Министерство науки и высшего образования Российской Федерации  
Федеральное государственное автономное учреждение высшего образования  
«Пермский национальный исследовательский политехнический университет»

Электротехнический факультет  
Кафедра «Информационные технологии и автоматизированные системы»

**О Т Ч Ё Т  
по лабораторной работе**

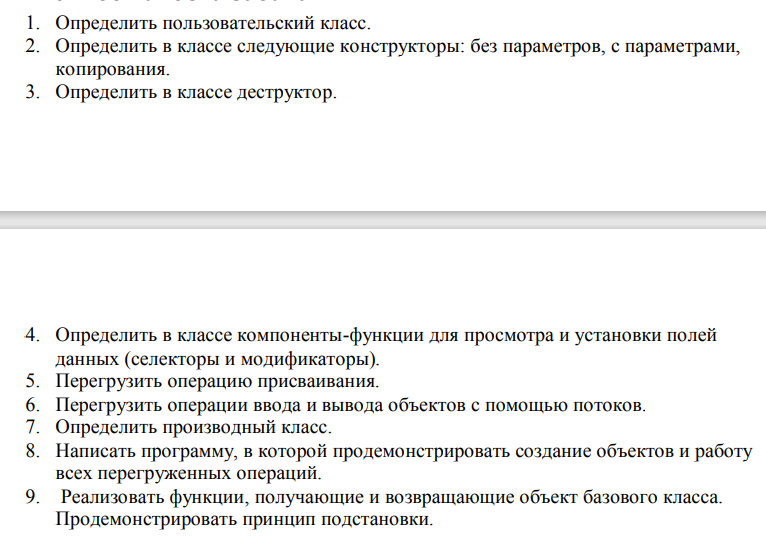
Дисциплина: «Основы алгоритмизации и программирования»

Выполнил  
студент группы ИВТ-21-1б  
Мифтахов И.Г

Проверил  
ст. п. кафедры ИТАС  
Яруллин Д.В.

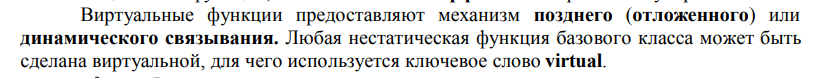
Пермь 2022

Постановка задачи:

 Анализ задачи:

Пользовательский класс – triad, три числа типа int, которые при наследовании в класс date станут днями, месяцами, годами. Все находится в public секции. Использовано 3 стандартных конструктора, а также инициализация объектов при помощи сеттеров. Реализована перегрузка оператора++, для префиксной и постфиксной формы. Реализован метод вывода на экран данных(show()). Также для потомка(date) реализован метод добавления n количества дней к дате в public секции. Если дней больше 31, начинаем считать количество месяцев, которые нужно будет прибавить, аналогичная операция для месяцев, в конце считаем количество лет и добавляем к итоговой дате. Инициируем вывод ф-ией show().

Ответы на вопросы:

1. Для передачи полей потомкам, чтоб лишний раз не объявлять их в потомственном классе.
2. Полный доступ к методам и полям в public секции.
3. Private не наследуются, доступ возможен только в классе-предке.
4. Protected наследуется как private.
5. Class BaseClass: NewClass.
6. Конструкторы не наследуются.
7. Деструкторы не наследуются.
8. Сначала объект самого базового класса и дальше по старшинству.
9. Аналогично обычному деструктору.
10. 
11. Конструкторы нет, деструкторы да.
12. Да, виртуальность наследуется.
13. Класс потомок имеет доступ ко всем возможностям класса предка.
14. Закрытое наследование – наследование только реализации.
15. Везде, где можно использован объект базового класса, можно использовать вместо него и объект производного.
16. Name и Stage, private age не наследуется, post наследуется в private секции.
17. Student() {age=none; name=’none’;}; Employee() {post=’none’;}; Teacher() {stage=none;};
18. Student(int valueAge, string ValueName) {this->age= valueAge; this->name= ValueName;}; Employee(string ValuePost) {this->post= ValuePost;}; Teacher(int ValueStage) {this->stage= ValueStage;};
19. Student(const Student&other) {this->age=other.age; this->name= other.name;}; Employee(const Employee&other) {this->post= other.post;}; Teacher(const Teacher&other) {this->stage= other.stage;};
20. Student & operator=( const Student&other)

{

this->age=other.age;

this->name= other.name;

return \*this;

}

Employee & operator=( const Employee &other)

{

this->post= other.post; return \*this;

