***Ministerul Educaţiei,Tineretului si Sportului a Republicii Moldova***

### **Universitatea Tehnică a Moldovei**

***Facultatea Calculatoare, Informatică si Microelectronică***

Referat

Lucrarea de laborator Nr.5

***Tema:* Moştenirea şi compoziţia**

**Varianta 10**

## A efectuat

## studentul grupei TI-161

***Grosu Elena***

***A verificat***

***Asist. Univ. Botnaru S.***

**Tema: Moştenirea multiplă**

**Scopul lucrării:**

* Studierea regulilor de determinare a moştenirii multiple;
* Studierea avantajelor şi neajunsurilor moştenirii multiple;
* Probleme legate de utilizarea moştenirii multiple;
* Studierea rezolvării problemelor;

**Întrebări de control:**

**Care sunt avantajele mostenirii multiple?**

– Mostenirea multipla reprezinta mostenirea a 2 sau mai multe clase , ceea ce nu ne permite mostenirea obisnuita.

**Care sunt problemele la realizarea moștenirei multiple?**

- Una din problemele mostenirii multiple consta in acea ca atunci cind ambele clase pe care le mostenim au acelasi parametru . Fiind ambele mostenite compilatorul nu stie cu care sa opereze atunci cind dorim s-o initializam sau schimbam.

**Clasifcați problemele de mostenirea multiplă** -

1. Au aceeasi parametri
2. Sunt clase inrudite

**Cum se rezolva problemele de moștenirea multiplă?**

1. Indicam clasa( c.A::x=10; c.B::x=15; )
2. Se utilizeaza forma de mostenire virtuala

**De ce este util mostenirea virtuala?**

**Sarcina**

Pentru toate variantele este necesar de creat două programe, care ar ilustra ambele exemple date mai sus de moştenire multiplă.

Varianta 10

1. Să se creeze, o ierarhie de moştenire: vehicul, camion – maşina de teren.
2. Să se creeze, o ierarhie de moştenire: automobil - vehicul, camion – maşină de teren.

#include <iostream>

#include <cstring>

#include <stdio.h>

#include <process.h>

#include <stdlib.h>

#include <conio.h>

#include<ctime>

#include "windows.h"

using namespace std;

class Vehicul{

protected:

char \*nume;

char \*producator;

int pret;

public:

Vehicul(char \*c,char \*n,int p)

{

producator=new char[strlen(c)+1];

strcpy(producator,c);

nume=new char[strlen(n)+1];

strcpy(nume,n);

pret=p;

}

~Vehicul()

{

delete[] producator;

delete[] nume;

}

void data1()

{

cout<<"Compania care produce : "<<producator;

cout<<endl<<"Nume : "<<nume;

cout<<endl<<"Pret : "<<pret;

}

};

class Camion{

protected:

int capacitate;

char \*dimensiune; //usor, mediu, gabarit

public:

Camion(int cap, char \*dim)

{

capacitate=cap;

dimensiune=new char[strlen(dim)+1];

strcpy(dimensiune,dim);

}

void data2()

{

cout<<endl<<"Aceasta este un camion de capacitatea : "<<capacitate;

cout<<endl<<"si dimensiunea : "<<dimensiune;

}

~Camion()

{

delete[] dimensiune;

}

};

class Masina\_de\_teren:public Vehicul, public Camion

{

int an\_prod;

char \*tara;

public:

Masina\_de\_teren(char \*c,char \*n, int p,int cap, char \*dim,int x, char \*t):Vehicul(c,n,p),Camion(cap,dim)

{

an\_prod=x;

tara=new char[strlen(t)+1];

strcpy(tara,t);

}

~Masina\_de\_teren()

{

delete [] tara;

}

void Afisare()

{

Vehicul:data1();

Camion:data2();

cout<<endl<<"Anul de producere : "<<an\_prod;

cout<<endl<<"Tarea de producere : "<<tara;

}

} ;

int main()

