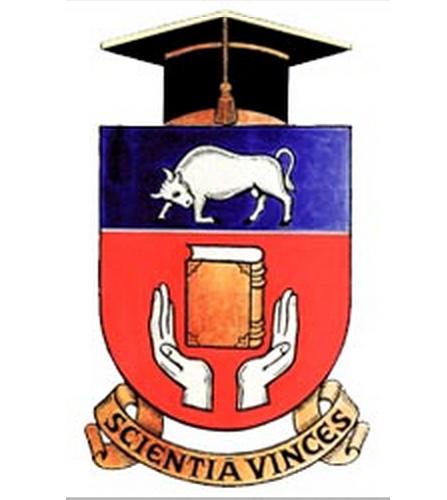
Universitatea de Stat din Tiraspol

Facultatea de Fizica Matematica si Informatica



Laboratorul #4

**Elaborat de**: Dordea Pavel, student gr.2i

**Verificat**: dr.conf.univ. Ala Gasnaș

**Chișinău, 2021**

**3. Moştenire simplă**

*Problemă exemplu* În calitate de clasă de *bază* se consideră clasa *binar*. În calitate de date vor fi două valori reale. Metodele vor fi:

1. constructorul cu şi fără parametri;
2. metodele de citire şi afisare.

Creaţi clasa *dreptunghi*, derivata clasei *binar*. Pentru clasa dreptunghi vor fi implementate metodele de determinare a suprafeţei şi a perimetrului.

De la tastatură se citesc datele despre *n* dreptunghiuri. Elaboraţi un program  prin intermediul căruia la ecran se vor afişa datele despre toate dreptunghiurile, dreptunghiul cu suprafaţa maximă şi dreptunghiul cu perimetrul maxim.

1. În calitate de clasă de *bază* se va crea clasa *triad*. date: trei numere reale;

metode: *constructorul* cu/fără parametri, *citire* şi *afisare*.

1. Creaţi clasa *triunghi*, derivata clasei *triad*. Pentru clasa triunghi vor fi implementate metode pentru determinare a suprafeţei, a perimetrului şi o metodă prin intermediul căreia se va verifica dacă datele introduse pot fi lungimile laturilor unui triunghi.

De la tastatură se citesc datele despre *n* triunghiuri. Elaboraţi un program prin intermediul căruia, la ecran se vor afişa datele despre toate triunghiurile, triunghiul cu suprafaţă maximă şi triunghiul cu perimetru maxim.

**Codul C++ :**

#include <iostream>

#include <math.h>

using namespace std;

class triad {

protected:

double lat\_1, lat\_2, lat\_3;

public:

triad(){

lat\_1 = 0;

lat\_2 = 0;

lat\_3 = 0;

}

triad(double a1, double a2, double a3){

lat\_1 = a1;

lat\_2 = a2;

lat\_3 = a3;

}

void citire();

void afisare();

};

class triunghi:public triad {

public:

int verificare();

double area();

double perimetru();

triunghi():triad(){};

triunghi(double a1, double a2, double a3):triad(a1,a2,a3){

lat\_1 = a1;

lat\_2 = a2;

lat\_3 = a3;

}

};

void triad:: citire(){

cout << "Introduceti prima latura: ";

cin >> lat\_1;

cout << "Introduceti a doua latura: ";

cin >> lat\_2;

cout << "Introduceti a treia latura: ";

cin >> lat\_3;

}

void triad:: afisare(){

cout << "Prima latura: " << lat\_1 << " cm\n" << endl;

cout << "A doua latura: " << lat\_2 << " cm\n" << endl;

cout << "A treia latura: " << lat\_3 << " cm\n" << endl;

}

int triunghi:: verificare(){

if (lat\_1 + lat\_2 > lat\_3 && lat\_1 + lat\_3 > lat\_2 && lat\_3 + lat\_2 > lat\_1)

return 1;

else

return 0;

}

double triunghi::perimetru(){

return lat\_1 + lat\_2 + lat\_3;

}

double triunghi::area(){

double p = (lat\_1 + lat\_2 + lat\_3) / 2.0;

double s;

s = p \* (p - lat\_1) \* (p - lat\_2) \* (p - lat\_3);

s = pow(s,0.5);

return s;

}

main(){

triunghi tr[300];

int i,n;

cout <<"Introduceti n:";

cin >> n;

for (i=0; i<n; i++){

tr[i].citire();

cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

tr[i].afisare();

cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

}

double sMax = tr[0].area();

int sMax\_index = 0;

for (i=1; i<n; i++){

if (tr[i].area()>sMax){

sMax\_index = i;

sMax = tr[i].area();

}

}

double pMax = tr[0].perimetru();

int pMax\_index = 0;

for (i=1; i<n; i++){

if (tr[i].perimetru()>pMax){

pMax\_index = i;

pMax = tr[i].perimetru();

}

}

cout << "Perimetrul maximal:" << endl;

cout << pMax << endl;

cout << "Triunghiu cu perimterul maxim:" << endl;

tr[pMax\_index].afisare();

cout << "Aria maxima:" << endl;

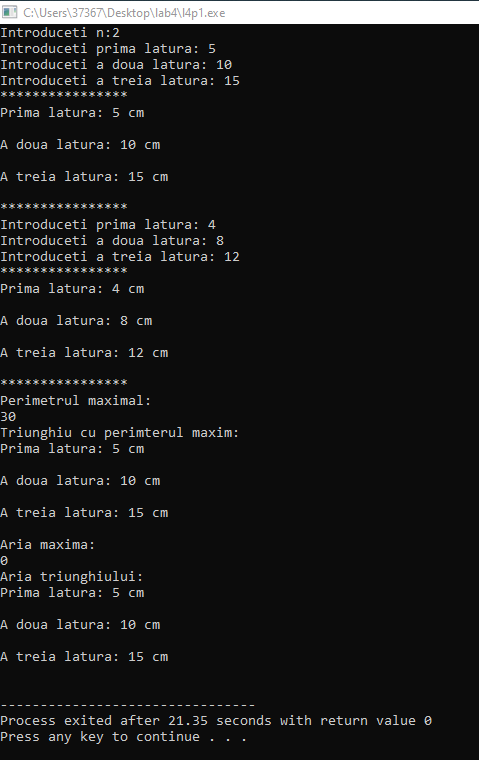
cout << sMax << endl;

cout << "Aria triunghiului:" << endl;

tr[sMax\_index].afisare();

}

**Screenshot-uri:**



1. Creaţi clasa *paralelogram*, derivata clasei *triad*. Pentru clasa paralelogram vor fi implementate metodele de determinare a suprafeţei şi a perimetrului.

De la tastatură se citesc datele despre *n* paralelograme. Elaboraţi un program prin intermediul căruia la ecran se vor afişa datele despre toate figurile,  paralelogramul cu suprafaţă maximă, paralelogramul cu perimetru maxim.

