(int cod,string tip, string mec,string imp, float pret, float pret\_tva, float tva){//datele care sunt noi sunt tiparite in fisier;

inf.open("data.txt",ios::app);//deschidem fisierul data.txt in mod append

if(!inf.is\_open()){//verificam daca s-a putut deschide fisierul

cout << "Fisierul nu a putut fi deschis!" << endl;

return;

}

else{//fisierul a fost deschis

cout << "Fisierul este deschis!" << endl;

}

inf <<" " << cod << " " << tip << " " << mec << " " << imp << " " << pret << " " ;//printam in fisier

inf << tva << " " << pret\_tva << " " << endl;//printam in fisier

inf.close();//inchidem fisierul;

}

void citire\_piese(){//citim piesele

//cand se face selectia 3. din meniu se face apel la aceasta functie si se citesc informatiile

int c;

string t,m,imp;

float pr,p\_tva,tva;

cout << "Introduceti codul piesei: " << endl;

cin >> c;

cout << "Introduceti tipu piesei: " << endl;

cin >> t;

cout << "Din ce sistem mecanic face parte piesa: " << endl;

cin >> m;

cout << "Importator: " << endl;

cin >> imp;

cout << "Pretul fara tva: "<< endl;

cin >> pr;

cout << "Cota tva: " << endl;

cin >> tva;

p\_tva=pr+((pr\*tva)/100);//calculam pretul cu tot cu tva

adauga\_dupa(c,t,m,imp,pr,p\_tva,tva);//bagam in lista

print\_to\_file(c,t,m,imp,pr,p\_tva,tva);//printam in fisier;

}

void afisare\_meniu(){//aici e inima programului, meniul si legaturi intre toate functiile

int x;

string m;

x=-1;//initializarea lui x cu o valoare oarecare pt comparatie indicat sa fie initializat cu orice valoare diferita de 0;

while(x!=0){

meniu();//afisam meniul

cin >> x;//citim x;

switch (x)

{

case 0://cazul 0 iesim din aplicatie

cout << "Iesire..." << endl;

break;

case 1://preia informatiile din fisier si creaza lista;

get\_from\_file();

break;

case 2://afisarea listei intregi

afisare\_lista\_intreaga();

break;

case 3://adaugare piesa

citire\_piese();

break;

case 4://cautare dupa cod

int c;

cout << "Introduceti codul: " << endl;

cin >> c;

cautare\_dupa\_cod(c);

break;

case 5://cautare dupa mecanism

cout << "Introduceti mecanismul ";

cin >> m;

cautare\_dupa\_mec(m);

break;

case 6://cautare dupa importator;

cout << "Introduceti importatorul: ";

cin >> m;

cautare\_dupa\_imp(m);

break;

case 7://cautare dupa tip;

cout << "Introduceti tipu: " << endl;

cin >> m;

cautare\_dupa\_tip(m);

break;

default://in cazul in care x are alta valoarea decat (0,7);

cout << "Optiune gresita" << endl;

break;

}

}

}

void cautare\_dupa\_cod(int c){//functia care face cautarea dupa cod;

o\_piesa \*p;

p=varf;

int n=0;//n este pt a stii daca s-a gasit codul dorit sau nu;

while(p!=NULL){//citim toata lista(cazul in care exista 2 intrari cu acelasi cod sa apara ambele

if(c==p->cod){//daca codul cerut este acelasi cu un cod din lista

afisare\_inf(p);//afisam informatiile

n=1;//n se schimba pt a nu intra pe ramura if de mai jos

}

p=p->urm;

}

if(n==0){//daca n este 0 codul cautat nu exista sau ceva nu este in regula;

cout << "Codul nu a fost gasit!" << endl;

}

}

void afisare\_inf(piese \*p){//afisam toate informatiile

//apelul catre aceasta functie este facut doar din functiile de cautare si trimitem ca parametru un anumit nod din lista

cout << "Codul: ";

cout << p->cod << endl;

cout << "Tip: ";

cout << p->tip << endl;

cout << "Mecanism: ";

cout << p->mecanism << endl;

cout << "Importator: " ;

cout << p->importator << endl;

cout << "Pret fara tva: " ;

cout << p->pret << endl;

cout << "Pret cu tva: ";

cout << p->tva << endl;

}

void cautare\_dupa\_mec(string mec){//cautarea dupa mecanism

piese \*p;

p=varf;

int n=0;

while(p!=NULL){//parcurgem toata lista

if(mec== p->mecanism){//daca am gasit ceva afisam

afisare\_inf(p);//aici

n=1;

}

p=p->urm;

}

if(n==0){//n=0 inseamna ca nu este informatia ceruta sau ceva nu este in regula;

cout << "Informatia dorita nu a fost gasita!" << endl;

}

}

void cautare\_dupa\_imp(string imp){//cautarea dupa importator

//exact acelasi principiu ca la cautarea dupa mecanism doar ca reperul este importatorul;

piese \*p;

p=varf;

int n=0;

while(p!=NULL){

if(imp==p->importator){

afisare\_inf(p);

n=1;

}

p=p->urm;

}

if(n==0){

cout << "Informatia dorita nu a fost gasita" << endl;

}

}

void cautare\_dupa\_tip(string t){//cautarea dupa tip, exact la fel ca si cautarile dupa mecanism si importator;

o\_piesa \*p;

p=varf;

int n=0;

while(p!=NULL){

if(t==p->tip){

afisare\_inf(p);

n=1;

}

p=p->urm;

}

if(n==0){

cout << "Informatia ceruta nu a fost gasita" << endl;

}

}

void afisare\_lista\_intreaga(){//aici afisam toata lista;

piese \*p;

p=varf;

if(p==NULL){//daca varful este null inseamna ca lista este vida;

cout << "Lista este goala!" << endl;

}

else{//daca intra pe aceasta ramura inseamna ca avem ceva in lista;

while(p!= NULL){//afisam informatiile din lista;

cout << p->cod << " " << p->tip << " " << p->mecanism << " " << p->importator << " " << p->pret << " " ;

cout << p->tva << " " << p->pret\_tva << " " << endl;

p=p->urm;

}

}

}

int main(){//programul principal;

creaza();//apel catre functia creaza varful devine NULL avem repere pt a crea lista

afisare\_meniu();//afisam meniu si ne facem de cap cu acest mic program;

system("pause");

return 0;

