**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Программирование на основе классов и шаблонов»

Отчет по домашнему заданию



|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-21Б |  | преподаватель каф. ИУ5 |
| Михалёв Я.М. |  | Козлов А.Д. |
| Подпись и дата: |  | Подпись и дата: |
|  |  |  |

Москва, 2022 г.

**Постановка задачи**

На телефонной станции картотека абонентов, содержащая сведения о телефонах и их владельцах, организована в виде линейного списка и хранится в файле.

Написать программу, которая:

* обеспечивает начальное формирование картотеки
* производит вывод всей картотеки
* вводит номер телефона, дату и время разговора
* выводит извещение на оплату телефонного разговора

Программа должна обеспечивать диалог с помощью меню и контроль ошибок при вводе

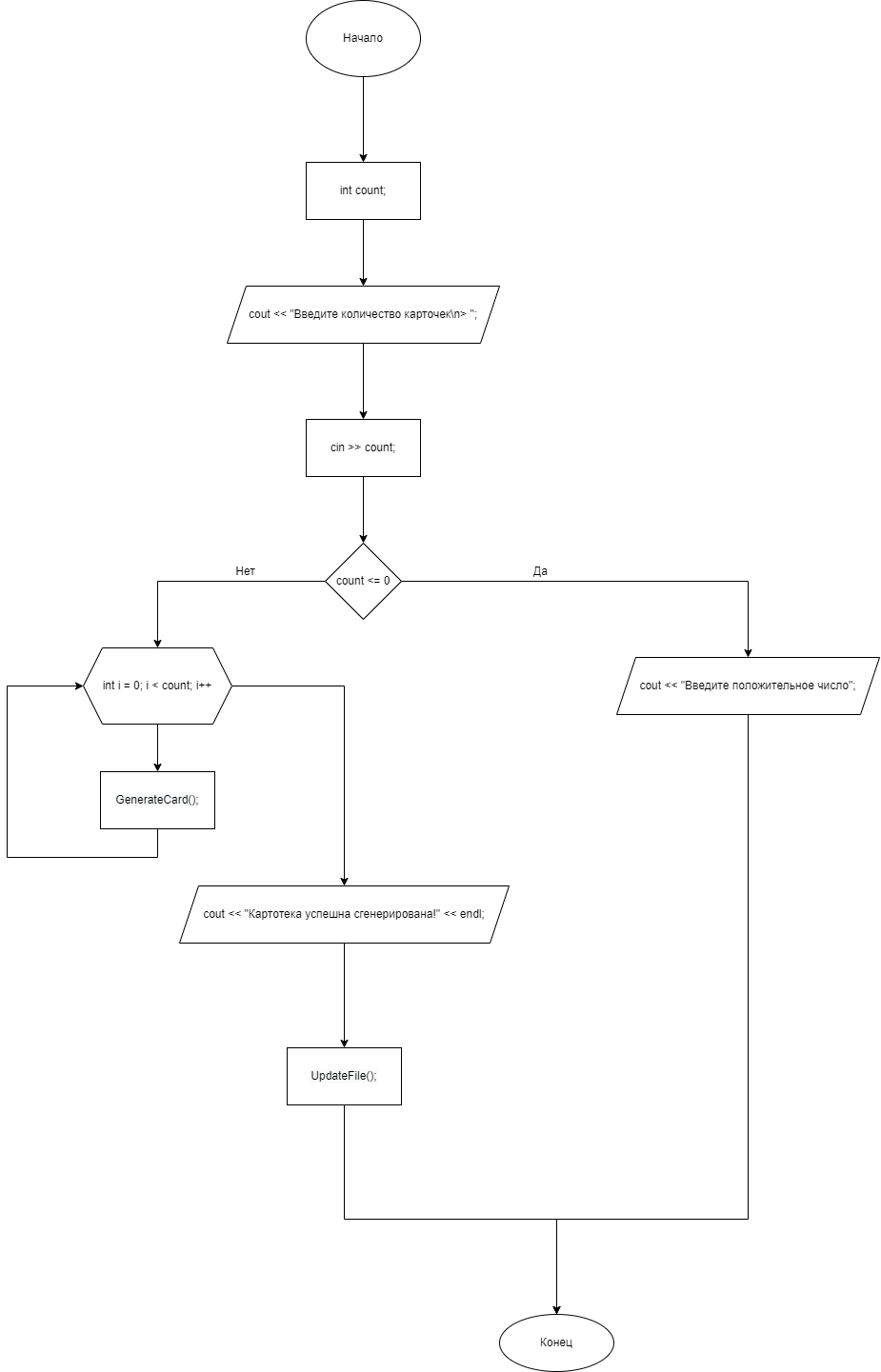
**Разработка алгоритма**

Класс Card

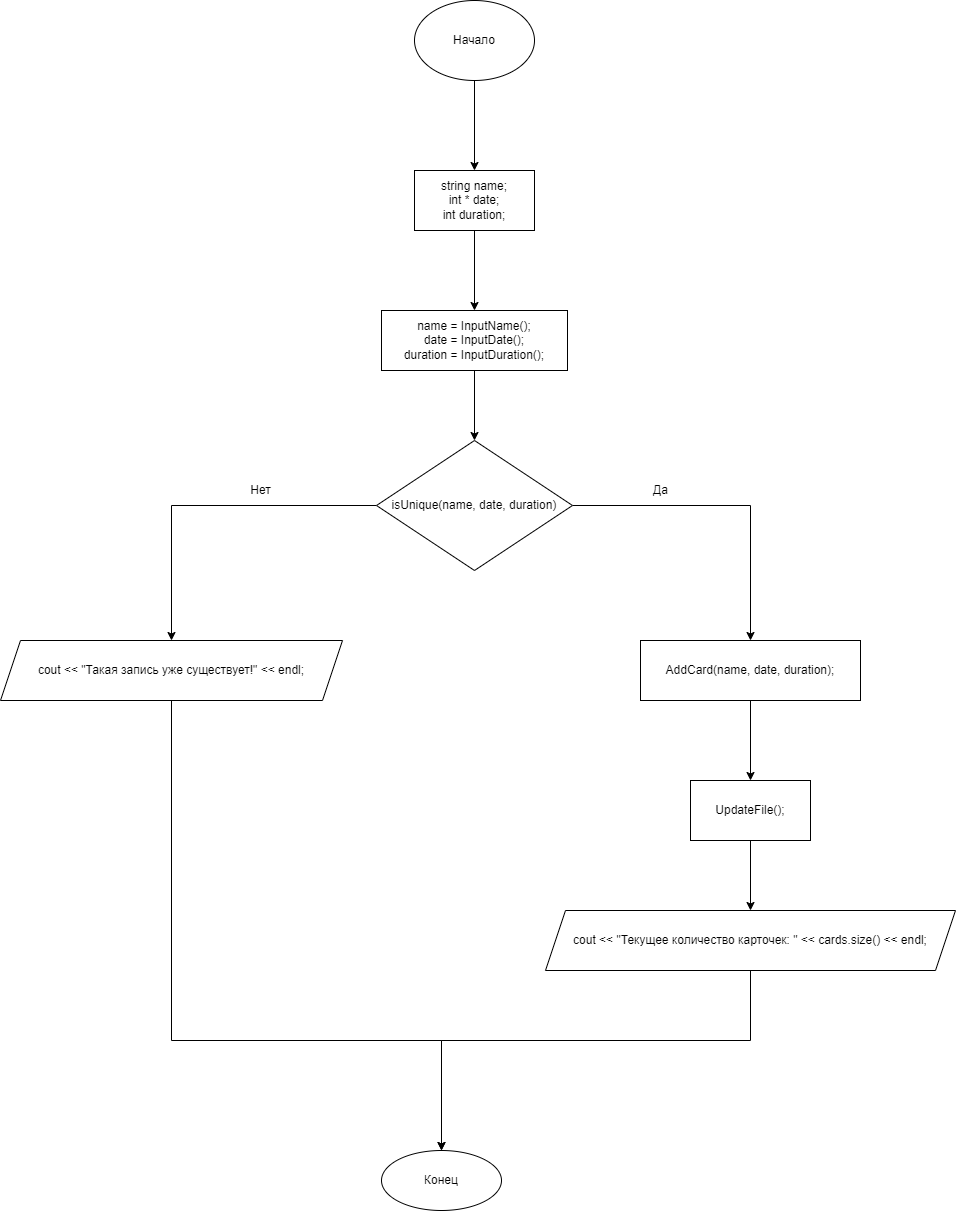
* **Члены - данные (private)**
  + string name– имя абонента
  + int\* date = new int[3] - дата разговора
  + int time– время разговора
* **Конструкторы и деструктор**
  + Конструктор с параметрами string name, int date[], int time
* **Методы**
  + string getName() – возвращает имя абонента
  + int\* getDate() - возвращает дату разговора
  + int getTime() – возвращает время разговора
  + void ShowNotification () – выводит извещение на оплату телефона
* **Перегруженные операторы**
  + “<<” - оператор вывода в консоль

