**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ КЫРГЫЗСКОЙ РЕСПУБЛИКИ**

**КЫРГЫЗСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСТИТЕТ**

**им. И.Раззакова**

**ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ**

Кафедра «**Программное обеспечение компьютерных систем**»

Направление: 710400 «**Программная инженерия**»

Дисциплина: ««**Объектно-ориентированное программирование**»»

ОТЧЕТ

Лабораторная работа №3

Выполнил: Кудайбердиев Эрлан

Группа: ПИ-2-21

Проверил: Мусабаев Э. Б.

Бишкек – 2023

**Задание №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.

**Код программы**

#include <iostream>

#include <ctime>

#include <string>

#define CLK\_TCK CLOCKS\_PER\_SEC

using namespace std;

class Timer {

public:

int seconds;

public:

Timer(int sec) {

seconds = sec;

}

Timer(int min, int sec) {

seconds = min \* 60 + sec;

}

Timer(string time) {

int delimiterPos = time.find(':');

int min = stoi(time.substr(0, delimiterPos));

int sec = stoi(time.substr(delimiterPos + 1));

seconds = min \* 60 + sec;

}

void run() {

cout << "Начинается обратный отсчет таймера!" << endl;

while (seconds > 0) {

cout << seconds << " с" << endl;

seconds--;

clock\_t start\_time = clock();

while (clock() < start\_time + CLK\_TCK);

}

cout << "Время вышло!!!!!!!" << endl;

}

};

int main()

{

setlocale(LC\_ALL, "rus");

int seconds, min;

string time;

short n;

cout << "Выберите действие: \n 1) Ввести начальное значения таймера в секундах \n 2) Ввести начальное значения таймера в минутах и секундах \n 3) Ввести начальное значения таймера в виде строки(0:20) ";

cout << "\nДействие: ";

cin >> n;

if (n == 1) {

cout << "\nВведите начальное значения таймера в секундах: ";

cin >> seconds;

Timer timer1(seconds);

timer1.run();

}

else if (n == 2) {

cout << "\nВведите начальное значения таймера в минутах и секундах: ";

cout << "\nВведите минуту: ";

cin >> min;

cout << "\nВведите секунду: ";

cin >> seconds;

Timer timer2(min, seconds);

timer2.run();

}

else if (n == 3) {

cout << "\nВведите начальное значения таймера в виде строки(0:20): ";

cin >> time;

Timer timer3(time);

timer3.run();

}

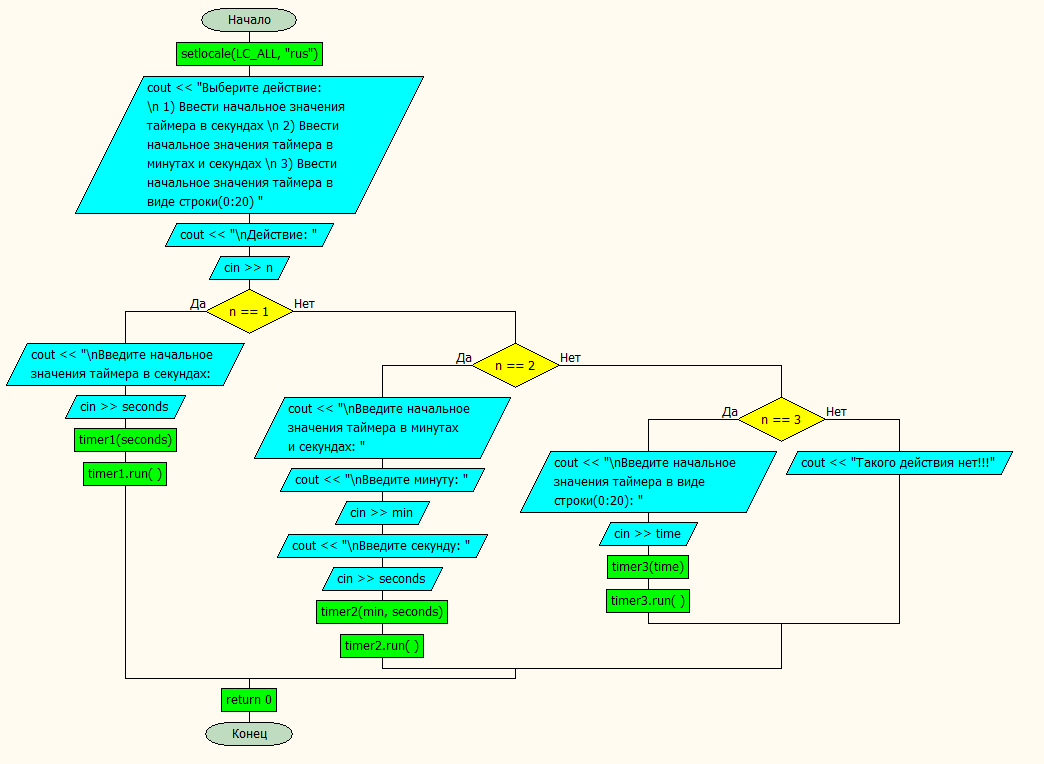
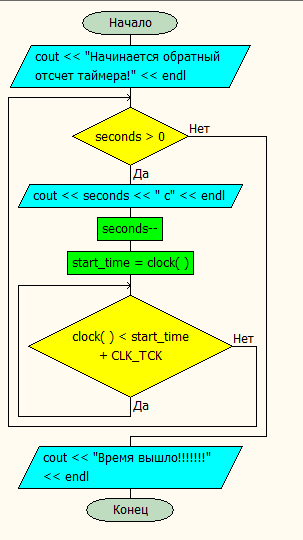
else { cout << "Такого действия нет!!!"; }

