Задание.

Цель работы: научится наследовать классы объектов.

Задания

Во всех заданиях реализовать вывод на экран, методы получения значений полей и методы установки значений полей, а также необходимые конструкторы (если это не указано в задании явно). Конструкторы и методы обязательно должны проверять параметры на допустимость; в случае неправильных данных — выводить сообщение об ошибке и заканчивать работу. Преобразование в строку реализовать в виде функции преобразования toString().

Для демонстрации работы с объектами нового типа во всех заданиях требуется написать главную функцию. В программе должны присутствовать различные способы создания объектов и массивов объектов. Программа должна демонстрировать использование всех функций и методов.

Продемонстрировать принцип подстановки.

15. Создать класс Triad (тройка чисел); определить методы увеличения полей на1. Определить класс-наследник Time с полями: час, минута, секунда. Переопределить методы увеличения полей на 1 и определить методы увеличенияна и секунд и минут.

Код.

Triad.h

```
#include <iostream>
#include <string>

using namespace std;

class Triad
{
   int number1 = 0, number2 = 0, number3 = 0;

   public:
        Triad(int n1, int n2, int n3);

        // number1 methods
        int get_number1();
        void set_number1(int new_val);
        void increase_number1();
        // number2 methods
        int get_number2();
        void set_number2(int new_val);
        void increase_number2();
        // number3 methods
        int get_number3();
        void set_number3(int new_val);
        void increase_number3();
```

```
        // перезапись всего остального.
        Triad operator++();
        Triad operator++(int);
        // перезапись ввода/вывода.
        friend ostream& operator<<(ostream& stream, const Triad &a);
        friend istream& operator>>(istream& stream, Triad &a);
};
```

Triad.cpp

```
#include "triad.h"

using namespace std;

Triad::Triad(int n1, int n2, int n3)
{
    number1 = n1;
    number2 = n2;
    number3 = n3;
}

int Triad::get_number1()
{
    return this->number1;
}

void Triad::set_number1(int new_val)
{
    this->number1 = new_val;
}

void Triad::increase_number1()
{
    this->number1++;
}

int Triad::get_number2()
{
    return this->number2;
}

void Triad::set_number2(int new_val)
```

```
{
    this->number2 = new_val;
}

void Triad::increase_number2()
{
    this->number2++;
}

int Triad::get_number3()
{
    return this->number3;
}

void Triad::set_number3(int new_val)
{
    this->number3 = new_val;
}

void Triad::increase_number3()
{
    this->number3++;
}

Triad Triad::operator++()
{
    this->number1 += 1;
    this->number2 += 1;
    this->number3 += 1;
    return *this;
}

Triad Triad::operator++(int)
{
    this->number1 += 1;
    this->number2 += 1;
    this->number3 += 1;
    return *this;
}

ostream& operator<<(ostream& stream, const Triad &a)
{
```

```
    return (stream << "(" << a.number1 << ":" << a.number2 << ":" <<
a.number3 << ")");
}

istream &operator>>(istream &stream, Triad &a)
{
    string flag;
    cout << "Enter number1(enter 'o' if do not want to change number1):
";
    stream >> flag;
    if (flag != "o") a.number1 = stoi(flag);
    cout << "Enter number2(enter 'o' if do not want to change number2):
";
    stream >> flag;
    if (flag != "o") a.number2 = stoi(flag);
    cout << "Enter number3(enter 'o' if do not want to change number3):
";
    stream >> flag;
    if (flag != "o") a.number3 = stoi(flag);
    return stream;
}
```

Time.h

```
#include "triad.h"

class Time : public Triad
{
    int hours, minutes, seconds;
    public:
        Time(int n1, int n2, int n3);

        // hours
        void increase_number1();
        // minutes
        void increase_number2();
        // seconds
        void increase_number3();

        // перезапись инкремента.
        // здесь это операции увеличения и секунд, и минут.
        Time operator++();
```

```
        Time operator++(int);
        // ввод и вывод.
        friend ostream& operator<<(ostream& stream, const Time &a);
        friend istream& operator>>(istream& stream, Time &a);
};
```

Time.cpp

```
#include "time.h"

using namespace std;

Time::Time(int n1, int n2, int n3) : Triad (n1, n2, n3)
{
    if (n1 >= 0 && n2 >= 0 && n3 >= 0)
    {
        hours = n1;
        minutes = n2;
        seconds = n3;
        return;
    }
    cout << "Incorrect values were gotten! Default values were setted!";
    hours = 0;
    minutes = 0;
    seconds = 0;
}

void Time::increase_number1()
{
    this->hours += 1;
    this->set_number1(this->hours);
}

void Time::increase_number2()
{
    if (this->minutes != 60)
    {
        this->minutes += 1;
        this->set_number2(this->minutes);
        return;
    }
    this->increase_number1();
```

```
    this->minutes = 0;
    this->set_number2(0);
}

void Time::increase_number3()
{
    if (this->seconds != 60)
    {
        this->seconds += 1;
        this->set_number3(this->seconds);
        return;
    }
    this->increase_number2();
    this->seconds = 0;
    this->set_number3(0);
}

Time Time::operator++()
{
    this->increase_number3();
    return *this;
}

Time Time::operator++(int)
{
    this->increase_number3();
    return *this;
}

ostream &operator<<(ostream &stream, const Time &a)
{
    stream << "[";
    if (a.hours < 10) stream << "0";
    stream << a.hours << ":";
    if (a.minutes < 10) stream << "0";
    stream << a.minutes << ":";
    if (a.seconds < 10) stream << "0";
    stream << a.seconds << "]";
    return stream;
    // return (stream << "[" << a.hours << ":" << a.minutes << ":" <<
a.seconds << "]");
}
```

```
istream &operator>>(istream &stream, Time &a)
{
   string flag;
   cout << "Enter number1(enter 'o' if do not want to change number1):
";
   stream >> flag;
   if (flag != "o")
   {
       if (stoi(flag) >= 0) a.hours = stoi(flag);
       else cout << "Incorrect values! Hours has not changed!";
   }
   cout << "Enter number2(enter 'o' if do not want to change number2):
";
   stream >> flag;
   if (flag != "o")
   {
       if (stoi(flag) < 60 && stoi(flag) >= 0) a.minutes = stoi(flag);
       else cout << "Incorrect values! Minutes has not changed!";
   }
   cout << "Enter number3(enter 'o' if do not want to change number3):
";
   stream >> flag;
   if (flag != "o")
   {
       if (stoi(flag) < 60 && stoi(flag) >= 0) a.seconds = stoi(flag);
       else cout << "Incorrect values! Seconds has not changed!";
   }
   return stream;
}
```

Main.cpp

```
#include "time.h"

using namespace std;

int main()
{
   Time timer(0, 0, 0);
   Triad test(0, 0, 0);
   for (int i = 0; i <= 60; i++)
```

```
    {
        timer++;
    }
    cout << timer << "\n";
    cout << "Minute has left!\n";
    // демонстрация принципа подстановки.
    test = timer;
    cout << test << endl;
    return 0;
}
```

Результат.