Laborator 6 – Moștenire simplă

Tema: Studentul și prima sa mașină 🚗

Autor: ing. Eduard Donea

Poveste

După ce și-a luat permisul de conducere, Andrei își cumpără prima lui mașină second-hand. Deși nu știe foarte multe despre ea, vrea să își facă o mică aplicație care să îl ajute să țină evidența vehiculelor și a caracteristicilor lor. Pentru asta, va construi o clasă generală `Vehicul`, din care va deriva o clasă mai specifică `Masina`.

Acest laborator are scopul de a exersa conceptul de **moștenire simplă** și de a arăta cum putem **extinde o clasă de bază** cu funcționalități noi, fără a rescrie tot codul.

Scopul laboratorului

- înțelegerea relației de tip „**is-a**” (ex: o mașină este un vehicul);
- folosirea specificatorului de acces `protected`;
- apelarea constructorilor din clasa de bază în lista de inițializare;
- utilizarea corectă a operatorilor de copiere și atribuire în clasa derivată;
- afișarea corectă a informațiilor folosind operatorul `<<`.

Structura fișierelor

- **Vehicul.h** – declararea clasei de bază `Vehicul`;
- **Vehicul.cpp** – implementarea clasei `Vehicul`;
- **Masina.h** – declararea clasei derivate `Masina`;
- **Masina.cpp** – implementarea clasei `Masina`;
- **main.cpp** – testarea moștenirii și a metodelor.

Fișierul `Vehicul.h`

```h
#ifndef VEHICUL_H
#define VEHICUL_H

// Includeti bibliotecile necesare aici

class Vehicul {
protected:
    float vitezaMaxima;
    char* marca;

public:
    Vehicul();
    Vehicul(float vitezaMaxima, const char* marca);
    Vehicul(const Vehicul& v);
    Vehicul& operator=(const Vehicul& v);
    ~Vehicul();

    float getVitezaMaxima() const;
    char* getMarca() const;
    void setVitezaMaxima(float viteza);
    void setMarca(const char* marcaNoua);

    friend ostream& operator<<(ostream& out, const Vehicul& v);
};

#endif
```

Fișierul `Masina.h`

```h
#ifndef MASINA_H
#define MASINA_H

#include "Vehicul.h"

class Masina : public Vehicul {
    int numarLocuri;
    char* combustibil; // ex: benzina, motorina, electric

public:
    Masina();
    Masina(float viteza, const char* marca, int locuri, const char*
    ↪  combustibil);
    Masina(const Masina& m);
    Masina& operator=(const Masina& m);
    ~Masina();

    int getNumarLocuri() const;
    char* getCombustibil() const;

    void setNumarLocuri(int locuri);
    void setCombustibil(const char* tip);

    friend ostream& operator<<(ostream& out, const Masina& m);
};

#endif
```

Fișierul `main.cpp` (model)

OBS: Este doar un model, vă puteți scrie main-ul vostru cum doriți.

```cpp
#include "Masina.h"

int main() {
    cout << "=== Testare mostenire simpla Vehicul -> Masina ===\\n\\n";

    Vehicul v1(180, "Yamaha");
    cout << "Vehicul:\\n" << v1 << endl;

    Masina m1(220, "BMW", 5, "benzina");
    cout << "Masina:\\n" << m1 << endl;

    Masina m2 = m1;
    cout << "Masina copiata:\\n" << m2 << endl;

    Masina m3;
    m3 = Masina(130, "Dacia", 5, "GPL");
    cout << "Masina asignata:\\n" << m3 << endl;

    return 0;
}
```

Cerințe

1. Implementați complet clasa `Vehicul` cu constructori, operatori și metoda de afișare.
2. Implementați complet clasa `Masina` care moștenește clasa `Vehicul`.
3. Apelați constructorul din clasa părinte în lista de inițializare a constructorului din clasa derivată.
4. Testați operatorul de copiere și operatorul `<<` pentru afișare.
5. Structurați corect codul în fișiere și adăugați comentarii explicative.

OBS: Notarea la acest laborator va fi înlocuită de rezultatele obținute la testul de la final.

Compilare și rulare

```
g++ -g Vehicul.cpp Masina.cpp main.cpp -o lab6
valgrind ./lab6
```

Bonus GitHub Classroom

Studenții pot obține până la **3p bonus** dacă încarcă toate laboratoarele până la finalul semestrului, respectiv **1p bonus** dacă încarcă mai mult de jumătate dintre ele.

Pași pentru încărcarea laboratorului:

1. Accesați cursul de pe OCW – Programare Orientată pe Obiecte, secțiunea **Resurse** → **Bonus GitHub laborator**.
2. Acolo veți găsi linkul **GitHub Classroom** pentru Laboratorul 6:

   `https://classroom.github.com/a/ZHG6I0Xp`

3. După acceptare, veți primi propriul repository, ex. `laborator-6-NumePrenume`.
4. Clonați repository-ul local:

```
git clone
↪  git@github.com:Laborator-POO-2025-2026/laborator-6-NumePrenume.git
cd laborator-6-NumePrenume
```

5. După ce ați adăugat fișierele `Vehicul.h`, `Vehicul.cpp`, `Masina.h`, `Masina.cpp` și `main.cpp`, urmează comenzile:

```
git add .
git commit -m "Mesaj de commit"
git push origin main
```

Atenție: laboratoarele copiate între studenți sunt considerate nevalide, iar niciunul dintre cei doi participanți nu va primi bonusul. Repository-urile sunt monitorizate automat de sistemul GitHub Classroom.