Laborator 4 - 322ABa

Clasa `Jucator`

22.10.2025

Scopul Laboratorului

Acest laborator se concentrează pe managementul memoriei dinamice în C++. Vom învăța să gestionăm corect alocarea și dealocarea memoriei pentru membrii de tip pointer, implementând corect constructorul de copiere (deep copy) și destructorul. Toate acestea vor fi aplicate pe o clasă `Jucator`.

---

1 Pasul 1: Definirea Clasei `Jucator` (.h)

Creați fișierul header `Jucator.h`. Acesta va conține toate directivele `#include` necesare. Clasa va conține un pointer `char*` pentru username, care va necesita alocare dinamică.

Copiați codul de mai jos în fișierul `Jucator.h`:

```cpp
#ifndef JUCATOR_H
#define JUCATOR_H

#include <iostream>
#include <cstring> // Pentru strlen si strcpy

class Jucator {
private:
    char* username; // Username alocat dinamic
    int scor;
    static int contorJucatori;

public:
    // Constructor default
    Jucator();

    // Constructor cu parametri
    Jucator(const char* username, int scor);

    // Constructor de copiere (ESENTIAL pentru deep copy)
    Jucator(const Jucator& other);

    // Destructor (ESENTIAL pentru a elibera memoria)
    ~Jucator();

    // Metoda de afisare (const)
    void afisare() const;

    // Metoda pentru a adauga puncte la scor
    void adaugaScor(int puncte);

    // Metoda statica
    static int getContorJucatori();
};

#endif // JUCATOR_H
```

2 Pasul 2: Implementarea Clasei `Jucator` (.cpp)

Creați fișierul `Jucator.cpp`. Acesta trebuie să includă **doar** `"Jucator.h"`. Implementați logica pentru fiecare funcție conform cerințelor.

Copiați acest schelet în `Jucator.cpp` și completați secțiunile `TODO`:

```cpp
#include "Jucator.h"

// Initializarea membrului static
int Jucator::contorJucatori = 0;

// Constructor default
Jucator::Jucator() {
    // TODO: Initializati scorul cu 0.
    // TODO: Alocati memorie pentru username si copiati un string gol ""
    //       sau initializati-l cu nullptr.
    // TODO: Incrementati contorul.
}

// Constructor cu parametri
Jucator::Jucator(const char* username, int scor) {
    // TODO: Initializati this->scor cu valoarea primita.
    // TODO: Alocati memorie pentru this->username (strlen(username) + 1).
    // TODO: Copiati continutul din 'username' in 'this->username' (strcpy).
    // TODO: Incrementati contorul.
    // TODO: Afisati un mesaj (ex: "Jucator creat: ...").
}

// Constructor de copiere (DEEP COPY)
Jucator::Jucator(const Jucator& other) {
    // TODO: Copiati scorul din 'other' in 'this'.
    // TODO: Alocati *memorie noua* pentru this->username.
    // TODO: Copiati continutul numelui din 'other' in 'this->username'.
    //       NU faceti `this->username = other.username;` (shallow copy)
    // TODO: Incrementati contorul.
    // TODO: Afisati un mesaj (ex: "Clona jucatorului: ...").
}

// Destructor
Jucator::~Jucator() {
    // TODO: Eliberati memoria alocata pentru 'username' folosind delete[].
    // TODO: Decrementati contorul.
    // TODO: Afisati un mesaj (ex: "Jucator sters: ...").
}

// Metoda de afisare
void Jucator::afisare() const {
    // TODO: Afisati username-ul si scorul jucatorului.
}

// Metoda pentru a adauga puncte la scor
void Jucator::adaugaScor(int puncte) {
    // TODO: Adunati 'puncte' la membrul 'scor'.
}

// Metoda statica
int Jucator::getContorJucatori() {
    // TODO: Returnati valoarea contorului.
}
```

3 Pasul 3: Cerințe pentru `main.cpp`

Creați un fișier `main.cpp` în care să testați clasa. Programul vostru trebuie să demonstreze:

- Crearea a doi jucători folosind constructorul cu parametri.
- Afișarea numărului de jucători activi folosind metoda statică.
- Crearea unui al treilea jucător folosind constructorul de copiere.
- Modificarea scorului unui jucător folosind metoda `adaugaScor`.
- Afișarea tuturor jucătorilor pentru a verifica starea lor.
- Urmărirea mesajelor de la constructori/destructor pentru a vedea ciclul de viață al obiectelor.

Instrucțiuni generale

- Acceptați invitația pe GitHub Classroom pentru laboratorul 4, link: https://classroom.github.com/a/e5IrDPgO
- După accept, clonați repository-ul pe calculator:
  `git clone <link_repository>`
- Creați fișierele `main.cpp`, `Jucator.h` și `Jucator.cpp`.
- Toată logica clasei `Jucator` trebuie implementată în `Jucator.cpp`, cu definiția în `Jucator.h`.
- În `main.cpp` scrieți doar apelurile și testarea funcționalității clasei.
- La final, urcați modificările:
  - `git add .`
  - `git commit -m "Laborator 4"`
  - `git push`