TEMA LAB 3

CURIBAN SIMONA

Problema 2

#include<iostream>

#include<conio.h>

using namespace std;

//clasa de baza

class Mamifer {

char denumire[30];

public:

//constructorul clasei de baza

Mamifer(char \*denumire)

{

strcpy(Mamifer::denumire, denumire);

}

void afisareMamifer()

{

cout << "Numele mamiferului este:" << denumire << endl;

}

};

//clasa derivata

class Cangur :public Mamifer {

char regiune[30];

int varsta;

public:

//constructorul clasei derivate

Cangur(char \*denumire, char \*regiune, int varst) :Mamifer(denumire)

{

strcpy(Cangur::regiune, regiune);

Cangur::varsta = varst;

}

void afisareCg()

{

afisareMamifer();

cout << "Regiunea:" << regiune << "Varsta:" << varsta << endl;

}

};

//clasa derivata

class Urs :public Mamifer {

char zona[30];

int greutate;

public:

//constructorul clasei derivate

Urs(char \*denumire, char \*zona, int greut) : Mamifer(denumire)

{

strcpy(Urs::zona, zona);

Urs::greutate = greut;

}

void afisareU()

{

afisareMamifer();

cout << "Zona:" << zona << "Greutatea:" << greutate << endl;

}

};

int main()

{ //crearea unui obiect

Cangur cgr("Cangur brun", "Australia", 3);

Urs ur("Panda", "sudul Chinei", 100);

cgr.afisareCg();

ur.afisareU();

getch();

return 0;

}

Problema 4

#include<iostream>

#include<conio.h>

#include<string>

using namespace std;

class Persoana {

string nume;

string prenume;

int varsta;

public:

//constructor vid folosit pentru declararea vectorului

Persoana() {}

//constructor pentru initializarea variabilelor

Persoana(string nume, string prenume, int varsta)

{

this->nume = nume;

this->prenume = prenume;

this->varsta = varsta;

}

string nume1()

{

return this->nume;

}

//functie de afisare

void afisarePers()

{

cout << "Numele persoanei este:" << this->nume << endl;

cout << "Prenumele persoanei este:" << this->prenume << endl;

cout << "Varsta persoanei este:" << this->varsta << endl;

}

};

class Angajat : public Persoana {

unsigned int salar;

string departament;

public:

//constructor vid folosit pentru declararea vectorului

Angajat() {}

//constructor pentru initializarea variabilelor

Angajat(string nume, string prenume, int varsta, string departament, unsigned int salar) : Persoana(nume, prenume, varsta)

{

this->departament = departament;

this->salar = salar;

}

//functia de afisare

void afisareAng()

{

afisarePers();

cout << "Salarul angajatului este:" << this->salar << endl;

cout << "Departamentul angajatului este:" << this->departament << endl;

}

};

class Profesor : public Angajat {

int grad\_didactic;

string materia;

public:

//constructor vid folosit pentru declararea vectorului

Profesor() {};

//constructor pentru initializarea variabilelor

Profesor(string nume, string prenume, int varsta, int salar, string departament, unsigned int grad\_didactic, string materia) : Angajat (nume, prenume, varsta, salar, departament )

{

this->grad\_didactic = grad\_didactic;

this->materia = materia;

}

//functia de afisare

void afisareProf()

{

afisareAng();

cout << "Gradul didactic al profesorului este:" << this->grad\_didactic << endl;

cout << "Materia predata de profesor este:" << this->materia << endl;

}

};

class Inginer : public Angajat {

unsigned int vechime;

string functie;

public:

//constructor vid folosit pentru declararea vectorului

Inginer() {};

//constructor pentru initializarea variabilelor

Inginer(string nume, string prenume, int varsta , unsigned int salar, string departament, unsigned int vechime, string functie) : Angajat (nume, prenume, varsta, salar, departament)

{

this->vechime = vechime;

this->functie = functie;

}

//functie de afisare

void afisareIng()

{

afisareAng();

cout << "Vechimea este:" << this->vechime << endl;

cout << "Functia inginerului este:" << this->functie << endl;

}

};

class Student :public Persoana {

unsigned int media;

string specializare;

public:

//constructor vid pentru declararea vectorului

Student() {}

//constructor pentru initializarea variabilelor

Student(string nume, string prenume, int varsta, unsigned int media, string specializare) :Persoana(nume, prenume, varsta)

{

this->media = media;

this->specializare = specializare;

}

//functia de afisare

void afisareStud()

{

afisarePers();

cout << "Media studentului este:" << this->media << endl;

cout << "Specializarea studentului este:" << this->specializare << endl;

}

};

class StudentLicenta :public Student {

int AnStudiu;

string optional;

public:

//constructor vid pentru declararea vectorului

StudentLicenta() {};

//constructor pentru initializarea variabilelor

StudentLicenta(string nume, string prenume, int varsta, unsigned int media, string specializare, int AnStudiu, string optional) :Student(nume, prenume, varsta, media, specializare)

{

this->AnStudiu = AnStudiu;

this->optional = optional;

}

//functia de afisare

void afisareStudL()

{

afisareStud();

cout << "Anul de studiu al studentului este:" << this->AnStudiu << endl;

cout << "Optionalul ales de student este:" << this->optional << endl;

}

};

class StudentMaster : public Student {

int an;

string domeniu;

public:

//constructor vid pentru declararea vectorului

StudentMaster() {};

//constructor pentru initializarea variabilelor

StudentMaster(string nume, string prenume, int varsta, unsigned int media, string specializarea, int an, string domeniu) :Student(nume, prenume, varsta, media, specializarea)

{

this->an = an;

this->domeniu = domeniu;

}

//functia de afisare

void afisareStudM()

{

afisareStud();

cout << "Anul de studiu este:" << this->an << endl;

cout << "Domeniul ales este:" << this->domeniu << endl;

}

};

class StudentDoctorand :public Student {

string profil;

public:

//constructor vid pentru declararea vectorului

StudentDoctorand() {};

//constructor pentru initializarea variabilelor

StudentDoctorand(string nume, string prenume, int varsta, unsigned int media, char \*specializarea, string profil) :Student(nume, prenume, varsta, media, specializarea)

{

this->profil = profil;

}

//functia de afisare

void afisareStudD()

{

afisareStud();

cout << "Profilul ales este:" << this->profil << endl;

}

};

int main()

{

int opt, nr1 = -1, nr2 = -1, nr3 = -1, nr4 = -1, nr5 = -1, varsta, AnStudiu, an,salar, media, grad\_didactic, vechime, i, i1, i2, i3, i4, i5, z1 = 0, z2 =- 0, z3 = 0, z4 = 0, z5 = 0;

Profesor prof[50];

Inginer ingr[50];

StudentLicenta sl[50];

StudentMaster sm[50];

StudentDoctorand sd[50];

string nume, prenume, departament, materie, specializare, profil, domeniu, functie, optional;

do {

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

cout << "1.Adaugare profesor" << endl;

cout << "2.Adaugare inginer" << endl;

cout << "3.Adaugare student licenta" << endl;

cout << "4.Adaugare student master" << endl;

cout << "5.Adaugare student doctorand" << endl;

cout << "6.Afisare" << endl;

cout << "7.Stergeti profesor" << endl;

cout << "8.Stergeti inginer" << endl;

cout << "9.Stergeti student licenta" << endl;

cout << "10.Stergeti student master" << endl;

cout << "11.Stergeti student doctorand" << endl;

cout << "12.Cautati profesor" << endl;

cout << "13.Cautati inginer" << endl;

cout << "14.Cautati student licenta" << endl;

cout << "15.Cautati student master" << endl;

cout << "16.Cautati student doctorand" << endl;

cin >> opt;

switch (opt)

{

case 0:exit(0);

break;

case 1:cout << "Introduceti numarul de profesori:" << endl;

cin >> nr1;

z1 = nr1;

do {

cout << "Nume profesor:" << endl;

cin >> nume;

cout << "Prenumele profesorului:" << endl;

cin >> prenume;

cout << "Varsta profesorului:" << endl;

cin >> varsta;

cout << "Gradul didactic al profesorului:" << endl;

cin >> grad\_didactic;

cout << "Materia:" << endl;

prof[nr1] = Profesor(nume, prenume, varsta, salar, departament, grad\_didactic, materie);

nr1--;

} while (nr1 != 0);

break;

case 2:

break; nr2++;

cout << "Dati numarul de ingineri:" << endl;;

cin >> nr2;

z2 = nr2;

do {

cout << "Nume inginer:" << endl;

cin >> nume;

cout << "Prenume inginer:" << endl;

cin >> prenume;

cout << "Varsta inginer:" << endl;

cin >> varsta;

cout << "Vechime:" << endl;

cin >> vechime;

cout << "Functie:" << endl;

cin >> functie;

ingr[nr2] = Inginer(nume, prenume, varsta, salar, departament, vechime, functie);

nr2--;

