**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**федеральное государственное образовательное бюджетное учреждение высшего образования**

**«Вологодский государственный университет»**

**(ВоГУ)**

* 1. **ОБЪЕКТНО-ОРЕНТИРОВАННОЕ программирование**
  2. **лабораторнАЯ РАБОТА №1**

**«Разработка классов на С++»**

**Направление подготовки: 09.03.01 Информатика и вычислительная техника**

**Направленность (профиль): Программное обеспечение средств вычислительной техники и автоматизированных систем**

**Форма обучения: очная**

**Институт: Математики, естественных и компьютерных наук**

**Кафедра: Автоматики и вычислительной техники**

**Группа: 4Б09 ПО-41**

**Студент: Махонин М.Н.**

**Руководитель: Сорокин А.Н.**

Вологда

2021 г.

**Цель работы:** научиться разрабатывать классы на языке С++.

**Вариант 2**

1. Разработать класс CTime для работы со значениями времени. Класс должен содержать следующие элементы.

Поля (скрытые):

* часы
* минуты
* секунды

Методы:

* конструктор по умолчанию, заполняющий все поля нулевыми значениями; конструктор, параметрами которого являются заданные значения для часов, минут и секунд; конструктор-копировщик
* методы-функции для задания значений времени и получения часов, минут и секунд
* метод-функция для вывода времени на экран в общепринятом формате «чч:мм:сс» (можно предусмотреть и сокращенный вывод «чч:мм»)
* методы-функции для выражения заданного отрезка времени в секундах и для перевода заданного количества секунд в переменную типа CTime

2. Реализовать для разработанного класса методы-операторы:

* +, - (плюс и минус) - для сложения и вычитания значений времени, при этом все данные должны быть типа CTime
* ++, --(два плюса или два минуса подряд) – уменьшение или увеличение значения времени на 1 секунду
* == (два знака равенства подряд) - для сравнения двух значений времени, результатом этой операции должен быть 0, если значения неравны, и любое ненулевое значение в противном случае

3. Разработать программу, которая демонстрировала бы работоспособность всех методов данного класса.

#include <iostream>

#include <cmath>

using namespace std;

class CTime

{

public:

CTime() {

\_hour = 0;

\_minute = 0;

\_second = 0;

}

CTime(int hour, int minute, int second) {

if (hour > 23 || minute > 60 || second > 60) {

cout << "Неккоректное время" << endl;

}

else {

\_hour = hour;

\_minute = minute;

\_second = second;

}

}

CTime(const CTime& time) {

\_hour = time.\_hour;

\_minute = time.\_minute;

\_second = time.\_second;

}

void SetHour(int hour) {

if (hour > 23) {

cout << "Неккоректное время" << endl;

}

else

{

\_hour = hour;

}

}

void SetMinute(int minute) {

if (minute > 60) {

cout << "Неккоректное время" << endl;

}

else

{

\_minute = minute;

}

}

void SetSecond(int second) {

if (second > 60) {

cout << "Неккоректное время" << endl;

}

else

{

\_second = second;

}

}

int GetHour() {

return \_hour;

}

int GetMinute() {

return \_minute;

}

int GetSecond() {

return \_second;

}

void ShowTime() {

cout << \_hour << ':' << \_minute << ':' << \_second << endl;

}

int TimeToSecond(CTime time1, CTime time2) {

int hour = abs(time1.\_hour - time2.\_hour);

int minute = abs(time1.\_minute - time2.\_minute);

int second = abs(time1.\_second - time2.\_second);

return hour \* 360 + minute \* 60 + second;

}

CTime SecondToTime(int second) {

int hour = (second / 360) % 24;

int minute = (second % 360) / 60;

int \_second = (second % 360) % 60;

return CTime(hour, minute, \_second);

}

CTime operator + (CTime time2) {

int hour = (this->\_hour + time2.\_hour) % 24;

int minute = (this->\_minute + time2.\_minute) % 60;

int second = (this->\_second + time2.\_second) % 60;

return CTime(hour, minute, second);

