***Министерство образования Республики Беларусь***

***Учреждение образования***

***«Брестский государственный технический университет»***

***Кафедра ИИТ***

**Лабораторная работа №3**

**По дисциплине ППвИС за III семестр**

**Тема: «Ссылочный тип. Инициализация классов, конструкторы и деструкторы»**

**Выполнил:**

Студент группы ИИ-15 (1)

2-го курса

Волк И. А.

**Проверил:**

Михняев А.Л.

Брест 2018

ЦЕЛЬ РАБОТЫ

1. Изучение использования ссылочного типа в пользовательских классах.

2. Программирование классов с использованием конструкторов (по умолчанию, с параметрами, конструктора копирования) и деструкторов.

СПИСОК ЗАДАЧ

Лекционные примеры по использованию ссылочного типа.

Индивидуальные задания по разработке классов.

*Выбранная предметная область:* игровая компания.

*Описание тестового примера.*

В качестве тестовых входных данных будем использовать следующие:

CompanyName::name = “valve”;

CompanyName::equity = 2500000000;

Game::name = “Dota 2”;

Game::yearOfRelease = 2013;

Game::price = 0;

Game::name = “CS:GO”;

Game::yearOfRelease = 2012;

Game::price = 8.49;

*Описание классов:*

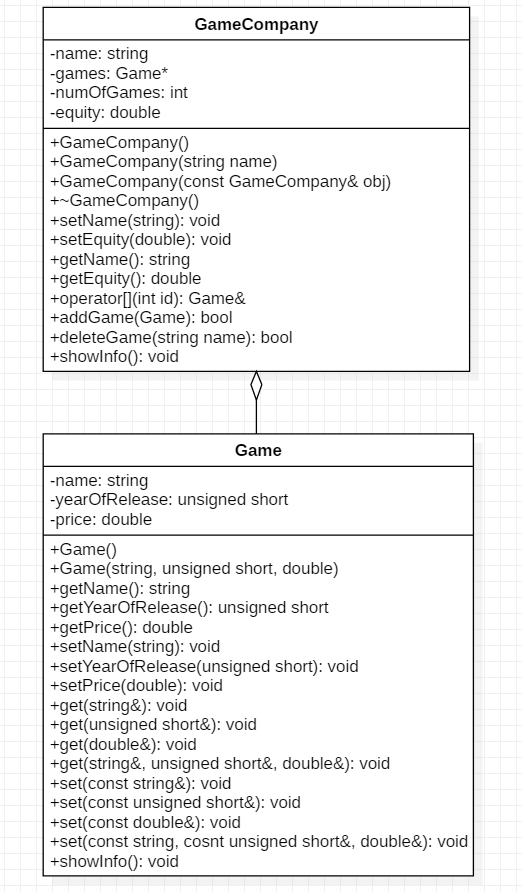
1. Имя: Game

Назначение: хранит информацию об игре, а именно: имя, дату выпуска и текущую цену.

2. Имя: GameCompany

Назначение: хранит информацию о компании по разработке игр, а именно: имя, капитал и игры, разработанные этой компанией.

UML:



*Листинг кода программы:*

// game.h

#pragma once

#include <string>

class Game

{

std::string name;

unsigned short yearOfRelease;

double price;

public:

Game();

Game(std::string name, unsigned short yearOfRelease, double price);

std::string getName();

unsigned short getYearOfRelease();

double getPrice();

void setName(std::string newName);

void setYearOfRelease(unsigned short newYearOfRelease);

void setPrice(double newPrice);

void get(std::string& str);

void get(unsigned short& u\_sh);

void get(double& d);

void get(std::string& str, unsigned short& u\_sh, double& d);

void set(const std::string& str);

void set(const unsigned short& u\_sh);

void set(const double& d);

void set(const std::string& str, const unsigned short& u\_sh, const double& d);

void showInfo();

};

//game.cpp

#include "game.h"

#include <iostream>

Game::Game()

{

this->name = "without\_name";

this->yearOfRelease = 0;

this->price = 0;

}

Game::Game(std::string name, unsigned short yearOfRelease, double price)

{

this->name = name;

this->yearOfRelease = yearOfRelease;

this->price = price;

}

std::string Game::getName()

{

return this->name;

}

unsigned short Game::getYearOfRelease()

{

return this->yearOfRelease;

}

double Game::getPrice()

{

return this->price;

}

void Game::setName(std::string newName)

{

this->name = newName;

}

void Game::setYearOfRelease(unsigned short newYearOfRelease)

{

this->yearOfRelease = newYearOfRelease;

}

void Game::setPrice(double newPrice)

{

this->price = newPrice;

}

void Game::get(std::string& str)

{

str = this->name;

}

void Game::get(unsigned short& u\_sh)

{

u\_sh = this->yearOfRelease;

}

void Game::get(double& d)

{

d = this->price;

}

void Game::get(std::string& str, unsigned short& u\_sh, double& d)

{

str = this->name;

u\_sh = this->yearOfRelease;

d = this->price;

