5. РАЗРАБОТКА ВНУТРЕННИХ СПЕЦИФИКАЦИЙ ПРОГРАММЫ

5.1. Разработка структуры программы

Структура программы представлена на рис. 5.1 в виде иерархической схемы.

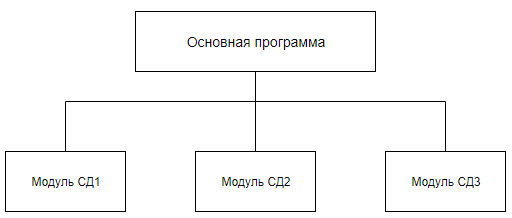


Рис. 5.1. Структура программы

где:

Модуль СД1 — основан на структуре ЛСД 1 (рис. 5.2).

Модуль СД2 — основан на структуре ЛСД 2 (рис. 5.3).

Модуль СД3 — основан на структуре ЛСД 3 (рис. 5.4).

5.2. Разработка представления данных на уровне модели памяти

Представление данных на уровне модели памяти имеет вид, представленный на рис. 5.2, рис. 5.3, рис. 5.4.

Участок 1

Участок 2

Сотр 1

Сотр 2

Сотр 3

Сотр 4

Рис. 5.2. Схема представления ЛСД 1

Город 1

\*

\*

Участок 1

\*

\*

Участок 2

\*

Х

Город 2

\*

\*

Участок 3

\*

Х

Город 3

Х

\*

Участок 4

\*

Х

Рис. 5.3. Схема представления ЛСД 2

Участок 1

Участок 2

Водитель 1

Водитель 2

Водитель 3

Рис. 5.4. Схема представления ЛСД 3

6. ОПИСАНИЕ ПРОГРАММЫ

6.1. Вводная часть

Программа реализует запросы к информационной системе, содержащей сведения о «Служба ГИБДД», которые хранятся на момент выполнения в памяти ЭВМ. Сведения в память ЭВМ заносятся с клавиатуры.

6.2. Функциональное назначение

Программа предназначена для реализации запросов к информационной базе. Она не предусматривает редактирования существующих данных. Для работы программы требуются следующие устройства:

— flash-накопитель;

— свободная оперативная память не менее 200 Кб;

— дисплей;

— клавиатура.

При сбое в работе устройств, программа прекращает свою работу.

6.3. Описание информации

Входные данные поступают с клавиатуры.

Ввод данных осуществляется в следующей структурной последовательности:

Сотрудник:

-ФИО;

-Дата рождения;

-Должность;

Участок:

- Номер;

- Территориальное расположение;

Город:

- Название

Водитель:

- ФИО;

- Дата рождения;

Автомобильный номер:

-Дата получения;

- Код региона регистрации;

-Страна.

При работе программы выводится меню, описанное в п. 4.2.

Перечень сообщений, используемых в работе программы при запросах, представлен в пункте 4.2.

Перечень сообщений, используемых в работе программы представлен в таблице 6.1.

Таблица 6.1

Сообщения программы

|  |  |  |
| --- | --- | --- |
| Текст сообщения | Содержание сообщения | Действие |
| Введите ФИО сотрудника | Требуется ввести ФИО сотрудника для выполнения запроса | Введите ФИО сотрудника для выполнения запроса |
| Введите номер участка | Требуется ввести номер участка для выполнения запроса | Введите номер участка для выполнения запроса |
| Введите название города | Требуется ввести название города для выполнения запроса | Введите название города для выполнения запроса |
| Введите ФИО водителя | Требуется ввести ФИО водителя для выполнения запроса | Введите ФИО водителя для выполнения запроса |
| Введите автомобильный номер | Требуется автомобильный номер для выполнения запроса | Введите автомобильный номер для выполнения запроса |
| Сотрудник не найден | Необходимый по запросу сотрудник не найден | Нажмите любую клавишу, чтобы вернуться в главное меню |
| Номер участка не найдено | Необходимое по запросу номер участка не найден | Нажмите любую клавишу, чтобы вернуться в главное меню |
| Город не найден | Необходимый по запросу город не найден | Нажмите любую клавишу, чтобы вернуться в главное меню |
| Водитель не найден | Необходимый по запросу водитель не найден | Нажмите любую клавишу, чтобы вернуться в главное меню |
| Автомобильный номер не найден | Необходимый по запросу автомобильный номер не найден | Нажмите любую клавишу, чтобы вернуться в главное меню |

6.4. Описание логики программы

Схема основной программы приведена на рис. 6.1.