**Схема алгоритма**

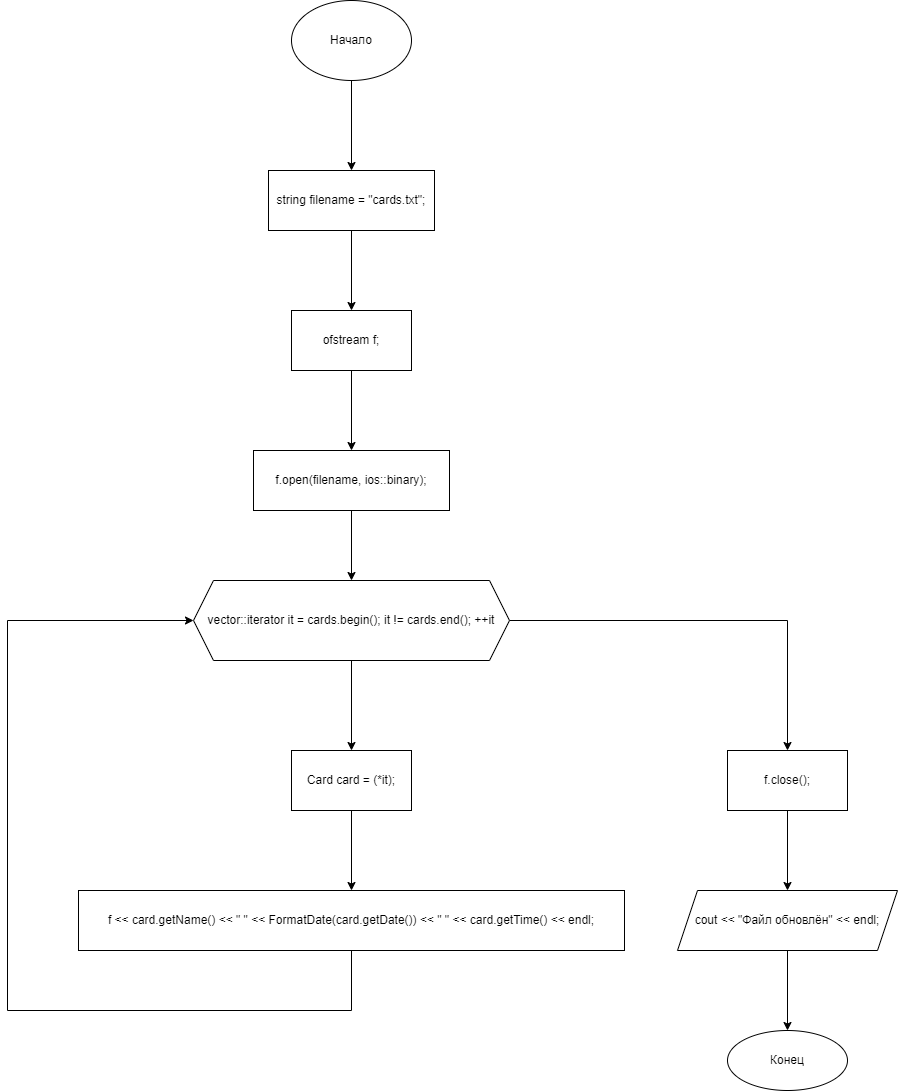
GenerateCards



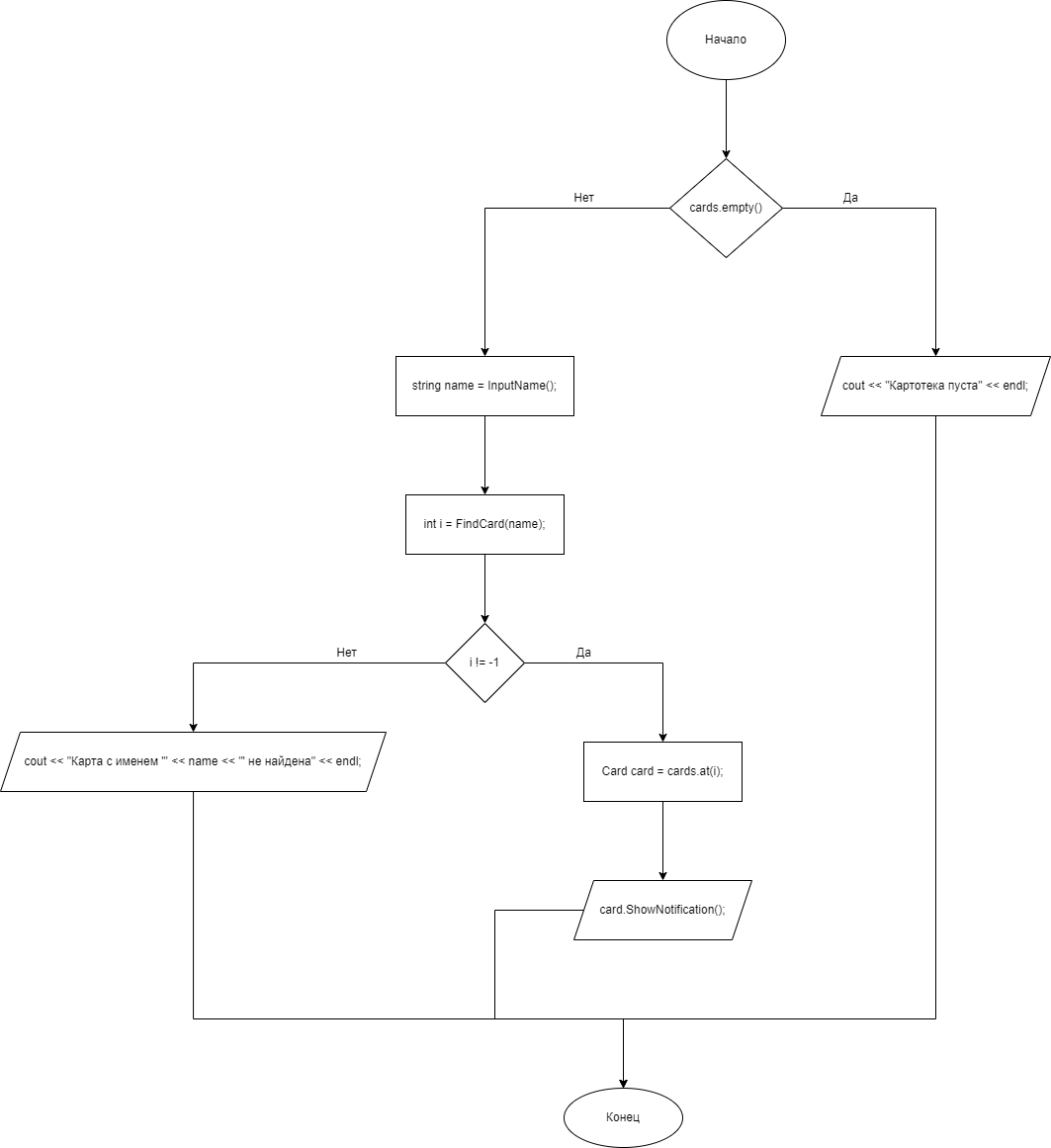
InputCard



UpdateFile



ShowNotification



**Текст программы**

Card.h

#include <iostream>

#include <iomanip>

#include <fstream>

#include <vector>

#include <string>

#include "Utils.h"