**Codul C++ :**

#include <iostream>

#include <math.h>

using namespace std;

class triad {

protected:

double lat\_1, lat\_2,inalt;

public:

triad(){

lat\_1 = 0;

lat\_2 = 0;

inalt = 0;

}

triad(double a1, double a2,double a3){

lat\_1 = a1;

lat\_2 = a2;

inalt = a3;

}

void citire();

void afisare();

};

class paralelogram:public triad {

public:

double area();

double perimetru();

paralelogram():triad(){};

paralelogram(double a1, double a2,double a3):triad(a1,a2,inalt){

lat\_1 = a1;

lat\_2 = a2;

inalt = a3;

}

};

void triad::citire(){

cout << "Introduceti baza paralelogramului: " <<endl;

cin >> lat\_1;

cout << "Introduceti laturile laterale: " <<endl;

cin>>lat\_2;

cout << "Introduceti inaltimea: " <<endl;

cin >> inalt;

}

void triad::afisare(){

cout << "Baza paralelogramului: " << lat\_1 <<endl;

cout << "Laturile laterale: " <<lat\_2 <<endl;

cout << " Inaltimea introdusa: " <<inalt <<endl;

}

double paralelogram::area(){

return lat\_1 \* inalt;

}

double paralelogram::perimetru(){

return 2 \* (lat\_1 + lat\_2);

}

int main(){

paralelogram paral[100];

int i,n;

cout << "Dati N: " <<endl;

cin >> n;

for (i=0 ; i<n;i++){

paral[i].citire();

cout << "/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/" <<endl;

}

for (i=0 ; i <n ;i++){

paral[i].afisare();

cout<< "Datele despre Paralelogramele introduse:" <<endl;

cout<< "/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/" <<endl;

}

double sMax = paral[0].area();

int sMax\_index = 0;

for (i=1; i<n; i++){

if (paral[i].area()>sMax){

sMax\_index = i;

sMax = paral[i].area();

}

}

double pMax = paral[0].perimetru();

int pMax\_index = 0;

for (i=1; i<n; i++){

if (paral[i].perimetru()>pMax){

pMax\_index = i;

pMax =paral[i].perimetru();

}

}

cout << "Perimetrul maximal:" << endl;

cout << pMax << endl;

cout << "Paralelogram cu perimterul maxim:" << endl;

paral[pMax\_index].afisare();

cout << "Aria maxima:" << endl;

cout << sMax << endl;

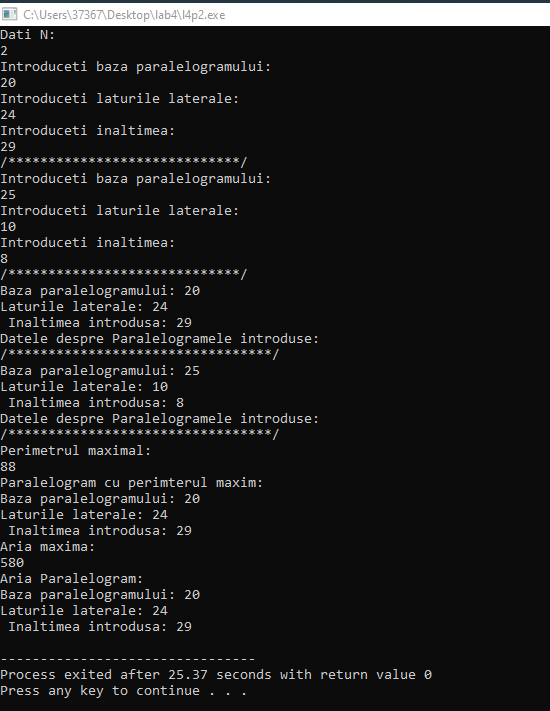
cout << "Aria Paralelogram:" << endl;

paral[sMax\_index].afisare();

return 0;

}

**Screenshot-uri:**



1. Creaţi clasa *trapez\_isoscel*, derivata clasei *triad*. Pentru clasa trapez isoscel vor fi implementate metodele de determinare a suprafeţei şi a perimetrului.

De la tastatură se citesc datele despre *n* trapeze isoscele. Elaboraţi un program prin intermediul căruia la ecran se vor afişa datele despre toate figurile, trapezul cu suprafaţă maximă, trapezul cu perimetru maxim.  
**Codul C++ :**

#include <iostream>

#include <math.h>

using namespace std;

class triad {

protected:

double lat\_1, lat\_2,b\_mare,b\_mic;

public:

triad(){

lat\_1 = 0;

lat\_2 = 0;

b\_mare = 0;

b\_mic = 0;

}

triad(double a1, double a2, double a3, double a4){

lat\_1 = a1;

lat\_2 = a2;

b\_mare = a3;

b\_mic = a4;

}

void citire();

void afisare();

};

class trapez\_isoscel:public triad {

public:

double inalt();

double aria();

double perimetru();

trapez\_isoscel():triad(){};

trapez\_isoscel(double a1, double a2,double a3,double a4,double a5):triad(a1,a2,a3,a4){

lat\_1 = a1;

lat\_2 = a2;

b\_mare = a3;

b\_mic = a4;

}

};

void triad::citire(){

cout << "Introduceti baza mica a trapezulu isoscel: " <<endl;

cin >> b\_mic;

cout << "Introduceti baza mare a trapezulu isoscel: " <<endl;

cin>>b\_mare;

cout << "Introduceti laturile trapezulu isoscel:" <<endl;

cin>>lat\_1;

cout << "Introduceti laturile trapezului isoscel: "<<endl;

cin >> lat\_2;

}

void triad::afisare(){

cout << "Baza mica a trapezului isoscel: " << b\_mic <<endl;

cout<<"Baza mare a trapezului isoscel: " <<b\_mare <<endl;

cout << "Laturile trapezului isoscel: " <<lat\_1 <<endl;

cout << "Laturile trapezului isoscel: " <<lat\_2 <<endl;

}

double trapez\_isoscel::inalt(){

double diff = (b\_mare - b\_mic) / 2.0;

double h = lat\_1 \* lat\_2 - diff \* diff;

h = pow(h,0.5);

return h;

}

double trapez\_isoscel::aria(){

return (b\_mare + b\_mic) \* inalt() / 2;

}

double trapez\_isoscel::perimetru(){

return b\_mare + b\_mic + lat\_1 + lat\_2;

}

int main(){

trapez\_isoscel trap[100];

int i,n;

cout << "Dati N: " <<endl;

cin >> n;

for (i=0 ; i<n;i++){

trap[i].citire();

cout << "/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/" <<endl;

}

for (i=0 ; i <n ;i++){

trap[i].afisare();

cout<< "/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/" <<endl;

cout<< "Datele despre Trapezul Isoscel introduse:" <<endl;

cout<< "/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/" <<endl;

}

double sMax = trap[0].aria();

int sMax\_index = 0;

for (i=1; i<n; i++){

if (trap[i].aria()>sMax){

sMax\_index = i;

sMax = trap[i].aria();

}

}

double pMax = trap[0].perimetru();

int pMax\_index = 0;

for (i=1; i<n; i++){

if (trap[i].perimetru()>pMax){

pMax\_index = i;

pMax =trap[i].perimetru();

}

}

cout << "Perimetrul maximal:" << endl;

cout << pMax << endl;

cout << "Trapezul isoscel cu perimterul maxim:" << endl;

trap[pMax\_index].afisare();

cout << "Aria maxima:" << endl;

cout << sMax << endl;

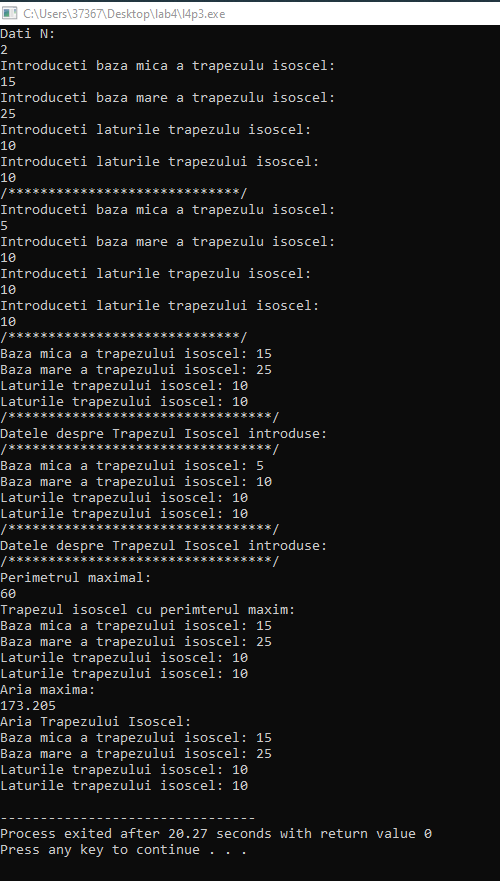
cout << "Aria Trapezului Isoscel:" << endl;

trap[sMax\_index].afisare();

return 0;

}

**Screenshot-uri:**



1. Creaţi clasa *prismă* (baza este dreptunghi), derivata clasei *triad*. Pentru clasa prismă vor fi implementate metodele de determinare a suprafeţei totale, laterale şi de determinare a volumului.

