Ministerul Educației și Cercetării al Republicii Moldova
Colegiul Universității Tehnice a Moldovei
Administrarea Aplicațiilor Web

RAPORT

Lucrare de laborator nr.3

Disciplina: Asistență pentru programarea orientată pe obiecte

Tema: Sistem de gestionare a parcării

A efectuat: Tintiuc Cătălin

Chişinău 2025

Codul în format text

```cpp
#include <iostream>
#include <fstream>
#include <vector>
#include <string>
using namespace std;

class Vehicul {
protected:
    string nrInmatriculare;
    int oraIntrare, oraIesire;
public:
    Vehicul(string nr, int intrare, int iesire) : nrInmatriculare(nr),
oraIntrare(intrare), oraIesire(iesire) {}
    virtual ~Vehicul() {}
    int calculTimpParcare() { return oraIesire - oraIntrare; }
    virtual double calculTarif() = 0;
    virtual void afisare() {
        cout << "Numar: " << nrInmatriculare << ", Timp: " <<
calculTimpParcare() << " ore, Tarif: " << calculTarif() << " lei\n";
    }
};

class Masina : public Vehicul {
public:
    Masina(string nr, int intrare, int iesire) : Vehicul(nr, intrare, iesire)
{}
    double calculTarif() override { return calculTimpParcare() * 5; }
};

class Motocicleta : public Vehicul {
public:
    Motocicleta(string nr, int intrare, int iesire) : Vehicul(nr, intrare,
iesire) {}
    double calculTarif() override { return calculTimpParcare() * 3; }
};

class Camion : public Vehicul {
public:
    Camion(string nr, int intrare, int iesire) : Vehicul(nr, intrare, iesire)
{}
    double calculTarif() override { return calculTimpParcare() * 10; }
};

class Bicicleta : public Vehicul {
public:
    Bicicleta(string nr, int intrare, int iesire) : Vehicul(nr, intrare,
iesire) {}
```

```
    double calculTarif() override {
        int timp = calculTimpParcare();
        return (timp <= 2) ? 0 : (timp - 2) * 2;
    }
};

void citireFisier(vector<Vehicul*>& vehicule, const string& numeFisier) {
    ifstream file(numeFisier);
    string tip, nr, oraIntrareStr, oraIesireStr;
    int oraIntrare, oraIesire;
    while (file >> tip >> nr >> oraIntrareStr >> oraIesireStr) {
        oraIntrare = stoi(oraIntrareStr.substr(0, 2));
        oraIesire = stoi(oraIesireStr.substr(0, 2));

        if (tip == "Masina") vehicule.push_back(new Masina(nr, oraIntrare,
oraIesire));
        else if (tip == "Motocicleta") vehicule.push_back(new Motocicleta(nr,
oraIntrare, oraIesire));
        else if (tip == "Camion") vehicule.push_back(new Camion(nr,
oraIntrare, oraIesire));
        else if (tip == "Bicicleta") vehicule.push_back(new Bicicleta(nr,
oraIntrare, oraIesire));
    }
}

int main() {
    vector<Vehicul*> vehicule;
    citireFisier(vehicule, "parcare.txt");
    for (auto v : vehicule) {
        v->afisare();
        delete v;
    }
    return 0;
}
```

Codul C++

```cpp
#include <iostream>
#include <fstream>
#include <vector>
#include <string>
using namespace std;

class Vehicul {
protected:
    string nrInmatriculare;
    int oraIntrare, oraIesire;
public:
    Vehicul(string nr, int intrare, int iesire) : nrInmatriculare(nr), oraIntrare(intrare), oraIesire(iesire) {}
    virtual ~Vehicul() {}
    int calculTimpParcare() { return oraIesire - oraIntrare; }
    virtual double calculTarif() = 0;
    virtual void afisare() {
        cout << "Numar: " << nrInmatriculare << ", Timp: " << calculTimpParcare() << " ore, Tarif: " << calculTarif() << " lei\n";
    }
};

class Masina : public Vehicul {
public:
    Masina(string nr, int intrare, int iesire) : Vehicul(nr, intrare, iesire) {}
    double calculTarif() override { return calculTimpParcare() * 5; }
};
```

```cpp
class Motocicleta : public Vehicul {
public:
    Motocicleta(string nr, int intrare, int iesire) : Vehicul(nr, intrare, iesire) {}
    double calculTarif() override { return calculTimpParcare() * 3; }
};

class Camion : public Vehicul {
public:
    Camion(string nr, int intrare, int iesire) : Vehicul(nr, intrare, iesire) {}
    double calculTarif() override { return calculTimpParcare() * 10; }
};

class Bicicleta : public Vehicul {
public:
    Bicicleta(string nr, int intrare, int iesire) : Vehicul(nr, intrare, iesire) {}
    double calculTarif() override {
        int timp = calculTimpParcare();
        return (timp <= 2) ? 0 : (timp - 2) * 2;
    }
};
```

```cpp
void citireFisier(vector<Vehicul*>& vehicule, const string& numeFisier) {
    ifstream file(numeFisier);
    string tip, nr, oraIntrareStr, oraIesireStr;
    int oraIntrare, oraIesire;
    while (file >> tip >> nr >> oraIntrareStr >> oraIesireStr) {
        oraIntrare = stoi(oraIntrareStr.substr(0, 2));
        oraIesire = stoi(oraIesireStr.substr(0, 2));

        if (tip == "Masina") vehicule.push_back(new Masina(nr, oraIntrare, oraIesire));
        else if (tip == "Motocicleta") vehicule.push_back(new Motocicleta(nr, oraIntrare, oraIesire));
        else if (tip == "Camion") vehicule.push_back(new Camion(nr, oraIntrare, oraIesire));
        else if (tip == "Bicicleta") vehicule.push_back(new Bicicleta(nr, oraIntrare, oraIesire));
    }
}

int main() {
    vector<Vehicul*> vehicule;
    citireFisier(vehicule, "parcare.txt");
    for (auto v : vehicule) {
        v->afisare();
        delete v;
    }
    return 0;
}
```

Fișierul parcare.txt

```
Masina B123XYZ 8:00 12:00.
Motocicleta C456ABC 10:00 13:00.
Camion D789DEF 7:00 11:00.
Bicicleta E101GHI 9:00 12:00.
Bicicleta F202JKL 14:00 18:00.
```

Rezultatul compilării

```
 Numar: B123XYZ, Timp: 4 ore, Tarif: 20 lei
 Numar: C456ABC, Timp: 3 ore, Tarif: 9 lei
 Numar: D789DEF, Timp: 4 ore, Tarif: 40 lei
 Numar: E101GHI, Timp: 3 ore, Tarif: 2 lei
 Numar: F202JKL, Timp: 4 ore, Tarif: 4 lei
```