Министерство образования и науки Кыргызской Республики

Кыргызский государственный технический университет

им. И.Раззакова

Институт информационных технологий

Кафедра «Программное обеспечение компьютерных систем»

Направление: 710400 «Программная инженерия»

Дисциплина: «Объектно-ориентированное программирование с++»

**ОТЧЕТ**

По лабораторной работе № 3

Тема: «Конструктор и деструктор»

Выполнила: студент группы

ПИ(б)-2-21 Кадырбек кызы Айзат

Проверила: Мусабев Э.Б.

Бишкек – 2024

**Задания №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:***

#include <iostream>

#include <ctime>

#include <string>

#include <Windows.h>

using namespace std;

class Timer { // Создаем класс Timer

private:

int seconds;

public:

// Конструктор класса Timer с одним аргументом, который принимает количестао секунд

// и использует его для инициализации приватного поля seconds

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

// Конструктор с инициализацией времени в формате "мм:сс"

Timer(const string& timeString) { //принимает время в виде строки, представляющей количество минут и секунд, разделенных символом ':'

size\_t delimiterPos = timeString.find(":");

int minutes = stoi(timeString.substr(0, delimiterPos));

int seconds = stoi(timeString.substr(delimiterPos + 1));

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

}

// Конструктор с инициализацией времени в минутах и секундах

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

// Функция запуска таймера

void run() {

while (seconds > 0) {

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

Sleep(1000);

--seconds; //уменьшает на единицу

}

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

}

};

int main() {

setlocale(LC\_ALL, "rus");

cout << "Выберите способ инициализации таймера:" << endl;

cout << "1. Введите количество секунд: ";

int secondsInput;

cin >> secondsInput;

Timer timer1(secondsInput);

cout << "2. Введите время в формате \"мм:сс\": ";

string timeInput;

cin >> timeInput;

Timer timer2(timeInput);

cout << "3. Введите минуты и секунды: ";

int minutesInput, secondsInput2;

cin >> minutesInput >> secondsInput2;

Timer timer3(minutesInput, secondsInput2);

cout << "\nЗапуск таймера 1:" << endl;

timer1.run();

cout << "\nЗапуск таймера 2:" << endl;

timer2.run();

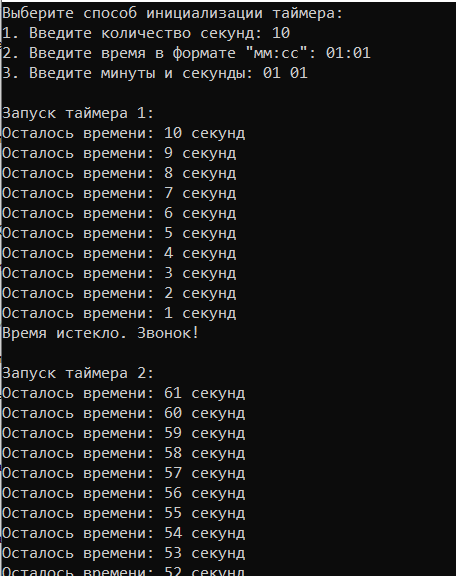
cout << "\nЗапуск таймера 3:" << endl;

timer3.run();

return 0;

}

***Результат:***



**Задания №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:***

#include <iostream>

#include <string>

#include <iomanip>

#include <Windows.h>

using namespace std;

class Nomenclature { // Объявление класса Nomenclature

private:

string productName;

int productOptPrice;

int productRoznPrice;

int productCount;

public:

Nomenclature() : productName(""), productOptPrice(0), productRoznPrice(0), productCount(0) {}

void Output() {

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

cout << "Оптовая цена: " << productOptPrice << endl;

cout << "Розничная цена: " << productRoznPrice << endl;

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

cout << "Чистая прибыль товара: ";

cleanPrice(); cleanPrice(); // Вызов метода расчета чистой прибыли

cout << endl;

}

void OutputNomenclature(short productNumber) {

cout << "| " << productNumber << " | " << setw(19) << productName << " | " << setw(19) << productOptPrice << " | " << setw(21) << productRoznPrice

<< " | " << setw(32) << productCount << " | " << setw(23);

cleanPrice();

cout << " | " << endl;

}

void cleanPrice() {

int OptPrice = productOptPrice \* productCount;

int RozSale = productRoznPrice \* productCount;

int cleanPrice = RozSale - OptPrice;

cout << cleanPrice;

