Universitatea Tehnică a Moldovei

Facultatea Calculatoare, Informatică şi Microelectronică

Departamentul Ingineria Software și Automatică

RAPORT

LUCRARE DE LABORATOR NR. 1

LA DISCIPLINA „PROGRAMAREA ÎN LIMBAJUL C++”

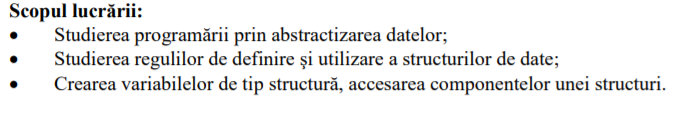
**Tema:** Structura – mecanism de abstractizare

Varianta 6

A efectuat : st.gr.SI-191 Brinzan Andrian

A verificat : V. Mititelu

Chișinău 2020



**Întrebări de control:**

1. Definiţi noţiunea – tip abstract de date.

**R:** Structura reprezinta un tip abstract de date deoarece prin intermediul acesteia putem implementa elemente ce pot avea tipuri de date diferite, ci nu numai de un singur fel.

2. Cum se defineşte o structură?

**R:** Este o multime de date grupate conform unei cerinte prevazute de catre programator pentru reprezentarea si prelucrarea datelor.

Structura se definteste astfel: struct <denumirea structurii>{

// definirea elementelor structurii };

3. Prin ce se deosebeşte structura de alte tipuri de date?

**R:** Structura poate fi comparata cu Array, dar este mai complexa:

Structura poate utiliza tipuri de date abstracte, pe cand array are date de acelasi tip.

4. Cum se defineşte o variabilă de tip structură?

**R:** Pentru definirea variabilelor este suficienta utilizarea numelui de structura. Variabilele unei structure pot fi de tip predefinit sau definit de utilizator.

5. Cînd se utilizează punctul, cînd - săgeata?

**R:**  Punctul este utilizat in cazul in care este nevoie sa facem referire la componentele unei structuri.

6. Care sunt deosebirile între structura din limbajul C şi C++?

**R:** In C o structura nu poate avea membri ai functiei in structura, C++ poate;

In C nu putem initializa tipul de dat structura direct in C, in C++ putem;

In C++ structura poate avea membri statici;

7. O structură poate oare să conţină altă structură?

**R:** Nu

8. O structură poate oare să conţină pointer spre ea însăşi?

**R:** Da

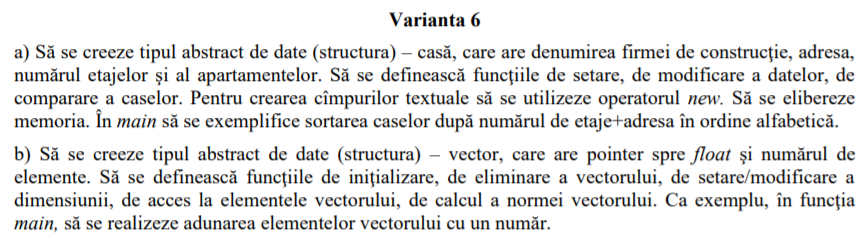
9. Poate oare să fie creată dinamic o variabilă de tip structură?

**R:**  Da, si poate fi eliminata pe parcurs.

10. Explicaţi cum are loc apelul prin referinţă.

**R:** Variabilele se transfmit functiilor prin parametri. Petru modificarea variabilei ea se transmite prin referinta.

**Sarcina lucrarii**:



#include <iostream>

#include <cstring>

using namespace std;

struct Company{

char \*denumire;

char \*adresa;

int etaj;

}\*listaComp;

void citireComp(Company &p1){

char temp[30];

cout << "Introduceti denumirea companiei: ";

cin >> temp;

p1.denumire = new char[strlen(temp)+1];

strcpy(p1.denumire, temp);

cout << "Introduceti adresa companiei: ";

cin >> temp;

p1.adresa = new char[strlen(temp)+1];

strcpy(p1.adresa, temp);

cout << "Introduceti numarul de etaje al acestui apartament ";

cin >> p1.etaj;

}

void afisareComp(Company const &p1){

cout << "Compania: " << p1.denumire << " ," << " str." << p1.adresa << ", Etajul: " << p1.etaj << endl;

}

void elimComp(Company &p1){

if(p1.denumire){

delete []p1.denumire;

p1.denumire = nullptr;

}

}

void setDenumire(Company &p1, char \*denumire){

if(p1.denumire)

delete []p1.denumire;

p1.denumire = new char[strlen(denumire)+1];

strcpy(p1.denumire, denumire);

}

void setAdresa(Company &p1, char \*adresa){

if(p1.adresa)

delete []p1.adresa;

p1.adresa = new char[strlen(adresa)+1];

strcpy(p1.adresa, adresa);

}

void setEtaj(Company &p1, int v){

p1.etaj = v;

}

void compareByEtajeAdresa(Company const &p1, Company const &p2){

if(p1.etaj > p2.etaj)

cout << p1.denumire << " are mai multe etaje. El are " << p1.etaj << " etaje." << endl;

else if(p1.etaj < p2.etaj)

cout << p1.denumire << " are mai putin etjaje. El are " << p1.etaj << " etaje." << endl;

else

cout << p1.denumire << " au acelasi numar de etaje " << p2.denumire <<". Ambele apartamentu au cate " << p1.denumire << " etaje." << endl;

}

int citireListaComp(){

int n;

cout << "Introduceti numarul de companii: ";

cin >> n;

listaComp = new Company[n];

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

citireComp(listaComp[i]);

return n;

}

void afisareListaComp(int n){

if(n == 0){

cout << "Lista este vida. Mai intai adaugati companii in lista" << endl;

return;

}

cout << "Lista companiilor: " << endl;

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

afisareComp(listaComp[i]);

}

void sortByEtaje(int n){

bool changed;

Company temp;

do{

changed = false;

for(int i=0; i<n-1; i++){

if(listaComp[i].etaj > listaComp[i+1].etaj){

temp = listaComp[i];

listaComp[i] = listaComp[i+1];

listaComp[i+1] = temp;

changed = true;

}

}

}while(changed);

}

void sortByAdresa(int n){

bool changed;

Company temp;

do{

changed = false;

for(int i=0; i<n-1; i++){

if(strcmp(listaComp[i].adresa, listaComp[i+1].adresa) > 0){

temp = listaComp[i];

listaComp[i] = listaComp[i+1];

listaComp[i+1] = temp;

changed = true;

}

}

}while(changed);

}

void elimListaComp(int n){

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

elimComp(listaComp[i]);

delete []listaComp;

}

int main()

{

Company p;

citireComp(p);

afisareComp(p);

elimComp(p);

cout << "Compania dupa modificare: " << endl;

setDenumire(p, "NewTOWN");

setAdresa(p, "Luceafarul 18");

setEtaj(p,7);

afisareComp(p);

Company p2;

citireComp(p2);

compareByEtajeAdresa(p, p2);

int n = citireListaComp();

afisareListaComp(n);

sortByAdresa(n);

cout << "Lista companiilor sortata dupa adresa este: " << endl;

afisareListaComp(n);

sortByEtaje(n);

cout << "Lista persoanelor sortata dupa numarul de etaje este: " << endl;

afisareListaComp(n);

elimListaComp(n);

return 0;

}

**OUTPUT:**

