Министерство образования и науки Российской Федерации  
Федеральное‌ ‌государственное‌ ‌бюджетное‌ ‌образовательное‌ ‌учреждение‌

высшего‌ ‌образования‌

**«Пермский национальный исследовательский  
политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**О Т Ч Ё Т**

**по лабораторной работе №18.5**

Дисциплина: «Основы теории алгоритмов и структуры данных»  
Тема: Наследование. Виртуальные функции. Полиморфизм.

Вариант 14

Выполнил:

студент группы РИС-20-2б

Вичугов Алексей Дмитриевич

Проверила:

доцент кафедры ИТАС

Полякова Ольга Андреевна

Пермь, 2021

**Цель работы**

1. Изучить построение функции хеширования и алгоритмов хеширования данных.

2. Создание иерархии классов с использованием простого наследования и абстрактного класса.

3. Изучение полиморфизма и виртуальных методов.

**Постановка задачи**

1. Определить абстрактный класс.

2. Определить иерархию классов, в основе которой будет находиться абстрактный класс.

3. Определить класс Вектор, элементами которого будут указатели на объекты иерархии классов.

4. Перегрузить для класса Вектор операцию вывода объектов с помощью потоков.

5. В основной функции продемонстрировать перегруженные операции и полиморфизм Вектора.

6. Задание варианта: Базовый класс: человек. Имя - string. Возраст - int. Создать производный класс Teacher, имеющий поля subject - string и hours - int. Определить методы изменения полей, а также увеличения и уменьшения часов.

**Анализ задачи**

1. Описание класса:

#include <iostream>

#include <string>

using namespace std;

class Person{

public:

string name;

int age;

virtual void Init(istream& stream)=0;

virtual void Show(ostream& stream)=0;

};

class Teacher:public Person{

public:

string subject;

int hours;

Teacher();

Teacher(string n, string s, int a, int i);

Teacher(Teacher &t);

void Init(istream& stream);

void Show(ostream& stream);

};

class Sportsman:public Person{

public:

string sport;

int massa;

Sportsman();

Sportsman(string n, string s, int a, int i);

Sportsman(Sportsman &s);

void Init(istream& stream);

void Show(ostream& stream);

};

class Vector{

public:

Person \*obj;

Vector();

Vector(Person &p);

friend istream& operator >>(istream& ist, Vector &v);

friend ostream& operator <<(ostream &ost, Vector &v);

};

2. Определение компонентных функций:

#include "LabaClass5.h"

Teacher::Teacher(){

name="No name";

age=0;

subject="No subject";

hours=0;

}

Teacher::Teacher(string n, string s, int a, int i){

name=n;

age=a;

subject=s;

hours=i;

}

Teacher::Teacher(Teacher &t){

name=t.name;

age=t.age;

subject=t.subject;

hours=t.hours;

}

void Teacher::Init(istream &stream){

cout << "Input name: ";

stream >> name;

cout << "Input age: ";

stream >> age;

cout << "Input subject: ";

stream >> subject;

cout << "Input hours: ";

stream >> hours;

}

void Teacher::Show(ostream &stream){

stream << "Teacher: " << endl << "Name: " << name << endl << "Age: " << age << endl << "Subject: " << subject << endl << "Hours: " << hours << endl;

}

Sportsman::Sportsman(){

name="No name";

age=0;

sport="No sport";

massa=0;

}

Sportsman::Sportsman(string n, string s, int a, int i){

name=n;

age=a;

sport=s;

massa=i;

}

Sportsman::Sportsman(Sportsman &s){

name=s.name;

age=s.age;

sport=s.sport;

massa=s.massa;

}

void Sportsman::Init(istream &stream){

cout << "Input name: ";

stream >> name;

cout << "Input age: ";

stream >> age;

cout << "Input sport: ";

stream >> sport;

cout << "Input weight of sportsman: ";

stream >> massa;

}

void Sportsman::Show(ostream &stream){

stream << "Sportsman: " << endl << "Name: " << name << endl << "Age: " << age << endl << "Sport: " << sport << endl << "Massa: " << massa << endl;

}

Vector::Vector(){

obj=NULL;

}

Vector::Vector(Person &p){

obj=&p;

}

3. Определение глобальных функций:

istream& operator >>(istream &ist, Person \*obj){

obj->Init(ist);

}

ostream& operator <<(ostream &ost, Person \*obj){

obj->Show(ost);

}

istream& operator >>(istream &ist, Vector &v){

v.obj->Init(ist);

}

ostream& operator <<(ostream &ost, Vector &v){

v.obj->Show(ost);

}

4. Функция main():

main(){

Teacher fteacher;

cout << &fteacher;

cin>>&fteacher;

Vector v(fteacher);

cout << v;

Sportsman fs;

v.obj=&fs;

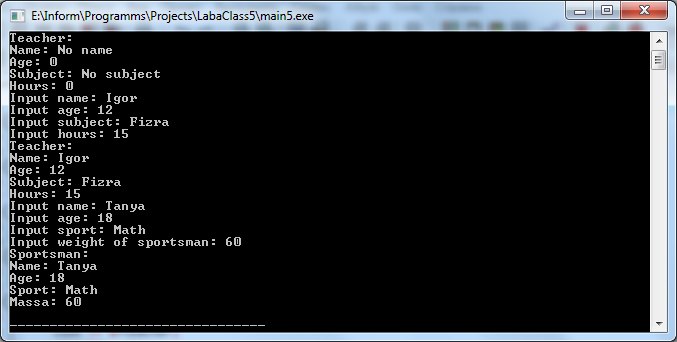
cin >> v;

cout << v;

return 0;

}

**Результаты работы программы**

****

**Ответы на вопросы**

1. Метод, который содержит признак "=0" вместо тела. Чисто виртуальный метод должен обязательно переопределяться в производных классах.

2. Имеющий чисто виртуальный метод.

3. Для представления общих понятий, которые предполагается конкретизировать в производных классах.

4. Функции, работающие с объектами разных классов в пределах иерархии.

5. Полиморфизм требует наличия виртуальной функции, в отличие от принципа подстановки.

6. Абстрактный класс – животное, виртуальная функция – издавать звук. Производные классы – кошка (мяукать), мышь (пищать), собака (лаять).

7. class Animal {public: virtual void Voice {cout << “Speak:”;}};  
class Dog : public Animal {public: void Voice {cout << “Gav”;}};

8. При необходимости по разному реализовать одну и ту же функцию в разных классах.