De la tastatură se citesc datele despre *n* prisme. Elaboraţi un program prin intermediul căruia, la ecran se vor afişa datele despre toate prismele, prisma cu suprafaţă maximă, prisma cu volumul maxim.

**Codul C++ :**

#include <iostream>

#include <math.h>

using namespace std;

class triad {

protected:

double lung,lat,inalt;

public:

triad(){

lung = 0;

lat = 0;

inalt = 0;

}

triad(double a1, double a2, double a3){

lung = a1;

lat = a2;

inalt = a3;

}

void citire();

void afisare();

};

class prisma:public triad {

public:

double a\_lat();

double a\_baz();

double volum();

double suprafata();

prisma():triad(){};

prisma(double a1, double a2,double a3):triad(a1,a2,a3){

lung = a1;

lat = a2;

inalt = a3;

}

};

void triad::citire(){

cout<<"Introduceti lungimea Prismei: " <<endl;

cin >> lung;

cout<<"Introduceti latimea Prismei:" <<endl;

cin >>lat;

cout << "Introduceti inaltimea Prismei: " <<endl;

cin >> inalt;

}

void triad::afisare(){

cout << "Lungimea Prismei este: " << lung <<endl;

cout << "Latimea Prismei este: " << lat <<endl;

cout << "Inaltimea Prismei este: " << inalt <<endl;

}

double prisma::a\_lat(){

return 2 \* lat \* inalt + 2 \* lung \* inalt;

}

double prisma::a\_baz(){

return lung \* lat;

}

double prisma::volum(){

return lung \* lat \* inalt;

}

double prisma::suprafata(){

return a\_lat() + a\_baz() \* 2;

}

int main(){

prisma pr[100];

int i,n;

cout << "Dati N: " <<endl;

cin >> n;

for (i=0 ; i<n;i++){

pr[i].citire();

cout << "/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/" <<endl;

}

for (i=0 ; i <n ;i++){

pr[i].afisare();

cout<< "/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/" <<endl;

cout<< "Datele despre Prismele introduse:" <<endl;

cout<< "/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/" <<endl;

}

double sMax = pr[0].volum();

int sMax\_index = 0;

for (i=1; i<n; i++){

if (pr[i].volum()>sMax){

sMax\_index = i;

sMax = pr[i].volum();

}

}

double pMax = pr[0].suprafata();

int pMax\_index = 0;

for (i=1; i<n; i++){

if (pr[i].suprafata()>pMax){

pMax\_index = i;

pMax =pr[i].suprafata();

}

}

cout << "Prisma cu volumul maxim:" << endl;

pr[sMax\_index].afisare();

cout<< "/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/" <<endl;

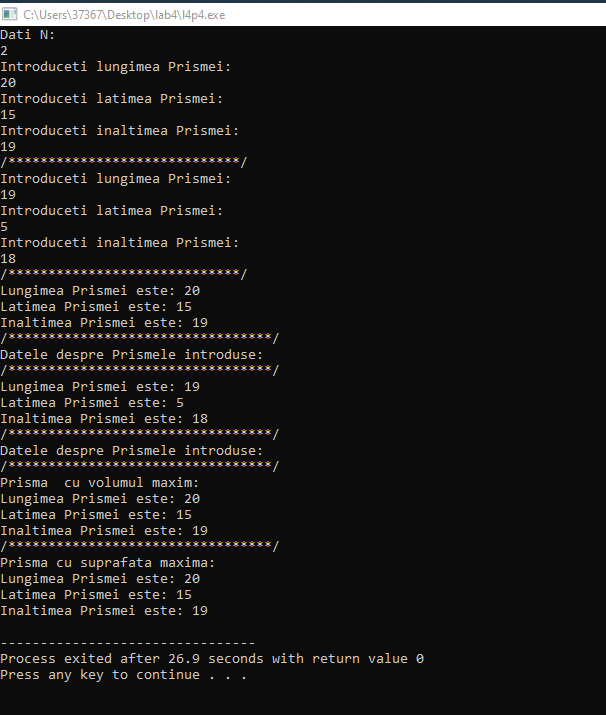
cout << "Prisma cu suprafata maxima:" << endl;

pr[sMax\_index].afisare();

return 0;

}

**Screenshot-uri:**



1. Creaţi clasa *piramidă* (baza este dreptunghi), derivata clasei *triad*. Pentru clasa piramidă vor fi implementate metodele de determinare a suprafeţei totale,  laterale şi de determinare a volumului.

De la tastatură se citesc datele despre *n* piramide. Elaboraţi un program prin intermediul căruia, la ecran se vor afişa datele despre toate piramidele, piramida cu suprafaţă maximă, piramida cu volumul maxim.

1. În calitate de clasă de *bază* se va crea clasa *persoana*. date: *nume*, *prenume*, *anul*, *cnp* (codul personal de 13 cifre); metode : *citire*, *afişare, virsta*.

**Codul C++ :**

#include <iostream>

#include <math.h>

using namespace std;

class triad {

protected:

double lung,lat,inalt;

public:

triad(){

lung = 0;

lat = 0;

inalt = 0;

}

triad(double a1, double a2, double a3){

lung = a1;

lat = a2;

inalt = a3;

}

void citire();

void afisare();

};

class piramida:public triad {

public:

double a\_lat();

double a\_tot();

double volum();

double apotema();

piramida():triad(){};

piramida(double a1, double a2,double a3):triad(a1,a2,a3){

lung = a1;

lat = a2;

inalt = a3;

}

};

void triad::citire(){

cout<<"Introduceti lungimea Piramidei: " <<endl;

cin >> lung;

cout<<"Introduceti latimea Piramidei:" <<endl;

cin >>lat;

cout << "Introduceti inaltimea Piramidei: " <<endl;

cin >> inalt;

}

void triad::afisare(){

cout << "Lungimea Piramidei este: " << lung <<endl;

cout << "Latimea Piramidei este: " << lat <<endl;

cout << "Inaltimea Piramidei este: " << inalt <<endl;

}

double piramida::apotema(){

float diagonala = lung \* lung + lat \* lat;

diagonala = pow(diagonala,0.5);

float semiDiag = diagonala / 2.0;

float apot = inalt \* inalt + semiDiag \* semiDiag;

apot = pow(apot,0.5);

return apot;

}

double piramida::a\_lat(){

return 2 \* (lung \* lat) \* apotema() / 2.0;

}

double piramida::a\_tot(){

return a\_lat() + lung + lat;

}

double piramida::volum(){

return lung \* lat \* inalt / 3.0;

}

int main(){

piramida pr[100];

int i,n;

cout << "Dati N: " <<endl;

cin >> n;

for (i=0 ; i<n;i++){

pr[i].citire();

cout << "/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/" <<endl;

