****

**MINISTERUL EDUCAȚIEI, CULTURII ȘI CERCETĂRII**

**AL REPUBLICII MOLDOVA**

**Universitatea Tehnică a Moldovei**

**Facultatea Calculatoare, Informatică şi Microelectronică**

**Departamentul Ingineria Software și Automatică**

**Raport**

**pentru lucrarea de laborator Nr.1**

***la cursul de “Programarea orientata pe obiecte”***

Efectuat: Studentul gr. SI-191 **Pelin Bogdan**

Verificat:  *conf. Univ* **Mititelu Vitalii**

**Chișinău – 20****20**

***Tema: Structura – mecanism de abstractizare***

**Scopul lucrării:**

• Studierea programării prin abstractizarea datelor;

• Studierea regulilor de definire şi utilizare a structurilor de date;

• Crearea variabilelor de tip structură, accesarea componentelor unei structuri.

**Varianta 7**

а) Să se creeze tipul abstract de date (structura) – monitor, care are denumirea firmei producătoare, dimensiunea în inch, numărul de culori şi rezoluţie. Să se definească funcţiile de setare, de modificare a datelor, de comparare a monitoarelor. Pentru crearea cîmpurilor textuale să se utilizeze operatorul new. Să se elibereze memoria. În main să se exemplifice căutarea monitorului potrivit după dimensiune şi alte caracteristici.

b) Să se creeze tipul abstract de date (structura) – vector, care are pointer spre char şi numărul de elemente. Să se definească funcţiile de iniţializare, de eliminare a vectorului, de setare/modificare a dimensiunii, de acces la elementele vectorului, de calcul a produsului elementelor negative ale vectorului. Ca exemplu, în funcţia main, să se realizeze înmulţirea a doi vectori.

* **A)**

#include<conio.h>

#include<stdio.h>

#include<string.h>

#include<iostream>

#include<stdlib.h>

using namespace std;

struct monitor

{

char \*nume\_monitor;

float size;

int nr\_color;

int resolution;

}

monitors[10];

int\* i = new int(0);

void cautare\_nume(monitor\*monitors)

{

system("cls");

char cuv[64];

int c=0;

cout<<endl<<"Numele monitorului cautat: ";

cin>>cuv;

cout<<endl<<"Rezultatele cautarii sunt: ";

for(int j=0;j<(\*i);j++)

{

if((strcmp(monitors[j].nume\_monitor,cuv))==0)

{

cout<<endl<<monitors[j].nume\_monitor<<" "<<monitors[j].size<<" "<<monitors[j].nr\_color<<" " <<monitors[j].resolution;

c=-1;

}

}

if(c!=-1)

cout<<endl<<"\n\n\n\n\n\n\t\t\t\t\t"<<"Nu a fost gasit nici un monitor cu asa nume";

getch();

}

void cautare\_size(monitor\*monitors)

{

system("cls");

float c=0;

cout<<endl<<"Marimea cautata: ";

cin>>c;

cout<<endl<<"Rezultatele cautarii sunt: ";

for(int j=0;j<(\*i);j++){

if(monitors[j].size==c)

{

cout<<endl<<monitors[j].nume\_monitor<<" "<<monitors[j].size<<" "<<monitors[j].nr\_color<<" "<<monitors[j].resolution;

c=-1;

}

}

if(c!=-1)

cout<<endl<<"\n\n\n\n\n\n\t\t\t\t\t"<<"Nu a fost gasit nici un monitor cu asa parametri";

getch();

}

monitor init(monitor\* monitors)

{

char new\_monitor[64];

cout<<"\nDenumirea monitorului: ";

cin>>new\_monitor;

if(monitors[\*i].nume\_monitor) delete[] monitors[\*i].nume\_monitor;

monitors[\*i].nume\_monitor=new char[strlen(new\_monitor)+1];

strcpy(monitors[\*i].nume\_monitor,new\_monitor);

cout<<"\nMarimea ecranului: ";

cin>>monitors[\*i].size;

cout<<"\nNr de culori: ";

cin>>monitors[\*i].nr\_color;

cout<<"\nResolutia: ";

cin>>monitors[\*i].resolution;

(\*i)++;

return \*monitors;

}

void free(monitor\*monitors){

int g;

for(g=0;g<\*i;g++){

if(monitors[g].nume\_monitor) delete[] monitors[g].nume\_monitor;

monitors[g].nume\_monitor=NULL;

(\*i)=0;

}

}

void compare(monitor\*monitors){

system("cls");;

int j;

monitor \*temp;

for(int jj=0;jj<(\*i);jj++){

for(j=0;j<(\*i);j++){

temp=new monitor[sizeof(monitors[j])];

if(monitors[j+1].size>monitors[j].size){

\*temp=monitors[j];

monitors[j]=monitors[j+1];

monitors[j+1]=\*temp;

delete[] temp;

}

}

}

j=0;

while(monitors[j].nume\_monitor)

{

cout<<endl<<j+1<<"." << monitors[j].nume\_monitor;

cout<<" "<<monitors[j].size;

cout<<"\t"<<monitors[j].nr\_color;

cout<<"\t"<<monitors[j].resolution;

j++;

}

getch();