}

void Input() {

cout << "Введите наименование товара: ";

getline(cin, productName);

cout << "Введите оптовую цену товара: ";

cin >> productOptPrice;

cout << "Введите розничную цену товара: ";

cin >> productRoznPrice;

cout << "Введите количество товарных единиц на складе: ";

cin >> productCount;

cin.ignore();

}

~Nomenclature() {

// Деструктор

cout << "Товар " << productName << " удален" << endl;

}

};

int main() {

setlocale(LC\_ALL, "");

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int N = 0; // кол-во записей о товарах

while (N <= 0) {

cout << "Сколько записей о товарах Вы хотите ввести? " << endl;

cin >> N; // Ввод количества записей

if (N <= 0) {

cout << "Ошибка! Значение кол-ва записей должно быть больше 0. Повторите ввод. " << endl;

}

}

Nomenclature\* product = new Nomenclature[N]; // Выделение памяти для массива объектов класса Nomenclature

short choice;

bool flinc = false;

menu:

cout << setw(27) << "-------Меню--------" << endl;

cout << "1 - Заполнить информацию о товарах\n2 - Посмотреть информацию о товарах\n3 - Удалить товар\n4 - Завершить программу\n";

cout << setw(29) << "-------------------\n\n";

cin >> choice;

cin.ignore();

switch (choice) {

case 1:

for (int i = 0; i < N; i++) {

cout << "Пожалуйста, введите информацию о товаре № " << i + 1 << ":\n" << endl;

product[i].Input();

cout << "Итого чистая прибыль товара будет равна: ";

product[i].cleanPrice();

cout << "\nИнформация о товаре № " << i + 1 << " заполнена!\n" << endl;

}

cout << "\nИнформация о " << N << " товарах заполнена!\n" << endl;

flinc = true;

goto menu;

case 2:

system("cls");

if (flinc) {

cout << "| " << "№" << " | " << setw(18) << "Наименование товара" << " | " << setw(19) << "Оптовая цена товара" << " | " << setw(15) << "Розничная цена товара"

<< " | " << setw(18) << "Кол-во товарных единиц на складе" << " | " << setw(21) << "Чистая прибыль с товара" << " | " << endl;

for (int i = 0; i < N; i++) {

product[i].OutputNomenclature(i + 1);

}

cout << endl;

goto menu;

}

else {

cout << "\nВы не ввели информацию о товарах!\n" << endl;

goto menu;

}

case 3:

// Удаление всех объектов перед завершением программы

delete[] product; // освобождаем память

cout << "\nПрограмма завершена!\n";

break;

default:

cout << "Вы выбрали неверный пункт меню!\n" << endl;

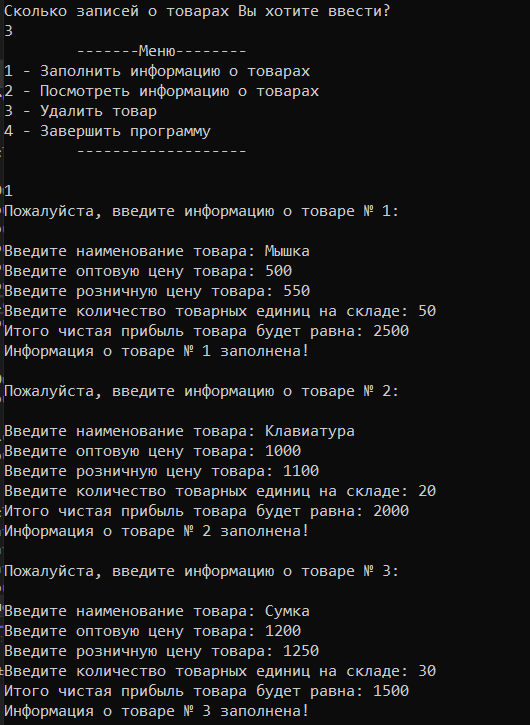
goto menu;

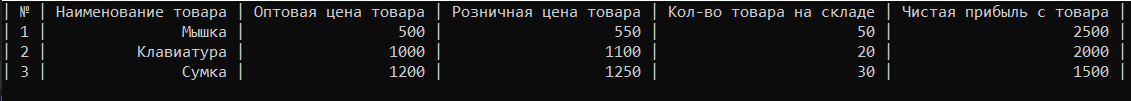
}

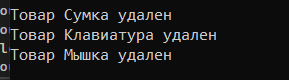
return 0;