}

for (i=0 ; i <n ;i++){

cout<< "/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/" <<endl;

cout<< "Datele despre Piramidele introduse:" <<endl;

cout<< "/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/" <<endl;

pr[i].afisare();

}

double sMax = pr[0].volum();

int sMax\_index = 0;

for (i=1; i<n; i++){

if (pr[i].volum()>sMax){

sMax\_index = i;

sMax = pr[i].volum();

}

}

double pMax = pr[0].a\_tot();

int pMax\_index = 0;

for (i=1; i<n; i++){

if (pr[i].a\_tot()>pMax){

pMax\_index = i;

pMax =pr[i].a\_tot();

}

}

cout << "Piramida cu volumul maxim:" << endl;

cout<< "/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/" <<endl;

pr[sMax\_index].afisare();

cout << "Piramida cu suprafata maxima:" << endl;

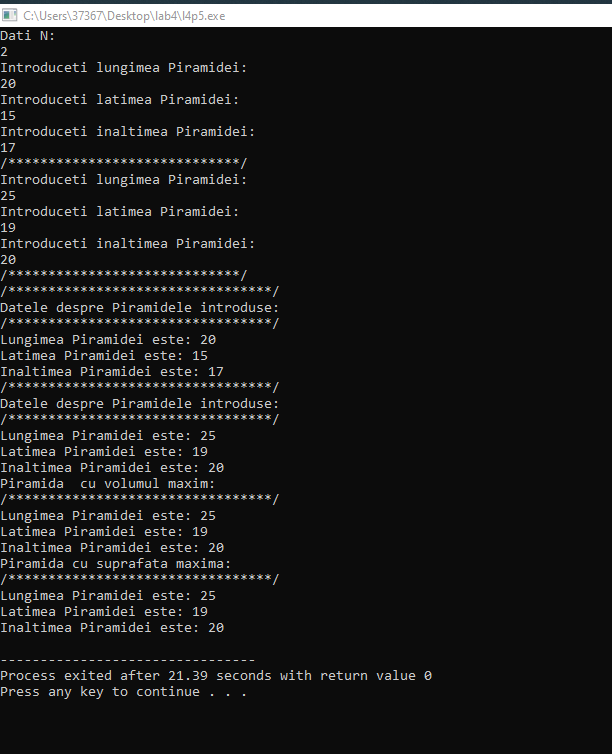
cout<< "/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/" <<endl;

pr[sMax\_index].afisare();

return 0;

}

**Screenshot-uri:**



1. Creaţi clasa *Salariat*, derivata clasei *persoana*. Pentru clasa salariat vor fi adăugate:

date: numărul de ore lucrate, plata pentru o oră, anul angajării;

metode: vor fi redefinite metodele de *citire* şi *afişare*, *salariu*. Salariul va fi determinat conform formulei : *nr\_ore\*plata*.

De la tastatură se citesc datele despre *n* salariaţi. Elaboraţi un program prin intermediul căruia la ecran se vor afişa datele despre toţi salariaţii, salariatul cel mai tînăr şi salariatul cu salariu maxim.

**Codul C++ :**

#include <iostream>

#include <iterator>

#include <iomanip>

#include <vector>

#include <cmath>

#include <ctime>

#include <string>

using namespace std;

class Persoana{

protected:

string nume = "Pavel";

string prenume = "Dordea";

string cnp = "7777777777777";

int an\_nas = 2000;

public:

Persoana(){};

Persoana(string str1, string str2, string str3, int an){

nume = str1;

prenume = str2;

cnp = str3;

an\_nas = an;

}

void citire();

void afisare();

int varsta();

};

class Salariat:public Persoana{

private:

int nr\_ore;

float plata;

int an\_ang;

public:

Salariat(int nr,float plata2,int an2):Persoana(){

nr\_ore = nr;

plata = plata2;

an\_ang = an2;

};

void citire1();

void afisare1();

float salariu();

};

void Persoana::citire(){

cout << "\nIntroduceti numele angajatului: ";

cin >> nume;

cout << "Introduceti prenumele angajatului: ";

cin >> prenume;

cout << "Introduceti codul personal al angajatului: ";

cin >> cnp;

cout << "Introduceti anul nasterii: ";

cin >> an\_nas;

}

void Persoana::afisare(){

cout << "\nNumele angajatului: " << nume << endl;

cout << "Prenumele angajatului: " << prenume << endl;

cout << "Codul personal al angajatului: " << cnp << endl;

cout << "Anul nasterii: " << an\_nas << endl;

cout << "Varsta angajatului: " << varsta() << endl;

}

int Persoana::varsta(){

return 2021 - an\_nas;

}

float Salariat::salariu(){

return nr\_ore \* plata;

}

void Salariat::citire1(){

cout << "Introduceti numarul de ore lucrate de catre salariat: ";

cin >> nr\_ore;

cout << "Introduceti plata pentru o ora: ";

cin >> plata;

cout << "Introduceti anul angajarii: ";

cin >> an\_ang;

}

void Salariat::afisare1(){

cout << "\nNumarul de ore lucrate: " << nr\_ore << "\n";

cout << "Plata pentru o ora: " << plata << " lei\n";

cout << "Anul angajarii: " << an\_ang << "\n";

cout << "Salariul: " << salariu() << " lei\n";

}

int main() {

/\* int n;

cout << "n=";

cin >> n;

vector <Salariat> lista;

vector <Salariat>::iterator it;

vector <Salariat>::iterator it2;

vector <Salariat>::iterator it3;

for (int i = 0;i < n; i++) {

Salariat ob;

ob.citire();

ob.citire1();

lista.push\_back(ob);

}

int index = 1;

for (it = lista.begin();it != lista.end(); it++) {

cout << "\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n";

cout << "Datele salariatului numarul " << index << " :\n";

it -> afisare();

it -> afisare1();

index++;

}

it = lista.begin();

float salMax = it->salariu();

it++;

for (it; it != lista.end(); it++) {

if (it -> salariu() > salMax) {

salMax = it -> salariu();

it2 = it;

}

}

cout << "\n\nSalariul maxim este de " << salMax << " lei\n";

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n";

cout << "Datele salariatului cu salariu maxim: \n";

it2 -> afisare();

it2 -> afisare1();

it = lista.begin();

int anMin = it->varsta();

it++;

for (it; it != lista.end(); it++) {

if (it -> varsta() < anMin) {

anMin = it -> varsta();

it3 = it;

}

}

cout << "\n\nVarsta celui mai tanar angajat este: " << anMin << " \n";

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n";

cout << "Datele celui mai tanar salaria: \n";

it3 -> afisare();

it3 -> afisare1();\*/

Salariat ob2(100,100,2015);

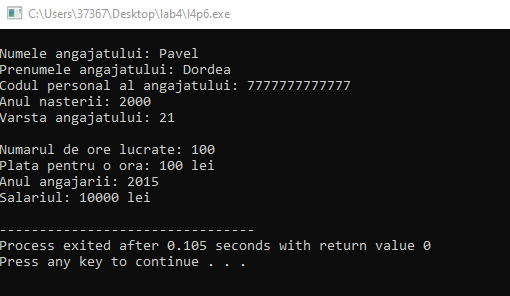
ob2.afisare();

ob2.afisare1();

return 0;

}

**Screenshot-uri:**



1. Creaţi clasa *Student*, derivata clasei *persoana*. Pentru clasa student vor fi adăugate:

date: media notelor curente, grupa;

metode: vor fi redefinite metodele de *citire* şi *afişare*, *bursa*. Bursa va fi determinată conform fomulei: *media\*75 lei,* dacă media este mai mare decît 7.5, în  caz contrar va fi 0.