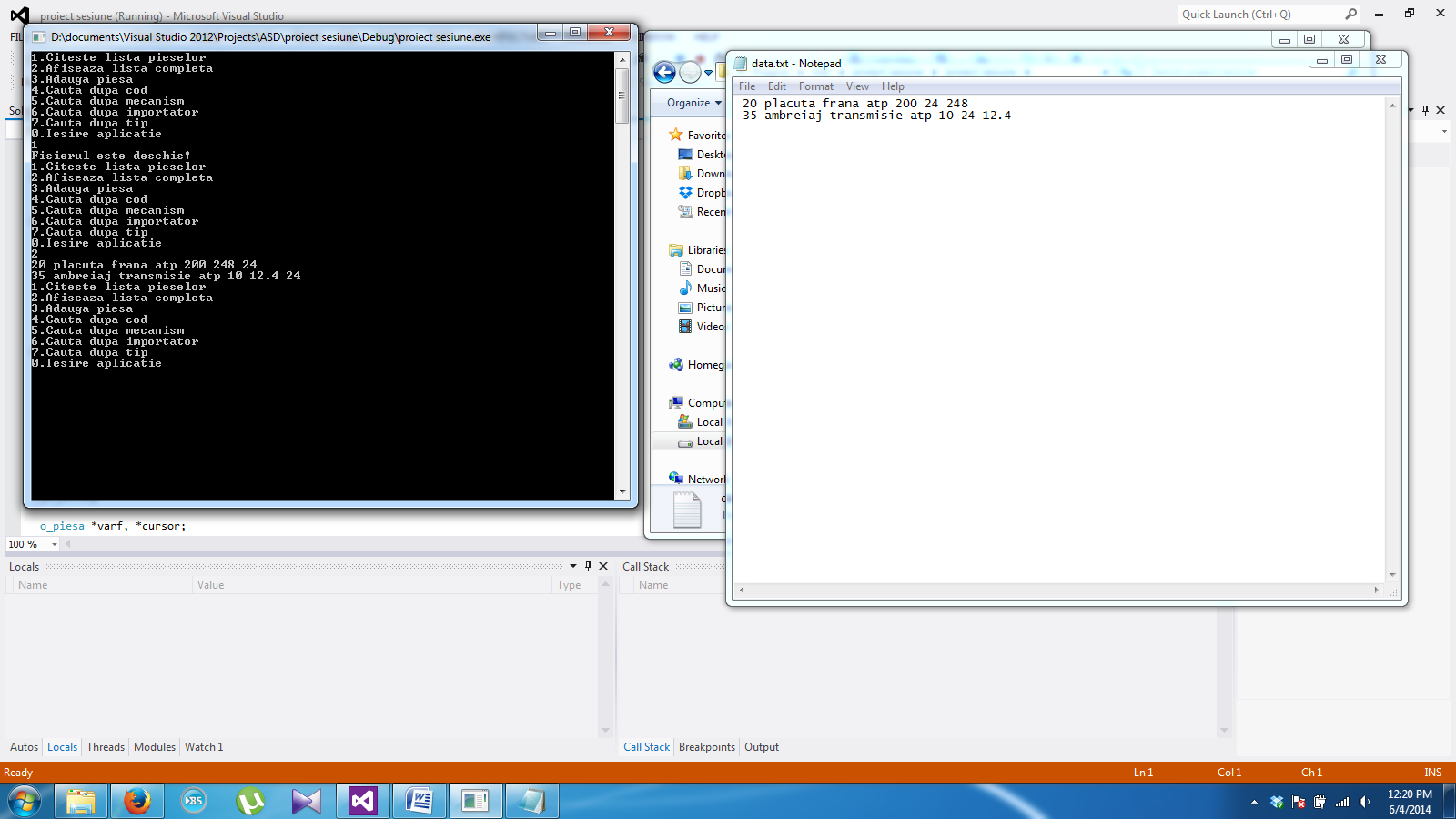
}

Testarea programului:

Ca si date de test am ales pentru inceput verificarea functiilor de adaugare in lista, cautare in lista, scriere si citire in fisier.

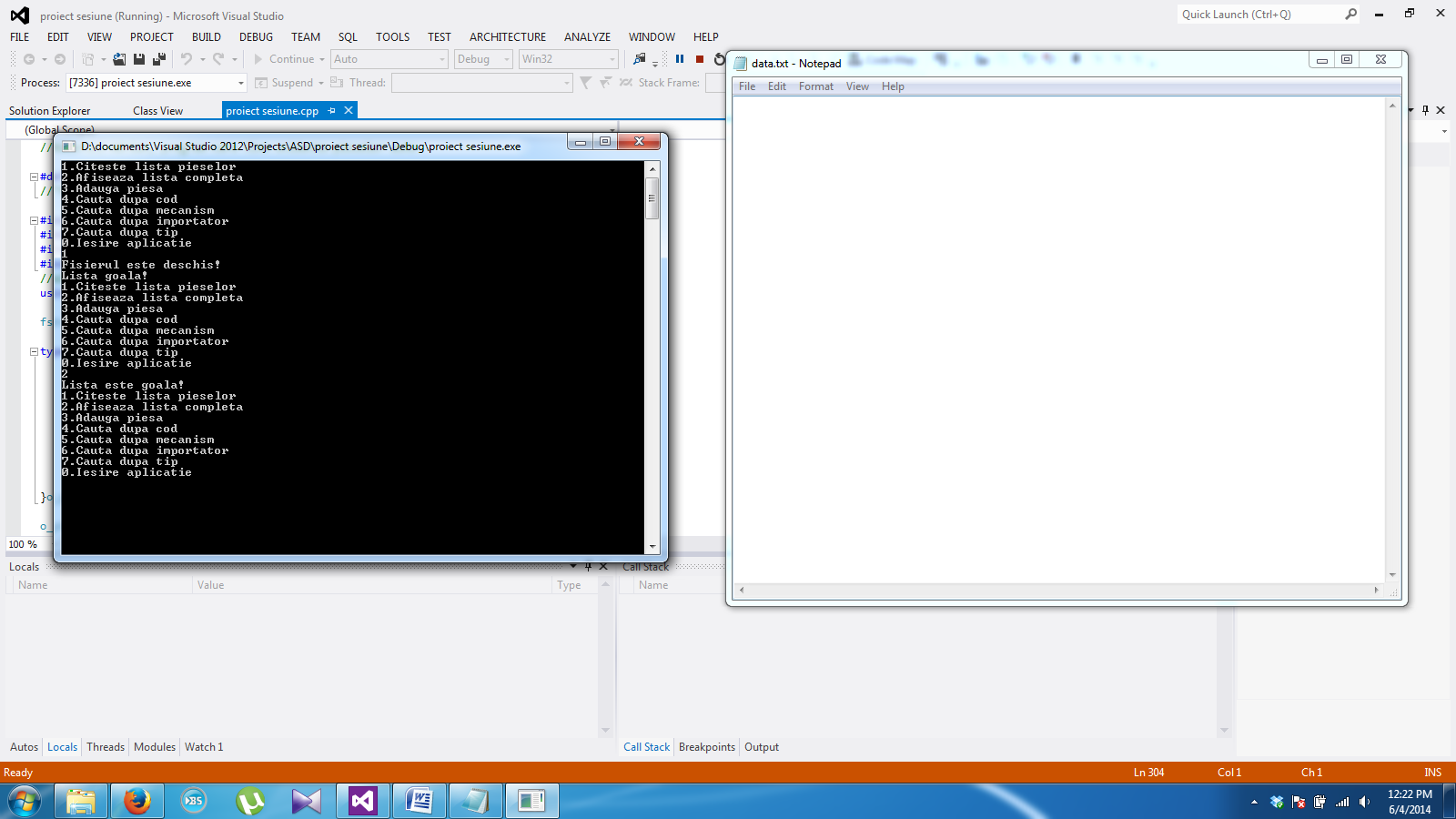
Primul test:

Fisierul data.txt are 2 linii, verificam daca cele 2 linii se pun in lista asa cum trebuie sau avem erori.