class Card

**{**

private**:**

std**::**string name**;**

int**\*** date **=** **new** int**[**3**];**

int time**;**

public**:**

Card**(**std**::**string name**,** int date**[],** int time**)**

**{**

**this->**name **=** name**;**

**this->**date**[**0**]** **=** date**[**0**];**

**this->**date**[**1**]** **=** date**[**1**];**

**this->**date**[**2**]** **=** date**[**2**];**

**this->**time **=** time**;**

**}**

std**::**string getName**()** **{** **return** **this->**name**;** **}**

int**\*** getDate**()** **{** **return** **this->**date**;** **}**

int getTime**()** **{** **return** **this->**time**;** **}**

void ShowNotification**()**

**{**

float price **=** **this->**time **\*** 2.5 **/** 60 **\*** 10**;**

std**::**cout **<<** std**::**fixed **<<** std**::**setprecision**(**2**)** **<<** "Абонент " **<<** **this->**name **<<** " должен заплатить " **<<** price **<<** " рублей за разговор " **<<** FormatDate**(this->**date**)** **<<** std**::**endl**;**

**};**

friend void **operator<<** **(**std**::**ostream**&** out**,** Card card**)**

**{**

out **<<** card**.**name **<<** " " **<<** FormatDate**(**card**.**date**)** **<<** " " **<<** FormatTime**(**card**.**time**)** **<<** std**::**endl**;**

**};**

**};**

Utils.h

#pragma once

#include <string>

std**::**string FormatDate**(**int**\*** date**);**

std**::**string FormatTime**(**int time**);**

std**::**string GetRandomName**();**

std**::**string InputName**();**

int**\*** GetDate**(**std**::**string raw**);**

int**\*** InputDate**();**

int InputDuration**();**

bool CheckDate**(**std**::**string raw**);**

int GetRandomInt**(**int min**,** int max**);**

int**\*** GetRandomDate**();**

Utils.cpp

#include "Utils.h"

#include <string>

#include <vector>

#include <iostream>

**using** **namespace** std**;**

string FormatDate**(**int**\*** date**)**

**{**

**return** to\_string**(**date**[**0**])** **+** "." **+** to\_string**(**date**[**1**])** **+** "." **+** to\_string**(**date**[**2**]);**

**}**

string FormatTime**(**int time**)**

**{**

string res**;**

**if** **(**time **>=** 3600**)** res **+=** to\_string**(**time **/** 3600**)** **+** "h "**;**

**if** **(**time **>=** 60**)** res **+=** to\_string**(**time **/** 60 **%** 60**)** **+** "m "**;**

**if** **(**time **>=** 0**)** res **+=** to\_string**(**time **%** 60**)** **+** "s"**;**

**return** res**;**

**}**

int**\*** GetDate**(**string raw**)**

**{**

int**\*** date **=** **new** int**[**3**];**

int day**,** month**,** year**;**

int pos **=** 0**;**

pos **=** raw**.**find**(**"."**);**

day **=** stoi**(**raw**.**substr**(**0**,** pos**));**

raw**.**erase**(**0**,** pos **+** 1**);**

pos **=** raw**.**find**(**"."**);**

month **=** stoi**(**raw**.**substr**(**0**,** pos**));**

raw**.**erase**(**0**,** pos **+** 1**);**

pos **=** raw**.**find**(**"."**);**

year **=** stoi**(**raw**.**substr**(**0**,** pos**));**

date**[**0**]** **=** day**;**

date**[**1**]** **=** month**;**

date**[**2**]** **=** year**;**

**return** date**;**

**}**

string GetRandomName**()**

**{**

vector**<**string**>** names**{** "John"**,** "Robert"**,** "John"**,** "Mary"**,** "Patricia"**,** "Jennifer" **};**

vector**<**string**>** surnames**{** "Smith"**,** "Johnson"**,** "Williams"**,** "Brown"**,** "James"**,** "Miller" **};**

