** Ministerul Educaţiei Republicii**

**Moldovei**

**Universitatea Tehnică a Moldovei**

Catedra: Calculatoare

**Raport**

Lucrare de laborator nr.3

## Tema: Supraincarcarea operatorilor

Varianta 8

A efectuat: Nicolenco Eugeniu, Gr. C-162

A verificat: Lector.Univ. Constantin Plotnic

2017

**Scopul lucrării**: familiarizarea studenţilor cu noţiunile operatori supraîncărcaţi.

**8.** Scrieţi un program care efectuează următoarele operaţii asupra unităţilor de lungime (de exemplu, metri, centimetri, milimetri):

* adunarea,
* înmulţirea cu un coeficient,
* scăderea,
* împărţirea la un coeficient,

Clasa trebuie să fie absolut funcţională, adică să conţină toţi constructorii necesari şi destructorul.

**Programa cod:**

#include <iostream>

#include <math.h>

#include <string.h>

#define Endl endl

using namespace std;

class Operatie{

public:

int km,m,cm,mm;

Operatie(int metri, int centim, int milim){

m=metri;

cm=centim;

mm=milim;

};

Operatie(){

km=0;

m=0;

cm=0;

mm=0;

};

Operatie& operator =(const Operatie &a){

m=a.m;

cm=a.cm;

mm=a.mm;

return \*this;

}

void Egalare(){

if(mm>=10){

do{

mm-=10;

cm++;

}while(mm>=10);

}

if(cm>=100){

do{

cm-=100;

m++;

}while(cm>=100);

}

if(m>=1000){

do{

m-=1000;

km++;

}while(m>=1000);

}

}

void Afisare(){

cout<<"\nLungimea finala este : ";

if(!km){

cout<<"\n\t\t\t\t\t\t"<<m<<" metri; "<<cm<<" cm; "<<mm<<" mm\n";

}else{

cout<<"\n\t\t\t\t\t"<<km<<" km; "<<m<<" metri; "<<cm<<" cm; "<<mm<<" mm";

}

}

void Citire(){

km=0;

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

cout<<"Dati metri: "; cin>>m;

cout<<""<<"Dati cm: "; cin>>cm;

cout<<""<<"Dati mm: "; cin>>mm;

this->Egalare();

}

friend Operatie operator+(Operatie a,Operatie b);

};

Operatie operator +(Operatie a, Operatie b){

Operatie temp;

temp.m=a.m+b.m;

temp.cm=a.cm+b.cm;

temp.mm=a.mm+b.mm;

temp.Egalare();

return temp;

}

Operatie operator -(const Operatie &a, const Operatie &b){

Operatie temp;

temp.m=a.m-b.m;

temp.cm=a.cm-b.cm;

temp.mm=a.mm-b.mm;

temp.Egalare();

return temp;

}

Operatie operator \*(Operatie a, int z){

Operatie temp;

temp.km=a.km\*z;

temp.m=a.m\*z;

temp.cm=a.cm\*z;

temp.mm=a.mm\*z;

temp.Egalare();

return temp;

}

Operatie operator /(Operatie a, int z){

Operatie temp,aux;

if(a.m%z!=0){

aux.cm=a.m%z;

}

if(a.cm%z!=0){

aux.mm=a.cm%z;

}

temp.m=a.m/z;

temp.cm=a.cm/z+aux.cm;

temp.mm=a.mm/z+aux.mm;

temp.Egalare();

return temp;

}

Operatie choose(int one,Operatie x1,Operatie x2,Operatie x3,Operatie x4,Operatie x5){

Operatie t;

switch(one){

case 1:t=x1;

break;

case 2:t=x2;

break;

case 3:t=x3;

break;

case 4:t=x4;

break;

case 5:t=x5;

break;

}

return t;

}

void menu(){

cout<<"\tMENU"<<Endl;

cout<<"1-Adunarea lungimilor"<<endl;

cout<<"2-Scaderea Lungimilor"<<endl;

cout<<"3-Inmultirea lungimilor"<<endl;

cout<<"4-Impartirea lungimilor"<<endl;

cout<<"5-Afisarea lungimilor introduse"<<endl;

cout<<"6-Terminarea programului"<<endl;

}

void Lungimi(Operatie n1,Operatie n2,Operatie n3,Operatie n4){

cout<<"\nLungimea 1 este : ";

if(!n1.km){

cout<<"\t\t"<<n1.m<<" metri; "<<n1.cm<<" cm; "<<n1.mm<<" mm";

}else{

cout<<"\t\t"<<n1.km<<" km; "<<n1.m<<" metri; "<<n1.cm<<" cm; "<<n1.mm<<" mm";

}

cout<<"\nLungimea 2 este : ";

if(!n2.km){

cout<<"\t\t"<<n2.m<<" metri; "<<n2.cm<<" cm; "<<n2.mm<<" mm";

}else{

cout<<"\t\t"<<n2.km<<" km; "<<n2.m<<" metri; "<<n2.cm<<" cm; "<<n2.mm<<" mm";

}

cout<<"\nLungimea 3 este : ";

if(!n3.km){

cout<<"\t\t"<<n3.m<<" metri; "<<n3.cm<<" cm; "<<n3.mm<<" mm";

}else{

cout<<"\t\t"<<n3.km<<" km; "<<n3.m<<" metri; "<<n3.cm<<" cm; "<<n3.mm<<" mm";