De la tastatură se citesc datele despre *n* studenţi. Elaboraţi un program prin intermediul căruia la ecran se vor afişa datele despre toţi studenţii, studentul cel mai tînăr, studentul cu bursa maximă.

**Codul C++ :**

#include <iostream>

#include <iterator>

#include <iomanip>

#include <vector>

#include <cmath>

#include <ctime>

#include <string>

using namespace std;

class Persoana{

protected:

string nume = "Dumitru";

string prenume = "Munteanu";

string cnp = "7777777777777";

int an\_nas = 2001;

public:

Persoana(){};

Persoana(string str1, string str2, string str3, int an){

nume = str1;

prenume = str2;

cnp = str3;

an\_nas = an;

}

void citire();

void afisare();

int varsta();

};

class Student:public Persoana{

private:

float media;

string grupa;

public:

Student():Persoana(){};

Student(float med, string group):Persoana(){

media = med;

grupa = group;

};

void citire1();

void afisare1();

float bursa();

};

void Persoana::citire(){

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

cin >> nume;

cout << "Introduceti prenumele studentului: ";

cin >> prenume;

cout << "Introduceti codul personal al studentului: ";

cin >> cnp;

cout << "Introduceti anul nasterii: ";

cin >> an\_nas;

}

void Persoana::afisare(){

cout << "\nNumele studentului: " << nume << endl;

cout << "Prenumele studentului: " << prenume << endl;

cout << "Codul personal al studentului: " << cnp << endl;

cout << "Anul nasterii: " << an\_nas << endl;

cout << "Varsta studentului: " << varsta() << endl;

}

int Persoana::varsta(){

return 2021 - an\_nas;

}

float Student::bursa(){

if (media > 7.5 && media <= 10) {

return media \* 75;

}

return 0;

}

void Student::citire1(){

cout << "Introduceti media studentului: ";

cin >> media;

cout << "Introduceti grupa studentului: ";

cin >> grupa;

}

void Student::afisare1(){

cout << "\nMedia studentului: " << media << "\n";

cout << "Grupa studentului: " << grupa << "\n";

cout << "Bursa studentlui: " << bursa() << " lei\n";

}

int main() {

int n;

cout << "n=";

cin >> n;

vector <Student> lista;

vector <Student>::iterator it;

vector <Student>::iterator it2;

vector <Student>::iterator it3;

for (int i = 0;i < n; i++) {

Student ob;

ob.citire();

ob.citire1();

lista.push\_back(ob);

}

int index = 1;

for (it = lista.begin();it != lista.end(); it++) {

cout << "\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n";

cout << "Datele studentului numarul " << index << " :\n";

it -> afisare();

it -> afisare1();

index++;

}

it = lista.begin();

it2 = lista.begin();

float bursaMax = it->bursa();

it++;

for (it; it != lista.end(); it++) {

if (it -> bursa() > bursaMax) {

bursaMax = it -> bursa();

it2 = it;

}

}

cout << "\n\nBursa maxima este de " << bursaMax << " lei\n";

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n";

cout << "Datele studentului cu bursa maxima: \n";

it2 -> afisare();

it2 -> afisare1();

it = lista.begin();

it3 = lista.begin();

int anMin = it->varsta();

it++;

for (it; it != lista.end(); it++) {

if (it -> varsta() < anMin) {

anMin = it -> varsta();

it3 = it;

}

}

cout << "\n\nVarsta celui mai tanar student este de: " << anMin << " ani\n";

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n";

cout << "Datele celui mai tanar student: \n";

it3 -> afisare();

it3 -> afisare1();

Student ob2(7.52, "Grupa 2i");

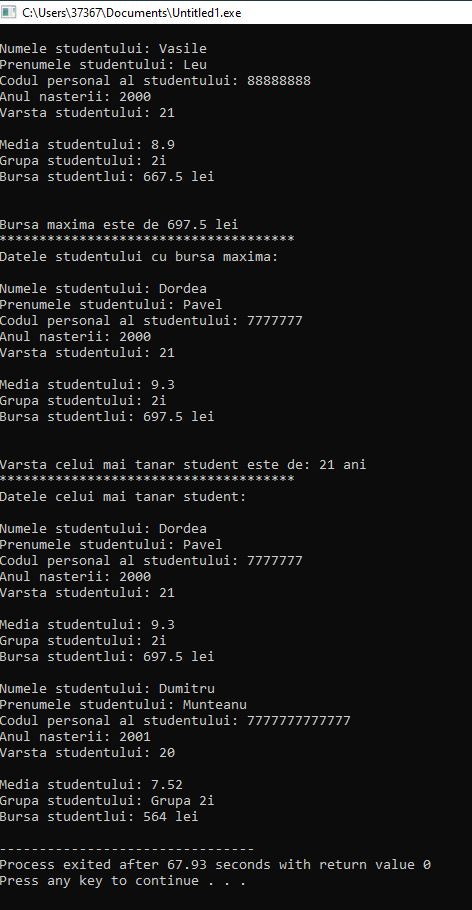
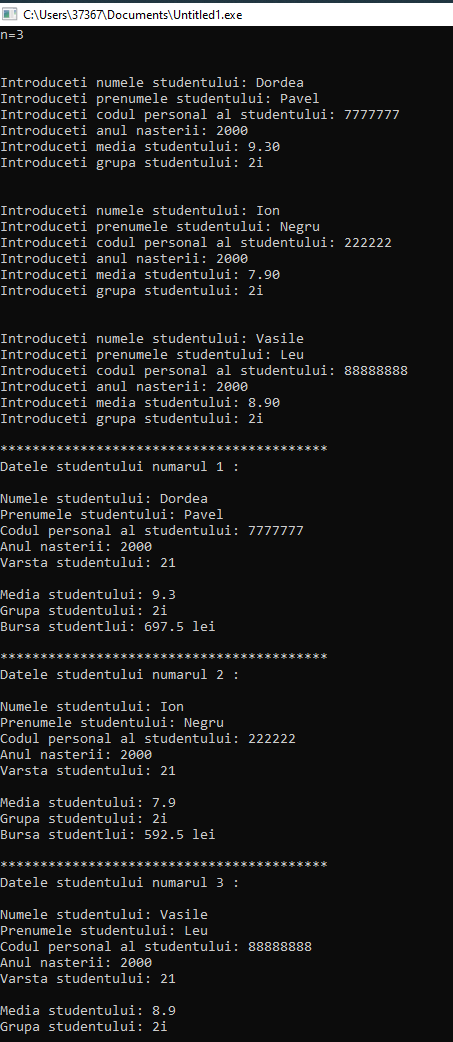
ob2.afisare();

ob2.afisare1();

return 0;

}

**Screenshot-uri:**



1. Creaţi clasa *Sofer*, derivata clasei *persoana*. Pentru clasa şofer vor fi adăugate:

date: distanţa parcursă (în km), plata pentru 1 km;

metode: vor fi redefinite metodele de *citire* şi *afişare*, *plata*. Prin intermediul metodei *Plata*, se va determina costul unei curse efectuate de către şofer după  formula: *distanta\*plata pentru 1 km*.

De la tastatură se citesc datele despre *n* şoferi. Elaboraţi un program prin intermediul căruia la ecran se vor afişa datele despre toţi şoferii, şoferul cel mai tînăr, şoferul cu plata maximă.

