Ministerul Educaţiei și Cercetării

al Republicii Moldova

Universitatea Tehnică a Moldovei

RAPORT

# Lucrarea de laborator Nr4

**LA DISCIPLINA „PROGRAMAREA ORIENTATĂ PE OBIECTE”**

A efectuat: st. gr. TI-211 Popa Cătălin

A verificat: Lisnic Inga

UTM, Chișinău 2021

**Tema:** Moştenirea şi compoziţia

**Scopul lucrării:**

* studierea moştenirii, avantajele şi dezavantajele;
* studierea compoziţiei;
* studierea regulilor de definire a moştenirii şi compoziţiei;
* studierea formelor de moştenire;
* studierea iniţializatorilor;
* principiul de substituţie;

moştenirea şi compoziţia – ce să alegem.

Varianta 10

а) De creat o ierarhie a claselor *transport* – *transport aerian – elicopter*. Determinaţi fluxul de ieşire şi fluxul de intrare, constructorul de copiere, operatorul de atribuire prin funcţia corespunzătoare a clasei de bază.

#include <iostream>

#include <string>

using namespace std;

class transport

{

protected:

    int num = 0;

    int num\_transport;

    int num\_elicoptere;

public:

    transport()

    {

        num\_transport = 10;

        num\_elicoptere = 20;

    }

    transport(int n)

    {

        num\_transport = n;

        num\_elicoptere = n;

    }

    void transport1()

    {

        int t = num\_elicoptere \* num\_transport;

        cout << "\nNumarul de trasport este: " << t;

    }

    transport(transport &clone)

    {

        num\_elicoptere = clone.num\_elicoptere;

        num\_transport = clone.num\_transport;

    }

    transport& operator = (const transport& c);

};

transport &transport :: operator=(const transport &c)

{

    num\_elicoptere = c.num\_elicoptere;

    num\_transport = c.num\_transport;

}

class transport\_aerian : public transport

{

public:

    void transport\_aerian\_afisare()

    {

        cout << "\nNumarul de trasport: " << num\_transport;

    }

};

class elicoptere : public transport\_aerian

{

public:

    void elicoptere\_afisare()

    {

        cout << "\nNumarul de elicoptere este: " << num\_elicoptere;

    }

};

int main()

{

    cout << "\nCrearea claselor si definirea: \n";

    elicoptere a1;

    a1.transport\_aerian\_afisare();

    a1.elicoptere\_afisare();

    a1.transport1();

    cout << "\n\nConstructorul de copiere:\n";

    elicoptere a2(a1);

    a2.transport\_aerian\_afisare();

    a2.elicoptere\_afisare();

    a2.transport1();

    cout << "\n\nSupraincarcarea operatorului de atribuire + constructor de copiere:\n";

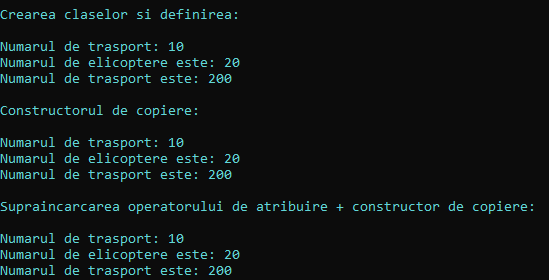
    elicoptere a3 = a1;

    a3.transport\_aerian\_afisare();

    a3.elicoptere\_afisare();

    a2.transport1();

}



b) Definiţi clasa *element chimic*, care conţine informaţia despre denumirea elementului şi proprietăţile ei chimice. Determinaţi clasa *medicamente*, care conţine diferite elemente chimice şi diferite cantităţi. Determinaţi constructorii, funcţiile fluxului de ieşire şi alte funcţii necesare.

#include <conio.h>

#include <iostream>

using namespace std;

class element\_chimic

{

protected:

    int num\_elements = 3;

    string name\_element;

    string simbol;

    int numar\_ordine;

    string serie\_chimica;

    int densitate;

    int grupa;

    int perioada;

public:

    element\_chimic()

    {

        for (int i = 0; i < num\_elements; i++)

        {

            name\_element = "Fier";

            simbol = "Fe";

            numar\_ordine = 26;

            serie\_chimica = "Metal";

            densitate = 7860;

            grupa = 8;

            perioada = 4;

        }

    }

    void citire()

    {

        for (int i = 0; i < num\_elements; i++)

        {

            cout << "\nIntroduceti numele elementului: \n";

            name\_element = "Fier";

            cout << "\nIntroduceti simbolul: \n";

            simbol = "Fe";

            cout << "\nIntroduceti numarul de ordine: \n";

            numar\_ordine = 26;

            cout << "\nSeria chimica: \n";

            serie\_chimica = "Metal";

            cout << "\nDensitatea: \n";

            densitate = 7860;

            cout << "\nGrupa: \n";

            grupa = 8;

            cout << "\nPerioada: \n";

            perioada = 4;

        }

    }

    void afisare()

    {

        cout << "\nDenumirea elementului: " << name\_element;

        cout << "\nSimbolul: " << simbol;

        cout << "\nNumarul de ordine: " << numar\_ordine;

        cout << "\nSeria chimica: " << serie\_chimica;

        cout << "\nDensitatea elementului: " << densitate;

        cout << "\nGrupa: " << grupa;

        cout << "\nPerioada: " << perioada << endl;

    }

};

class medicamente : public element\_chimic

{

protected:

public:

    void baza()

    {

        for (int i = 0; i < num\_elements; i++)

        {

            afisare();

        }

    }

};

int main()

{

    medicamente a1;

    a1.baza();

}