}

CTime operator - (CTime time2) {

int hour = abs(this->\_hour + time2.\_hour);

int minute = abs(this->\_minute + time2.\_minute);

int second = abs(this->\_second + time2.\_second);

return CTime(hour, minute, second);

}

CTime operator ++(int) {

CTime temp(\*this);

this->\_second++;

if (this->\_second > 60) {

this->\_minute++;

this->\_second = 0;

}

if (this->\_minute > 60) {

this->\_hour++;

this->\_minute = 0;

}

if (this->\_hour > 23) {

this->\_hour = 0;

}

return temp;

}

CTime operator --(int) {

CTime temp(\*this);

if (\_second == 0) {

if (\_minute == 0) {

if (\_hour == 0) {

\_hour = 23;

\_minute = 59;

\_second = 59;

}

\_hour--;

\_minute = 59;

\_second = 59;

}

\_minute--;

\_second = 59;

}

\_second--;

return temp;

}

CTime& operator ++() {

this->\_second++;

if (this->\_second > 60) {

this->\_minute++;

this->\_second = 0;

}

if (this->\_minute > 60) {

this->\_hour++;

this->\_minute = 0;

}

if (this->\_hour > 23) {

this->\_hour = 0;

}

return \*this;

}

CTime& operator --() {

if (\_second == 0) {

if (\_minute == 0) {

if (\_hour == 0) {

\_hour = 23;

\_minute = 59;

\_second = 59;

}

\_hour--;

\_minute = 59;

\_second = 59;

}

\_minute--;

\_second = 59;

}

\_second--;

return \*this;

}

bool operator == (const CTime& time2) {

return (this->\_hour == time2.\_hour && this->\_minute == time2.\_minute &&

this->\_second == time2.\_second);

}

private:

int \_hour;

int \_minute;

int \_second;

};

void main() {

setlocale(LC\_ALL, "Russian");

CTime time1(12, 35, 40);

CTime time2(6, 50, 0);

time1.ShowTime();

time1.SetMinute(20);

time2.SetSecond(70);

time1.ShowTime();

cout << time2.GetHour() << ':' << time2.GetMinute() << endl;

CTime time3 = time1 + time2;

time3.ShowTime();

if (time3 == time1) {

cout << "time3 == time1" << endl;

}

else {

cout << "time3 != time1" << endl;

}

time1++;

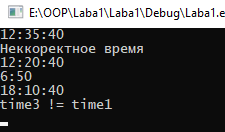
++time1;

time1.ShowTime();

int a;

cin >> a;

}



***Контрольные вопросы***

**1. Что понимается под термином «класс»? Каково соотношение понятий «класс» и «экземпляр класса»?**

Класс — это тип данных, определяемый пользователем, который похож на структуру или запись, так как содержит поля с данными, но кроме полей в описание класса входят также функции, которые предназначены для обработки полей. Такие функции называются методами класса. Поля и методы называются элементами класса.

Экземпляр класса является объектом данного класса и через него можно с ним взаимодействовать.

**2. Что понимается под термином «конструктор»? В каких случаях вызывается конструктор копирования и конструктор преобразования класса?**

Конструкторы — это особые методы, которые выполняются автоматически при создании объекта. При этом конструктор имеет то же самое имя, что и класс.

Конструктор преобразования вызывается, когда необходимо задать начальные значения полей класса.

Конструктор копирования вызывается в случаях, когда нужно получить полную копию объекта. В C++ полная копия объекта нужна в трех случаях:

* В момент объявления нового объекта и его инициализации данными другого объекта с помощью оператора =.
* Когда нужно передать объект в функцию как параметр-значение. В этом случае создается полная копия объекта.
* Когда нужно вернуть объект из функции по значению. В этом случае также создается полная копия объекта.

**3. Что понимается под термином «деструктор»? В каких случаях применяется деструктор класса?**

Деструкторы — это особые методы, которые выполняются автоматически при уничтожении объекта. Деструкторы имеют то же самое имя, но со знаком ~. Деструктор вызывается в том случае когда программа покидает область видимости, после этого происходит чистка полей класса.