Structuri

1. Declarați structura **Student** cu următoarele câmpuri: **nume** (char\*), **varsta** (int), **note** (int\*), **nrNote** (int)
2. In **main** se va declara o varibila de tip char[] care va conține numele unui student și un vector de tip int[] care va conține notele studentului (10, 9).
3. Declarați în **main** o variabila de tip **Student**
4. Inițializați câmpurile din variabila de tip Student cu ajutorul variabilelor declarate anterior. Utilizați o valoare la alegere pentru vârstă.
5. Sa se declare o funcție care asigură afișarea câmpurilor din variabila de tip Student
6. Se declara o funcție care inițializează un obiect de tip Student cu valori primite ca parametrii. Funcția returnează studentul.
7. In **main** se va declara o noua variabila de tip **Student**
8. Se declara o metoda care primește un parametru de tip **Student** (transmis prin valoare) și asigura preluarea datelor de la tastatură. Metoda returnează **void**.
9. Apelați metoda declarata anterior pentru a inițializa datele variabilei de tip **Student**.
10. Ce se observă? Ce modificări sunt necesare?
11. Declarați o metodă care primește un parametru de tip **Student** (transmis prin valoare) și afișează informațiile acestuia.
12. Declarați o metoda care primește un parametru de tip **Student** (transmis prin valoare) și afișează numele și media studentului.
13. Apelați metoda pentru variabila de tip Student declarată în **main**.
14. Adăugați în structura **Student** un pointer care să poată fi încărcat cu adresa unei metode care primește ca parametru un obiect de tip Student.
15. Apelați metoda de calcul al mediei prin intermediul pointerului la metoda.
16. Se definește structura grupa cu campurile: nr\_grupa (int), nr\_studenti (int), studenti (Student\*)
17. Se declara o funcție care inițializează un obiect de tip Grupa cu valori primite ca parametrii. Functia returneaza grupa creata. Vectorul de studenți este creat in **main** si dat ca parametru acestei funcții, precum si numărul de studenți.

|  |
| --- |
| #include <iostream>  **using** **namespace** std**;**  struct TheVector  **{**  double elem**[**10**];**  int dim**;**  **};**  void aduna**(**TheVector x**,** TheVector y**,** TheVector **&**rez**)**  **{**  **for** **(**int i **=** 0**;** i**<**x**.**dim**;** i**++)**  rez**.**elem**[**i**]** **=** x**.**elem**[**i**]** **+** y**.**elem**[**i**];**  **}**  void main**()**  **{**  TheVector a1**,** a2**,** a3**;**  a1**.**dim **=** 3**;**  a2**.**dim **=** 3**;**  a3**.**dim **=** 3**;**  **for** **(**int i **=** 0**;** i **<** a1**.**dim**;** i**++)**  **{**  cout**<<**"a1.elem["**<<**i**<<**"]="**;**  cin**>>**a1**.**elem**[**i**];**  cout**<<**"a2elem["**<<**i**<<**"]="**;**  cin**>>**a2**.**elem**[**i**];**  **}**  aduna**(**a1**,** a2**,** a3**);**  **for** **(**int i **=** 0**;** i **<** 3**;** i**++)**  cout**<<**endl**<<**"a3.elem["**<<**i**<<**"] = "**<<**a3**.**elem**[**i**];**  **}** |

Exercitiul 3

|  |
| --- |
| #include <iostream>  **using** **namespace** std**;**  struct TheVector  **{**  double elem**[**10**];**  int dim**;**  void**(\***fPrelucrare**)(**TheVector**,** TheVector**,** TheVector**&);**  **};**  void aduna**(**TheVector x**,** TheVector y**,** TheVector **&**rez**)**  **{**  **for** **(**int i **=** 0**;** i**<**x**.**dim**;** i**++)**  rez**.**elem**[**i**]** **=** x**.**elem**[**i**]** **+** y**.**elem**[**i**];**  **}**  void main**()**  **{**  TheVector a1**,** a2**,** a3**;** a1**.**dim **=** 3**;**  a2**.**dim **=** 3**;** a3**.**dim **=** 3**;**    a1**.**fPrelucrare **=** aduna**;**  **for** **(**int i **=** 0**;** i **<** a1**.**dim**;** i**++)**  **{**  cout **<<** "a1.elem[" **<<** i **<<** "]="**;**  cin **>>** a1**.**elem**[**i**];**  cout **<<** "a2elem[" **<<** i **<<** "]="**;**  cin **>>** a2**.**elem**[**i**];**  **}**  //aduna(a1, a2);  a1**.**fPrelucrare**(**a1**,** a2**,** a3**);**    **for** **(**int i **=** 0**;** i **<** 3**;** i**++)**  cout **<<** endl **<<** "a3.elem[" **<<** i **<<** "] = " **<<** a3**.**elem**[**i**];**  **}** |

Exercitiul 4

|  |
| --- |
| #include <iostream>  **using** **namespace** std**;**  class TheVector  **{**  private**:**  double elem**[**10**];**  int dim**;**  public**:**  TheVector aduna**(**TheVector x**,** TheVector y**);**  void init**(**int dimp**);**  void afiseaza**(**TheVector x**);**  **};**  TheVector TheVector**::**aduna**(**TheVector x**,** TheVector y**)**  **{**  int i **=** 0**;**  TheVector rez**;**  **for** **(**int i**=**0**;** i**<**x**.**dim**;**i**++)**  rez**.**elem**[**i**]** **=** x**.**elem**[**i**]** **+** y**.**elem**[**i**];** //merge chiar dk este private cum sunt in interiorul obiectului  **return** rez**;**  **}**  void TheVector**::**init**(**int dimp**)**  **{**  **this->**dim **=** dimp**;**  **for(**int i**=**0**;** i**<this->**dim**;** i**++)**  **{**  cout**<<**"elem:"**<<**i**<<**"= "**;** cin**>>this->**elem**[**i**];**  **}**  **}**  void TheVector**::**afiseaza**(**TheVector x**)**  **{**  **for(**int i**=**0**;** i**<**x**.**dim**;**i**++)** cout**<<**"/n"**<<**x**.**elem**[**i**];**  **}**  void main**()**  **{**  TheVector a1**,**a2**,**a3**;**  a1**.**init**(**3**);**  a2**.**init**(**3**);**  a3**.**init**(**3**);**  a3 **=** a1**.**aduna**(**a1**,**a2**);**  a3**.**afiseaza**(**a3**);**  **}** |