**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ**

**КЫРГЫЗСКОЙ РЕСПУБЛИКИ**

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

**ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ**

Кафедра: **Программное обеспечение компьютерных систем**

Дисциплина «**Объектно-Ориентированное Программирование**»

Отчет

Лабораторная работа №3

**Тема: Конструктор и деструктор**

Выполнил: студент группы ПИ-2-21

Камиев Абулхаир

Проверил: Мусабаев Э. Б.

**Бишкек 2024**

**Задание №1**

1. Разработайте программу с классом **Timer**, в которой таймер действует как таймер обратного отсчета. При создании объекта типа **timer** ему присваивается начальное значение времени. В результате вызова функции **run()** таймер начинает отсчет в сторону уменьшающихся значений, пока не достигнет значения 0, после чего зазвонит звонок. **Создайте 3 варианта конструктора, т.е. конструктор перегрузите 3 раза** для того, чтобы можно было указывать время в секундах с помощью целого числа или строки, или в минутах и секундах, если указываются два целых числа.

Эта программа использует библиотечную функцию **clock(),** возвращающую число тиков, прошедших с момента запуска программы. Поделив это значение на макрос **CLK\_TCK**, получаем значение в секундах. Прототипы для **clock**() и **CLK\_TCK** содержатся в заголовочном файле time.h.

#include <iostream>

#include <ctime>

#include <string>

using namespace std;

const int CLK\_TCK\_CONST = CLOCKS\_PER\_SEC;

class Timer {

private:

int seconds;

public:

// Конструктор, принимающий время в секундах

Timer(int secs) {

seconds = secs;

}

// Конструктор, принимающий строку времени в формате "мм:сс"

Timer(const string& timeStr) {

int minutes = stoi(timeStr.substr(0, 2));

int seconds = stoi(timeStr.substr(3, 2));

this->seconds = minutes \* 60 + seconds;

}

// Конструктор, принимающий минуты и секунды

Timer(int minutes, int seconds) {

this->seconds = minutes \* 60 + seconds;

}

// Функция запуска таймера

void run() {

clock\_t start = clock();

while (seconds > 0) {

// Получаем текущее время

clock\_t current = clock();

double elapsed\_secs = double(current - start) / CLK\_TCK\_CONST;

// Если прошла одна секунда, уменьшаем счетчик секунд

if (elapsed\_secs >= 1.0) {

seconds--;

start = current;

// Выводим текущее количество оставшихся секунд

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

}

}

cout << "Звонок!" << endl;

}

};

int main() {

// Создание объекта Timer с указанием времени в секундах

Timer timer1(10);

timer1.run();

// Создание объекта Timer с указанием времени в формате "мм:сс"

Timer timer2("01:01");

timer2.run();

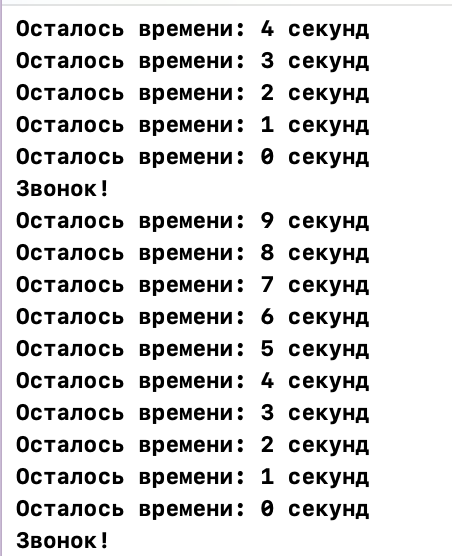
// Создание объекта Timer с указанием минут и секунд

Timer timer3(1, 5);

timer3.run();

return 0;

}



**Задание №2**

2. Создайте класс **Nomenclature**, описывающий товары на складе магазина. Закрытыми элементами класса будут: название товара, оптовая цена, розничная наценка и количество товаров на складе. Включите в класс открытые функции подсчета возможного чистого дохода при продаже этого товара и вывода всех данных о товаре на экран. Для инициализации и удаления объектов класса используйте конструкторы и деструкторы.