Начало

Конец

input

ch == 0

Вывод меню

Ввод ch

Да

Нет

Рис. 6.1. Схема основной программы

6.5. Используемые подпрограммы

Программа использует следующие подпрограммы:

first\_module — устанавливает связи между участками и сотрудниками.

second\_module — устанавливает связи между городами и участками.

third\_module — связи между участками и водителями.

7. ОПИСАНИЕ ПОДПРОГРАММЫ first\_module

7.1. Вводная часть

Для реализации first\_module удобно создать отдельную подпрограмму, которая будет считывать данные из файла.

7.2. Функциональное назначение

Данный модуль устанавливает связи между участками и сотрудниками: на каком участке работает каждый сотрудник, сколько сотрудников работает на конкретном участке.

7.3. Описание информации

Входные данные:

номер участка (формат - числовой от 1 до 1000, тип - целочисленный).

Выходные данные:

ФИО сотрудника (формат - текстовый, тип - строковый).

7.4. Описание логики

Схема подпрограммы first\_module представлена на рис. 7.1.

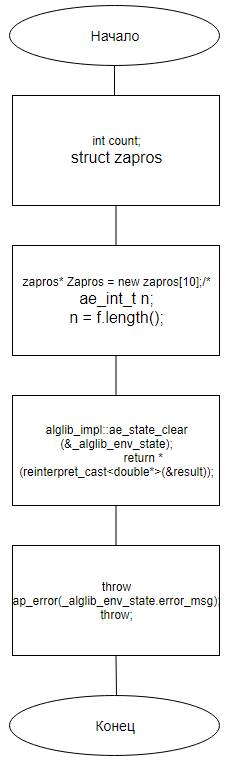


Рис. 7.1. Схема подпрограммы first\_module

8. ОПИСАНИЕ ПОДПРОГРАММЫ second\_module

8.1. Вводная часть

Для реализации ввода данных в базу удобно создать отдельную подпрограмму.

8.2. Функциональное назначение

Данный модуль устанавливает связи между городами и участками: в каком городе находится каждый участок, сколько участков находится в конкретном городе.

8.3. Описание информации

Входные данные:

название города (формат - текстовый, тип - строковый)

Выходные данные:

номер участка (формат - числовой от 1 до 1000, тип - целочисленный)

8.4. Описание логики

Схема подпрограммы second\_module представлена на рис. 8.1.



Рис. 8.1. Схема подпрограммы second\_module

9. ОПИСАНИЕ ПОДПРОГРАММЫ third\_module

9.1. Вводная часть

Для реализации вывода результатов запросов удобно создать отдельную подпрограмму, которая будет выводить данные на экран.

9.2. Функциональное назначение

Данный модуль устанавливает связи между участками и водителями: на каком участке задержан каждый водитель, сколько водителей задержано на конкретном участке.

9.3. Описание информации

Входные данные:

номер участка (формат - числовой от 1 до 1000, тип - целочисленный).

Выходные данные:

ФИО водителя (формат - текстовый, тип - строковый).

9.4. Описание логики

Схема подпрограммы third\_module представлена на рис. 9.1 .