}

void Game::set(const std::string& str)

{

this->name = str;

}

void Game::set(const unsigned short& u\_sh)

{

this->yearOfRelease = u\_sh;

}

void Game::set(const double& d)

{

this->price = d;

}

void Game::set(const std::string& str, const unsigned short& u\_sh, const double& d)

{

this->name = str;

this->yearOfRelease = u\_sh;

this->price = d;

}

void Game::showInfo()

{

std::cout << "Name: " << this->name.c\_str() << std::endl;

std::cout << "Year of release: " << this->yearOfRelease << std::endl;

std::cout << "Price: " << this->price << std::endl;

}

//gamecompany.h

#pragma once

#include <string>

#include "game.h"

class GameCompany

{

std::string name;

Game \* games;

int numOfGames;

double equity;

public:

GameCompany();

GameCompany(std::string name);

GameCompany(const GameCompany& obj);

~GameCompany();

void setName(std::string);

void setEquity(double assets);

std::string getName();

double getEquity();

Game& operator[](int id);

bool addGame(Game game);

bool deleteGame(std::string name);

void showInfo();

};

//gamecompany.cpp

#include "gamecompany.h"

#include <iostream>

GameCompany::GameCompany()

{

this->name = "without\_name";

this->games = NULL;

this->numOfGames = 0;

this->equity = 0;

}

GameCompany::GameCompany(std::string name)

{

this->name = name;

this->numOfGames = 0;

this->games = NULL;

this->equity = 0;

}

GameCompany::GameCompany(const GameCompany& obj)

{

this->name = obj.name;

this->numOfGames = obj.numOfGames;

this->games = new Game[this->numOfGames];

for (int i = 0; i < this->numOfGames; i++)

{

this->games[i] = obj.games[i];

}

this->equity = obj.equity;

}

GameCompany::~GameCompany()

{

std::cout << "GameCompany::~GameCompany()" << std::endl;

delete[] this->games;

}

void GameCompany::setName(std::string name)

{

this->name = name;

}

void GameCompany::setEquity(double assets)

{

if (assets < 0)

{

this->equity = 0;

return;

}

this->equity = assets;

}

std::string GameCompany::getName()

{

return this->name;

}

double GameCompany::getEquity()

{

return this->equity;

}

Game& GameCompany::operator[](int id)

{

try

{

if (id < 0)

throw 1;

if (id >= this->numOfGames)

throw 2;

return this->games[id];

}

catch (int e)

{

std::cout << "GameCompany::operator[](int)\t\tError - " << e << std::endl;

std::cin.get();

exit(0);

}

}

bool GameCompany::addGame(Game game)

{

Game \*games\_copy = new Game[this->numOfGames];

for (int i = 0; i < this->numOfGames; i++)

games\_copy[i] = this->games[i];

delete[] games;

this->numOfGames++;

games = new Game[this->numOfGames];

for (int i = 0; i < (this->numOfGames - 1); i++)

games[i] = games\_copy[i];

games[this->numOfGames - 1] = game;

return true;

}

bool GameCompany::deleteGame(std::string name)

{

int id = -1;

for (int i = 0; i < this->numOfGames; i++)

if (this->games[i].getName() == name)

{

id = i;

break;

}

if (id == -1)

return false;

Game \*games\_copy = new Game[this->numOfGames];

for (int i = 0; i < this->numOfGames; i++)

games\_copy[i] = this->games[i];

delete[] games;

this->numOfGames--;

games = new Game[this->numOfGames];

for (int i = 0; i < id; i++)

games[i] = games\_copy[i];

for (int i = id + 1; i < (this->numOfGames + 1); i++)

games[i - 1] = games\_copy[i];

return true;

}

void GameCompany::showInfo()

{

std::cout << "Name of company: " << this->name << std::endl;

std::cout << "Equity: " << this->equity << std::endl;

std::cout << "\tGames: " << std::endl;

for (int i = 0; i < this->numOfGames; i++)

std::cout << i + 1 << ". " << this->games[i].getName() << std::endl;

}

// main.cpp

#include <iostream>

#include "gamecompany.h"

int main()

{

{

GameCompany valve("valve");

valve.setEquity(2500000000);

Game dota2("Dota 2", 2013, 0);

valve.addGame(dota2);

std::string name;

unsigned short yearOfRelease;

double price;

valve[0].get(name);

valve[0].get(yearOfRelease);

valve[0].get(price);

std::cout << name.c\_str() << std::endl;

std::cout << yearOfRelease << std::endl;

std::cout << price << std::endl;

valve.showInfo();

valve.deleteGame("Dota 2");

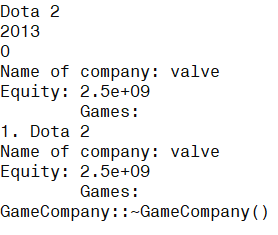
valve.showInfo();

}

std::cin.get();

}

Листинг результата:



Вывод: изучил использование ссылочного типа в пользовательских классах, конструкторов и деструктора.