Министерство науки и высшего образования Российской Федерации

Новосибирский государственный технический университет

Кафедра вычислительной техники

Лабораторная работа №1

По дисциплине «Программирование»

**.**

Преподаватель: Булатов А.Д.  
Группа:  АВТ-042  
Студент: Лукьянов А.О.

# Постановка задачи

Реализовать класс в соответствии с вариантом. Класс должен обеспечивать набор методов для работы с данными. Создать перегруженные конструкторы: конструктор копирования, конструктор по умолчанию, конструктор с параметрами. Реализовать в классах поля с динамическим выделением памяти для хранения некоторых данных. Создать деструктор для освобождения памяти. Изучить порядок вызова конструкторов и деструктора. Обязательно добавить в класс статические члены.

Составить демонстрационную программу, реализующую вызов всех методов класса. Для демонстрационной программы использовать отдельный модуль. Программу построить с использованием проекта. Посмотреть работу программы в отладчике, обратить внимание на представление данных. Построить программу без отладочной информации. Обратить внимание на размер программы.

Вариант 1:

Постpоить класс Дата-Вpемя. Класс должен включать следующие поля: день, месяц, год, часы, минуты, строковое представление даты. Класс должен обеспечивать пpостейшие функции для pаботы с данными класса: увеличение/уменьшение на 1 день, час, минуту, секунду и т.д., методы изменения объекта, вывода на экран объекта.

# Листинг программы

Содержимое “DT.h”

#pragma once

#include <iostream>

#include <string>

using namespace std;

class DT

{

public:

DT();

~DT();

DT(short valueday, short valuemonth, short valueyear, short valuehour, short valueminute);

DT(const DT & ob);

private:

static int val;

short day;

short month;

short year;

short hour;

short minute;

public:

char\* SR();

int PlusDay();

int PlusMonth();

int PlusYear();

int PlusHour();

int PlusMinute();

void Write();

int MinusDay();

int MinusMonth();

int MinusYear();

int MinusHour();

int MinusMinute();

void SetDay(int day1);

void SetMonth(int month1);

void SetYear(int year1);

void SetHour(int hour1);

void SetMinute(int minute1);

int GetDay();

int GetMonth();

int GetYear();

int GetHour();

int GetMinute();

};

Содержимое “DT.cpp”

#include "DT.h"

#include <iostream>

#include <string>

using namespace std;

DT::DT()

{

year = 0;

day = 1;

month = 1;

hour = 0;

minute = 0;

}

DT::DT(short valueday, short valuemonth, short valueyear, short valuehour, short valueminute)

{

if (valueday < 32 && valueday > 0)

{

day = valueday;

}

else

{

day = 1;

}

if (valuemonth < 13 && valuemonth >0)

{

month = valuemonth;

}

else

{

month = 1;

}

if (valuehour < 25 && valuehour >= 0)

{

hour = valuehour;

}

else

{

hour = 1;

}

if (valueminute < 61 && valueminute >= 0)

{

minute = valueminute;

}

else

{

minute = 1;

}

if (valueyear > 0)

{

year = valueyear;

}

else

{

year = 1;

}

}

DT::DT(const DT& ob)

{

day = ob.day;

month = ob.month;

year = ob.year;

hour = ob.hour;

minute = ob.minute;

}

DT::~DT()

{

cout << " Destructing DT "<<endl;

}

void DT::SetDay(int day1) {

if (day1 > 31 || day1 < 0) { cout << "Error"; }

else{day = day1;}

}

void DT::SetMonth(int month1) {

if (month1 > 12 || month1 < 0) { cout << "Error"; }

else { month = month1; }

}

void DT::SetYear(int year1) {

if (year1 < 0) { cout << "Error"; }

else { year = year1; }

}

void DT::SetHour(int hour1) {

if (hour1 > 24 || hour1<0) { cout << "Error"; }

else { hour = hour1; }

}

void DT::SetMinute(int minute1) {

if (minute1 > 24 || minute1 < 0) { cout << "Error"; }

else { minute = minute1; }

}

int DT::GetDay() { return day; };

int DT::GetMonth() { return month; };

int DT::GetYear() { return year; };

int DT::GetHour() { return hour; };

int DT::GetMinute() { return minute; };

int DT::PlusDay()

{

day++;

return day;

}

int DT::PlusMonth()

{

month++;

return month;

}

int DT::PlusYear()

{

year++;

return year;

}

int DT::PlusHour()

{

hour++;

return hour;

}

int DT::PlusMinute()

{

minute++;

return minute;

}

int DT::MinusDay()

{

day--;

return day;

}

int DT::MinusMonth()

{

month--;

return month;

}

int DT::MinusYear()

{

year--;

return year;

}

int DT::MinusHour()

{

hour--;

return hour;

}

int DT::MinusMinute()

{

minute--;

return minute;

}

char\* DT::SR()

{

char\* stroka= new char[255];

sprintf\_s(stroka,25, "%d.%d.%d.%d.%d", minute, hour, day, month, year);

return stroka;

}

void DT::Write()

{

cout << SR();

}

Содержимое “Test.h”

#pragma once

using namespace std;

class Test

{

public:

void TestPlus();

void TestMinus();

void TestExcep();

};

Содержимое “Test.cpp”

#include "Test.h"

#include "DT.h"

#include <iostream>

#include <cassert>

void Test::TestPlus()

{

DT a = DT(1,1,1,1,1);

assert(a.PlusDay()==2);

assert(a.PlusHour() == 2);

assert(a.PlusMinute() == 2);

assert(a.PlusMonth() == 2);

assert(a.PlusYear() == 2);

cout << "TestPlus completed";

}

void Test::TestMinus()

{

DT a = DT(2, 2, 2, 2, 2);

assert(a.MinusDay() == 1);

assert(a.MinusHour() == 1);

assert(a.MinusMinute() == 1);

assert(a.MinusMonth() == 1);

assert(a.MinusYear() == 1);

cout << "TestMinus completed";

}

void Test::TestExcep()

{

DT a = DT(0, -1, 0, -1, -1);

assert(a.GetDay() == 1);

assert(a.GetMonth() == 1);

assert(a.GetHour() == 1);

assert(a.GetMinute() == 1);

assert(a.GetYear() == 1);

cout << "TestExcep completed";

}

Содержимое “ LB1.cpp”

#include <iostream>

#include "DT.h"

#include "Test.h"

using namespace std;

int main()

{

setlocale(LC\_ALL, "ru");

Test t = Test();

t.TestPlus();

t.TestExcep();

t.TestMinus();

} **Вывод:**  
Изучил структуру класса, механизм создания и использования, описание членов-данных класса и методов доступа к ним, возможность инициализации объектов класса с помощью конструкторов и уничтожение их с помощью деструкторов. Построил класс для работы с датой и временем. Класс обеспечивает простейшие функции для работы с данными класса: сложение, вычитание.  
Использование классов намного уменьшает объем кода и создает удобную структурированную систему. В данной лабораторной работе использование классов очень облегчило написание программы.