////№2

#include <iostream>

#include <string>

class Nomenclature {

private:

std::string productName;

double wholesalePrice;

double retailMarkup;

int quantity;

public:

// Конструктор

Nomenclature(const std::string& name, double wholesalePrice, double retailMarkup, int quantity)

: productName(name), wholesalePrice(wholesalePrice), retailMarkup(retailMarkup), quantity(quantity) {}

// Деструктор

~Nomenclature() {}

// Функция для расчета чистого дохода при продаже

double calculateProfit() const {

double retailPrice = wholesalePrice \* (1 + retailMarkup);

return (retailPrice - wholesalePrice) \* quantity;

}

// Функция для вывода данных о товаре на экран

void displayProduct() const {

std::cout << "Наименование товара: " << productName << std::endl;

std::cout << "Оптовая цена: $" << wholesalePrice << std::endl;

std::cout << "Розничная наценка: " << retailMarkup \* 100 << "%" << std::endl;

std::cout << "Колличество товара на складе: " << quantity << std::endl;

}

};

int main() {

// Пример использования класса

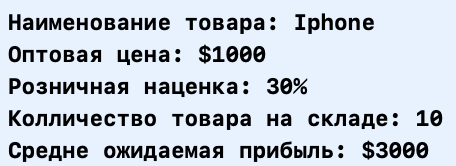
Nomenclature product1("Iphone", 1000, 0.3, 10);

product1.displayProduct();

std::cout << "Средне ожидаемая прибыль: $" << product1.calculateProfit() << std::endl;

return 0;

}

****

**Задание №3**

3. Создайте класс **Soft**, который содержит информацию об установленном программном обеспечении. Закрытыми элементами класса будут: название программы, разработчик, занимаемый объем, дата завершения лицензии. Включите в класс открытые функции подсчета количества дней до завершения лицензии и вывода всех данных об установленном программном обеспечении на экран. Для инициализации и удаления объектов класса используйте конструкторы и деструкторы.

//3

#include <iostream>

#include <string>

#include <chrono>

class Soft {

private:

std::string name;

std::string developer;

double size;

std::chrono::system\_clock::time\_point licenseExpirationDate;

public:

// Конструктор класса

Soft(const std::string& name, const std::string& developer, double size, const std::chrono::system\_clock::time\_point& expirationDate)

: name(name), developer(developer), size(size), licenseExpirationDate(expirationDate) {

std::cout << "Object of class Soft is being created." << std::endl;

}

// Деструктор класса

~Soft() {

std::cout << "Object of class Soft is being destroyed." << std::endl;

}

// Функция для подсчета количества дней до завершения лицензии

int daysUntilLicenseExpiration() const {

auto now = std::chrono::system\_clock::now();

std::chrono::duration<int> diff = std::chrono::duration\_cast<std::chrono::days>(licenseExpirationDate - now);

return diff.count();

}

// Функция для вывода всех данных об установленном программном обеспечении на экран

void printInfo() const {

std::cout << "Name: " << name << std::endl;

std::cout << "Developer: " << developer << std::endl;

std::cout << "Size: " << size << " MB" << std::endl;

std::cout << "Days until license expiration: " << daysUntilLicenseExpiration() << std::endl;

}

};

int main() {

// Пример использования класса Soft

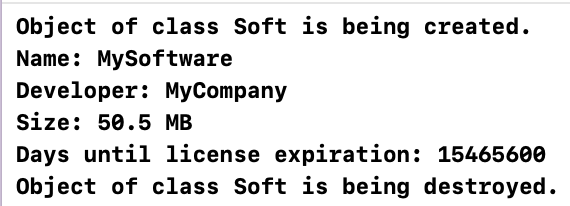
std::chrono::system\_clock::time\_point expirationDate = std::chrono::system\_clock::now() + std::chrono::hours(24 \* 30 \* 6); // Лицензия на 6 месяцев

Soft mySoftware("MySoftware", "MyCompany", 50.5, expirationDate);

mySoftware.printInfo();

return 0;

}

****