}

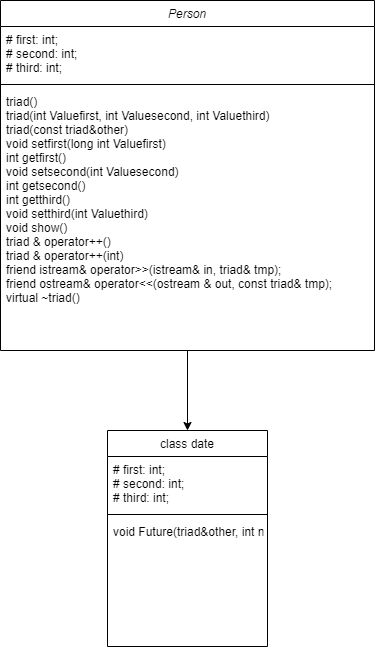
Teacher & operator=( const Teacher &other)

{

this->stage= other.stage; return \*this;

}

Блок-схема:



Код программы:

#include<iostream>

using namespace std;

class triad

{

public:

int first;

int second;

int third;

triad()

{

first = 10;

second = 06;

third = 2003;

cout << "Вызван конструктор без параметров для обьекта " << this << " с числами " << first << "." << second << "." << third << endl;

}

triad(int Valuefirst, int Valuesecond, int Valuethird)

{

first = Valuefirst;

second = Valuesecond;

third = Valuethird;

cout << "Вызван конструктор с параметрами для обьекта " << this << " с числами " << first << "." << second << "." << third << endl;

}

triad(const triad&other)

{

this->first = other.first;

this->second = other.second;

this->third = other.third;

cout << "Вызван конструктор копирования для обьекта " << this << " с числами " << first << "." << second << "." << third << endl;

}

void setfirst(long int Valuefirst)

{

first = Valuefirst;

}

int getfirst()

{

return first;

}

void setsecond(int Valuesecond)

{

second = Valuesecond;

}

int getsecond()

{

return second;

}

int getthird()

{

return third;

}

void setthird(int Valuethird)

{

third =Valuethird;

}

void show()

{

cout << "Вызван иной способ инициализации обьекта " << this << " с числами " << first << "." << second << "." << third << endl;

}

triad & operator++()

{

this->first += 1;

this->second += 1;

this->third += 1;

return \*this;

}

triad & operator++(int)

{

this->first += 1;

this->second += 1;

this->third += 1;

return \*this;

}

friend istream& operator>>(istream& in, triad& tmp);

friend ostream& operator<<(ostream & out, const triad& tmp);

virtual ~triad()

{

cout << "Вызван деструктор для обьекта " << this << endl;

}

};

istream& operator>>(istream& in, triad& tmp)

{

cout << "Введите первое число: "; in >> tmp.first;

cout << "Введите второе число: "; in >> tmp.second;

cout << "Введите третье число: "; in >> tmp.third;

return in;

}

ostream& operator<<(ostream& out, const triad& tmp)

{

return (out << "(" << tmp.first << "." << tmp.second << "."<<tmp.third);

}

class date : public triad

{

public:

void Future(triad&other, int n)

{

int day; int month; int year; int countM=0; int countY=0;

day = other.first + n;

while (day > 31)

{

day = day - 31;

countM++;

}

month = other.second + countM;

while (month > 12)

{

month = month - 12;

countY++;

}

year = other.third + countY;;

other.first = day; other.second = month; other.third = year;

}

};

int main()

{

setlocale(LC\_ALL, "Rus");

triad first;

triad second(05, 02, 2002);

triad third(second);

triad fourth;

fourth.setfirst(19);

fourth.setsecond(12);

fourth.setthird(1999);

fourth.getfirst();

fourth.getsecond();

fourth.getthird();

fourth.show();

date birth;

birth++;

birth.show();

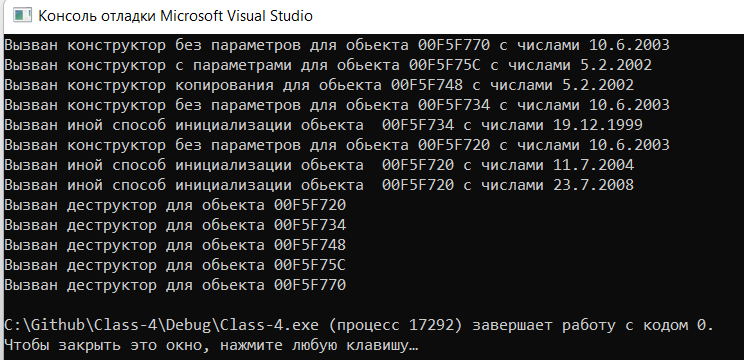
birth.Future(birth, 1500);

birth.show();

return 0;

}

Скриншоты результатов:



Анализ результатов:

Программа работает корректно. Выводятся поля созданные с помощью конструкторов без параметров, с параметрами, копирования, с помощью сеттеров и объект класса наследования, к которому прибавлено 1500 дней.