return 0;

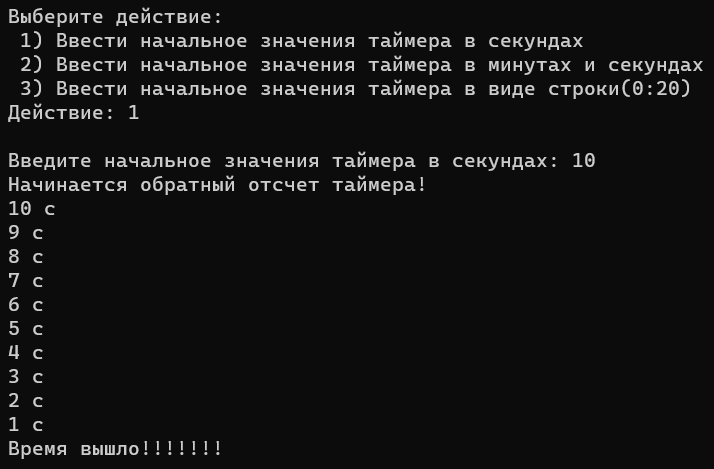
}

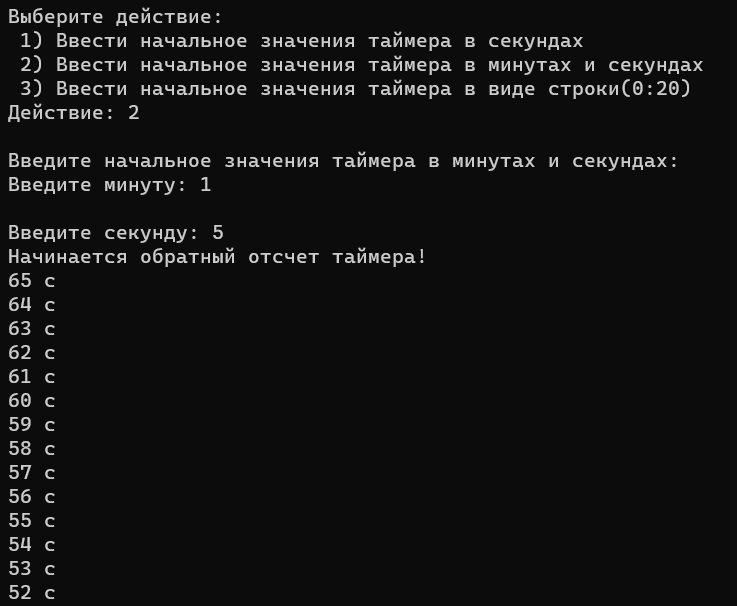
**Блок-схема**

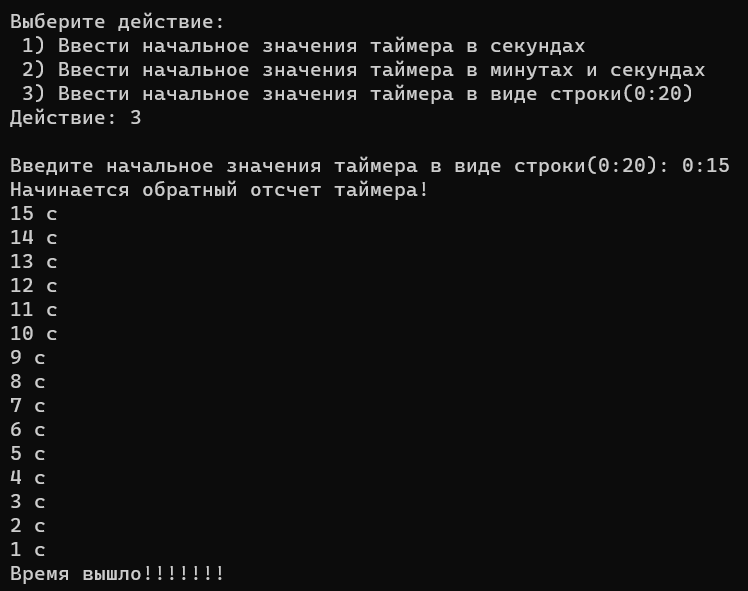
Главная функция main(): Функция run():