}

***Результат:***







**Задания №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:***

#include <iostream>

#include <string>

#include <ctime>

#include <chrono>

#include <iomanip>

#include <sstream>

#include <Windows.h>

using namespace std;

class Soft { // Объявление класса Soft

private:

string name;

string developer;

double memory;

string expirationDate;

public:

// Конструктор по умолчанию

Soft() : name(""), developer(""), memory(0.0), expirationDate("") {}

// Функция для установки информации о программном обеспечении

void setInfo(const string& name, const string& developer, double memory, const string& expirationDate) {

this->name = name;

this->developer = developer;

this->memory = memory;

this->expirationDate = expirationDate;

}

// Функция для подсчета количества дней до завершения лицензии

int daysUntilExpiration() {

// Получаем текущую дату

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

time\_t now\_c = chrono::system\_clock::to\_time\_t(now);

// Разбиваем дату завершения лицензии на компоненты (день, месяц, год)

stringstream ss(expirationDate);

int day, month, year;

char dot;

ss >> day >> dot >> month >> dot >> year;

// Формируем объект времени для даты завершения лицензии

tm expDate = {};

expDate.tm\_mday = day;

expDate.tm\_mon = month - 1; // Месяцы начинаются с 0

expDate.tm\_year = year - 1900; // Годы отсчитываются от 1900

// Преобразуем даты в time\_point

auto expirationTime = chrono::system\_clock::from\_time\_t(mktime(&expDate));

// Вычисляем разницу во времени между текущей датой и датой завершения лицензии

auto difference = expirationTime - now;

// Переводим разницу в количество дней и округляем до целого значения

return chrono::duration\_cast<chrono::hours>(difference).count() / 24;

}

// Функция для вывода данных

void displayInfoTableFormat(int index) {

cout << left << setw(8) << index << "|";

cout << setw(19) << name << "|";

cout << setw(19) << developer << "|";

cout << setw(13) << memory << " Мб" << "|";

cout << setw(15) << expirationDate << "|";

cout << endl;

}

// Деструктор

~Soft() {

cout << "Программное обеспечение " << name << " удалено" << endl;

}

};

int main() {

setlocale(LC\_ALL, "rus");

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int numSoftware; // Переменная для хранения количества программного обеспечения

cout << "Сколько программного обеспечения вы хотите ввести? ";

cin >> numSoftware;

if (numSoftware <= 0) {

cout << "Некорректное количество, ошибка!" << endl;

return 0;

}

// Создаем массив объектов класса Soft

Soft\* softwareArray = new Soft[numSoftware];

for (int i = 0; i < numSoftware; ++i) {

string name, developer, expirationDate;

double memory;

cout << "\nВведите информацию о программном обеспечении №" << i + 1 << endl;

cout << "Введите название программы: ";

cin.ignore();

getline(cin, name);

cout << "Введите разработчика: ";

getline(cin, developer);

cout << "Введите занимаемый объем (Мб): ";

cin >> memory;

cout << "Введите дату завершения лицензии (дд.мм.гггг): ";

cin >> expirationDate;

softwareArray[i].setInfo(name, developer, memory, expirationDate);

}

int choice;

do {

cout << "\nМеню:\n";

cout << "1. Показать информацию о программном обеспечении\n";

cout << "2. Показать количество дней до завершения лицензии для всех программ\n";

cout << "3. Удалить программное обеспечение\n";

cout << "4. Выйти\n";

cout << "Выберите действие: ";

cin >> choice;

switch (choice) {

case 1:

cout << left << setw(8) << "№" ;

cout << setw(20) << "Название";

cout << setw(20) << "Разработчик";

cout << setw(20) << "Объем";

cout << setw(15) << "Дата окончания";

cout << endl;

for (int i = 0; i < numSoftware; ++i) {

softwareArray[i].displayInfoTableFormat(i + 1);

}

break;

case 2:

// Выводим количество дней до истечения лицензии для каждого программного обеспечения

for (int i = 0; i < numSoftware; ++i) {

cout << "Осталось " << softwareArray[i].daysUntilExpiration() << " дней до завершения лицензии для ПО № "<< i+1 <<endl;

}

break;

case 3:

// Удаление всех объектов перед завершением программы

delete[] softwareArray; // освобождаем память

cout << "\nПрограмма завершена!\n"; // Вывод сообщения о завершении программы

break;

default:

cout << "Некорректный ввод. Попробуйте снова.\n";

break;

}

} while (choice != 4);

return 0;

}

***Результат:***

