Министерство образования и науки Кыргызской Республики

Кыргызский государственный технический университет

им. И.Раззакова

Факультет информационных технологий

Кафедра «Программное обеспечение компьютерных систем»

Направление: 710400 «Программная инженерия»

Дисциплина: «Объектно – ориентированное программирование»

**ОТЧЕТ**

По лабораторной работе №3.

Тема: «Конструктор и деструктор»

                                                                       Выполнила: студент группы

ПИ(б)-2-21 Марат кызы Жылдыз

Проверила: Мусабаев Э. Б.

**Бишкек – 2024**

**Лабораторная работа №3.**

**Тема: Конструктор и деструктор.**

**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 программы:**

#include<iostream>

#include<Windows.h>

#include<stdlib.h>

#include<time.h>

using namespace std;

class timer

{

int seconds;

public:

timer(char\* t)

{

seconds = atoi(t);

}

timer(int t)

{

seconds = t;

}

timer(int min, int sec)

{

seconds = min \* 60 + sec;

}

void run()

{

clock\_t t1, t2;

t1 = t2 = clock() / CLK\_TCK;

while (seconds) {

if (t1 / CLK\_TCK + 1 <= (t2 = clock()) / CLK\_TCK) {

seconds--;

t1 = t2;

cout << seconds << " ";

}

}

cout << "\n\a";

}

};

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

timer a(10), b(20), c(1, 30);

cout << "Обратный отсчет 10 секунд :" << endl;

a.run();

cout << "Обратный отсчет 20 секунд :" << endl;

b.run();

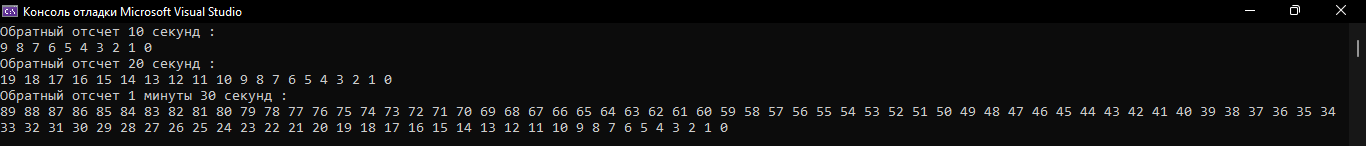
cout << "Обратный отсчет 1 минуты 30 секунд :" << endl;

c.run();

return 0;

}

**Результат программы:**



**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<Windows.h>

#include<iomanip>

using namespace std;

class Nomenclature

{

private:

// Закрытые поля класса

string Nameproduct;

int Oprice;

int Rprice;

int Kproduct = 0;

int kol = 0;

public:

// Инициализация поля класса нулями

Nomenclature() :Nameproduct(""), Oprice(0), Rprice(0), Kproduct(0)

{}

// Метод ввода данных

void input()

{

cout << "Название товара: ";

cin >> Nameproduct;

cout << "Оптовая цена (сом): ";

cin >> Oprice;

cout << "Розничная наценка (сом): ";

cin >> Rprice;

cout << "Количество товара: " << endl;

cin >> Kproduct;

kol++;

}

// Деструктор

~Nomenclature()

{

cout << endl << "Успешно выполнено!" << endl;

}

// Метод для нахождения чистого дохода от каждого товара

void Searchprice(int n, int\* Sum)

{

float Ch\_O = 0, Ch\_R = 0, Val\_R = 0, Val\_O = 0;

// Умножаем оптовую цену и количество товара

Val\_O = Oprice \* Kproduct;

Val\_R = Rprice \* Kproduct;

// От полученного результата отнимаем налоговый платеж.

Ch\_O = Val\_O - n;

Ch\_R = Val\_R - n;

// Вывод дохода

cout << "Доход от товара (Оптовой цены): " << "'" << Nameproduct << "'" << ", составляет : " << Ch\_O << endl;

cout << "Доход от товара (Розничной цены): " << "'" << Nameproduct << "'" << ", составляет : " << Ch\_R << endl;

// Записываем доход в переменную

\*Sum = Ch\_O;

\*Sum = Ch\_R;

}

// Вывод данных

void Show()