Dupa cum se vede si in poza informatiile au fost afisate corect dupa ce s-a citit fisierul text.

Urmatorul test ar fi sa vedem daca apare sau nu mesajul "fisier gol".



Mesajul este afisat, partea aceasta este ok.

Urmatoarele teste ar fi sa vedem functiile de cautare daca fac ceea ce trebuie sau nu, pentru aceasta vom adauga date in lista noastra(care acuma este goala).

Am adaugat datele:

20 placuta frana atp 200 24 248

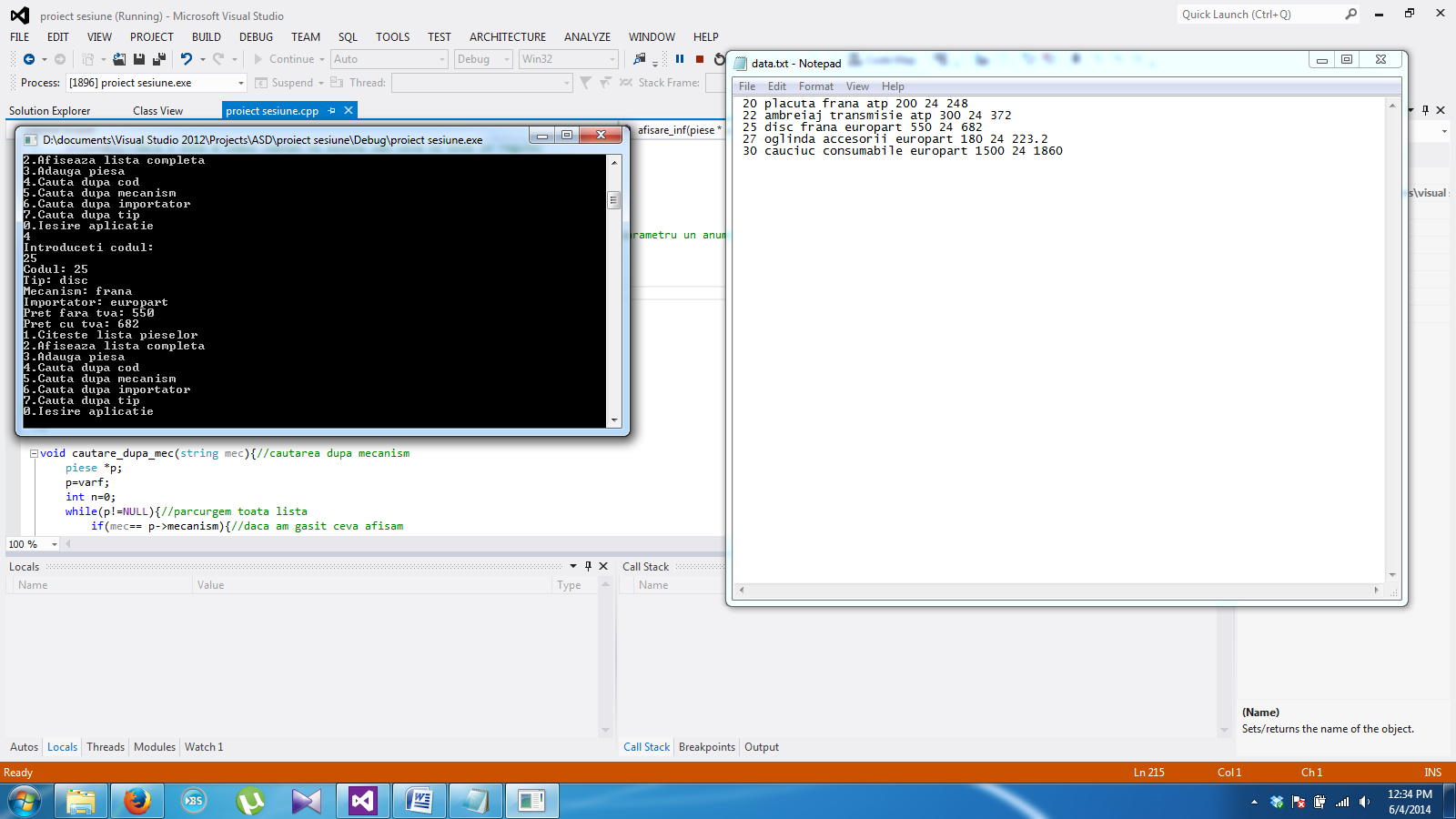
22 ambreiaj transmisie atp 300 24 372

25 disc frana europart 550 24 682

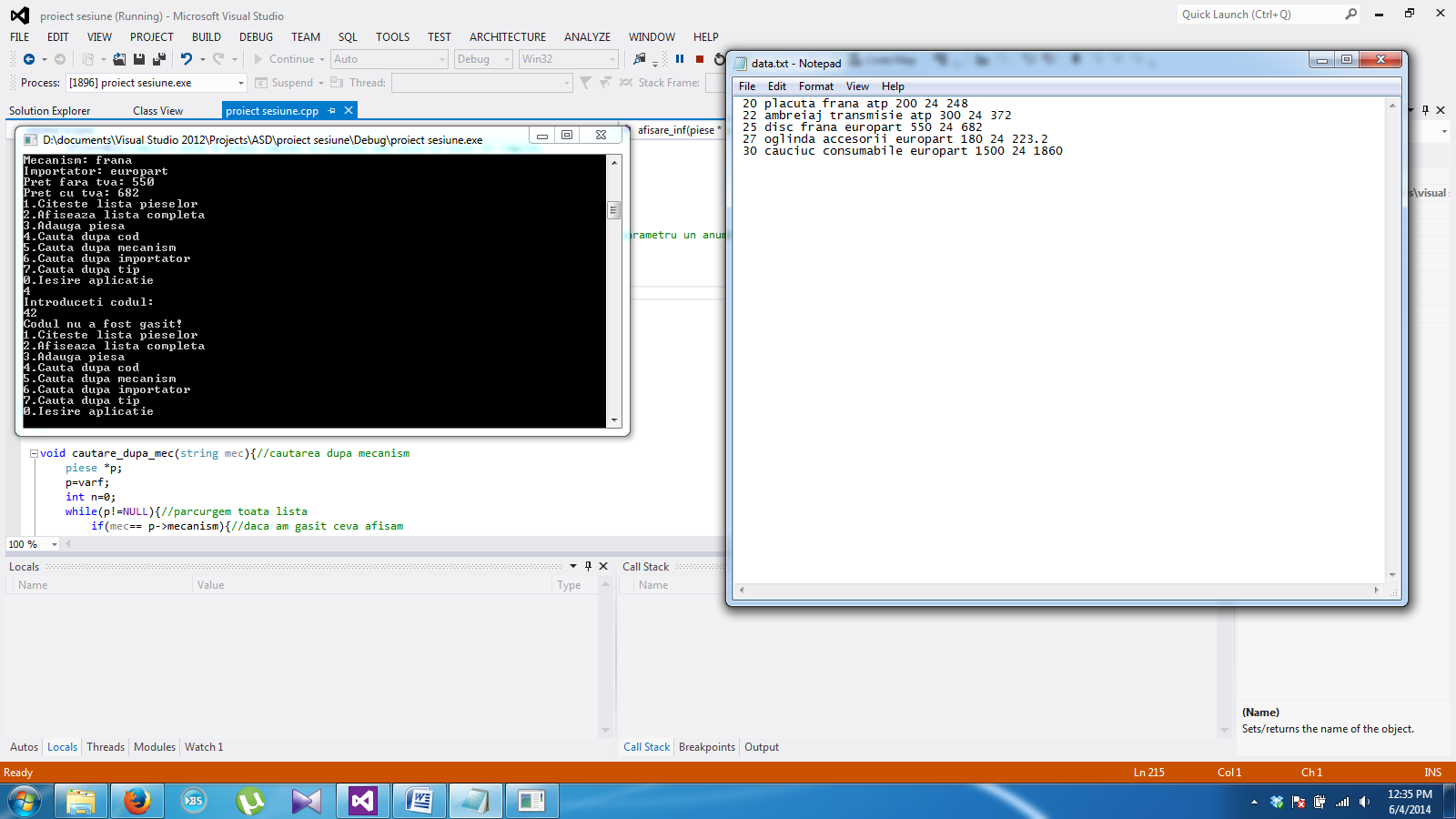
27 oglinda accesorii europart 180 24 223.2

30 cauciuc consumabile europart 1500 24 1860

Dupa care am selectat cautarea dupa cod, introducat codul 25:



Acuma cautam un cod care nu exista in lista noasttra, de exemplu 42:



Cautarea dupa cod functioneaza exact cum trebuie.

Acuma testam cautarea dupa mecanism, lista fiind aceeasi:

20 placuta frana atp 200 24 248

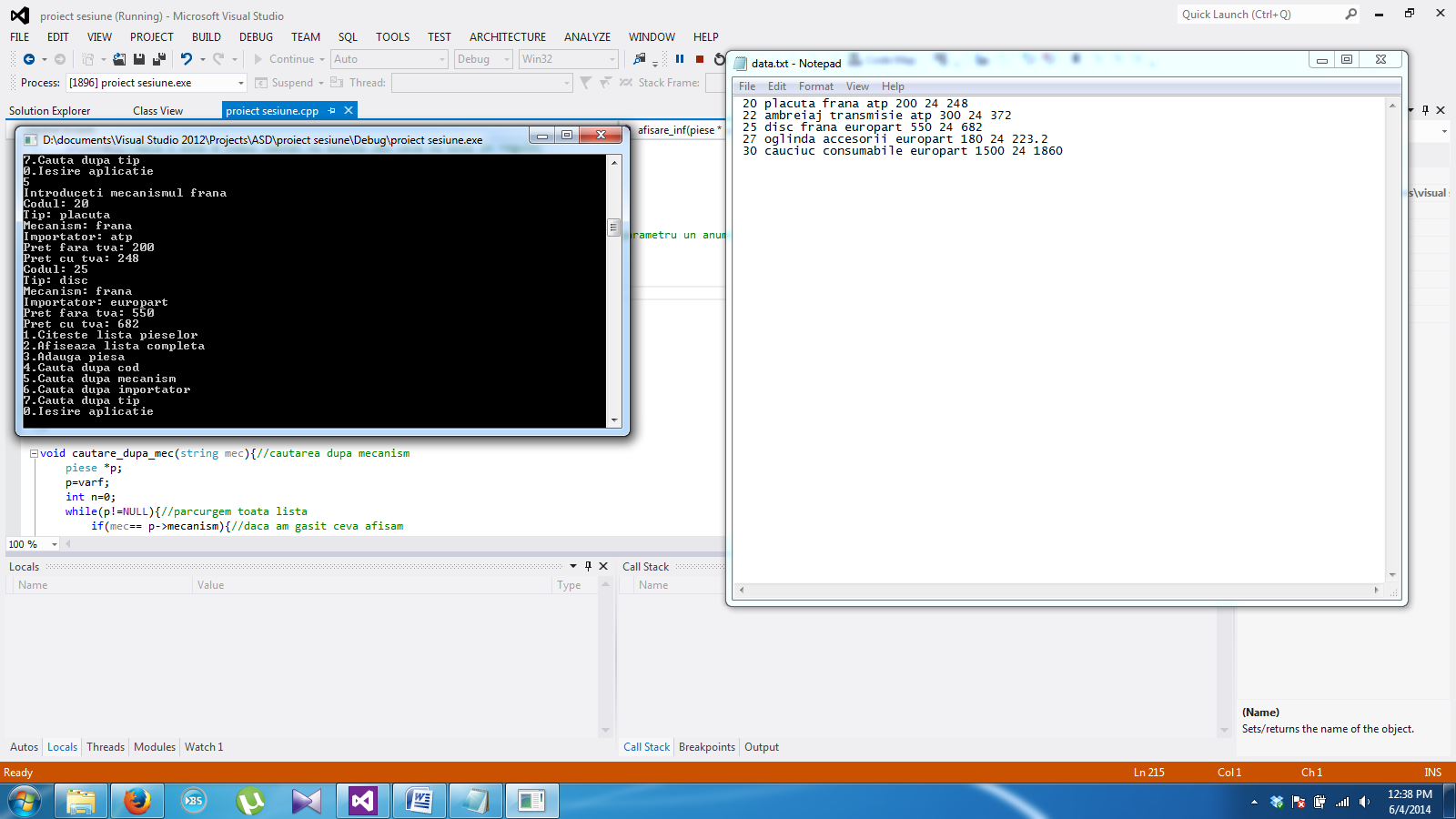
22 ambreiaj transmisie atp 300 24 372

25 disc frana europart 550 24 682

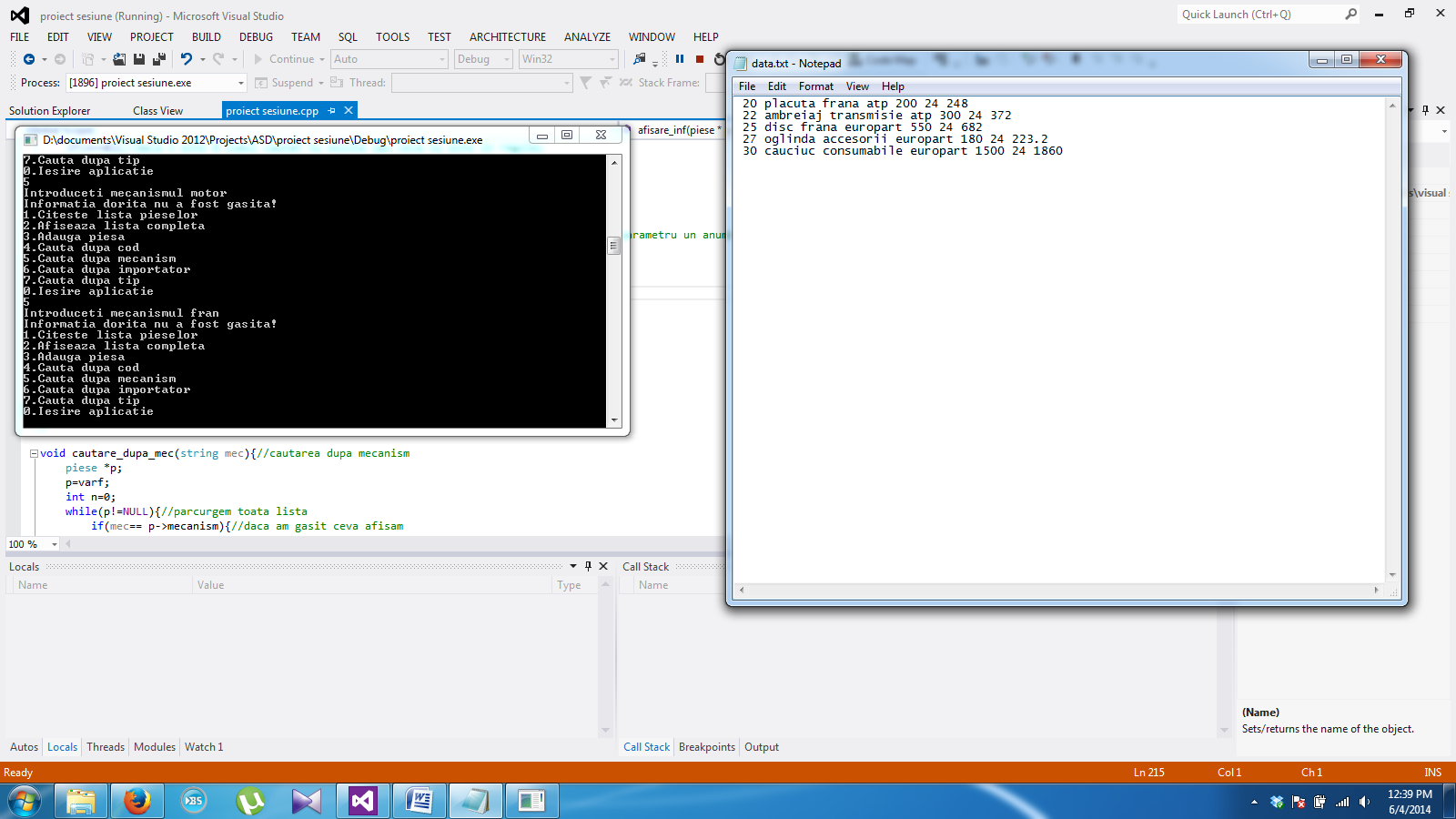
27 oglinda accesorii europart 180 24 223.2

30 cauciuc consumabile europart 1500 24 1860