**Результат:**







**Задание №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.

**Код программы:**

#include <iostream>

#include <string>

using namespace std;

class Nomenclature {

private:

string itemName;

double wholesalePrice;

double retailMarkup;

int quantity;

public:

Nomenclature(string name, double wholesale, double markup, int qty) :

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

~Nomenclature() {

cout << "\nТовар '" << itemName << "' был удален!!!" << endl;

}

double calculateProfit() {

return quantity \* (retailMarkup \* wholesalePrice);

}

void display() {

cout << "\nНазвания товара: " << itemName << endl;

cout << "Оптовая цена: " << wholesalePrice <<" сом" << endl;

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

cout << "Количество товаров на складе: " << quantity << endl;

cout << "Чистый доход: " << calculateProfit() << " сом" << endl;

}

};

int main() {

setlocale(LC\_ALL, "rus");

Nomenclature item1("Холодильник", 800, 0.2, 10);

item1.display();

Nomenclature item2("Кровать", 489, 0.4, 7);

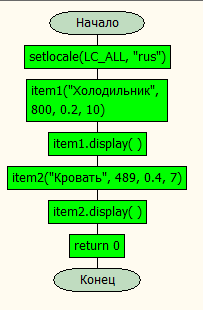
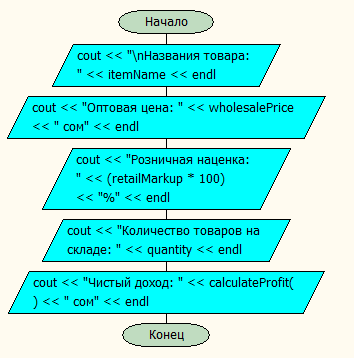
item2.display();

return 0;

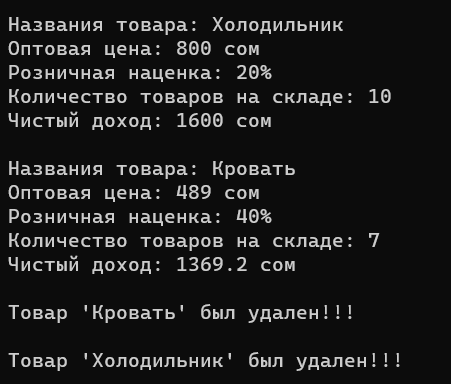
}

**Блок-схема**

Главная функция main(): Функция display():



**Результат:**



**Задание №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.

**Код программы:**

#include <iostream>

#include <string>

#include <ctime>

using namespace std;

class Soft {

private:

string name;

string developer;

float size;

time\_t licenseExpirationDate;

public:

Soft(const string& name, const string& developer, float size, time\_t expirationDate)

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

~Soft() {

cout << "\nПрограммное обеспечение '" << name << "' был удален!!!" << endl;

}

int daysUntilExpiration() const {

time\_t currentTime;

time(&currentTime);

double secondsLeft = difftime(licenseExpirationDate, currentTime);

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

}

void displayInfo() const {

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

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

cout << "Занимаемый объем: " << size << " МБ" << endl;

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

}

};

int main() {

setlocale(LC\_ALL, "rus");

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

Soft software("Paint", "Эрлан", 10.5, expirationDate);

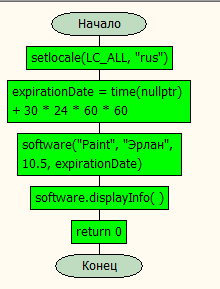
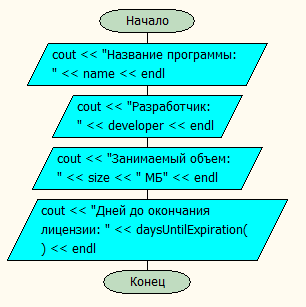
software.displayInfo();

return 0;

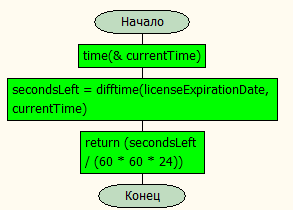
}

**Блок-схема**

Главная функция main(): Функция displayInfo():



Функция daysUntilExpiration():



**Результат:**

