МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

“БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”

КАФЕДРА ИИТ

ОТЧЁТ

по лабораторной работе №1

**«КЛАССЫ И ОБЪЕКТЫ В С++»**

Выполнил:

студент 2 курса

группы ПО-9

Мисиюк Алексей Сергеевич

Проверил:

Козик И. Д.

labrab1.cpp

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

#include "Ship.h"

/\*

\* Ход работы:

\* 1. Определить пользовательский класс в соответствии с вариантом задания (смотри приложение).

\* 2. Определить в классе следующие конструкторы: без параметров, с параметрами, копирования.

\* 3. Определить в классе деструктор.

\* 4. Определить в классе компоненты-функции для просмотра и установки полей данных.

\* 5. Определить указатель на компоненту-функцию.

\* 6. Определить указатель на экземпляр класса.

\* 7. Написать демонстрационную программу, в которой создаются и разрушаются объекты

\* пользовательского класса и каждый вызов конструктора и деструктора сопровождается

\* выдачей соответствующего сообщения (какой объект какой конструктор или деструктор вызвал).

\* 8. Показать в программе использование указателя на объект и указателя на компоненту-функцию.

\*/

int main()

{

Ship\* armidale\_class = new Ship();

char\* new\_name = new char[16];

strcpy(new\_name, "Patrol boat #1\0");

armidale\_class->setName(new\_name);

std::cout << armidale\_class->getName() << std::endl

<< armidale\_class->getType() << std::endl

<< armidale\_class->getDisplacement() << std::endl;

delete armidale\_class;

return 0;

}

Ship.cpp

#include "Ship.h"

std::string Ship::className() {

return "Ship";

}

// Конструкторы

Ship::Ship() {

this->name = new char[10];

strcpy(this->name, "New ship\0");

this->type = new char[10];

strcpy(this->type, "Armidale\0"); // Armidale class

this->displacement = 300;

std::cout << "Вызван конструктор по умолчанию класса " << this->className() << std::endl;

}

Ship::Ship(char\* name, char\* type, int displacement) {

this->name = new char[strlen(name)];

strcpy(this->name, name);

this->type = new char[strlen(type)];

strcpy(this->type, type);

this->displacement = displacement;

std::cout << "Вызван конструктор с параметрами класса " << this->className() << std::endl;

}

Ship::Ship(const Ship& original) {

this->name = new char[strlen(original.name)];

strcpy(this->name, original.name);

this->type = new char[strlen(original.type)];

strcpy(this->type, original.type);

this->displacement = original.displacement;

std::cout << "Вызван конструктор копирования класса " << this->className() << std::endl;

}

// Деструктор

Ship::~Ship() {

delete[] this->name;

delete[] this->type;

std::cout << "Вызван деструктор класса " << this->className() << std::endl;

}

// Сеттеры

void Ship::setName(char\* name) {

this->name = new char[strlen(name)];

strcpy(this->name, name);

}

void Ship::setType(char\* type) {

this->type = new char[strlen(type)];

strcpy(this->type, type);

}

void Ship::setDisplacement(int displacement) {

this->displacement = displacement;

}

// Геттеры

char\* Ship::getName() {

char\* tmp = new char[strlen(this->name)];

strcpy(tmp, this->name);

return tmp;

}

char\* Ship::getType() {

char\* tmp = new char[strlen(this->type)];

strcpy(tmp, this->type);

return tmp;

}

int Ship::getDisplacement() {

return this->displacement;

}

Ship.h

#pragma once

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

#include "string.h"

// Вариант #15

// КОРАБЛЬ

// имя – char\*

// водоизмещение – int

// тип – char\*

class Ship {

protected:

char\* name;

int displacement;

char\* type;

public:

// Имя класса

std::string className();

// Конструкторы

Ship(); // По умолчанию

Ship(char\* name, char\* type, int displacement = 0); // С параметрами

Ship(const Ship& original); // Конструктор копирования

// Деструктор

~Ship();

// Сеттеры

void setName(char\* name);

void setType(char\* type);

void setDisplacement(int displacement);

// Геттеры

char\* getName();

char\* getType();

int getDisplacement();

};

Результат: