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: Галета Н.Е.

# Цель работы

Изучить 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.

# 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:

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

# Листинг программы

Содержимое “Date.h”

#pragma once

#include <stdio.h>

#include <string>

using namespace std;

class Date

{

public:

static int a;

Date(int day, int month, int year, int hour, int minute);

Date();

Date(const Date& obj);

~Date();

void additionDay();

void additionMonth();

void additionYear();

void additionHour();

void additionMinute();

void subtractionDay();

void subtractionMonth();

void subtractionYear();

void subtractionHour();

void subtractionMinute();

void setDay(int day);

void setMonth(int mounth);

void setYear(int year);

void setHour(int hour);

void setMinute(int minute);

int getDay();

int getMonth();

int getYear();

int getHour();

int getMinute();

char\* ToString();

private:

int day, month, year, hour, minute;

};

Содержимое “Date.cpp”

#include "Date.h"

#include <iostream>

#include <stdio.h>

Date::Date() {

day = 0;

month = 0;

year = 0;

hour = 0;

minute = 0;

}

Date::Date(int day, int month, int year, int hour, int minute) {

if ((day < 31) && (day > 0)) {

this->day = day;

}

else {

this->day = 1;

}

if ((month < 13) && (month > 0)) {

this->month = month;

}

else {

this->month = 1;

}

this->year = year;

if ((minute < 61) && (minute > 0)) {

this->minute = minute;

}

else {

this->minute = 0;

}

if ((hour < 25) && (hour > 0)) {

this->hour = hour;

}

else {

this->hour = 0;

}

}

Date::Date(const Date& date) {//конструктор копирования

this->day = date.day;

this->month = date.month;

this->year = date.year;

this->minute = date.minute;

this->hour = date.hour;

}

Date::~Date()

{

cout << "Memory is cleared" << std::endl;

}

void Date::additionDay()

{

if (day+1 >= 30) throw "ERROR";

else{day++; }

}

void Date::additionMonth()

{

if (month+1 > 12) throw "ERROR";

month++;

}

void Date::additionYear()

{

year++;

}

void Date::additionHour()

{

if (hour+1 > 24) throw "ERROR";

hour++;

}

void Date::additionMinute()

{

if (minute+1 > 60) throw "ERROR";

minute++;

}

void Date::subtractionDay()

{

if (day - 1 < 1) {

setMonth(month - 1);

day = 29;

}

else {

day--;

}

}

void Date::subtractionMonth()

{

if (month - 1 < 1) {

setYear(year - 1);

month = 59;

}

else { month--; }

}

void Date::subtractionYear()

{

year--;

}

void Date::subtractionHour()

{

if (hour - 1 < 0) {

setDay(day - 1);

hour = 23;

}

else{hour--;

}

}

void Date::subtractionMinute()

{

//if (minute <= 0) throw "ERROR"

if (minute - 1 < 0) {

setHour(hour-1);

minute = 59;

}

else {

minute--;

}

}

void Date::setDay(int day)

{

if (day > 30 || day < 1) throw "ERROR";

this->day = day;

}

void Date::setMonth(int month)

{

if (month > 13 || month < 1) throw "ERROR";

this->month = month;

}

void Date::setYear(int year)

{

this->year = year;

}

void Date::setHour(int hour)

{

if (hour > 24 || hour < 0) throw "ERROR";

this->hour = hour;

}

void Date::setMinute(int minute)

{

if (minute > 60 || minute < 0) throw "ERROR";

this->minute = minute;

}

int Date::getDay()

{

return day;

}

int Date::getMonth()

{

return month;

}

int Date::getYear()

{

return year;

}

int Date::getHour()

{

return hour;

}

int Date::getMinute()

{

return minute;

}

char\* Date::ToString()

{

char\* string = new char[255];

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

return string;

}

Содержимое “Test.h”

#pragma once

using namespace std;

class Test

{

public:

void Test1();

void Test2();

void Test3();

void Test4();

};

Содержимое “Test.cpp”

#include "Test.h"

#include "Date.h"

#include <iostream>

#include <cassert>

using namespace std;

bool equal(Date& date, int a[5]) {

int i = 0;

bool k = false;

if ((a[0] == date.getDay()) && (a[1] == date.getMonth()) && (a[2] == date.getYear()) && (a[3] == date.getMinute()) && ((a[4] == date.getHour()))) {

k = true;

}

return k;

}

void Test::Test1()

{

cout << "First test\n";

int a[5] = { 3,7,2005, 34, 21 };

Date date1(3, 7, 2221, 21, 35);

cout<<date1.ToString();

cout << endl;

try{

date1.setYear(2005);

date1.subtractionMinute();

cout << date1.ToString();

}

catch (const char\* msg) {

cout << msg << endl;

}

bool t = equal(date1, a);

if (t == 1) {

cout << " true" << endl;

}

else{ cout << " false" << endl; }

cout << endl << endl;

}

void Test::Test2()

{

cout << "Second test\n";

int b[5] = { 4,3,2001, 3, 2 };

Date date2(5, 4, 2002, 3, 3);

cout << date2.ToString();

cout << endl;

try {

date2.subtractionHour();

date2.subtractionDay();

date2.subtractionMonth();

date2.subtractionYear();

cout << date2.ToString();

}

catch(const char\* msg) {

cout << msg << endl;

}

bool t2 = equal(date2, b);

if (t2 == 1) {

cout << " true" << endl;

}

else {

cout << " false" << endl;

}

cout << endl << endl;

}

void Test::Test3()

{

cout << "Third test\n";

Date date3(13, 11, 2002, 8, 9);

int c[5] = { 13,11,2002, 8, 9 };

cout << date3.ToString();

cout << endl;

try {

date3.setDay(50);

date3.additionDay();

date3.additionMonth();

date3.additionYear();

date3.additionMinute();

cout << date3.ToString();

}

catch (const char\* msg) {

cout << msg << endl;

}

bool t3 = equal(date3, c);

if (t3 == 1) {

cout << " true" << endl;

}

else{ cout << " false" << endl; }

cout << endl << endl;

}

void Test::Test4()

{

cout << "Fourth test\n";

Date date4(1, 10, 2002, 59, 60);

int d[5] = {29,8,2001, 3, 12 };

cout << date4.ToString();

cout << endl;

try {

date4.additionMinute();

date4.additionDay();

date4.additionMonth();

date4.additionYear();

cout << date4.ToString();

}

catch (const char\* msg) {

cout << msg << endl;

}

try {

date4.subtractionDay();

date4.subtractionMonth();

date4.subtractionYear();

date4.subtractionMinute();

cout << date4.ToString();

}

catch (const char\* msg) {

cout << msg << endl;

}

bool t4 = equal(date4, d);

if (t4 == 1) {

cout << " true" << endl;

}

else{ cout << " false" << endl; }

cout << endl << endl;

}

Содержимое “ СonsoleApplication3.cpp”

#include <iostream>

#include "Date.h"

#include "Test.h"

using namespace std;

int main()

{

setlocale(LC\_ALL, "ru");

Test t = Test();

t.Test1();

t.Test2();

t.Test3();

t.Test4();

char ch;

std::cin >> ch;

}

**Вывод:**  
Построил класс для работы с датой и временем и научился работать с основными принципами класса. Данный класс обеспечивает простейшие функции для работы с данными: прибавление 1, вычитание 1.  
Благодаря использованию класса создается удобно структурированная система. В данной лабораторной работе использование классов облегчило работу программы.