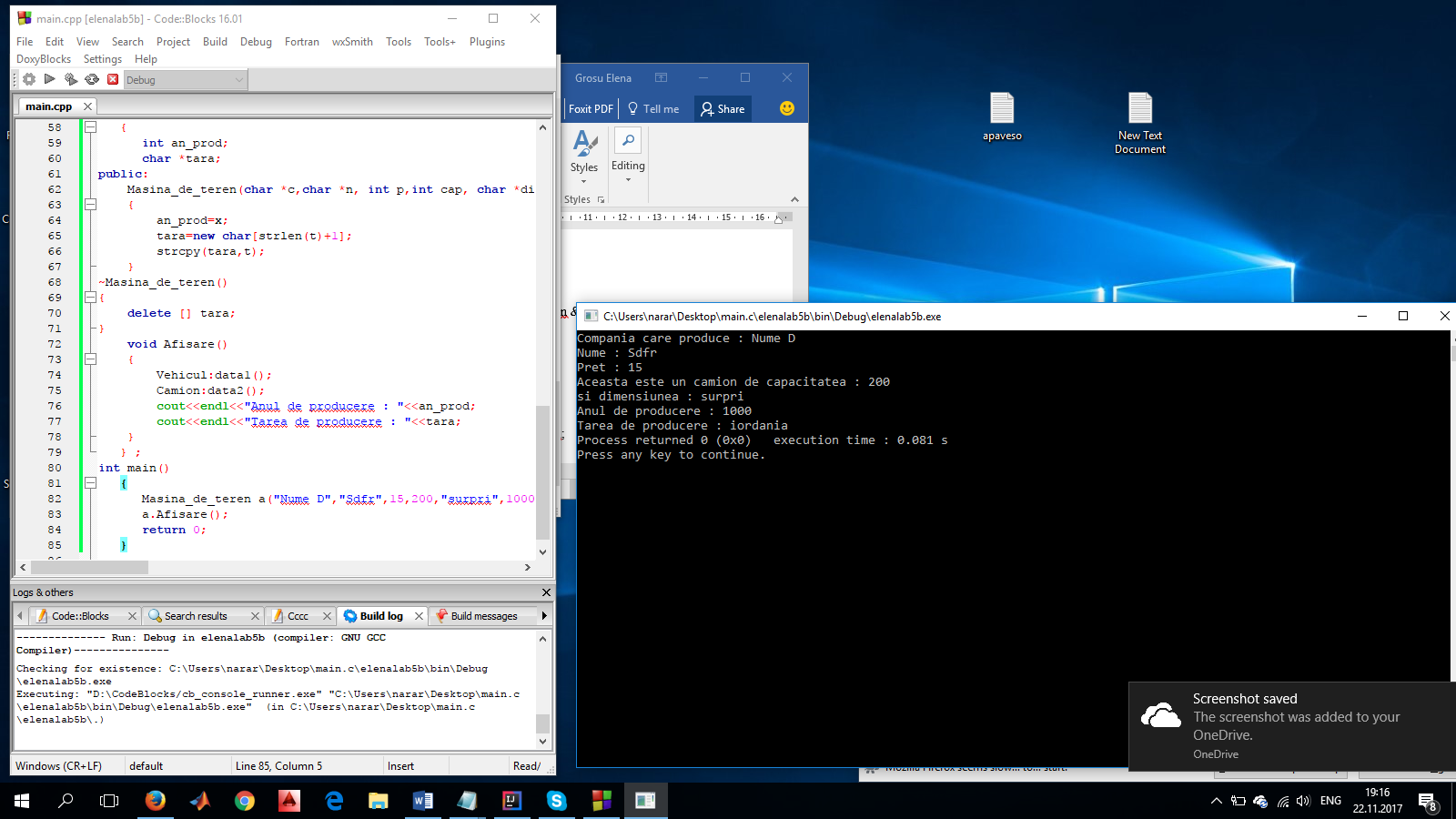
{

Masina\_de\_teren a("Nume D","Sdfr",15,200,"surpri",1000,"iordania") ;

a.Afisare();

return 0;

}



b)

#include <cstring>

#include <stdio.h>

#include <process.h>

#include <stdlib.h>

#include <conio.h>

#include<ctime>

#include "windows.h"

using namespace std;

class Automobil

{

protected:

char \*nume;

char \*producator;

int pret;

public:

Automobil(char \*n, char \*p, int pr)

{

nume=new char[strlen(n)+1];

strcpy(nume, n);

producator=new char[strlen(p)+1];

strcpy(producator, p);

pret=pr;

}

~Automobil(){delete []nume; delete[]producator;}

};

class vehicul:virtual public Automobil

{

protected:

char \*combustibil;

int nrpers;

vehicul(char \*cm,int nrp):Automobil(NULL,NULL,NULL)

{

nrpers=nrp;

combustibil=new char[strlen(cm)+1];

strcpy(combustibil,cm);

}

~vehicul(){delete[]combustibil;}

};

class camion:virtual public Automobil

{

protected:

int tonaj;

char \*compproprietar;

camion(int tj,char \*cmp):Automobil(NULL,NULL,NULL)

{

tonaj=tj;

compproprietar= new char[strlen(cmp)+1];

strcpy(compproprietar, cmp);

}

~camion(){delete[]compproprietar;}

};

class masina\_de\_teren:public vehicul, public camion

{

public:

int an\_producere;

masina\_de\_teren(int an=0,char \*n=NULL, char \*p=NULL,int pr=0,char \*cm=NULL, int nrp=0,int t=0,char \*cmp=NULL):

vehicul(cm,nrp),camion(t,cmp),Automobil(n,p,pr)

{

an\_producere=an;

}

friend ostream &operator<<(ostream &output,masina\_de\_teren &p )