**Codul C++ :**

#include <iostream>

#include <iterator>

#include <iomanip>

#include <vector>

#include <cmath>

#include <ctime>

#include <string>

using namespace std;

class Persoana{

protected:

string nume = "Pavel";

string prenume = "Dordea";

string cnp = "7777777777777";

int an\_nas = 2000;

public:

Persoana(){};

Persoana(string str1, string str2, string str3, int an){

nume = str1;

prenume = str2;

cnp = str3;

an\_nas = an;

}

void citire();

void afisare();

int varsta();

};

class Sofer:public Persoana{

private:

int nr\_km;

float plata;

public:

Sofer():Persoana(){};

Sofer(int nr,float plata2):Persoana(){

nr\_km = nr;

plata = plata2;

};

void citire1();

void afisare1();

float salariu();

};

void Persoana::citire(){

cout << "\nIntroduceti numele soferului: ";

cin >> nume;

cout << "Introduceti prenumele soferului: ";

cin >> prenume;

cout << "Introduceti codul personal al soferului: ";

cin >> cnp;

cout << "Introduceti anul nasterii: ";

cin >> an\_nas;

}

void Persoana::afisare(){

cout << "\nNumele soferului: " << nume << endl;

cout << "Prenumele soferului: " << prenume << endl;

cout << "Codul personal al soferului: " << cnp << endl;

cout << "Anul nasterii: " << an\_nas << endl;

cout << "Varsta soferului: " << varsta() << endl;

}

int Persoana::varsta(){

return 2021 - an\_nas;

}

float Sofer::salariu(){

return nr\_km \* plata;

}

void Sofer::citire1(){

cout << "Introduceti numarul de km parcursi: ";

cin >> nr\_km;

cout << "Introduceti plata pentru un km parcurs: ";

cin >> plata;

}

void Sofer::afisare1(){

cout << "\nNumarul de ore lucrate: " << nr\_km << "\n";

cout << "Plata pentru o ora: " << plata << " lei\n";

cout << "Salariul: " << salariu() << " lei\n";

}

int main() {

int n;

cout << "n=";

cin >> n;

vector <Sofer> lista;

vector <Sofer>::iterator it;

vector <Sofer>::iterator it2;

vector <Sofer>::iterator it3;

for (int i = 0;i < n; i++) {

Sofer ob;

ob.citire();

ob.citire1();

lista.push\_back(ob);

}

int index = 1;

for (it = lista.begin();it != lista.end(); it++) {

cout << "\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n";

cout << "Datele salariatului numarul " << index << " :\n";

it -> afisare();

it -> afisare1();

index++;

}

it = lista.begin();

it2 = lista.begin();

float salMax = it->salariu();

it++;

for (it; it != lista.end(); it++) {

if (it -> salariu() > salMax) {

salMax = it -> salariu();

it2 = it;

}

}

cout << "\n\nSalariul maxim este de " << salMax << " lei\n";

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n";

cout << "Datele soferului cu salariu maxim: \n";

it2 -> afisare();

it2 -> afisare1();

it = lista.begin();

it3 = lista.begin();

int anMin = it->varsta();

it++;

for (it; it != lista.end(); it++) {

if (it -> varsta() < anMin) {

anMin = it -> varsta();

it3 = it;

}

}

cout << "\n\nVarsta celui mai tanar sofer este de: " << anMin << " ani\n";

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n";

cout << "Datele celui mai tanar salariat: \n";

it3 -> afisare();

it3 -> afisare1();

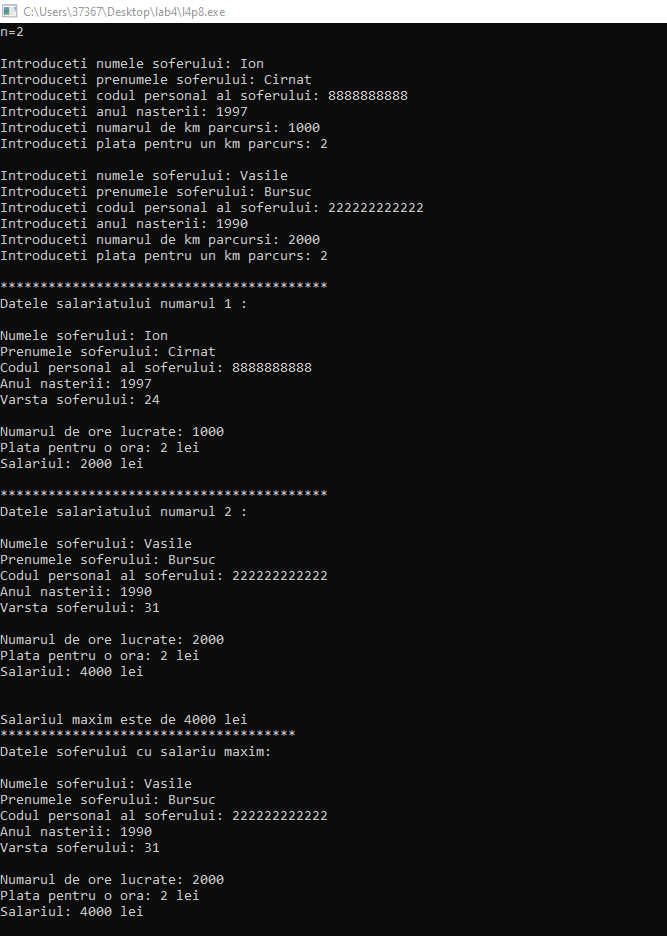
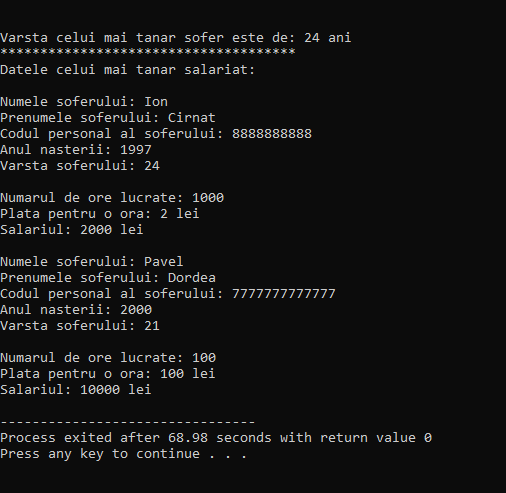
Sofer ob2(100,100);

ob2.afisare();

ob2.afisare1();

return 0;

}

**Screenshot-uri:** 

1. Creaţi clasa *Piramida*, derivata clasei *triunghi* (prob. 1). date: *înălţimea* piramidei,

metode: *citire* şi *afişare*, *suprafaţa\_totală* şi *volum*.

De la tastatură se citesc datele despre *n* piramide şi *m* triunghiuri. Elaboraţi un program prin intermediul căruia, la ecran se vor afişa datele despre toate piramidele (triunghiurile), piramida (triunghiul) cu suprafaţă maximă, piramida cu volumul maxim, triunghiul cu perimetrul minim.