int i **=** rand**()** **%** names**.**size**();**

string name **=** names**[**i**];**

int j **=** rand**()** **%** surnames**.**size**();**

string surname **=** surnames**[**j**];**

string fullname **=** name **+** " " **+** surname**;**

**return** fullname**;**

**}**

int GetRandomInt**(**int min**,** int max**)**

**{**

**return** min **+** **(**rand**()** **%** **static\_cast<**int**>(**max **-** min **+** 1**));**

**}**

int **\*** GetRandomDate**()**

**{**

int **\*** test **=** **new** int**[**3**];**

test**[**0**]** **=** GetRandomInt**(**1**,** 31**);**

test**[**1**]** **=** GetRandomInt**(**1**,** 12**);**

test**[**2**]** **=** GetRandomInt**(**1900**,** 2022**);**

**return** test**;**

**}**

bool CheckDate**(**string raw**)**

**{**

int day**,** month**,** year**;**

int pos **=** 0**;**

pos **=** raw**.**find**(**"."**);**

day **=** stoi**(**raw**.**substr**(**0**,** pos**));**

raw**.**erase**(**0**,** pos **+** 1**);**

pos **=** raw**.**find**(**"."**);**

month **=** stoi**(**raw**.**substr**(**0**,** pos**));**

raw**.**erase**(**0**,** pos **+** 1**);**

pos **=** raw**.**find**(**"."**);**

year **=** stoi**(**raw**.**substr**(**0**,** pos**));**

**return** **(**1 **<=** day **&&** day **<=** 31**)** **&&**

**(**1 **<=** month **&&** month **<=** 12**)** **&&**

**(**1 **<=** year **&&** year **<=** 2022**)** **?** **true** **:** **false;**

**}**

string InputName**()**

**{**

string name**;**

cout **<<** "Введите имя абонента\n> "**;**

cin**.**ignore**(**1000**,** '\n'**);**

getline**(**cin**,** name**);**

**return** name**;**

**}**

int**\*** InputDate**()**

**{**

string raw**;**

cout **<<** "Введите дату разговора\n> "**;**

cin **>>** raw**;**

**while** **(!**CheckDate**(**raw**))**

**{**

cout **<<** "Введена некорректная дата" **<<** endl**;**

cout **<<** "Введите дату разговора\n> "**;**

cin **>>** raw**;**

**}**

**return** GetDate**(**raw**);**

**}**

int InputDuration**()**

**{**

int duration**;**

cout **<<** "Введите время разговора\n> "**;**

cin **>>** duration**;**

**while** **(**duration **<** 0**)**

**{**

cout **<<** "Время разговора должно быть неотрицательным числом" **<<** endl**;**

cout **<<** "Введите время разговора\n> "**;**

cin **>>** duration**;**

**}**

**return** duration**;**

**}**

Main.cpp

#include <iostream>

#include <iomanip>

#include <fstream>

#include <vector>

#include <string>

#include "Utils.h"

#include "Card.h"

**using** **namespace** std**;**

vector**<**Card**>** cards**;**

void PrintCards**()**

**{**

**if** **(**cards**.**empty**())**

**{**

cout **<<** "Картотека пуста" **<<** endl**;**

**}**

**else**

**{**

cout **<<** "| Имя | Дата разговора | Время разговора |" **<<** endl**;**

**for** **(**vector**<**Card**>::**iterator it **=** cards**.**begin**();** it **!=** cards**.**end**();** **++**it**)**

**{**

Card card **=** **(\***it**);**

cout**.**width**(**20**);**

cout **<<** card**.**getName**();**

cout**.**width**(**20**);**

cout **<<** FormatDate**(**card**.**getDate**());**

cout**.**width**(**20**);**

cout **<<** FormatTime**(**card**.**getTime**());**

cout**.**width**(**20**);**

cout **<<** "\n"**;**

**}**

**}**

**}**

void UpdateFile**()**

**{**

string filename **=** "cards.txt"**;**

ofstream f**;**

f**.**open**(**filename**,** ios**::**binary**);**

**for** **(**vector**<**Card**>::**iterator it **=** cards**.**begin**();** it **!=** cards**.**end**();** **++**it**)**

**{**

Card card **=** **(\***it**);**

f **<<** card**.**getName**()** **<<** " " **<<** FormatDate**(**card**.**getDate**())** **<<** " " **<<** card**.**getTime**()** **<<** endl**;**

**}**

f**.**close**();**

cout **<<** "Файл обновлён" **<<** endl**;**

**}**

void AddCard**(**string name**,** int date**[**3**],** int duration**)**

**{**

Card card**(**name**,** date**,** duration**);**

cards**.**push\_back**(**card**);**

**}**

bool isUnique**(**string name**,** string date**,** int duration**)**

**{**

**for** **(**auto it **=** cards**.**begin**();** it **!=** cards**.**end**();** **++**it**)**

**{**

Card card **=** **(\***it**);**

**if** **(**card**.**getName**()** **==** name **&&** FormatDate**(**card**.**getDate**())** **==** date **&&** card**.**getTime**()** **==** duration**)** **return** **false;**

**}**

**return** **true;**

**}**

bool isUnique**(**string name**,** int**\*** date**,** int duration**)**

**{**

**for** **(**auto it **=** cards**.**begin**();** it **!=** cards**.**end**();** **++**it**)**

**{**

Card card **=** **(\***it**);**

**if** **(**card**.**getName**()** **==** name **&&** FormatDate**(**card**.**getDate**())** **==** FormatDate**(**date**)** **&&** card**.**getTime**()** **==** duration**)** **return** **false;**

**}**

**return** **true;**

**}**

void InputCard**()**

**{**

string name**;**

int **\*** date**;**

int duration**;**

name **=** InputName**();**

date **=** InputDate**();**

duration **=** InputDuration**();**

**if** **(**isUnique**(**name**,** date**,** duration**))**

**{**

AddCard**(**name**,** date**,** duration**);**

UpdateFile**();**

cout **<<** "Текущее количество карточек: " **<<** cards**.**size**()** **<<** endl**;**

**}**

**else**

**{**

cout **<<** "Такая запись уже существует!" **<<** endl**;**

**}**

**}**

int FindCard**(**string name**)**

**{**

int i **=** 0**;** // counter.