{

cout << "| " << setw(9) << Nameproduct << setw(8) << "|" << setw(8) << Oprice << setw(7) << "|" << setw(10) << Rprice << setw(10) << "|" << setw(10) << Kproduct << setw(10) << "|" << endl;

cout << "\*-----------------|--------------|-------------------|-------------------\*" << endl;

cout << endl;

}

};

int main()

{

// Кириллица

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int val = 0, k = 0;

int Sum = 0;

float Sum2 = 0;

// Количество элементов

Nomenclature Obj[10];

while (val != 4)

{

system("CLS");

cout << "\t\tМеню" << endl << endl;

cout << "\t1. Ввод данных" << "\n\t2. Вывод данных" << "\n\t3. Вывод дохода" << "\n\t4. Выход" << endl << endl;

cout << "Введите номер операции: ";

cin >> val;

if (val == 1)

{

Obj[k].input();

k++;

cout << "Данные записаны!" << endl;

system("pause");

}

else if (val == 2)

{

system("CLS");

cout << "\*------------------------------------------------------------------------\*" << endl;

cout << "|\t\t\t\tВсе товары |" << endl;

cout << "\*------------------------------------------------------------------------\*" << endl;

cout << "| Название товара | Оптовая цена | Розничная наценка | Количество товара |" << endl;

cout << "\*-----------------|--------------|-------------------|-------------------\*" << endl;

for (int i = 0; i < k; i++)

{

Obj[i].Show(); // Вывод данных

}

system("pause");

}

else if (val == 3)

{

int n = 0;

system("CLS");

cout << "Введите сумму налогового платежа: ";

cin >> n;

for (int i = 0; i < k; i++)

{

// Вызываем функция с двумя параметрами.

Obj[i].Searchprice(n, &Sum);

// Суммируем доходы для вычисления общего дохода от товаров

Sum2 += Sum;

}

// Переменную приравниваем к нулю

Sum2 = 0;

system("pause");

}

}

system("cls");

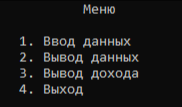
cout << "До свидания!" << endl;

system("pause");

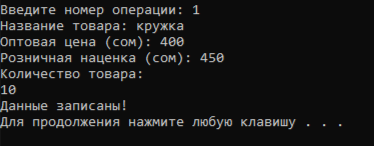
}

**Результат программы:**

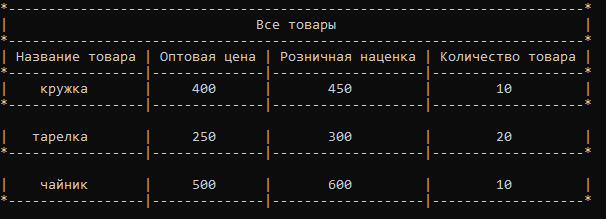
**Рис.1.Меню.**



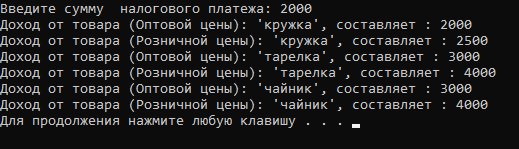
**Рис.2. Ввод данных.**



**Рис.3. Вывод данных.**



**Рис.4. Подсчет доходов.**



**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<Windows.h>

#include<iomanip>

#include<ctime>

#include<cstdlib>

#include <string>

#include <vector>

using namespace std;

class Soft

{

private:

// Закрытые поля класса

string SName;

string DName;

string SSize;

int Day;

int Month;

int Year;

public:

// Инициализируем поля класса нулями(конструктор по умолчанию)

Soft() : SName(""), DName(""), SSize(""), Month(0), Day(0), Year(0) {}

// Функция ввода

void input()

{

cout << "Введите название программы: ";

getline(cin >> ws, SName); // Ввод названия программы с учетом пробелов

cout << "Введите наименование разработчика: ";

getline(cin >> ws, DName); // Ввод наименования разработчика с учетом пробелов

cout << "Введите занимаемый объем: ";

getline(cin >> ws, SSize); // Ввод занимаемого объема с учетом пробелов

f:cout << "Введите дату завершения лицензии \*День, Месяц, Год\*: ";

// Ввод даты

cin >> Day >> Month >> Year;

// Проверка правильности введенных данных

if (Day > 31 || Month > 12)

{

// Если не правильно, то выводим сообщение

cout << "Введите день, не больше 31, месяц не больше 12!" << endl;

goto f;