}

void modificare(monitor \*monitors)

{

system("cls");

int j=0;

while(monitors[j].nume\_monitor){

cout<<endl<<j+1<<"." << monitors[j].nume\_monitor;

cout<<" "<<monitors[j].size;

cout<<"\t"<<monitors[j].nr\_color;

cout<<"\t"<<monitors[j].resolution;

j++;

}

cout<<endl<< "Modifica cimpul nr.: ";

int k=0; char n\_monitor[23];

cin>> k;

k-=1;

cout<<endl<<"Numele monitorului: ";

gets(n\_monitor);

cout<<endl<<"Marimea: ";

cin>>monitors[k].size;

cout<<endl<<"Nr de culori: ";

cin>>monitors[k].nr\_color;

cout<<endl<<"Resolution: ";

cin>>monitors[k].resolution;

if(monitors[k].nume\_monitor) delete[] monitors[k].nume\_monitor;

monitors[k].nume\_monitor=new char[strlen(n\_monitor)+1];

strcpy(monitors[k].nume\_monitor,n\_monitor);

getch();

}

void meniu(){

system("cls");;

int n=0;

cout<<" MENIU"<<endl;

cout<<" 1. Initializare"<<endl;

cout<<" 2. Modificare"<<endl;

cout<<" 3. Comparare"<<endl;

cout<<" 4. Eliberarea memoriei"<<endl;

cout<<" 5. Cautare dupa cimpul <<Denumire>>"<<endl;

cout<<" 6. Cautare dupa cimpul <<Marimea ecarn >>"<<endl;

cout<<" 7. Exit";

n=0;

n=getch();

n-=48;

switch (n){

case 1: system("cls"); init(monitors);

meniu();

case 2: if(\*i>0) modificare(monitors);

else{

system("cls");

cout<<"\n\n\n\n\n\n\n\t\tNu sunt elemente pentru modificare ";

}

getch();

meniu();

case 3:

if(\*i>0) compare(monitors);

else {system("cls");cout<<endl<<"\n\n\n\n\n\t\t\tNu sunt cimpuri pentru comparat";getch();}

meniu();

case 4:

system("cls");

free(monitors);

cout<<endl<<"\n\n\n\n\n\t\t\t\tMemoria a fos eliberata";

getch();

meniu();

case 5: cautare\_nume(monitors);

meniu();

case 6: cautare\_size(monitors);

meniu();

case 7: exit(0);

default: meniu();

}

}

int main(){

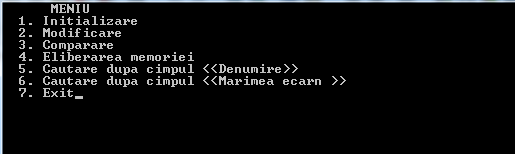
system("cls");

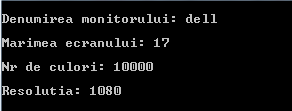
meniu();

getch();

}

* **Afisarea punctul a**

****

****

* **B)**

#include <iostream>

#include <stdio.h>

class vector{

public:

vector(){

this->array = NULL;

this->len = 0;

}

vector(char arr[], int arrayLen){

this->array = (char\*)malloc(arrayLen \* sizeof(char));

for(short unsigned int i = 0; i < arrayLen; i++){

this->array[i] = arr[i];

}

this->len = arrayLen;

}

~vector(){

free(this->array);

}

void printVector();

void changeSize(int a);

void changeElement(int a, char b);

void showElement(int a);

private:

char \*array;

int len;

};

void vector::printVector(){

printf("\nElementele vectorului: \n");

for(int i = 0;i < this->len; i++){

printf("%c ", this->array[i]);

}

printf("\n");

}

void vector::changeSize(int a){

if(a > 0){

this->array = (char\*)realloc(this->array, a\*sizeof(char));

}

else if(a == 0){

free(this->array);

}

this->len = a;

}

void vector::changeElement(int a, char b){

this->array[a] = b;

printf("\nElementul cu indexul %d a fost modificat\n", a);

}

void vector::showElement(int a){

printf("\nElementul cu index %i este %c\n" ,a, this->array[a]);

}

int main(){

char temp [] = {'a', 'b', 'x', 'y', '1'};

vector A(temp, 5);

A.printVector();

A.changeSize(6);

printf("Am alocat memorie pentru inca un element:");

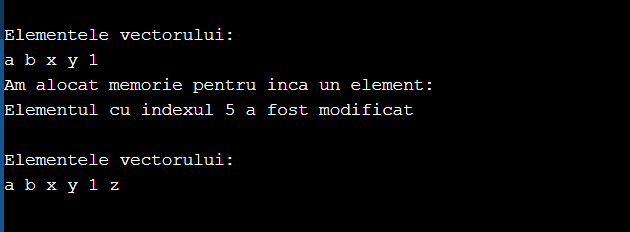
A.changeElement(5, 'z');

A.printVector();

return 0;

}

* **Afisarea punctul b**

****

**Concluzia:**

In acesata lucrare am facut cunostinta cu bazele limbajului de programare C++,anume ce reprezinta structura. Structura fiind o mulţime de date grupate, conform unei ierarhii, de obicei de tipuri diferite.