**for** **(**auto it **=** cards**.**begin**();** it **!=** cards**.**end**();** it**++,** i**++)**

**{**

**if** **(**cards**.**at**(**i**).**getName**()** **==** name**)**

**{**

**return** i**;**

**}**

**}**

**return** **-**1**;**

**}**

void LoadCards**()**

**{**

string line**;**

string name**;**

string surname**;**

string fullname**;**

int **\*** date**;**

int duration**;**

string filename **=** "cards.txt"**;**

ifstream file**(**filename**,** ios**::**binary**);**

int pos **=** 0**;**

**while** **(**file**)**

**{**

getline**(**file**,** line**);**

**if** **(**line**.**empty**())** **break;**

pos **=** line**.**find**(**" "**);**

name **=** line**.**substr**(**0**,** pos**);**

line**.**erase**(**0**,** pos **+** 1**);**

pos **=** line**.**find**(**" "**);**

surname **=** line**.**substr**(**0**,** pos**);**

fullname **=** name **+** " " **+** surname**;**

line**.**erase**(**0**,** pos **+** 1**);**

pos **=** line**.**find**(**" "**);**

string raw **=** line**.**substr**(**0**,** pos**);**

date **=** GetDate**(**raw**);**

line**.**erase**(**0**,** pos **+** 1**);**

pos **=** line**.**find**(**" "**);**

duration **=** stoi**(**line**.**substr**(**0**,** pos**));**

**if** **(**isUnique**(**fullname**,** raw**,** duration**))** AddCard**(**fullname**,** date**,** duration**);**

**}**

file**.**close**();**

cout **<<** "Картотека обновлена" **<<** endl**;**

cout **<<** "Текущее количество карточек: " **+** to\_string**(**cards**.**size**())** **<<** endl**;**

**}**

Card GetRandomCard**()**

**{**

int random **=** rand**()** **%** cards**.**size**();**

Card card **=** cards**[**random**];**

**return** card**;**

**}**

void GenerateCard**()**

**{**

string name **=** GetRandomName**();**

int**\*** date **=** GetRandomDate**();**

int duration **=** GetRandomInt**(**1**,** 600**);**

AddCard**(**name**,** date**,** duration**);**

**}**

void GenerateCards**()**

**{**

int count**;**

cout **<<** "Введите количество карточек\n> "**;**

cin **>>** count**;**

**if** **(**count **<=** 0**)**

**{**

cout **<<** "Введите положительное число"**;**

**}**

**else**

**{**

**for** **(**int i **=** 0**;** i **<** count**;** i**++)**

**{**

GenerateCard**();**

**}**

cout **<<** "Картотека успешна создана!" **<<** endl**;**

UpdateFile**();**

**}**

**}**

void ShowNotification**()**

**{**

**if** **(**cards**.**empty**())**

**{**

cout **<<** "Картотека пуста" **<<** endl**;**

**}**

**else**

**{**

string name **=** InputName**();**

int i **=** FindCard**(**name**);**

**if** **(**i **!=** **-**1**)**

**{**

Card card **=** cards**.**at**(**i**);**

card**.**ShowNotification**();**

**}**

**else**

**{**

cout **<<** "Карта с именем '" **<<** name **<<** "' не найдена" **<<** endl**;**

**}**

**}**

**}**

int Menu**()**

**{**

int choose**;**

cout **<<** "\n\n\nВведите число от 1 до 6: "**;**

**if** **(**cin **>>** choose**)**

**{**

**}**

**else**

**{**

cin**.**clear**();**

cin**.**ignore**(**1000**,** '\n'**);**

**}**

**return** choose**;**

**}**

int main**()**

**{**

system**(**"chcp 1251 > nul"**);**

setlocale**(**LC\_ALL**,** "rus"**);**

cout **<<** "1 - сгенерировать картотеку \n"