**Codul C++ :**

#include <iostream>

#include <iterator>

#include <iomanip>

#include <vector>

#include <cmath>

using namespace std;

class Triad{

public:

float lat\_1;

float lat\_2;

float lat\_3;

public:

Triad(){

lat\_1 = 0;

lat\_2 = 0;

lat\_3 = 0;

}

Triad(float l1, float l2, float l3){

lat\_1 = l1;

lat\_2 = l2;

lat\_3 = l3;

}

void citire();

void afisare();

};

class Triunghi:public Triad{

public:

Triunghi():Triad(){

};

Triunghi(float l1,float l2, float l3):Triad(l1,l2,l3){

};

int verify();

float perim();

float area();

};

class Piramida:public Triunghi{

private:

float height;

public:

void citire1();

void afisare1();

float aria();

float volum();

float apotema();

};

void Triad::citire(){

cout << "\nIntroduceti lungimea primei lature a bazei piramidei: ";

cin >> lat\_1;

cout << "Introduceti lungimea celei de a doua lature a bazei piramidei: ";

cin >> lat\_2;

cout << "Introduceti lungimea celei de a treia lature a bazei piramidei: ";

cin >> lat\_3;

}

void Triad::afisare(){

cout << "\nLungimea primei lature: " << lat\_1 << " cm\n";

cout << "Lungimea celei de a doua latura: " << lat\_2 << " cm\n";

cout << "Lungimea celei de a treia latura: " << lat\_3 << " cm\n";

}

void Piramida::citire1(){

cout << "Introduceti inaltimea piramidei: ";

cin >> height;

}

void Piramida::afisare1(){

cout << "Inaltimea piramidei: " << height << " cm\n";

cout << "Aria piramidei: " << aria() << " centimetri patrati\n";

cout << "Volumul piramidei: " << volum() << " centimetri cubi\n";

}

float Piramida::apotema(){

float diagonala = lat\_1 \* lat\_1 + lat\_2 \* lat\_2;

diagonala = pow(diagonala,0.5);

float semiDiag = diagonala / 2.0;

float apot = height \* height + semiDiag \* semiDiag;

apot = pow(apot,0.5);

return apot;

}

float Piramida::aria(){

return perim() \* apotema() / 2.0 + area();

}

float Piramida::volum(){

return area() \* height / 3.0;

}

int Triunghi::verify(){

if (lat\_1 + lat\_2 > lat\_3 && lat\_2 + lat\_3 > lat\_1 && lat\_1 + lat\_3 > lat\_2) {

return 1;

}

return 0;

}

float Triunghi::perim(){

return lat\_1 + lat\_2 + lat\_3;

}

float Triunghi::area(){

float p = perim() / 2.0;

float s = p \* (p - lat\_1) \* (p - lat\_2) \* (p - lat\_3);

s = pow(s , 0.5);

return s;

}

int main() {

int n;

cout << "n=";

cin >> n;

vector <Piramida> lista;

vector <Piramida>::iterator it, it2, it3;

for (int i = 0;i < n; i++) {

Piramida ob;

ob.citire();

ob.citire1();

lista.push\_back(ob);

}

int index = 1;

for (it = lista.begin();it != lista.end(); it++) {

cout << "\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n";

cout << "Datele piramidei " << index << " :\n";

it -> afisare1();

cout << "Aria " << fixed << setprecision(2) << it->aria() << " centimetri patrati\n";

cout << "Volumul " << fixed << setprecision(2) << it->volum() << " centimetri cubi\n";

index++;

}

it = lista.begin();

it2 = lista.begin();

float maxArea = it->aria();

it++;

for (it; it != lista.end(); it++) {

if (it -> aria() > maxArea) {

maxArea = it -> aria();

it2 = it;

}

}

cout << "\n\nAria maxima este " << maxArea << " centimetri patrati\n";

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n";

cout << "Datele piramidei cu aria maxima: \n";

it2 -> afisare1();

cout << "Aria " << fixed << setprecision(2) << it2->area() << " centimetri patrati\n";

cout << "Volumul " << fixed << setprecision(2) << it2->volum() << " centimetri cubi\n";

it = lista.begin();

it3 = lista.begin();

float maxVolum= it -> volum();

it++;

for (it; it != lista.end(); it++) {

if (it -> volum() > maxVolum) {

maxVolum = it -> volum();

it3 = it;

}

}

cout << "\n\nPerimetrul maxim este " << maxVolum << " cm\n";

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n";

cout << "Datele triunghiului cu perimetrul maxim: \n";

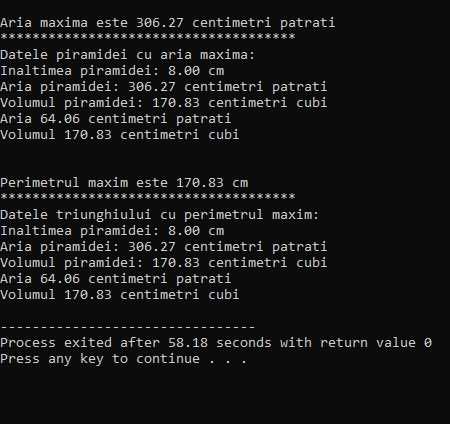
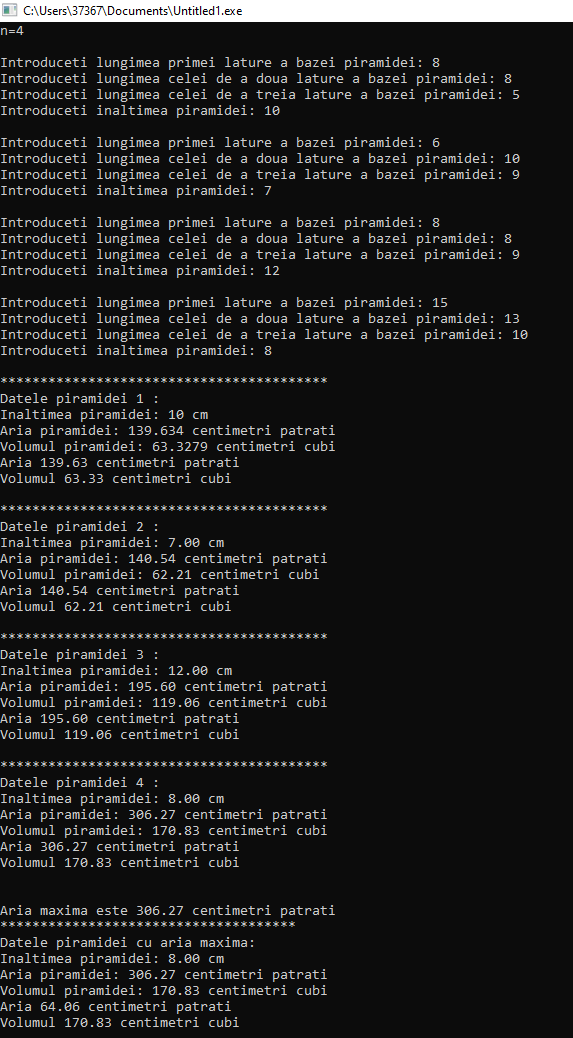
it3 -> afisare1();