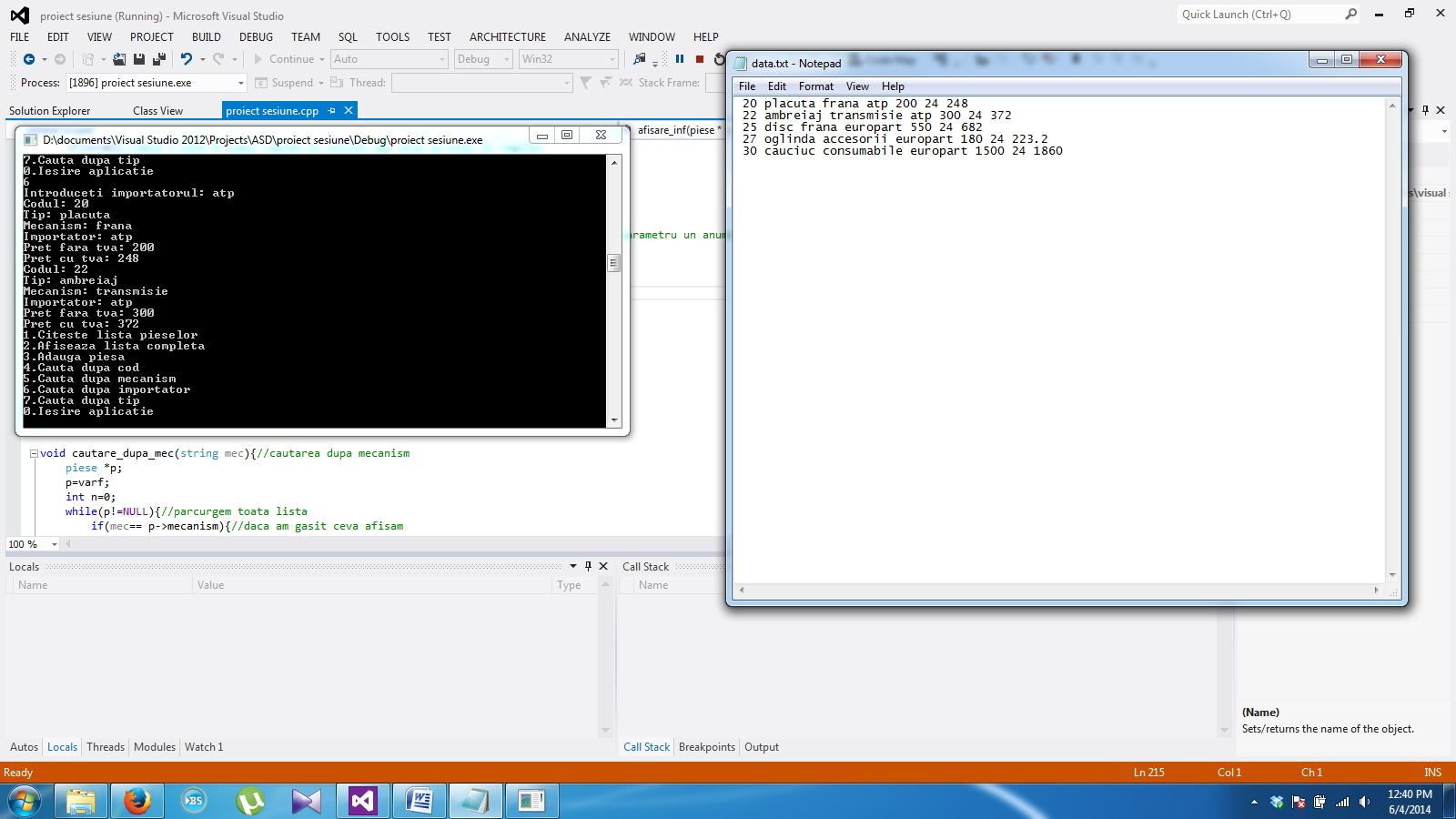
De exemplu in fisierul nostru text avem 2 piese care fac parte din sistemul de franare a masinii(placuta si disc) in momentul cand cautam dupa mecanism si criteriul de cautare este frana ar trebui sa avem informatiile despre ambele piese, vom vedea in imaginile ce urmeaza:



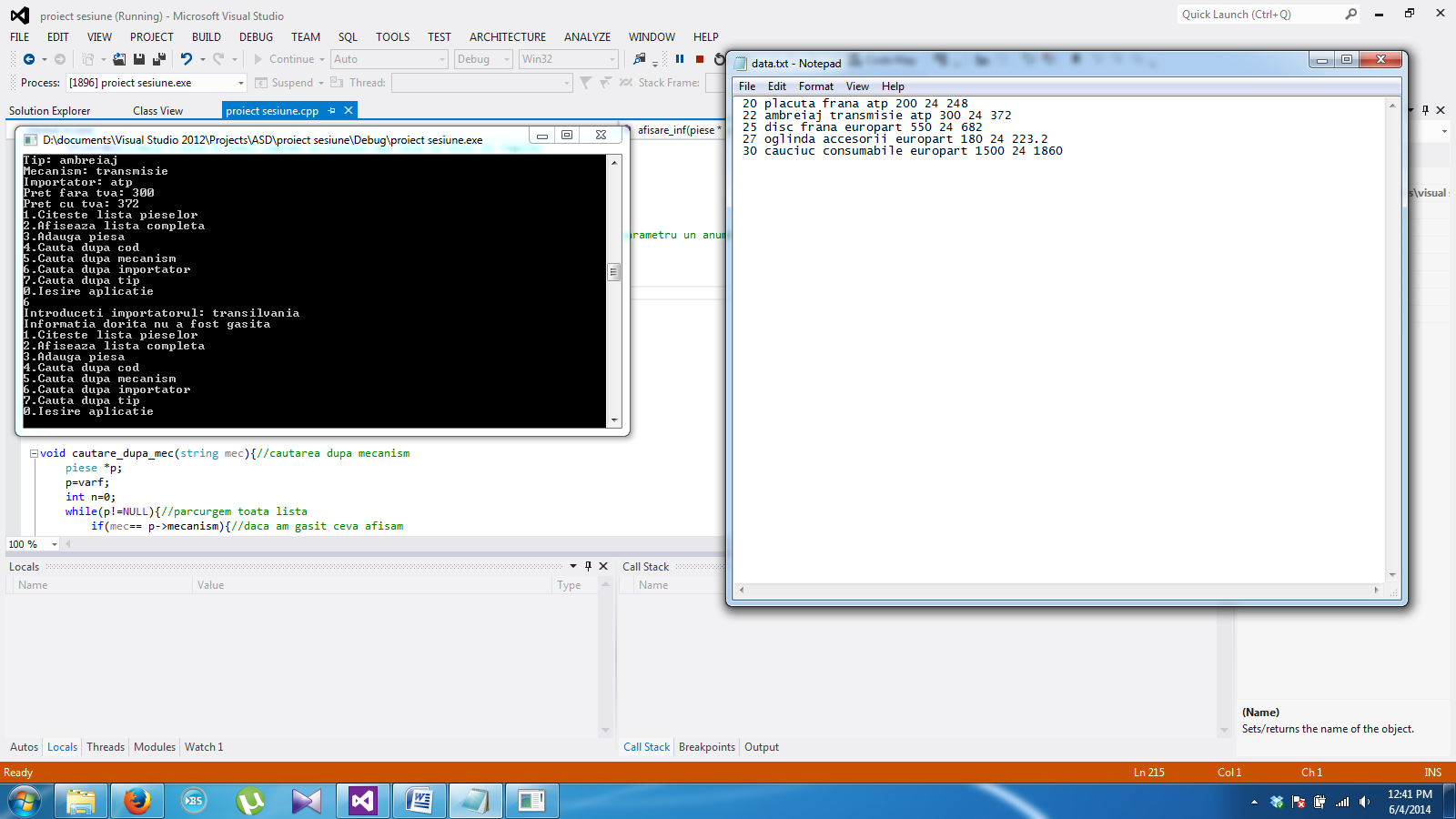
In poza se vedem ca avem informatiile despre ambele piese, acuma cautam ceva ce nu exista la noi in lista cum ar fi motor,sau daca scriem jumatate de cuvant:



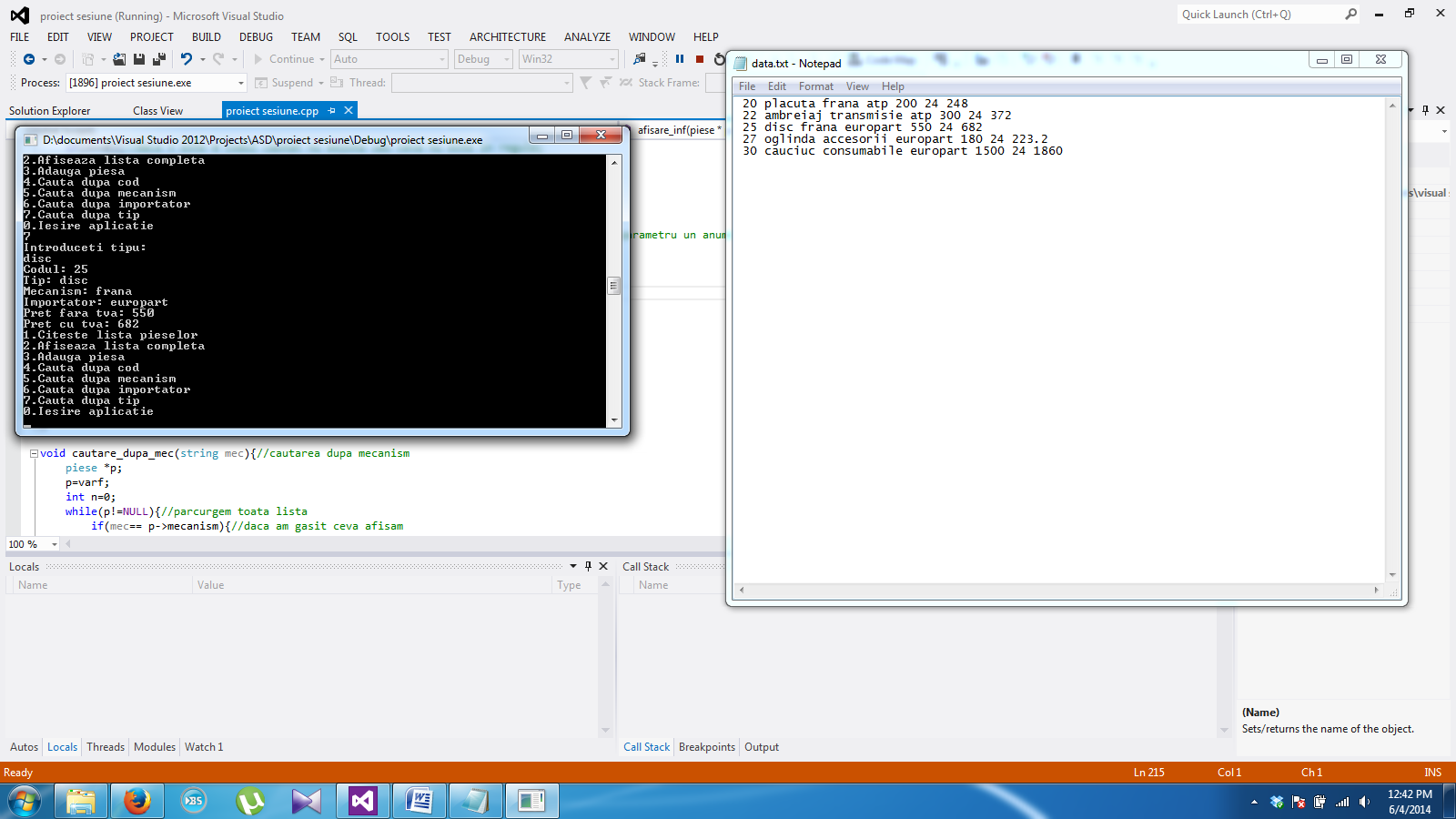
Urmatorul test va fi sa verificam functia de cautare dupa importator:



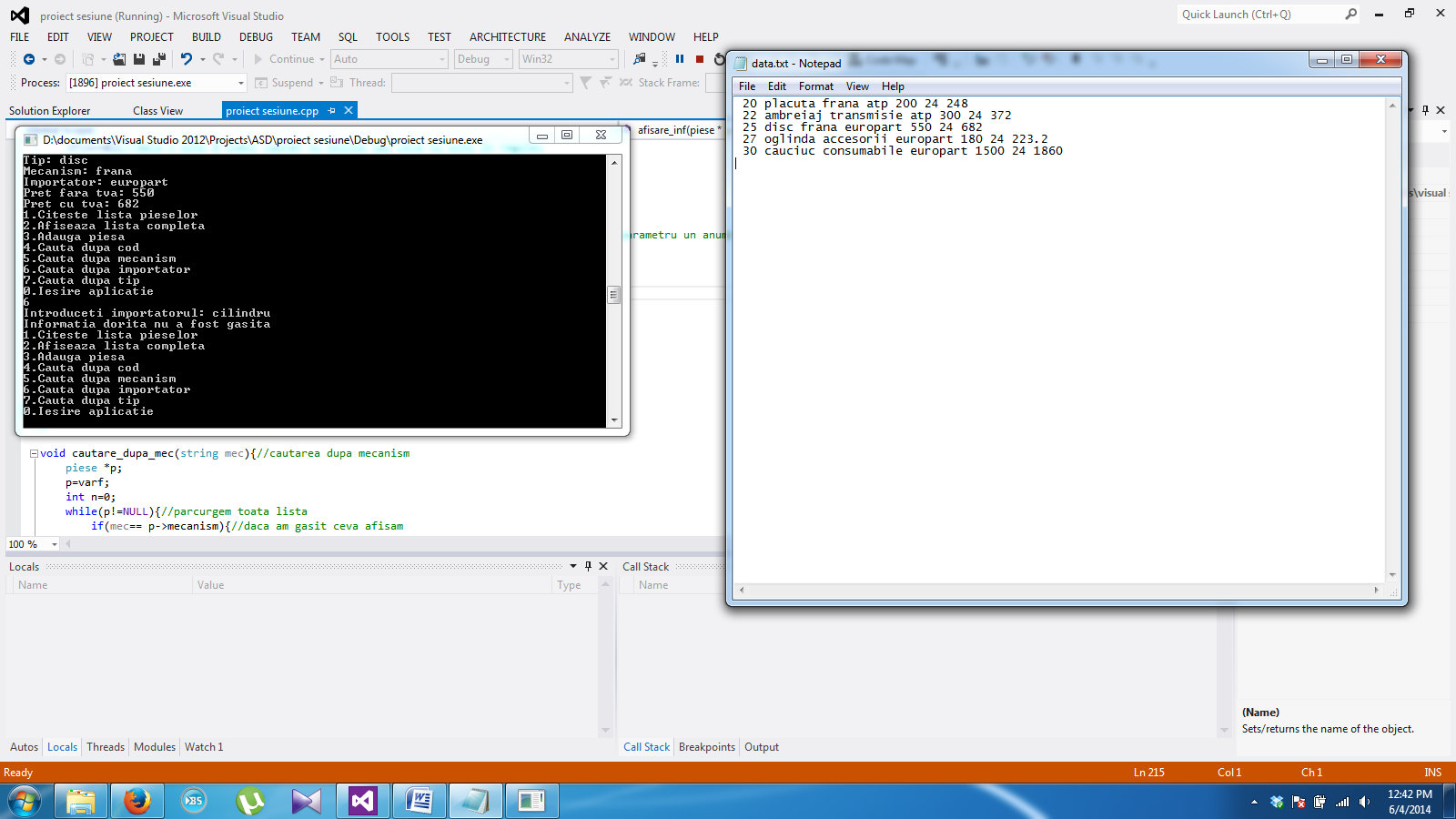
Iar daca importatorul nu este lista in noastra:



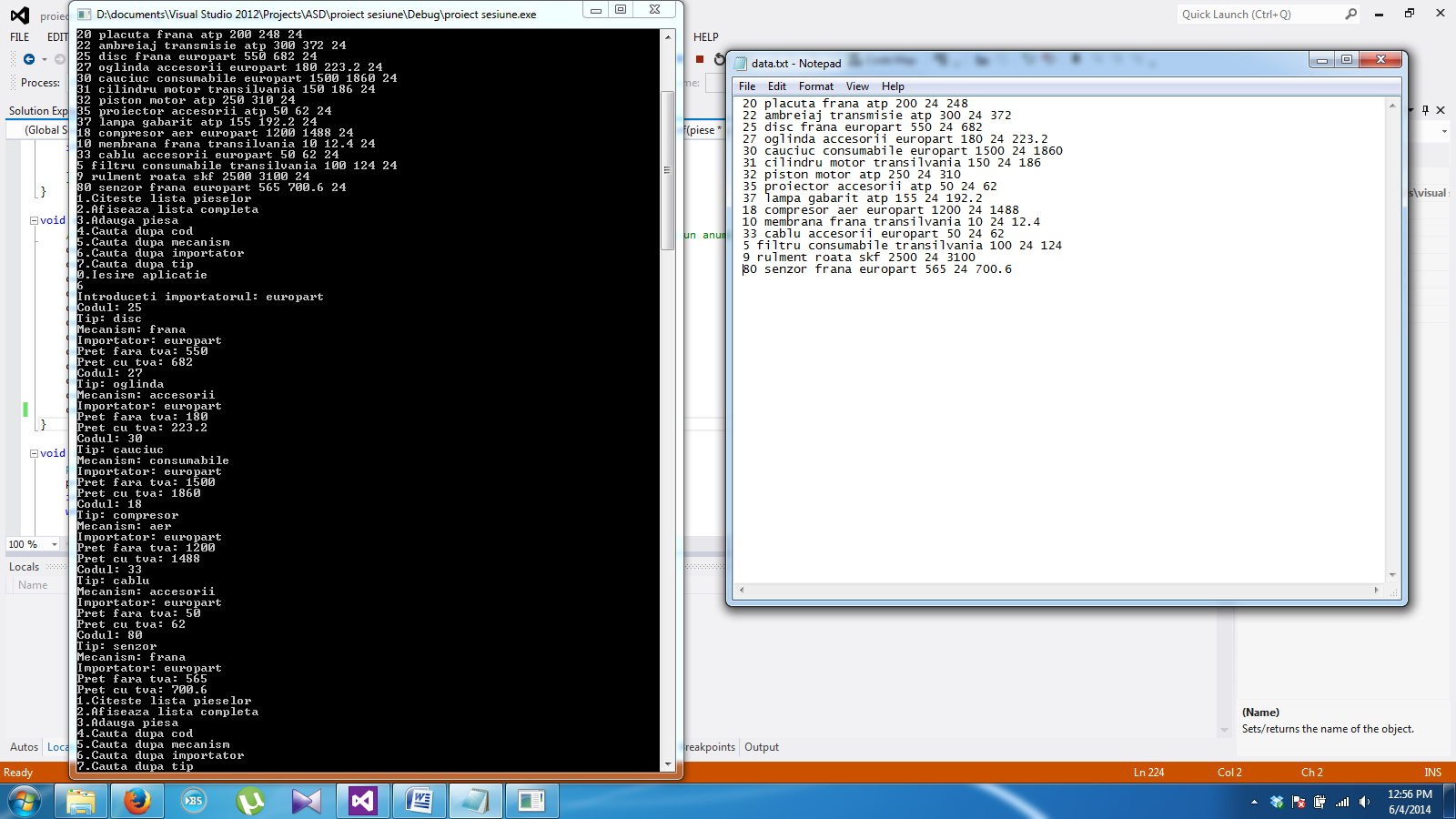
Ultima cautare, dupa tip:



Si daca nu avem acele informatii in lista:



Acuma verificam citirea/scrierea in/din fisier daca avem o lista cu 15-20 de noduri.



Functioneaza la fel de bine.

Documentatia programului - realizare si exploatare.

Realizarea programului:

Acest program salveaza intr-un fisier text o evidenta a unor piese. La inceput am creat structura cu piese cu variabilele necesare pentru a tine aceste informatii, o variabila intreaga pentru cod, 3 stringuri pt tip, mecanism si importator si 3 numbere in virgula mobila(float) pentru pret, pretul cu tva si cota tva.Dupa aceea am facut functia creaza() care initializeaza varful cu NULL, am facut facut functia de adaugare in lista si functia care citeste informatiile pentru fiecare nod. In acest punct m-am oprit si am verificat ca tot sa mearga bine si sa nu existe erori. Dupa care am continuat cu functiile de cautare, ca si baza a fost cursul de la Algoritmi si structuri de date(cautari in vector).Aici am verificat ca tot sa meaga bine si sa nu existe erori dupa care am facut functiile de citire/scriere din/in fisier si meniu. Am verificat totul si dupa am scos erorile aparute pe parcurs.

Exploatarea programului:

Programul are un meniu cu 8 alegeri posibile.

1."Citeste lista pieselor"-citirea listei cu piese, adica deschiderea si citirea fisierului data.txt, apoi fiecare rand se salveaza intr-o lista.Daca fisierul text este gol, va aparea mesajul "Lista este goala!", daca fisierul nu este gol va aparea mesajul "Fisierul a fost deschis cu succes".

Pentru a selecta aceasta optiune tastati 1 si apasati enter.

2."Afisare lista completa" - afisam lista incarcata in memoria ram, daca nu s-a selectat 1. din meniu sau nu s-a adaugat nici o piesa in lista, aceasta lista va aparea ca fiind goala.

Pentru a selecta aceasta optiune tastati 2 si apasati enter.

3."Adauga piesa" - aceasta optiune adauga o piesa in lista. Pentru a selecta tastati 3 si apasati enter. Dupa aceasta vi se vor cere urmatoarele informatii:

a) Codul piese - un nr intreg cuprins intre -32000 si +32000, ramane la alegerea dumneavoastra ce cod folositi pentru fiecare piesa.

b) Tipul piesei - ce piesa este, disc, placute, cablu, furtun.Indicat sa se foloseasca un singur cuvant.

c) Din ce sistem mecanic face parte piesa - sistemul de frana, transmisie, motor.

d) Pretul fara tva - se introduce pretul care nu contine tva

e)Cota tva - cat este valoarea tva-ului.

Dupa ce s-au introdus aceste date se vor salva in lista si se vor tipari in fisier.

Pentru urmatoarele 3 optiuni este indicat ca in lista sa existe date.

4."Cauta dupa cod" - este o cautare dupa cod in lista deja creata, doriti sa cautati o piesa si stiti doar codul selectati optiunea 4 si o sa apara toate informatiile despre piesa cu codul respectiv.

5."Cauta dupa mecanism" - exact la fel ca la punctul 4 doar criteriul de cautare este dupa mecanismul piesei.

6."Cauta dupa importator" - exact la fel ca la punctul 4 doar criteriul de cautare este dupa importator.

7."Cauta dupa tip" - exact la fel ca la punctul 4 doar criteriul de cautare este dupa tipul piesei.

0."Iesire aplicatie" - programul se va inchide.

Fisierul text se gaseste in acelasi folder cu acest program, il puteti citii oricand doriti fara sa deschideti aceasta aplicatie.