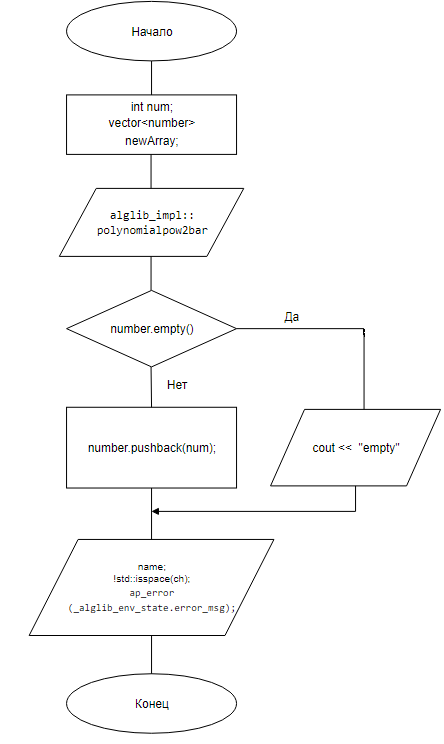


Рис. 9.1. Схема подпрограммы third\_module

10. ТЕСТИРОВАНИЕ ПРОГРАММЫ

10.1. Разработка плана тестирования и набора тестов

Для проверки работоспособности программы необходимо разработать такой тест, чтобы он проверил все возможные случаи, которые могут возникнуть при пользовании этой программой. Для этого составим набор тестов:

Запрос 1.

Введено:

Иванов Степан Геннадьевич

Ожидается:

а023кс

м263вт

р888рм

с912ан

Запрос 2.

Введено:

2

Ожидается:

о693вт

к491оа

Запрос 3.

Введено:

1

Ожидается:

Новиков

Васильев

Соколов

Запрос 4.

Введено:

1

Кожевников Иван Максимович

Ожидается:

Перевощиков

Путин

Запрос 5.

Введено:

3

Москва

Ожидается:

Белов

Ершов

Запрос 6.

Введено:

Санкт-Петербург

1

Ожидается:

Кутявин

Ялышев

Овчинников

Запрос 7.

Введено:

Пенза

Ожидается:

Участок 1

Участок 2

Запрос 8.

Введено:

Пермь

Григорьев Михаил Евгеньевич

Ожидается:

Участок 3

Участок 4

Запрос 9.

Введено:

Челябинск

к491оа

Ожидается:

Морозов

Калинин

Запрос 10.

Введено:

Антонов

Участок 3

Ожидается:

Ижевск

Уфа

10.2. Результаты тестирования

Распечатаем полные результаты, полученные по каждому из запросов.

Запрос 1.

|  |
| --- |
| Запрос 1.  Введите ФИО сотрудника:  Иванов Степан Геннадьевич  а023кс  м263вт  р888рм  с912ан |
| \_ |

Запрос 2.

|  |
| --- |
| Запрос 2.  Введите номер участка:  2  о693вт  к491оа |
| \_ |

Запрос 3.

|  |
| --- |
| Запрос 3.  Введите номер участка:  1  Новиков  Васильев  Соколов |

Запрос 4.

|  |
| --- |
| Запрос 4.  Введите номер участка:  1  Введите ФИО сотрудника:  Кожевников Иван Максимович  Перевощиков  Путин |
|  |

Запрос 5.

|  |
| --- |
| Запрос 5.  Введите номер участка:  3  Введите название города:  Москва  Белов  Ершов |
| \_ |

Запрос 6.

|  |
| --- |
| Запрос 6.  Введите название города:  Санкт-Петербург  Введите номер участка:  1  Кутявин  Ялышев  Овчинников |
|  |

Запрос 7.

|  |
| --- |
| Запрос 7.  Введите название города:  Пенза  Участок 1  Участок 2 |
|  |

Запрос 8.

|  |
| --- |
| Запрос 8.  Введите название города:  Пермь  Введите ФИО водителя:  Григорьев Михаил Евгеньевич  Участок 3  Участок 4 |
| \_ |

Запрос 9.

|  |
| --- |
| Запрос 9.  Введите название города:  Челябинск  Введите автомобильный номер:  к491оа  Морозов  Калинин |
| \_ |

Запрос 10.

|  |
| --- |
| Запрос 10.  Введите ФИО водителя:  Антонов  Введите номер участка:  3  Ижевск  Уфа |

ЗАКЛЮЧЕНИЕ

В ходе выполнения курсовой работы разработана информационно-справочная система «Служба ГИБДД». Данная система служит для облегчения работы со статистическими данными при обработке результатов. Используя данную информационную систему, можно существенно упростить поиск и просмотр основных данных в этой области.