{

output<<"Nume: "<< p.nume<<endl;

output<<"Producator:"<<p.producator <<endl;

output<<"Pret: "<<p.pret <<endl;

output<<"Combustibil: "<<p.combustibil <<endl;

output<<"Nr. de persoane: "<<p.nrpers <<endl;

output<<"Tonaj: "<<p.tonaj <<endl;

output<<"Compania proprietar:"<<p.compproprietar <<endl;

output<<"An de producere: "<<p.an\_producere <<endl;

output<<endl;

return(output);

}

};

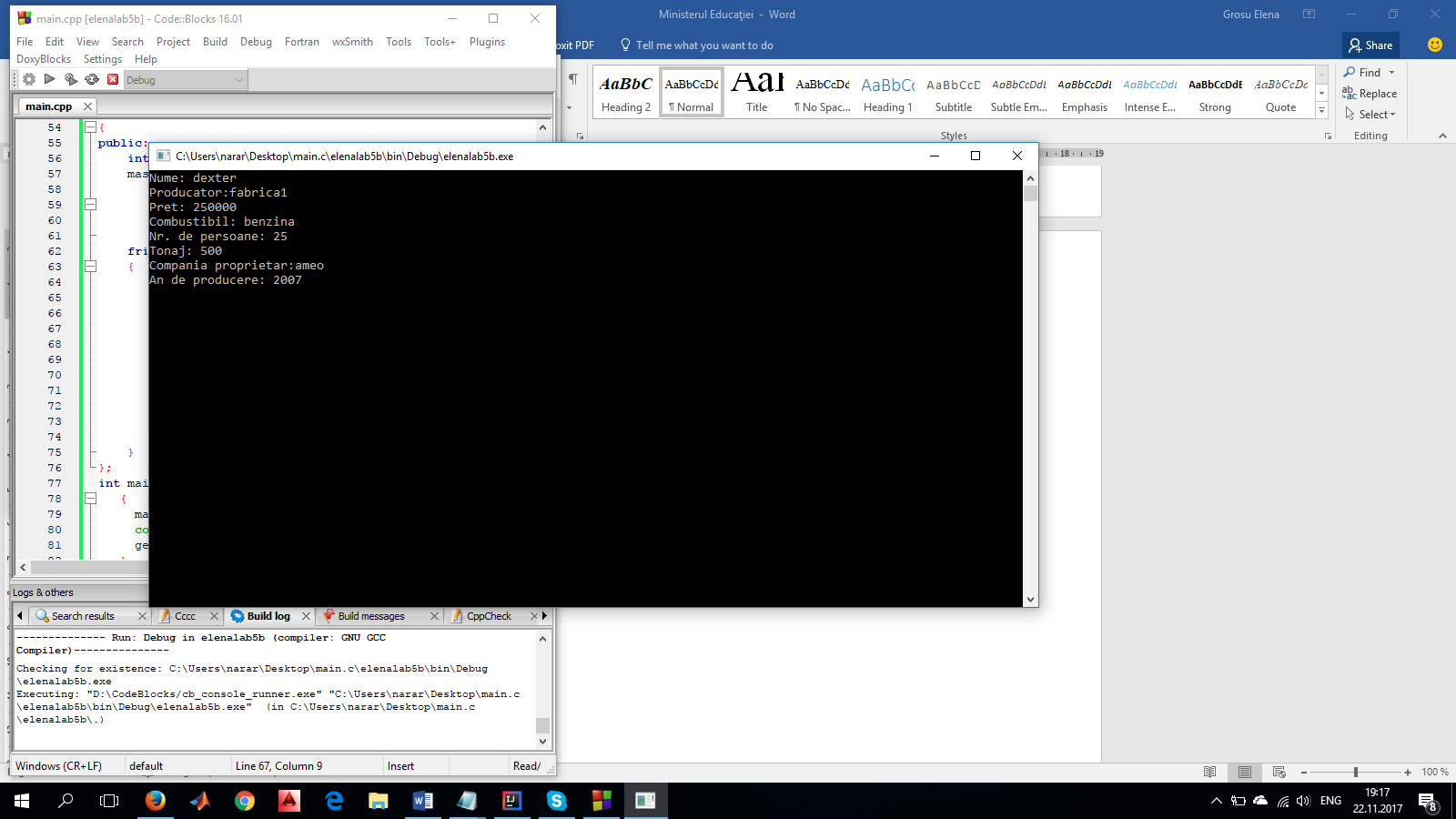
int main()

{

masina\_de\_teren y(2007,"dexter","fabrica1",250000,"benzina", 25,500,"ameo");

cout<<y;

getch(); }



**Concluzie:**

În urma efectuării acestei lucrări de laborator am făcut cunoștință cu principiile de lucru a moștenirii multiple, am învățat cum să construim o clasă derivată care are mai mult de o clasă de bază. Deasemenea am făcut cunoștință cu problema diamantului și cu modul de tratare a ambiguităților.