МИНОБРНАУКИ РОССИИ

федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

«Череповецкий государственный университет»

**Лабораторная работа № 2**

**«СОЗДАНИЕ И УНИЧТОЖЕНИЕ ОБЪЕКТОВ КЛАССА»**

**Выполнил:**

студент гр. 1ИВТпб-01-21оп

Климов А.Г.  
**Проверил:** преподаватель

Пышницкий К.М.  
Отметка о зачете:

Череповец

2017 год

**Цель работы:** изучить синтаксис и семантику определения и вызова конструкторов и деструкторов; приобрести практические навыки создания и уничтожения объектов; изучить особенности применения различных видов конструкторов.

**Вариант**

A 16. Осветительные приборы.

**Задания**

Модифицируйте абстрактный тип данных, реализованный по заданию раздела IV, согласно варианту задания (см. раздел III), заменив соответствующие компонентные функции на систему конструкторов и деструктор.

Проверьте работоспособность АТД на тестовом наборе данных.

**Ход работы**

**Текст программы:**

**ClassLight.h**

class light

{

public:// набор функций

light(); //конструктор по умолчанию

light(char \*nm1, char nm2, int i, float j); //конструктор с параметрами

light(const light &t); //конструктор с const, меняющий содержимое, к которому получает доступ (копирующий конструктор)

~light() { delete[]name1; } //деструктор

//

void input(char \*nm1, char nm2, int i, float j);

void print();

void prisv(light b);

bool sravn(light b);

void destr() { delete[]name1; } // уничтожение объектов

private:

char \*name1; // название осветительного прибора

char name2; // тип лампочки

int k; // размер цоколя

float d; // яркость света

};

**ClassLight.cpp**

#include "ClassLight.h"

#include <iostream>

#include <string>

void light::input(char \*nm1, char nm2, int i, float j)

{

name1 = new char[strlen(nm1) + 1];

strcpy(name1, nm1);

name2 = nm2;

k = i;

d = j;

}

void light::print()

{

std::cout<<"Название: " << name1 << std::endl;

std::cout<<"Цоколь: " << name2<<k << std::endl;

std::cout<<"Яркость света: " << d << std::endl;

std::cout << std::endl;

}

void light::prisv(light b)

{

delete[]name1;

name1 = new char[strlen(b.name1) + 1];

strcpy(name1, b.name1);

name2 = b.name2;

k = b.k;

d = b.d;

}

bool light::sravn(light b)

{

return ((strcmp(name1,b.name1)==0)&&(name2==b.name2)&&(k==b.k)&&(d==b.d))?true:false;

}

//конструкторы

light::light() {

name1 = new char[strlen("Unknown") + 1];

strcpy(name1, "Unknown");

name2 = 'E';

k = 0;

d = 0;

}

light::light(char \*nm1, char nm2, int i, float j) {

name1 = new char[strlen(nm1) + 1];

strcpy(name1, nm1);

name2 = nm2;

k = i;

d = j;

}

light::light(const light &t) {

name1 = new char[strlen(t.name1) + 1];

strcpy(name1, t.name1);

name2 = t.name2;

k = t.k;

d = t.d;

}

**Main.cpp**

#include <iostream>

#include "ClassLight.h"

using namespace std;

void main()

{

setlocale(LC\_ALL, "Russian");

//

//лр 1

light x, y, z;

x.input("Светильник1", 'E', 40, 70.5);

y.input("Светильник2", 'E', 27, 90.5);

z.input("Светильник3", 'E', 14, 50.5);

x.print();

y.print();

z.print();

cout << endl;

y.prisv(z);

x.print();

y.print();

z.print();

//y.destr();

y.input("Светильник35", 'E', 14, 50.5);

cout << y.sravn(z) << endl;

/\*x.destr(); // освобождение памяти

y.destr();

z.destr();\*/

//

//лр 2

light t1, t2("Светильник4", 'E', 40, 22.3),t3(t2);

t1.print();

t2.print();

t3.print();

system("pause");

}

**Результаты тестирования:**

*Входные данные (функции: конструктор по умолчанию, конструктор с параметрами, копирующий конструктор, деструктор):*

light t1, t2("Светильник4", 'E', 40, 22.3), t3(t2);

*Результат Рис.1:*

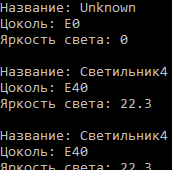


Рис. 1. Результат 1

**Ответы на контрольные вопросы:**

1. Каковы особенности синтаксиса и семантики конструкторов и деструкторов?

Конструктор (constructor) – это компонентная функция, предназначенная для создания объектов типа своего класса. Под созданием понимается инициализация компонентных данных и при необходимости распределения под них свободной памяти с помощью new. Имя конструктора совпадает с именем класса.

Деструктор (destructor) – это компонентная функция, предназначенная для уничтожения объектов типа своего класса. Под уничтожением понимается освобождение свободной памяти с помощью delete. Имя деструктора – это предваряемое символом «~» Имя класса.

2. В чем заключаются особенности применения конструкторов по умолчанию?

Конструктор, который может быть вызван без аргументов.

3. В каких случаях вызывается копирующий конструктор?

Копирующий конструктор вызывается в следующих случаях:

- если объект типа type инициализируется значением типа type;

- когда параметр типа type передается функции по значению;

- когда функция возвращает значение типа type.

При отсутствии копирующего конструктора эти действия по умолчанию выполняются поэлементно.

4. При каких условиях конструктор может использоваться для преобразования типа?

Конструктор с одним параметром автоматически является функцией преобразования типа параметра к типу класса. Использование explicit в начале объявления запрещает такое преобразование.

5. В каких ситуациях необходимы инициализаторы конструктора, синтаксис и семантика их использования?

Класс, содержащий элементы, тип которых требует конструктор, использует инициализаторы – разделенный запятыми список вызовов конструкторов, стоящий после двоеточия. Конструктор вызывается с помощью имени элемента, за которым в круглых скобках следует список аргументов. Элементы инициализируются в порядке объявления их в классе.

Класс, содержащий элементы, тип которых требует конструктор, использует инициализаторы – разделенный запятыми список вызовов конструкторов, стоящий после двоеточия. Конструктор вызывается с помощью имени элемента, за которым в круглых скобках следует список аргументов. Элементы инициализируются в порядке объявления их в классе.

Н а п р и м е р :

type :: type (int t): i(7), j(t) //инициализаторы

{// остальное тело конструктора}

Инициализаторы необходимы всегда для нестатических элементов констант или ссылок.

Конструкторы и деструкторы не наследуются. В отличие от деструкторов конструкторы не могут быть виртуальными.

**Вывод:** изучил синтаксис и семантику определения и вызова конструкторов и деструкторов; приобрел практические навыки создания и уничтожения объектов; изучил особенности применения различных видов конструкторов. Для этого был разработан класс “Осветительные приборы”. Для созданного класса было проведено тестирование.