Достоинством данной системы является автоматизация человеческого труда: экономия большого количество времени и получение точных и достоверных сведений.

СПИСОК ЛИТЕРАТУРЫ

1. Госавтоинспекция МВД России [Электронный ресурс]. URL: [https://ru.wikipedia.org/wiki](https://ru.wikipedia.org/wiki/%D0%93%D0%BE%D1%81%D0%B0%D0%B2%D1%82%D0%BE%D0%B8%D0%BD%D1%81%D0%BF%D0%B5%D0%BA%D1%86%D0%B8%D1%8F_%D0%9C%D0%92%D0%94_%D0%A0%D0%BE%D1%81%D1%81%D0%B8%D0%B8)/gosautoinspectsia (дата обращения: 20.12.2019).
2. Алгоритмы и структуры данных [Электронный ресурс]. URL: <https://stepik.org/course/63/promo> (дата обращения: 20.12.2019).
3. Логическая структура данных [Электронный ресурс]. URL: <https://helpiks.org/3-84717.html> (дата обращения: 20.12.2019).

ПРИЛОЖЕНИЕ 1

ТЕКСТ ПРОГРАММЫ

#include <iostream>

#include <algorithm>

#include <string>

#include <iomanip>

#include <map>

#include <conio.h>

#include <stdlib.h>

#include <list>

#include <vector>

#include <fstream>

#include <stdio.h>

using namespace std;

void first module() {

int count;

struct zapros

{

string name;

int len;

};

zapros\* Zapros = new zapros[10];

ae\_int\_t n;

n = f.length();

alglib\_impl::ae\_state\_init(&\_alglib\_env\_state);

try

{

double result = alglib\_impl::polynomialcalceqdist(a, b, const\_cast<alglib\_impl::ae\_vector\*>(f.c\_ptr()), n, t, &\_alglib\_env\_state);

alglib\_impl::ae\_state\_clear(&\_alglib\_env\_state);

return \*(reinterpret\_cast<double\*>(&result));

}

catch (alglib\_impl::ae\_error\_type)

{

throw ap\_error(\_alglib\_env\_state.error\_msg);

}

catch (...)

{

throw;

}

}

void second module() {

double c;

double s;

c = 0;

s = 1;

double\* mas = new double[10];

alglib\_impl::ae\_state \_alglib\_env\_state;

ae\_int\_t n;

n = a.length();

vector<bool> newArray;

for (int i = 0; i < universum.size(); i++) {

if (IsContain(universum[i], item)) {

newArray.push\_back(true);

}

else {

newArray.push\_back(false);

}

}

alglib\_impl::ae\_state\_init(&\_alglib\_env\_state);

try

{

alglib\_impl::polynomialpow2bar(const\_cast<alglib\_impl::ae\_vector\*>(a.c\_ptr()), n, c, s, const\_cast<alglib\_impl::barycentricinterpolant\*>(p.c\_ptr()), &\_alglib\_env\_state);

alglib\_impl::ae\_state\_clear(&\_alglib\_env\_state);

return;

}

catch (alglib\_impl::ae\_error\_type)

{

throw ap\_error(\_alglib\_env\_state.error\_msg);

}

catch (...)

{

throw;

}

mas = { 0 };

cout << endl;

}

void third module() {

int command = -1;

while (command != 0)

{

cin >> command;

first module();

switch (command)

{

case 1:

{

cout << "Иванов" << endl << endl << "1. а023кс" << endl << "2. м263вт" << endl << "3. р888рм" << endl << "4. с912ан" << endl << endl;

cout << "Смирнов" << endl << endl << "1. в538нк" << endl << "2. р834ту" << endl << endl;

cout << "Березин" << endl << endl << "1. е936см" << endl << "2. т165вв" << endl << "3. м579то" << endl << endl;

break;

}

case 2:

{

cout << "Участок 1" << endl << endl << "1. р834ту" << endl << "2. к729мт" << endl << "3. е777ее" << endl << endl;

cout << "Участок 2" << endl << endl << "1. о693вт" << endl << "2. к491оа" << endl << endl;

break;