} while (nr2 != 0);

case 3:cout << "Dati numarul de studenti:" << endl;;

cin >> nr3;

z3 = nr3;

do

{

cout << "Nume student:" << endl;

cin >> nume;

cout << "Prenume student:" << endl;

cin >> prenume;

cout << "Varsta student:" << endl;

cin >> varsta;

cout << "Anul destudiu:" << endl;

cin >> AnStudiu;

cout << "Optional:" << endl;

cin >> optional;

sl[nr3] = StudentLicenta(nume,prenume, varsta, media, specializare, AnStudiu, optional);

nr3--;

} while (nr3 != 0);

break;

case 4:cout << "Dati numarul de studenti:" << endl;;

cin >> nr4;

z4 = nr4;

do {

cout << "Nume student:" << endl;

cin >> nume;

cout << "Prenume student:" << endl;

cin >> prenume;

cout << "Varsta student:" << endl;

cin >> varsta;

cout << "Domeniul studentului:" << endl;

cin >> domeniu;

cout << "Anul de studiu:" << endl;

cin >> an;

sm[nr4] = StudentMaster(nume,prenume, varsta, media, specializare, an, domeniu);

nr4--;

} while (nr4 != 0);

break;

case 5:cout << "Dati numarul de studenti:" << endl;;

cin >> nr5;

z5 = nr5;

do {

cout << "Nume student:" << endl;

cin >> nume;

cout << "Prenume student:" << endl;

cin >> prenume;

cout << "Varsta student:" << endl;

cin >> varsta;

cout << "Profilul studentului:" << endl;

cin >> profil;

sd[nr5] = StudentDoctorand(nume,prenume, varsta, media, specializare, profil);

nr5--;

} while (nr5!= 0);

break;

case 6:for (i = 1; i <= z1; i++)

{

prof[i].afisareProf();

cout << endl;

}

for (i2 = 1; i2 <= z2; i2++)

{

ingr[i2].afisareIng();

cout << endl;

}

for (i3 = 1; i3 <= z3; i3++)

{

sl[i3].afisareStudL();

cout << endl;

}

for (i4 = 1; i4 <= z4; i4++)

{

sm[i4].afisareStudM();

cout << endl;

}

for (i5 = 1; i5 <= z5; i5++)

{

sd[i5].afisareStudD();

cout << endl;

}

break;

case 7:cout << "Dati numele profesorului pe care doriti sa-l stergeti:";

cin >> nume;

for (i = 0; i <= z1; i++)

if (prof[i].nume1() == nume)

{

for (i1 = i; i1 <= z1; i1++)

prof[i1] = prof[i1 + 1];

z1--;

cout << "S-a sters." << endl;

}

for (i1 = 1; i1 <= z1; i1++)

{

prof[i1].afisareProf();

cout << endl;

}

break;

case 8:cout << "Dati numele inginerului pe care doriti sa-l stergeti:";

cin >> nume;

for (i = 0; i <= z2; i++)

if (ingr[i].nume1() == nume)

{

for (i1 = i; i1 <= z2; i1++)

ingr[i1] = ingr[i1 + 1];

z2--;

cout << "S-a sters." << endl;

}

for (i1 = 1; i1 <= z2; i1++)

{

ingr[i1].afisareIng();

cout << endl;

}

break;

case 9:cout << "Dati numele studentului pe care doriti sa-l stergeti:";

cin >> nume;

for (i = 0; i <= z3; i++)

if (sl[i].nume1() == nume)

{

for (i1 = i; i1 <= z3; i1++)

sl[i1] = sl[i1 + 1];

z3--;

cout << "S-a sters." << endl;

}

for (i1 = 1; i1 <= z4; i1++)

{

sl[i1].afisareStudL();

cout << endl;

}

break;

case 10:cout << "Dati numele studentului pe care doriti sa-l stergeti:";

cin >> nume;

for (i = 0; i <= z4; i++)

if (sl[i].nume1() == nume)

{

for (i1 = i; i1 <= z4; i1++)

sm[i1] = sm[i1 + 1];

z4--;

cout << "S-a sters." << endl;

}

for (i1 = 1; i1 <= z4; i1++)

{

sm[i1].afisareStudM();

cout << endl;

}

break;

case 11:cout << "Dati numele studentului pe care doriti sa-l stergeti:";

cin >> nume;

for (i = 0; i <= z5; i++)

if (sl[i].nume1() == nume)

{

for (i1 = i; i1 <= z5; i1++)

sd[i1] = sd[i1 + 1];

z5--;

cout << "S-a sters." << endl;

}

for (i1 = 1; i1 <= z5; i1++)

{

sd[i1].afisareStudD();

cout << endl;

}

break;

case 12:cout << "Dati numele profesorului pe care doriti sa-l cautati:";

cin >> nume;

for (i = 0; i <= z1; i++)

if (nume == prof[i].nume1())

{

prof[i].afisareProf();

cout << "S-a gasit." << endl;

}

break;

case 13: cout << "Dati numele inginerului pe care doriti sa-l cautati:";

cin >> nume;

for (i = 0; i <= z2; i++)

if (nume == ingr[i].nume1())

{

ingr[i].afisareIng();

cout << "S-a gasit." << endl;

}

break;

case 14:cout << "Dati numele studentului pe care doriti sa-l cautati:";

cin >> nume;

for (i = 0; i <= z3; i++)

if (nume == sl[i].nume1())

{

sl[i].afisareStudL();

cout << "S-a gasit." << endl;

}

break;

case 15:cout << "Dati numele studentului pe care doriti sa-l cautati:";

cin >> nume;

for (i = 0; i <= z4; i++)

if (nume == sm[i].nume1())

{

sm[i].afisareStudM();

cout << "S-a gasit." << endl;

}

break;

case 16:cout << "Dati numele studentului pe care doriti sa-l cautati:";

cin >> nume;

for (i = 0; i <= z5; i++)

if (nume == sd[i].nume1())

{

sd[i].afisareStudD();

cout << "S-a gasit." << endl;

}

break;

default:cout << "Optiune gresita!";

break;

}

} while (1);

return 0;

}

Problema 5

#include <iostream>

#include <string>

using namespace std;

class Carte

{

string titlu;

public:

Carte(){};

Carte(string titlu)

{

this->titlu = titlu;

}

string nume1()

{

return this->titlu;

}

void afisare\_carte()

{

cout << "Numele cartii este:" << this->titlu << endl;

}

};

class FisaBiblioteca : public Carte

{

string nume;

public:

FisaBiblioteca()

{ }

FisaBiblioteca(string titlu, string nume) : Carte(titlu)

{

this->nume = nume;

}

void afisare\_fisa()

{

afisare\_carte();

cout << "Persoana care a imprumutat cartea este:" << this->nume << endl;

}

};

int main()

{

int opt, nr1 = -1, nr2 = -1,i, j, x = 0, y = 0;

string titlu, nume;

Carte carte[50];

FisaBiblioteca fisa[50];

do {

cout << "\n0.Iesire";

cout << "\n1.Adaugati o carte:";

cout << "\n2.Adaugati o fisa:";

cout << "\n3.Afisati cartile:";

cout << "\n4.Stergeti cartea:";

cout << "\n5.Cautati cartea:";

cout << "\nOptiunea dvs. este:"; cin >> opt;

switch (opt)

{

case 0: cout << "La revedere!"; exit(0);

break;

case 1:

cout << "Dati numarul de carti:" << endl; cin >> nr1;

x = nr1;

do

{

cout << "Nume carte:" << endl; cin >> titlu;

carte[nr1] = Carte(titlu);

nr1--;

} while (nr1 != 0);

break;

case 2:

cout << "Dati numarul de fise:" << endl; cin >> nr2;

y = nr2;

do

{

cout << "Numele persoanei din fisa:"; cin >> nume;

fisa[nr2] = FisaBiblioteca(titlu, nume);

nr2--;

} while (nr2 != 0);

break;

case 3:

for (i = 1; i <= x; i++)

{

carte[i].afisare\_carte();

cout << "\n";

}

for (i = 1; i <= y; i++)

{

fisa[i].afisare\_fisa();

cout << "\n";

}

break;

case 4:

cout << "Dati numele cartii pe care doriti sa o stergeti:"; cin >> titlu;

for (i = 0; i <= x; i++)

if (carte[i].nume1() == titlu)

{

for (j = i; j <= x; j++)

carte[j] = carte[j + 1];

x--;

cout << "S-a sters." << endl;

}

for (j = 1; j <= x; j++)

{

carte[j].afisare\_carte();

cout << "\n";

}

break;

case 5:

cout << "Dati numele cartii pe care doriti sa o cautati:"; cin >> titlu;

for (i = 0; i <= x; i++)

if (carte[i].nume1() == titlu)

{

carte[i].afisare\_carte();

cout << "S-a gasit." << endl;

}

break;

}

} while (1);

return 0;

}