Inleveropgave 1: Autoverhuur & Gameswinkel in C++.

Hussin Almoustafa

Studentnummer :1776495

Deel 1 : Autoverhuur

Er zijn 3 Classes gemaakt (Klant , Auto , Autoverhuur) waarbij ook functies worder gemaakt volgens de UML digram. Verder heb ik ook headerbestanden voor de classes gemaakt en de HPP bestanden kunnen variabelen, constanten en functies bevatten waarnaar door de broncode in hetzelfde project wordt verwezen. Alle fucties zijn Public gedefinieerd volgens de UML en de reden er achter is om te kunnen die fucties buiten de classes gebruiken.

Auto class :

Autohuur > Auto.cpp > ...

```cpp
#include <iostream>
#include "Auto.hpp"


Auto::Auto(std::string type , double prijs_per_dag){
    this -> type = type;
    this -> prijs_per_dag = prijs_per_dag;

};

void Auto:: set_prijs_per_dag(double prijs_per_dag){
    this -> prijs_per_dag = prijs_per_dag;
};

double Auto:: get_prijs_per_dag() const{
    return prijs_per_dag;
};

ostream& operator<<(ostream& os , const Auto& auto1){
    os <<" Auto :"<< auto1.type << "Prijs per dag is : " << auto1.prijs_per_dag<<endl;
};
```

Autohuur class :

Autohuur > Autohuur.cpp > ...

```cpp
#include <iostream>
#include "Autohuur.hpp"
#include "Auto.hpp"
#include "Klant.hpp"

AutoHuur::AutoHuur(Auto auto1, Klant huurder, int aantal_dagen ){
    this -> auto1 = auto1;
    this -> huurder = Klant;
    this -> aantal_dagen = aantal_dagen;

};

void AutoHuur::set_aantal_dagen(int aantal_dagen){
    this -> aantal_dagen = aantal_dagen;

};

void AutoHuur::set_gehuurde_auto(Auto auto1){
    this -> auto1 = auto1;
};

Auto AutoHuur::get_gehuurde_auto() const{
    return auto1;
};

void AutoHuur::set_huurder(Klant huurder){
    this-> huurder = huurder;
    };
```

Autohuur > Autohuur.cpp > set_huurder(Klant)

```cpp
25
26  void AutoHuur::set_huurder(Klant huurder){
27      this-> huurder = huurder;
28      };
29
30  Klant AutoHuur::get_huurder() const{
31      return huurder;
32
33  };
34
35  double AutoHuur::totaalprijs(){
36
37      double totaal_prijs_dagen = auto1.get_prijs_per_dag() * aantal_dagen;
38      double korting = (totaal_prijs_dagen - huurder.get_korting()) / 100 ;
39      double totaal = totaal_prijs_dagen - korting;
40      return totaal;
41
42  };
43
44  ostream& operator<<(ostream& os , const AutoHuur& autohuur){
45      os <<" Auto : "<< autohuur.get_gehuurde_auto() << " , Naam van de klant   " << autohuur.get_huurder()<< " , totaal prijs :  "<< autohuu
```

Klant class :

Autohuur > Klant.cpp > ...

```cpp
1   #include <iostream>
2   #include "Klant.hpp"
3
4   Klant::Klant(std::string naam){
5       this->naam = naam;
6       };
7
8   void Klant::set_korting_percentage(double korting_percentage){
9           this->korting_percentage = korting_percentage;
10      };
11
12  double Klant::get_korting() const{
13      return korting_percentage;
14  };
15
16  ostream& operator<<(ostream& os , const Klant& klant){
17      os <<" Klant naam :"<< klant.naam << "Met korting van : " << klant.get_korting()<<endl;
18  };
19
```

Deel 2 : Gamewinkel :

Voor opdracht gamewinkel zijn er maar 2 classes gemaakt (Persoon en Game) Een persoon heeft naam , bedget en games als variabelen. games die een persoon heeft is een type vector die is eigelijk sequence containers die arrays vertegenwoordigen die hun grootte kunnen veranderen tijdens runtime in C++ taal.

De regels om iemand game te kopen en verkopen zijn geimplenteerd volgens de opdracht er zijn ook headerbestanden gemaakt voor de classes.

Game class :

gamewinkel > Game.cpp > ...

```cpp

#pragma once
#include "Game.hpp"
#include <string>
#include <vector>
#include <iostream>
#include <ctime>
using std::ostream, std::endl , std::string;
using std::vector; using std::find;


Game::Game(string naam , int releasejaar , double prijs){
    this-> naam = naam;
    this-> releasejaar = releasejaar;
    this -> prijs = prijs;
};

double Game::get_prijs_na_korting() const {
    time_t curr_time = time(NULL);
    tm *tm_local = localtime(&curr_time);
    int currentYear = tm_local->tm_year + 1900;

    double new_prijs = this->prijs;
    for (int i = 0; i < (currentYear - this->releasejaar); i++) {
        new_prijs *= 0.7;
    };

    return new_prijs;
```

```cpp
double Game::get_prijs() const{
    return prijs;
};


int Game::get_releasejaar() const{
    return releasejaar;
};

string Game::get_naam() const{
    return naam;
};


ostream& operator<<(ostream& os , const Game& game){
os << game.get_naam() <<", uitgegeven in "<< get_releasejaar()<< "; nieuwprijs:" << game.get_prijs()<< " nu voor:  "<< game.get_prijs_na_kc
```

Persoon class :

gamewinkel > Persoon.cpp > verkoop(Game, Persoon)

```cpp
#pragma once

#include "Game.hpp"
#include <string>
#include <vector>
#include <iostream>
#include "Persoon.hpp"

using std::ostream, std::endl , std::string;
using std::vector; using std::find;


Persoon::Persoon(std::string naam , double budget){
    this -> naam = naam;
    this-> budget = budget;
};

string Persoon::get_naam() const {
    return naam;
};

void Persoon::set_budget(double budget) {
    budget = budget;
};

double Persoon::get_budget() const {
    return budget;
};

```

```
29
30  vector<Game> Persoon::getGames() const {
31      return games;
32  };
33
34
35  static Persoon::koop(Game game){
36      if(*find(games.begin(), games.end(), game)!= game && Persoon.get_budget() >= game.get_prijs_na_korting()) {
37          games.push_back(game);
38          std::cout << "Gelukt ";}
39      else
40      {
41          std::cout << "Niet gelukt";}
42  };
43
44  static Persoon::verkoop(Game g, Persoon koper){
45
46      if(*find(koper.games.begin(), koper.games.end(), g) != g && koop.budget >= g.get_prijs_na_korting()){
47          koper.koop(g);
48          games.erase(g);
49          std::cout << "Gelukt ";
50      }
51      else{
52          std::cout << "Niet gelukt ";}
53  };
54
55  ostream& operator<<(ostream& os , const Persoon& persoon){
56  os << persoon.get_naam() <<" heeft een budget van  "<< persoon.get_budget()<< "en bezit de volgende games:" <<persoon.getGames()<<endl;
```

Ik kon helaas de code niet runnen vanwegen een error in Visual stuido dus de code is geschreven zonder te kunnen testen.

voor het volledige code :

https://github.com/hassoonsy2/High-Performance-Programming