}

case 3:

{

cout << "Участок 1" << endl << endl << "1. Новиков" << endl << "2. Васильев" << endl << "3. Соколов" << endl << endl;

cout << "Участок 2" << endl << endl << "1. Петров" << endl << "2. Михайлов" << endl << "3. Кузнецов" << endl << endl;

break;

}

case 4:

{

cout << "Участок 1" << endl << endl << "Сотрудник Кожевников" << endl << endl << "Перевощиков" << endl << "Путин" << endl << endl;

cout << "Участок 2" << endl << endl << "Сотрудник Данилов" << endl << endl << "Кузьмин" << endl << "Еремин" << endl << "Афанасьев" << endl << endl;

break;

}

case 5:

{

cout << "Участок 1" << endl << endl << "Екатеринбург" << endl << endl << "Максимов" << endl << "Ведерников" << endl << "Орлов" << endl << endl;

cout << "Участок 2" << endl << endl << "Казань" << endl << endl << "Афанасьев" << endl << "Харламов" << endl << endl;

cout << "Участок 3" << endl << endl << "Москва" << endl << endl << "Белов" << endl << "Ершов" << endl << endl;

break;

}

case 6:

{

cout << "Санкт-Петербург" << endl << endl << "участок 1" << endl << endl << "Кутявин" << endl << "Ялышев" << endl << "Овчинников" << endl << endl;

cout << "Санкт-Петербург" << endl << endl << "участок 2" << endl << endl << "Ялышев" << endl << "Царев" << endl << "Хохряков" << endl << endl;

cout << "Рязань" << endl << endl << "участок 1" << endl << endl << "Самсонов" << endl << "Глебова" << endl << endl;

break;

}

case 7:

{

cout << "Москва" << endl << endl << "участок 1" << endl << endl << "Участок 2" << endl << "Участок 3" << endl << "Участок 4" << endl << "Участок 5" << endl << "Участок 6" << endl << "Участок 7" << endl << endl;

cout << "Сочи" << endl << endl << "участок 1" << endl << endl << "Участок 2" << endl << "Участок 3" << endl << "Участок 4" << endl << endl;

cout << "Пенза" << endl << endl << "участок 1" << endl << endl << "Участок 2" << endl << endl;

break;

}

case 8:

{

cout << "Пермь" << endl << endl << "Белоусов" << endl << endl << "Участок 1" << endl << "Участок 3" << endl << endl;

cout << "Пермь" << endl << endl << "Григорьев" << endl << endl << "Участок 3" << endl << "Участок 4" << endl << endl;

break;

}

case 9:

{

cout << "Воронеж" << endl << endl << "в538нк" << endl << endl << "Гогунский" << endl << "Пушин" << endl << "Фролов" << endl << endl;

cout << "Челябинск" << endl << endl << "к491оа" << endl << endl << "Морозов" << endl << "Калинин" << endl << endl;

cout << "Челябинск" << endl << endl << "с912ан" << endl << endl << "Калинин" << endl << "Гришин" << endl << endl;

break;

}

case 10:

{

cout << "Карпов" << endl << endl << "участок 2" << endl << endl << "Краснодар" << endl << "Новосибирск" << endl << "Владивосток" << endl << endl;

cout << "Логинов" << endl << endl << "участок 5" << endl << endl << "Самара" << endl << "Ярославль" << endl << "Калининград" << endl << endl;

cout << "Антонов" << endl << endl << "участок 3" << endl << endl << "Ижевск" << endl << "Уфа" << endl << endl;

break;

}

default:

break;

cout << "Введите команду" << endl;

}

}

}

int main()

