**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ**

**КЫРГЫЗСКОЙ РЕСПУБЛИКИ**

**КЫРГЫЗСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ им. И.Раззакова**

**ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ**

Кафедра: **Программное обеспечение компьютерных систем**

Курс «Объектно-ориентированное программирование»

**ОТЧЕТ**

**Лабораторная работа №3**

Выполнил: студент группы ПИ-3-21

Жайлообек уулу Улукбек

Проверил: Мусабаев Э.Б.

**Бишкек 2024**

# **Задание №1**

**Постановка задачи:**

Необходимо разработать программу с использованием класса **Timer**, который представляет собой таймер обратного отсчета. При создании объекта типа **Timer**, пользователю предоставляется возможность указать начальное значение времени. После вызова метода **run()**, таймер начинает отсчет в сторону уменьшающихся значений до тех пор, пока не достигнет значения 0, после чего звучит звонок.

Программа должна предоставлять три варианта конструктора для класса **Timer**:

1. Конструктор, принимающий начальное значение времени в секундах с помощью целого числа.
2. Конструктор, принимающий строку времени в формате "часы:минуты:секунды".
3. Конструктор, принимающий два целых числа - минуты и секунды.

Для реализации таймера используется библиотечная функция **clock()**, которая возвращает количество тиков, прошедших с момента запуска программы. Значение времени таймера рассчитывается путем деления этого значения на макрос **CLK\_TCK**, чтобы получить время в секундах. Прототипы для **clock()** и **CLK\_TCK** содержатся в заголовочном файле **time.h**.

**Исходный код программы на C++:**

#include <iostream>

#include <string>

#include <ctime>

using namespace std;

class Timer {

private:

int seconds;

public:

Timer(int initialSeconds) : seconds(initialSeconds) {}

Timer(const string& timeString) {

int hours, minutes, sec;

sscanf\_s(timeString.c\_str(), "%d:%d:%d", &hours, &minutes, &sec);

seconds = hours \* 3600 + minutes \* 60 + sec;

}

Timer(int minutes, int seconds) : seconds(minutes \* 60 + seconds) {}

void run() {

while (seconds > 0) {

cout << "Осталось времени: " << seconds << " секунд." << endl;

--seconds;

for (int i = 0; i < 100000000; ++i) {}

}

cout << "Звонок! Время вышло." << endl;

}

};

int main() {

setlocale(LC\_ALL, "Russian");

Timer timer1(60);

Timer timer2("00:05:30");

Timer timer3(3, 30);

cout << "Таймер 1:" << endl;

timer1.run();

cout << "\nТаймер 2:" << endl;

timer2.run();

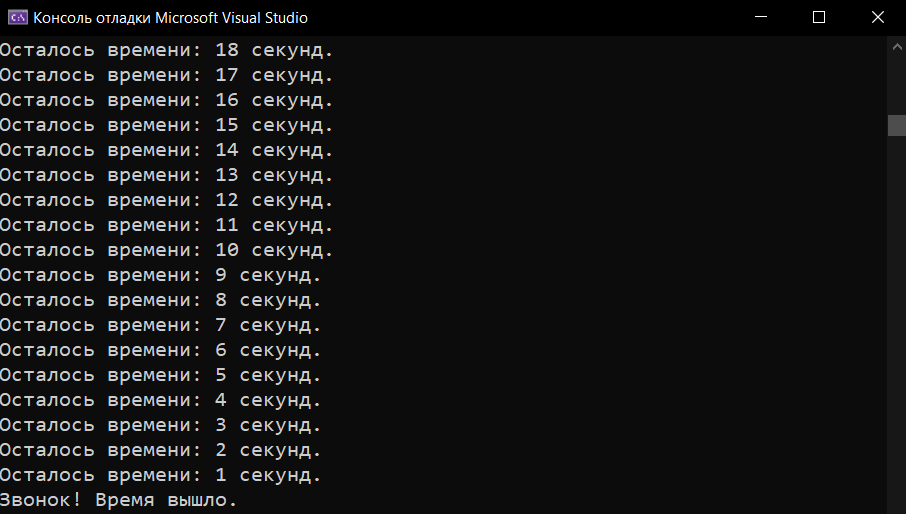
cout << "\nТаймер 3:" << endl;

timer3.run();

return 0;

}

**Тесты:**



# **Задание №2**

**Постановка задачи:**

Необходимо разработать программу с классом **Nomenclature**, который представляет собой описание товаров на складе магазина. Класс содержит закрытые элементы данных:

1. Название товара (**itemName**) - строка.
2. Оптовая цена (**wholesalePrice**) - вещественное число, представляющее цену закупки товара.
3. Розничная наценка (**retailMarkup**) - вещественное число, представляющее процентную наценку на товар.
4. Количество товаров на складе (**quantityInStock**) - целое число.