}

cout<<"\nLungimea 4 este : ";

if(!n1.km){

cout<<"\t\t"<<n4.m<<" metri; "<<n4.cm<<" cm; "<<n4.mm<<" mm";

}else{

cout<<"\t\t"<<n4.km<<" km; "<<n4.m<<" metri; "<<n4.cm<<" cm; "<<n4.mm<<" mm";

}

}

int main(){

int x1=0,x2=0,x3=0,x4=0,x=0;

int i=-1;

int y1=0,y2=0,y3=0,y4=0,y=0;

int terminat=1;

Operatie L1,L2,L3,L4,L5;

Operatie A1,A2,A3;

cout<<"Dati masurarile initiale\n";

cout<<"1 -";

L1.Citire();

cout<<"2 -";

L2.Citire();

cout<<"3 -";

L3.Citire();

cout<<"4 -";

L4.Citire();

while(terminat){

menu();

do{

cin>>i;

}while(i<=0||i>6);

switch(i){

case 1:cout<<"Alegeti lungimile de adaugat (5 pentru adaugarea la existenta)\n";

cin>>x;

while(x<1||x>5){

cout<<"Gresit : ";

cin>>x;

}

cout<<" si\n ";

cin>>y;

A1=choose(x,L1,L2,L3,L4,L5);

while(y<1||y>5){

cout<<"Gresit : ";

cin>>y;

}

A2=choose(y,L1,L2,L3,L4,L5);

L5=A1+A2;

cout<<"\t\t\t";

L5.Afisare();

break;

case 2:cout<<"Alegeti lungimile de scazut (5 pentru existenta)\n";

cin>>x;

while(x<1||x>5){

cout<<"Gresit : ";

cin>>x;

}

cout<<" si ";

cin>>y;

A1=choose(x,L1,L2,L3,L4,L5);

while(y<1||y>5){

cout<<"Gresit : ";

cin>>y;

}

A2=choose(y,L1,L2,L3,L4,L5);

L5=A1-A2;

L5.Afisare();

break;

case 3:cout<<"Alegeti lungimea de inmultit si multiplicatorul\n";

cin>>x;

while(x<1||x>5){

cout<<"Gresit : ";

cin>>x;

}

cout<<" si ";

cin>>y;

A1=choose(x,L1,L2,L3,L4,L5);

L5=A1\*y;

L5.Afisare();

break;

case 4:cout<<"Alegeti lungimea de impartit si fractia\n";

break;

cin>>x;

while(x<1||x>5){

cout<<"Gresit : ";

cin>>x;

}

cout<<" si ";

cin>>y;

A1=choose(x,L1,L2,L3,L4,L5);

L5=A1/y;

L5.Afisare();

break;

case 5:Lungimi(L1,L2,L3,L4);

break;

case 6:terminat=0;

break;

default:cout<<"Nedefinit";

break;

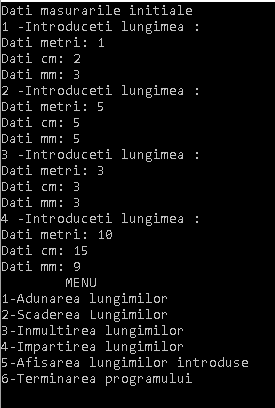
}

}

return 0;

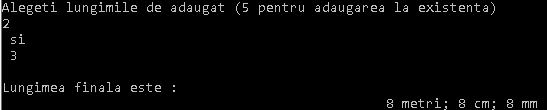
}

**Rezultatul Afisat:**



Drumul 1:

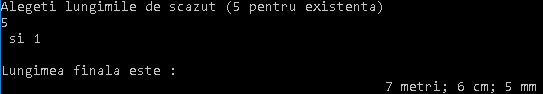
Adunarea :



La adunarea lungimii 2 si 3 sau aduna 3 si 5 (m,cm,mm)

Drumul 2:

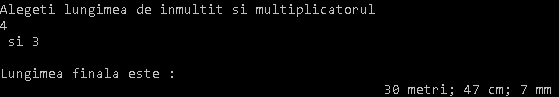
Scaderea:



La scaderea lungimii 1 din 5 sa scazut 8-1 metri,8-2cm si 8-3mm

Drumul 3:

Inmultirea:



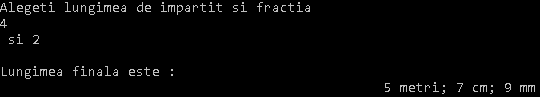
La inmultirea lungimii 4 cu multiplicatorul 3 am primit:

9\*3mm=27 = 2cm si 7mm;

15\*3=45(+2)cm 10\*3=30m;

Drumul 4:

Impartirea:



La imartirea 10 m la 2 primim 5m;

15cm primim 7cm si 5mm;

9mm primim 4mm si 5 mm de la cm;

Impartirea 2:





**Concluzie:** Comform cerintelor lucrarii au fost indeplinite constructori, operatori si metode de lucru cu unitatile de lungime. Programul ruleaza perfect si permite multe alte optiuni pentru a arata optimizat. Lungimile sunt adaptate dupa realitate, deci mm treci in cm, cm in m, si final m in km. Supraincarcarea operatorilor este o functiune foarte utila a limbajului C++ care usureaza scrierea programului si creaza o economie imensa de timp. Desi putem redefini (supraincarca) doar operatori deja existenti, aceasta nu creeaza incomoditati ci extinde planul de utilizare.