{

setlocale(LC\_ALL, "ru");

cout << "Главное меню" << endl << endl;

cout << "1. Выдать список НОМЕРОВ АВТОМОБИЛЕЙ, остановленных данным СОТРУДНИКОМ." << endl;

cout << "2. Выдать список НОМЕРОВ АВТОМОБИЛЕЙ, нарушивших на данном УЧАСТКЕ." << endl;

cout << "3. Выдать список СОТРУДНИКОВ, работающих на данном УЧАСТКЕ в определенный день. В определенный день каждый сотрудник работает только на одном участке." << endl;

cout << "4. Выдать список ВОДИТЕЛЕЙ, остановленных данным СОТРУДНИКОМ на данном УЧАСТКЕ В определенный день." << endl;

cout << "5. Выдать список СОТРУДНИКОВ, закрепленных на данном УЧАСТКЕ, отправленных на повышение квалификации в данный ГОРОД. Сотрудник может быть закреплен только на одном участке." << endl;

cout << "6. Выдать список ВОДИТЕЛЕЙ, задержанных на данном УЧАСТКЕ в данном ГОРОДЕ в определенный момент. В определенный момент водитель может быть задержан на одном участке." << endl;

cout << "7. Выдать список УЧАСТКОВ, находящихся в данном ГОРОДЕ." << endl;

cout << "8. Выдать список УЧАСТКОВ, на которых останавливали данного ВОДИТЕЛЯ в данном ГОРОДЕ." << endl;

cout << "9. Выдать список СОТРУДНИКОВ, задерживавших автомобиль с данным АВТОМОБИЛЬНЫМ НОМЕРОМ в данном ГОРОДЕ." << endl;

cout << "10. Выдать список ГОРОДОВ, в которых данного ВОДИТЕЛЯ задерживали на данном УЧАСТКЕ." << endl;

cout << "0. Выход из программы" << endl << endl;

cout << "Введите номер запроса" << endl;

second module();

third module();

system("pause");

return 0;

}

ПРИЛОЖЕНИЕ 2

РУКОВОДСТВО ПРОГРАММИСТА

1.Назначение программы

Данная программа написана на языке программирования С++ в среде разработки приложений Code::Blocks (распространяемой по лицензии GPL2). Программа состоит из одного модуля struct\_ies.cbp.

2. Условия применения программы

Для работы с программой необходимо:

— flash-накопитель

— IBM-совместимый персональный компьютер

— монитор

— свободная оперативная память не менее 256 кБ

— операционная система WindowsXP или выше.

3. Обращение к программе

Для запуска программы необходимо откомпилировать файл с исходным текстом программы struct\_ies.cbp и запустить двойным щелчком мыши исполняемый файл struct\_ies.ехе.

4. Входные и выходные данные

Входные данные поступают с клавиатуры. В контрольном примере использовались следующие входные данные:

Сотрудники: Иванов Степан Геннадьевич, Кожевников Иван Максимович

Водители: Григорьев Михаил Евгеньевич

Автомобильные номера: к491оа

Участки: участок 1, участок 2, участок 3

Города: Москва, Санкт-Петербург, Пенза, Пермь, Челябинск

ПРИЛОЖЕНИЕ 3

РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

1. Назначение программы

Данная программа предназначена для обращения к базе данных. С помощью этой программы и встроенных в неё запросов можно получить интересующую информацию.

2. Условия применения программы

Для работы с программой необходимо:

— flash-накопитель

— IBM-совместимый персональный компьютер

— монитор

— свободная оперативная память не менее 256 кБ

— операционная система WindowsXP или выше.

3. Пуск программы

Для запуска программы необходимо запустить двойным щелчком мыши исполняемый файл struct\_ies.ехе.

4. Сообщения пользователю

|  |  |
| --- | --- |
| Сообщение | Пояснение |
| Список сотрудников пуст | Нет ни одного сотрудника |
| Данного сотрудника нет в списке | В списке нет введённого сотрудника |
| Список водителей пуст | Нет ни одного водителя |
| Данного водителя нет в списке | В писке нет введенного водителя |
| Список автомобильных номеров пуст | Автомобильных номеров нет в списке |
| Данного автомобильного номера нет в списке | В списке нет введенного автомобильного номера |
| Список участков пуст | Нет ни одного участка |
| Данного участка нет в списке | В списке нет ни одного участка |
| Список городов пуст | Нет ни одного города |
| Данного города нет в списке | В списке нет введенного города |