}

}

// Функция для вычисления количества дней до завершения лицензии

int daysUntilExpiration()

{

// Получаем текущую дату

time\_t currentTime;

time(&currentTime);

tm localTime;

localtime\_s(&localTime, &currentTime);

int currentDay = localTime.tm\_mday;

int currentMonth = localTime.tm\_mon + 1; // tm\_mon начинается с 0

int currentYear = localTime.tm\_year + 1900; // tm\_year возвращает год с 1900

// Преобразуем дату окончания лицензии и текущую дату в количество дней с начала эпохи

time\_t expirationTime = dateSoft(Day, Month, Year);

time\_t currentTimeEpoch = dateSoft(currentDay, currentMonth, currentYear);

// Разница во времени в секундах

double differenceInSeconds = difftime(expirationTime, currentTimeEpoch);

// Переводим разницу в днях

int daysDifference = differenceInSeconds / (24 \* 60 \* 60);

// Возвращаем количество дней до истечения лицензии

return daysDifference;

}

// Функция для получения даты из введенной даты

time\_t dateSoft(int day, int month, int year)

{

tm tmp = tm();

tmp.tm\_mday = day;

tmp.tm\_mon = month - 1; // tm\_mon начинается с 0

tmp.tm\_year = year - 1900; // tm\_year возвращает год с 1900

return mktime(&tmp);

}

// Вывод данных о программном обеспечении

void Show()

{

cout << "\*--------------------------------------------------------------------------\*" << endl;

cout << "| Название программы | Имя разработчика | Занимаемый объем | Дата лицензии |" << endl;

cout << "\*--------------------|------------------|------------------|---------------\*" << endl;

cout << "| " << SName << setw(17) << "| " << DName << setw(10) << "| " << SSize << setw(16) << "| ";

// Условия для корректного вывода даты

if (Day < 10)

{

cout << "0" << Day << ".";

}

else

{

cout << Day << ".";

}

if (Month < 10)

{

cout << "0" << Month << ".";

}

else

{

cout << Month << ".";

}

cout << Year << setw(5) << "|";

cout << endl << endl;

}

};

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

// Создаем вектор для хранения объектов программного обеспечения

vector<Soft> softwareList;

int value = 0;

while (value != 4)

{

system("CLS");

cout << "\tМеню" << endl;

cout << "1) Ввод ПО" << "\n2) Вывод ПО" << "\n3) Вычислить количество дней до завершения лицензии" << endl;

cout << endl << "Введите номер операции: ";

cin >> value;

if (value == 1)

{

system("CLS");

Soft prog;

prog.input();

softwareList.push\_back(prog); // Добавляем объект программного обеспечения в вектор

cout << "Данные записаны!" << endl;

system("pause");

system("CLS");

}

else if (value == 2)

{

system("CLS");

cout << "\*--------------------------------------------------------------------------\*" << endl;

cout << "| Сведения о программном обеспечении |" << endl;

// Вывод информации о всех объектах программного обеспечения в векторе

for (size\_t i = 0; i < softwareList.size(); ++i)

{

softwareList[i].Show();

}

system("pause");

system("CLS");

}

else if (value == 3)

{

system("CLS");

cout << "Вычисление количества дней до завершения лицензии для всех ПО:" << endl;

for (size\_t i = 0; i < softwareList.size(); ++i)

{

int days = softwareList[i].daysUntilExpiration();

cout << "Для ПО с индексом " << i << ": " << days << " дней" << endl;

}

system("pause");

system("CLS");

}

}

system("CLS");

cout << "До свидания!" << endl;

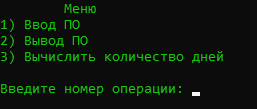
system("pause");

return 0;

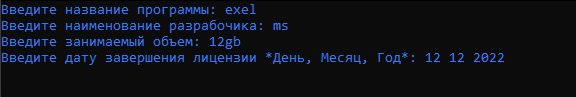
}

**Результат программы:**

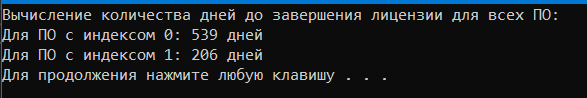
**Рис.1. Меню.**

****

**Рис.2. Ввод данных.**

****

**Рис.3.Вывод данных.**

****

**Рис.4. Подсчет количество дней.**