**<<** "2 - прочитать картотеку с файла \n"

**<<** "3 - вывести всю картотеку \n"

**<<** "4 - ввести номер телефона, дату и время разговора \n"

**<<** "5 - вывести извещение на оплату телефонного разговора \n"

**<<** "6 - выйти из программы \n\n\n"**;**

**while** **(true)** **{**

**switch** **(**Menu**())** **{**

**case** 1**:** GenerateCards**();** **break;**

**case** 2**:** LoadCards**();** **break;**

**case** 3**:** PrintCards**();** **break;**

**case** 4**:** InputCard**();** **break;**

**case** 5**:** ShowNotification**();** **break;**

**case** 6**:** **return** 0**;**

**default:**

cout **<<** "Введите число от 1 до 6" **<<** endl**;**

**break;**

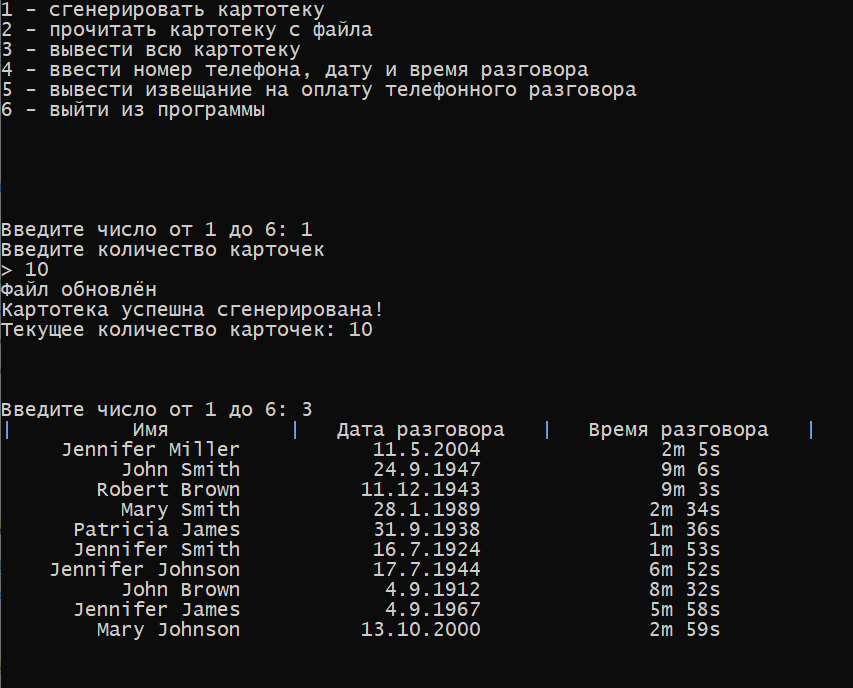
**}**

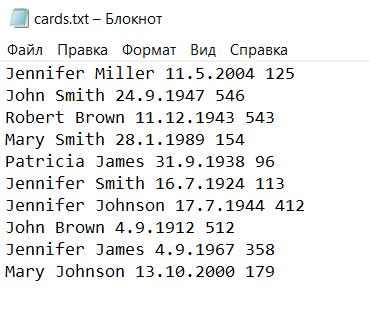
**}**

**}**

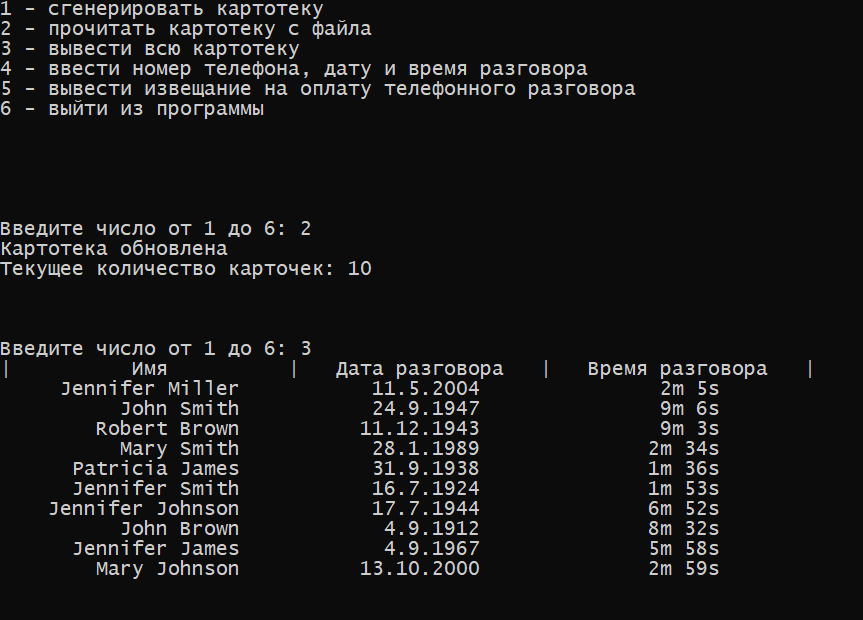
**Анализ результатов**

Создание и вывод картотеки

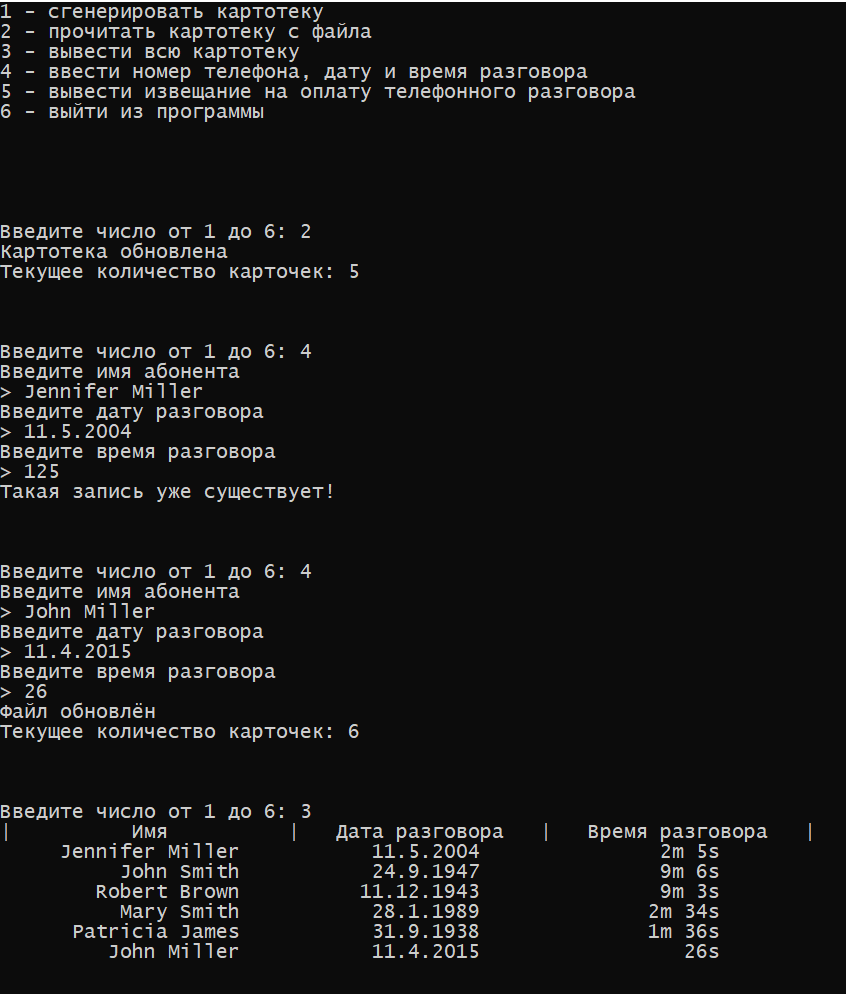




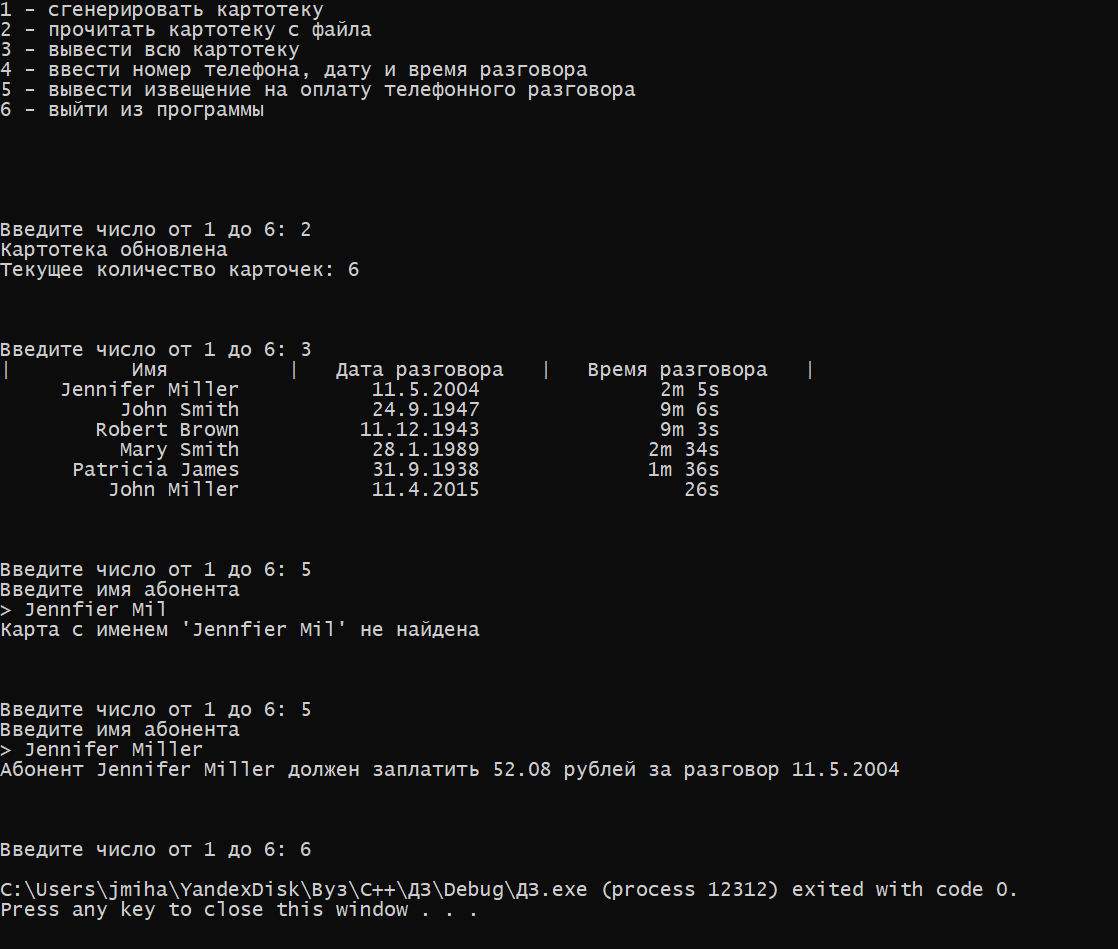
Чтение картотеки с файла



Создание новой карточки



Вывод извещения на оплату телефонного разговора



**Вывод**

Я научился

* Работать с конструктора и деконструкторами
* Перезагружать операторы
* Записывать и считывать информацию с файлов
* Работать с векторами