Открытые функции класса включают:

1. Конструктор, принимающий название товара, оптовую цену, розничную наценку и количество товаров на складе для инициализации объекта.
2. Деструктор, выводящий сообщение об удалении объекта с указанием названия товара.
3. Функция **calculateNetProfit()**, возвращающая возможный чистый доход при продаже всех товаров на складе. Чистый доход рассчитывается как произведение количества товаров на складе на разницу между розничной ценой и оптовой ценой.
4. Функция **displayItemInfo()**, выводящая на экран всю информацию о товаре, включая название, оптовую цену, розничную наценку, количество на складе и возможный чистый доход.

Программа должна создавать объекты класса **Nomenclature**, инициализировать их данными о товарах и выводить информацию о каждом товаре на экран.

**Исходный код программы на C++:**

#include <iostream>

#include <string>

using namespace std;

class Nomenclature {

private:

string itemName;

double wholesalePrice;

double retailMarkup;

int quantityInStock;

public:

Nomenclature(const string& name, double wholesale, double markup, int quantity)

: itemName(name), wholesalePrice(wholesale), retailMarkup(markup), quantityInStock(quantity) {}

~Nomenclature() {

cout << "Объект " << itemName << " удален." << endl;

}

double calculateNetProfit() const {

return quantityInStock \* (retailMarkup \* wholesalePrice);

}

void displayItemInfo() const {

cout << "Название товара: " << itemName << endl;

cout << "Оптовая цена: " << wholesalePrice << " руб." << endl;

cout << "Розничная наценка: " << retailMarkup << "%" << endl;

cout << "Количество на складе: " << quantityInStock << " шт." << endl;

cout << "Возможный чистый доход: " << calculateNetProfit() << " руб." << endl;

}

};

int main() {

setlocale(LC\_ALL, "Russian");

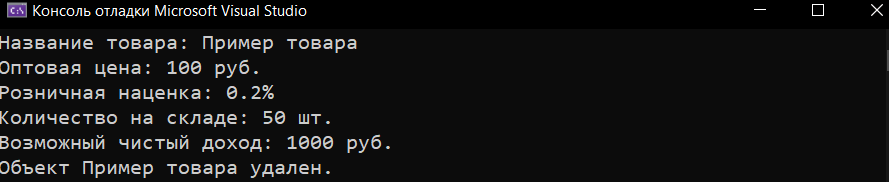
Nomenclature item("Пример товара", 100.0, 0.2, 50);

item.displayItemInfo();

return 0;

}

**Тесты:**



# **Задание №3**

**Постановка задачи:**

Необходимо создать класс **Soft**, который будет содержать информацию об установленном программном обеспечении. Закрытыми элементами класса будут:

1. Название программы (**programName**) - строка.
2. Разработчик (**developer**) - строка.
3. Занимаемый объем (**diskSpace**) - вещественное число, представляющее объем программы в гигабайтах.
4. Дата завершения лицензии (**licenseExpirationDate**) - объект типа **time\_t**, представляющий дату и время завершения лицензии.

Открытые функции класса включают:

1. Конструктор, принимающий название программы, разработчика, занимаемый объем и дату завершения лицензии для инициализации объекта.
2. Деструктор, выводящий сообщение об удалении объекта с указанием названия программы.
3. Функция **daysUntilExpiration()**, возвращающая количество дней до завершения лицензии на программное обеспечение.
4. Функция **displaySoftwareInfo()**, выводящая на экран всю информацию о программном обеспечении, включая название, разработчика, занимаемый объем, дату завершения лицензии и количество дней до завершения лицензии.

Программа должна создавать объекты класса **Soft**, инициализировать их данными о программном обеспечении и выводить информацию о каждом программном обеспечении на экран.

**Исходный код программы на C++:**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

#include <string>

#include <ctime>

using namespace std;

class Soft {

private:

string programName;

string developer;

double diskSpace;

time\_t licenseExpirationDate;

public:

Soft(const string& name, const string& dev, double space, const time\_t& expiration)

: programName(name), developer(dev), diskSpace(space), licenseExpirationDate(expiration) {}

~Soft() {

cout << "Удален объект " << programName << "." << endl;

}

int daysUntilExpiration() const {

time\_t now = time(nullptr);

double difference = difftime(licenseExpirationDate, now);

return static\_cast<int>(difference / (60 \* 60 \* 24));

}

void displaySoftwareInfo() const {

cout << "Название программы: " << programName << endl;

cout << "Разработчик: " << developer << endl;

cout << "Занимаемый объем: " << diskSpace << " Гб" << endl;

cout << "Дата завершения лицензии: " << ctime(&licenseExpirationDate);

cout << "Дней до завершения лицензии: " << daysUntilExpiration() << endl;

}

};

int main() {

setlocale(LC\_ALL, "Russian");

time\_t expirationDate = time(nullptr) + 30 \* 24 \* 60 \* 60;

Soft software("Примерное программное обеспечение", "Примерный разработчик", 2.5, expirationDate);

software.displaySoftwareInfo();

return 0;

}

**Тесты:**