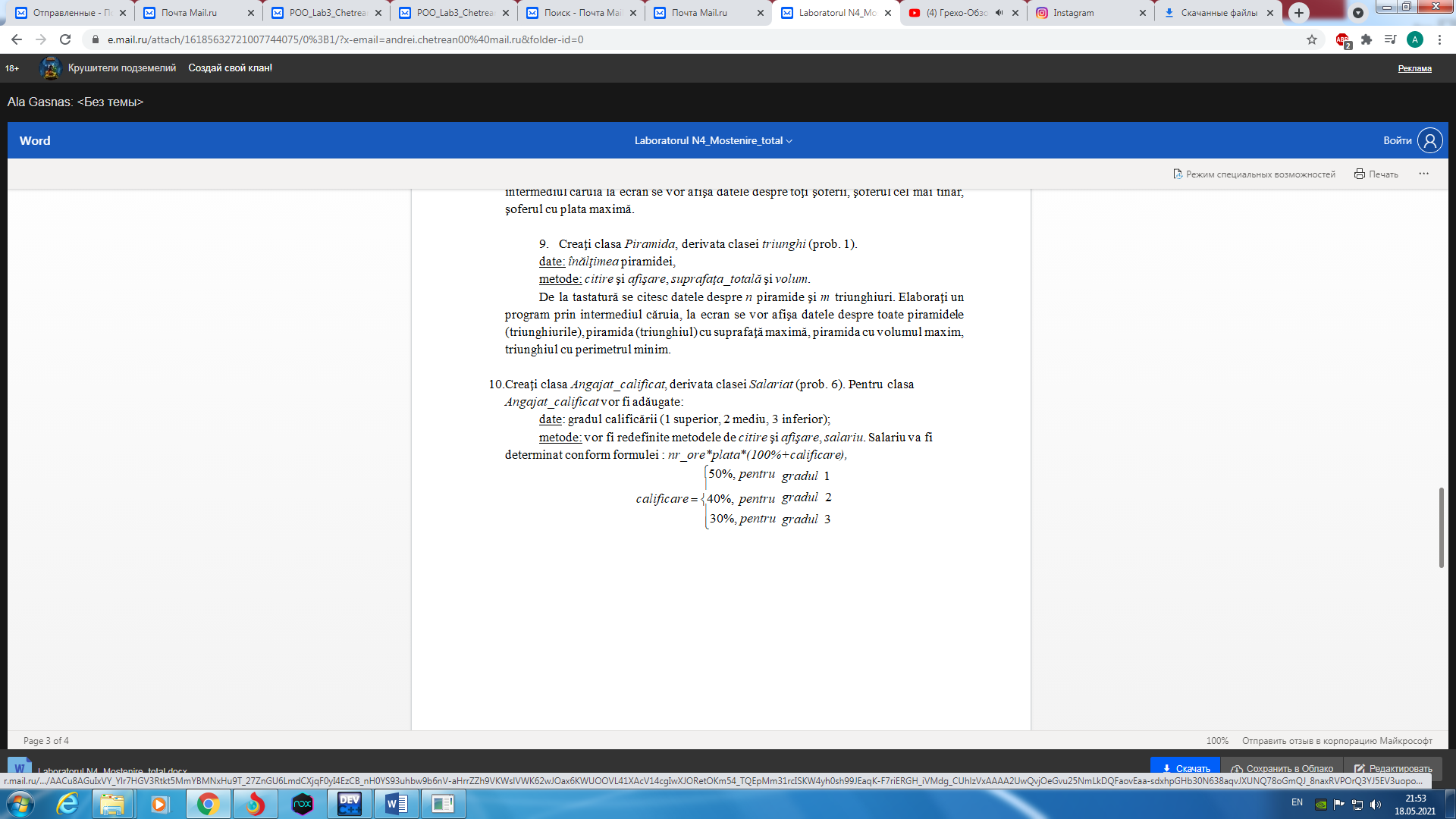
cout << "Aria " << fixed << setprecision(2) << it3->area() << " centimetri patrati\n";

cout << "Volumul " << fixed << setprecision(2) << it3->volum() << " centimetri cubi\n";

return 0;

}

**Screenshot-uri:**  




De la tastatură se citesc datele despre *n* angajaţi calificaţi. Elaboraţi un program prin intermediul căruia la ecran se vor afişa datele despre toţi angajaţii, angajatul cel mai tînăr şi angajatul cu salariu maxim.

**Codul C++ :**

#include <iostream>

#include <iterator>

#include <iomanip>

#include <vector>

#include <cmath>

#include <ctime>

#include <string>

using namespace std;

class Persoana{

public:

string nume = "Dumitru";

string prenume = "Munteanu";

string cnp = "7777777777777";

int an\_nas = 2001;

public:

Persoana(){};

Persoana(string str1, string str2, string str3, int an){

nume = str1;

prenume = str2;

cnp = str3;

an\_nas = an;

}

void citire();

void afisare();

int varsta();

};

class Salariat:public Persoana{

public:

int nr\_ore;

float plata;

int an\_ang;

public:

Salariat():Persoana(){};

Salariat(int nr,float plata2,int an2):Persoana(){

nr\_ore = nr;

plata = plata2;

an\_ang = an2;

};

void citire1();

void afisare1();

float salariu();

};

class Angajat\_calificat:public Salariat{

private:

int grad;

public:

void citire2();

void afisare2();

float salariu1();

};

void Persoana::citire(){

cout << "\nIntroduceti numele angajatului: ";

cin >> nume;

cout << "Introduceti prenumele angajatului: ";

cin >> prenume;

cout << "Introduceti codul personal al angajatului: ";

cin >> cnp;

cout << "Introduceti anul nasterii: ";

cin >> an\_nas;

}

void Persoana::afisare(){

cout << "\nNumele angajatului: " << nume << endl;

cout << "Prenumele angajatului: " << prenume << endl;

cout << "Codul personal al angajatului: " << cnp << endl;

cout << "Anul nasterii: " << an\_nas << endl;

cout << "Varsta angajatului: " << varsta() << endl;

}

int Persoana::varsta(){

return 2021 - an\_nas;

}

float Salariat::salariu(){

return nr\_ore \* plata;

}

void Angajat\_calificat::citire2(){

cout << "Introduceti gradul angajatului(1,2,3): ";

cin >> grad;

}

float Angajat\_calificat::salariu1() {

float spor = 1;

if (grad == 1) {

spor += 0.5;

} else if (grad == 2) {

spor += 0.4;

} else if (grad == 3) {

spor += 0.3;

}

return nr\_ore \* plata \* spor;

}

void Angajat\_calificat::afisare2(){

cout << "Gradul " << grad << endl;

cout << "Salariul " << salariu1() << " lei\n";

}

void Salariat::citire1(){

cout << "Introduceti numarul de ore lucrate de catre salariat: ";

cin >> nr\_ore;

cout << "Introduceti plata pentru o ora: ";

cin >> plata;

cout << "Introduceti anul angajarii: ";

cin >> an\_ang;

}

void Salariat::afisare1(){

cout << "\nNumarul de ore lucrate: " << nr\_ore << "\n";

cout << "Plata pentru o ora: " << plata << " lei\n";

cout << "Anul angajarii: " << an\_ang << "\n";

//cout << "Salariul: " << salariu() << " lei\n";

}

int main() {

int n;

cout << "n=";

cin >> n;

vector <Angajat\_calificat> lista;

vector <Angajat\_calificat>::iterator it;

vector <Angajat\_calificat>::iterator it2;

vector <Angajat\_calificat>::iterator it3;

for (int i = 0;i < n; i++) {

Angajat\_calificat ob;

ob.citire();

ob.citire1();

ob.citire2();

lista.push\_back(ob);

}

int index = 1;

for (it = lista.begin();it != lista.end(); it++) {

cout << "\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n";

cout << "Datele salariatului numarul " << index << " :\n";

it -> afisare();

it -> afisare1();

it -> afisare2();

index++;

}

it = lista.begin();

it2 = lista.begin();

float salMax = it->salariu1();

it++;

for (it; it != lista.end(); it++) {

if (it -> salariu1() > salMax) {

salMax = it -> salariu1();

it2 = it;

}

}

cout << "\n\nSalariul maxim este de " << salMax << " lei\n";

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n";

cout << "Datele salariatului cu salariu maxim: \n";

it2 -> afisare();

it2 -> afisare1();

it2 -> afisare2();

it = lista.begin();

it3 = lista.begin();

int anMin = it->varsta();

it++;

for (it; it != lista.end(); it++) {

if (it -> varsta() < anMin) {

anMin = it -> varsta();

it3 = it;

}

}

cout << "\n\nVarsta celui mai tanar angajat este de: " << anMin << " ani\n";

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n";

cout << "Datele celui mai tanar salariat: \n";

it3 -> afisare();

it3 -> afisare1();

it3 -> afisare2();

return 0;

}

**Screenshot-uri:**

