Федеральное государственное автономное образовательное учреждение

высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий

**ЛАБОРАТОРНАЯ РАБОТА 1**

По дисциплине “Технологии и методы программирования”

наименование дисциплины

Преподаватель Матковский И.В.

подпись, дата инициалы, фамилия

Студент КИ16-02 031619452 Семенихин В.В.

номер группы, зачетной книжки подпись, дата инициалы,фамилия

Красноярск 2019

Вариант лабораторной работы 4 – 5 – 1 – 3

Основное задание для составления программы:

|  |  |  |
| --- | --- | --- |
| Обобщенный артефакт, используемый в задании | Базовые альтернативы (уникальные параметры, задающие отличительные признаки альтернатив) | Параметры, общие для всех альтернатив |
| 4. Транспорт | 1. Самолеты (дальность полета – целое, грузоподъемность – целое) 2. Поезда (количество вагонов – целое) | Скорость – целое; Расстояние между пунктами отправления и назначения – целое |

Структутрный метод:

struct plane {

type key;

int range;

int c;

int spead;

int distance;

};

struct traine {

type key;

int count;

int spead;

int distance;

};

struct transport {

type key;

};

ООП метод:

class transport {

public:

static transport\* In(ifstream &ifst);

virtual void InData(ifstream &ifst) = 0; // ввод

virtual void Out(ofstream &ofst) = 0; // вывод

};

class plane : public transport {

int range;

int c;

int spead;

int distance;

public:

void InData(ifstream &ifst); // ввод

void Out(ofstream &ofst); // вывод

plane() {} // создание без инициализации.

};

class traine : public transport {

int count;

int spead;

int distance;

public:

void InData(ifstream &ifst); // ввод

void Out(ofstream &ofst); // вывод

traine() {} // создание без инициализации.

};

Способ организации контейнера

5. Контейнер на основе двунаправленного кольцевого списка.

Структурный метод:

struct container {

Node\* Top;

int count;

};

struct Node {

Node() {

this->Next = nullptr;

this->Prev = nullptr;

this->data = nullptr;

};

Node\* Next;

Node\* Prev;

transport\* data;

};

ООП метод:

class container

{

private:

class Node {

public:

Node();

Node\* Next;

Node\* Prev;

transport\* data;

};

Node\* Top;

int count;

public:

void In(ifstream &ifst); // ввод

void Out(ofstream &ofst); // вывод

void Clear(); // очистка контейнера от фигур

container(); // инициализация контейнера

~container() { Clear(); }

};

Организация модульной структуры программы

Клиентская часть программы (тестовая функция) размещается в отдельном модуле(**ConsoleApplication1.cpp** в структурной или **Program.cpp** в ООП). Основная часть кода, определяющего реализацию функций или методов, содержится в другом (едином) модуле(**pch.cpp**). Программные объекты, обеспечивающие связь между основной и клиентской частями (абстрактные типы данных, сигнатуры функций, описания классов), размещается в отдельном интерфейсном модуле (заголовочном файле)(**pch.h**).



В клиентской части программы (**ConsoleApplication1.cpp** в структурной или **Program.cpp** в ООП)

#include "pch.h"

Только в структурной программе:

namespace simple\_shapes { Объявление функций }

using namespace simple\_shapes;

В основной части кода (**pch.cpp):**

#include "pch.h"

Только в структурной программе:

namespace simple\_shapes { Объявление функций и их определение }

В заголовочном файле(**pch.h)**:

Прописаны все необходимые программные объекты

Организация обобщений в процедурной программе

3. Обобщение, использующее общую основу (базовую структуру), которая повторяется в каждой из отдельно прописанных специализаций.

struct plane {

type key;

int range;

int c;

int spead;

int distance;

};

struct traine {

type key;

int count;

int spead;

int distance;

};

struct transport {

type key;

};

transport \* In(ifstream &ifst)

{

transport \*s = new transport;

int key;

ifst >> key;

if (key == 1) {

plane\* p = new plane;

s = (transport\*)InPlane(\*p, ifst);

s->key = PLANE;

return s;

}

else if (key == 2)

{

traine\* t = new traine;

s = (transport\*)InTraine(\*t, ifst);

s->key = TRAINE;

return s;

}

else

return 0;

}

void Out(transport \*s, ofstream &ofst)

{

switch (s->key) {

case PLANE:

OutPlane((plane\*)s, ofst);

break;

case TRAINE:

OutTraine((traine\*)s, ofst);

break;

default:

cout << "Incorrect figure!" << endl;

}

}

Ссылки на репозитории на Github (1. Структурный, 2. ООП):

1.<https://github.com/dostoevsky-plus/-.git>

2. <https://github.com/dostoevsky-plus/proc-method.git>