МИНОБРНАУКИ РОССИИ

федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

«Череповецкий государственный университет»

**Лабораторная работа № 1**

**«РАЗРАБОТКА АБСТРАКТНЫХ ТИПОВ ДАННЫХ»**

**Выполнил:**

студент гр. 1ИВТпб-01-21оп

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

Пышницкий К.М.  
Отметка о зачете:

Череповец

2017 год

**Цель работы:** изучить способы определения классов, правила доступа к элементам; приобрести практические навыки работы с объектами класса; изучить принципы и механизмы создания абстрактных типов данных.

**Вариант**

A 16. Осветительные приборы.

**Задания**

Разработайте алгоритм и программу, реализующую абстрактный тип данных (АТД) – класс, согласно варианту задания (см. раздел III). Предусмотрите закрытую реализацию и открытый интерфейс. Интерфейс должен содержать псевдоконструкторы и псевдодеструктор, функции присваивания, вывода содержимого и обработки (сортировка, поиск, сравнение, арифметические действия и т.д.).

Проверьте работоспособность АТД на тестовом наборе данных.

**Ход работы**

**Текст программы:**

**ClassLight.h**

class light

{

public:// набор функций

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;

}

**Main.cpp**

#include <iostream>

#include "ClassLight.h"

using namespace std;

void main()

{

setlocale(LC\_ALL, "Russian");

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();

system("pause");

}

**Результаты тестирования:**

*Входные данные 1(функции: input,print):*

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

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

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

*Результат 1 Рис.1:*

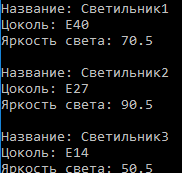


Рис. 1. Результат 1

*Входные данные 2 (функции: destr,input,prisv, print):*

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

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

y.prisv(z);

*Результат 2 Рис.2:*

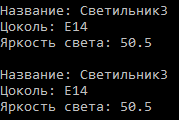


Рис. 2. Результат 2

*Входные данные 3 (функции: input, sravn, print):*

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

y.sravn(z);

*Результат 3 Рис.3:*



Рис. 3. Результат 3

**Ответы на контрольные вопросы:**

1. Дайте определение понятия «класс». Сформулируйте правила доступа к его элементам.

Класс – это гетерогенный агрегатный тип, имеющий элементы, данные и элементы функций, обеспечивающий сокрытие данных и наследование.

Структура класса имеет вид:

class имя\_типа {public: // элементы доступны и наследуются

…

protected: //элементы защищены и наследуются

…

private: //элементы закрыты, не наследуются

…

};

Спецификаторы доступа public, protected и private определяют степень открытости элементов, которые следуют за ними.

По умолчанию элементы структуры являются открытыми, элементы класса – закрытыми.

2. С какой целью в классе объединены компонентные данные и компонентные функции?

Функции, объявленные или определенные внутри класса, являются компонентными. Они имеют доступ к закрытым, защищенным и открытым элементам своего класса, могут быть перегруженными.

Компонентные функции вызываются с помощью оператора . или → , а также при объявлении объектов класса (конструкторы), при уничтожении объектов (деструкторы), при перегрузке операторов.

3. Каким образом осуществляется доступ к открытым и закрытым элементам?

При помощи компонентных функций.

4. Опишите назначение дружественных функций, назовите их разновидности.

Дружественная функция должна быть объявлена внутри объявления класса, по отношению к которому она является дружественной, с использованием спецификатора friend. Такая функция имеет доступ к закрытым элементам класса наряду с компонентными функциями. Дружественными могут быть как внешние, так и компонентные функции, в этом случае для указания имени функции в дружественном классе используется оператор разрешения области видимости :: .

Н а п р и м е р :

class A {-----

friend int B:: f( ); // f() - компонентная функция класса В, дружественная классу А

};

class C {----

friend class B; // все компонентные функции класса В являются дружественными классу С

};

class D {----

friend D f(D); // f() - внешняя функция, дружественная классу D

};

5. Что понимается под указателем this?

Ключевое слово this неявно объявляет указатель на себя. Может иметь место только в нестатических компонентных функциях. Заменяет объявление закрытого элемента:

class\_type \* const this.

6. Каковы особенности использования статических компонентных данных?

Статические элементы данных хранятся в одном месте, являются общими для всех объектов класса и существуют независимо от них.

7. В чем заключается синтаксис и семантика компонентных функций static и const?

Статическая компонентная функция доступа к компонентным данным не имеет, за исключением статических.

Обращение через оператор разрешения области видимости к статическим компонентным данным и функциям является предпочтительным:

class\_type :: статический\_элемент

class\_type :: статическая\_функция ( )

Именно так отражается тот факт, что элемент и функция являются статическими.

Компонентная функция, объявленная как const, не может изменять свои неявные аргументы, она получает неявный указатель:

const class\_type \* const this.

8. Каким образом могут изменяться компонентные данные объектов, объявленных константами?

Объекты класса могут быть объявлены как константы, и это гарантирует постоянство компонентных данных. Отказаться можно с помощью const\_cast или индивидуально для отдельных компонентных данных с использованием ключевого слова mutable.

9. Каковы особенности создания вложенных классов?

Классы могут быть созданы внутри блоков, в том числе внутри других классов, как часть реализации более сложной конструкции. Использование таких классов невозможно за пределами соответствующих блоков.

Вложение классов позволяет вкладывать определения функций - компонентная функция должна быть определена внутри своего локального класса, и на данную функцию нельзя ссылаться вне области видимости ее локального класса.

Пример:

class type {

public:

void f1 (int b) {x = b; y = 0;} // компонентная функция для доступа к x и y

void f2 (double k) {y = k\*x;}

static void f3 (int c) {z = c;} //статическая компонентная функция для доступа только к z

int f4 ( ) const {return x+y+z;} // f4 не может изменять x, y, z

friend int f5 (const type & t); // дружественная классу type функция, аналог f4

private :

int x; //элемент данных индивидуальный для

int y; // каждого объекта

static int z; //статический элемент данных, общий для всех объектов

};

int type :: z = 10; // должен быть определен и инициализирован вне класса

main()

{ type w1,w2; //объявление объектов

w1.f1(100); // инициализация w1: x = 100, y = 0, z = 10

w2.f1(200); // инициализация w2: x = 200, y = 0, z = 10

w1.f2(0.5); // вычисление y в w1

w2.f2(0.8); // вычисление y в w2

type :: f3(300); // вызов f3 для изменения z в w1 и w2

cout <<”w1 = ”<<w1.f4( ) // сумма x, y, z в w1

<<”w2 = ”<<f5(w2)<<endl; // сумма x, y, z в w2

}

**Вывод:** изучил способы определения классов, правила доступа к элементам; приобрёл практические навыки работы с объектами класса; изучил принципы и механизмы создания абстрактных типов данных. Для этого был разработан класс “Осветительные приборы”. Для созданного класса было проведено тестирование.