МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

### федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

(ГУАП)

КАФЕДРА ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ И СЕТЕЙ

Преподаватель

канд. техн. наук, доцент Л.Н. Бариков

Отчёт

по лабораторной работе №13

по дисциплине ОСНОВЫ ПРОГРАММИРОВАНИЯ

на тему: «Структуры и файлы структур»

Работу выполнил

студент гр. 4141 В.С. Сыворотнев

Санкт-Петербург

2022

***Цель лабораторной работы:*** *изучение структурной организации, способов доступа к элементам и других особенностей структур и файлов структур; изучение стандартных средств языка C/C++ для работы с бинарными файлами; совершенствование навыков процедурного программирования на языке C/C++ при решении задач обработки файлов.*

**31.**



***Описание процесса решения***

При решении задачи, а значит в тексте программы, кроме функции *main*() необходимо реализовать следующие функции:

- функцию проверки существования физического файла с конкретным именем;

- функцию создания исходного файла структур (базы данных) в соответствии с заданием;

- функцию добавления новых записей в существующий файл;

- функции сортировки содержимого бинарного файла по значению строкового поля;

- функцию поиска записей в файле по значению строкового поля с созданием файла результатов поиска;

- функции вывода содержимого файла базы данных и файла результатов поиска на экран.

Имена исходного файла и файла результатов поиска задаём константными строками. При необходимости у пользователя будет возможность создать файл с любым именем.

Поскольку работа будет идти с бинарными файлами, то для записи в файл будет использоваться функция *fwrite*(), а для чтения – функция *fread*().

Исполнение программы (функция *main*()) начинается с объявления и инициализации символьных массивов (строк), которые будут использоваться для хранения имён исходного *fname\_i* и файла результатов поиска *fname\_r*.

На экран выводится меню программы, и пользователю предлагается сделать выбор вида очередного действия (выбор конкретного пункта меню).

При выборе пункта 1 – «создание базы данных» - начинается создание бинарного файла структур, состав полей которых соответствует заданию. Этот процесс начинается с вызова функции проверки существования физического файла с указанным именем создаваемого исходного файла *doesFileExist*().

В этой функции делается попытка открытия физического файла с этим именем на чтение. Если такой файл уже существует (если он открылся на чтение), функция возвращает значение 1, в противном случае – 0.

Если физический файл с указанным именем уже существует, то пользователю об этом выдаётся сообщение «Файл базы данных студенческой канцелярии с именем …» и предлагается либо задать имя создаваемого файла другой базы данных, либо вернуться в меню программы. Результатом такого диалога является либо имя создаваемого исходного файла, которое хранится в строке *fname\_i*, либо возврат в меню программы.

Если файл создаётся, то после этого следует оператор открытия создаваемого файла с именем *fname\_i* на запись. Затем вызывается функция *addElement*(), и пользователь вводит значения полей структуры, которая будет очередной записью создаваемого файла, по выводимым на экран подсказкам. После задания значений всех полей очередной структуры с описанием конкретной турпоездки вводится новая запись фиксированного размера (этот размер определяется составом и типами полей структуры) в создаваемый бинарный файл с использованием функции *fwrite*().

Размер создаваемого файла при открытии не оговаривается, поэтому пользователю после ввода очередной записи в файл выводится запрос на продолжение или прекращение процесса создания файла («ввод с переспросом»).

При завершении процесса создания файла он закрывается с использованием функции *fclose*(), и происходит возврат в меню программы с выводом сообщения «Создание файла закончено. База данных готова к работе».

При выборе пункта 2 – «добавление новых записей» - все действия, выполняемые по пункту 1 меню программы, повторяются за исключением того, что добавление новых записей ведется в ранее созданный файл, имя которого задаёт пользователь. В этом случае этот существующий файл открывается на добавление новых записей в конец файла. При завершении процесса добавления новых записей происходит возврат в меню программы с выводом сообщения «Изменение файла закончено».

При выборе пункта 3 – «сортировка по ФИО друга» - пользователь переходит к процессу сортировки записей существующего файла по алфавиту содержимого одного из полей (сортировка по значению строкового поля). Эти действия можно осуществлять только для существующего физического файла. Поэтому вначале вызывается функция проверки существования физического файла с указанным именем *doesFileExist*(). В этой функции делается попытка открытия физического файла с этим именем на чтение и запись. Если файл не существует – возврат в меню программы с выводом сообщения об ошибке открытия файла. Если файл существует – происходит вызов функции *sortByName*().

В этой функции сортируемый файл открывается на чтение и запись. Открывается цикл сортировки *do..while*(*fl*), выполнение которого идет до тех пор, пока процесс сортировки необходимо продолжать. Файловый указатель устанавливается на запись с номером 0. Переменная-флаг *fl* принимает значение 0 (дальнейшая сортировка не требуется). Открывается цикл попарного считывания и сравнения значений полей двух соседних структур (записей) от начала файла до его конца. Поскольку сортировка выполняется по значению строкового поля, сравнение значений производится с использованием функции *strncmp*().

Если после сравнения значений выявлено, что записи в файле стоят неправильно, файловый указатель поднимается на две записи вверх, и в сортируемый файл идет запись вначале второй считанной записи, а потом – первой. Так записи в файле меняются местами. При этом флаг необходимости сортировки *fl* принимает значение 1 (цикл дальнейшей сортировки необходимо продолжать).

Если после сравнения значений выявлено, что записи в файле стоят правильно, файловый указатель поднимается на одну запись вверх, и считывается новая пара соседних записей. Это продолжается до достижения конца файла.

Если после просмотра и сравнения значений полей структур всех соседних записей файла выявлено, что была хотя бы одна перестановка записей в файле (значение флага необходимости продолжения процесса сортировки равно 1), то цикл сортировки *do..while*(*fl*) выполняется вновь. Если перестановок не было – цикл сортировки успешно завершен.

При завершении процесса сортировки записей происходит возврат в меню программы с выводом сообщения «Сортировка по названию тура закончена».

При выборе пункта 4 – «сортировка друзей по дате рождения» - выполняются все те же действия, что и в пункте 3 меню.

При завершении процесса сортировки записей происходит возврат в меню программы с выводом сообщения «Сортировка по году рождения закончена».

При выборе пункта 5 – «поиск друзей по имени супруга» - пользователь переходит к процессу поиска записей существующего файла по значению задаваемого им ключа поиска (фамилии) и формированию файла выборки записей. Эти действия можно осуществлять только для существующего физического файла с базой туров. Файл результатов выборки создаётся новый.

Поэтому вначале вызывается функция проверки существования физического файла с указанным именем *doesFileExist*(). В этой функции делается попытка открытия физического файла с этим именем на чтение. Если файл не существует – возврат в меню программы с выводом сообщения об ошибке открытия файла на чтение.

Затем делается попытка открытия создаваемого физического файла выборки с указанным именем на запись. Если при этом произошла ошибка (например, задано неверное имя файла) - возврат в меню программы с выводом сообщения об ошибке открытия файла на запись.

В случае успешных проверок вызывается функция *searchBySpouse*(). Файл с базой данных открывается на чтение, а файл результатов поиска открывается на запись. Пользователю предлагается ввести в виде значения ключа поиска названия супруга, информацию о друзьях которую он хотел бы просмотреть.

Все записи исходного файла поочерёдно считываются. Значение соответствующего поля считанной структуры сравнивается с ключевым значением с использованием функции *strncmp*(). Если сравнение произошло – запись в файл выборки.

После завершения просмотра всего содержимого исходного файла он закрывается функцией *fclose*(). Аналогично закрывается и файл выборки результатов. Происходит возврат в меню программы с выводом сообщения «Поиск по названию страны пребывания закончен».

При выборе пункта 6 – «просмотр базы данных» - выполняется проверка существования физического файла с заданным именем вызовом функции *doesFileExist*(). Если файл существует, вызывается функция *viewBase*(). Файл открывается на чтение. Выводится заголовок таблицы, в виде которой будут выводиться значения полей записей файла. Все записи файла поочерёдно читаются, и значения их полей выводятся в виде таблицы.

После завершения процесса файл закрывается.

При выборе пункта 7 – «просмотр базы данных поиска дурзей» - все действия полностью аналогичны выбору пункта меню 6.

При выборе пункта 8 – завершение работы программы.

Анализируем результаты теста и делаем выводы.

***Текст программы***

*/\*Бинарные файлы.*

*Ввод/вывод с использованием функций библиотек языка C*

*Программа работы с базой данных "Друзья"*

*Создание базы*

*Добавление новых записей*

*Просмотр базы*

*Поиск туров по ФИО супруга*

*Сортировка по ФИО друга в алфавитном порядке*

*Сортировка по году рождения*

*\*/*

#include <stdio.h>

#include <string.h>

#include <iostream>

using namespace std;

#define FNAME\_I "D:\\bd15.dat"    //имя файла с исходной базой

#define FNAME\_R "D:\\bd25.dat"    //имя файла с результатами поиска

typedef struct {

  char day[3];

  char month[3];

  char year[6];

} date;

typedef struct {

  char name[30];    // имя друга

  char address[15];    // адрес друга

  char phoneNumber[15]; // номер телефона друга

  date dateOfBirth; // дата рождения друга

  char nameOfSpouse[15]; // имя супруга

} friendInfo;

bool doesFileExist(const char\*, const char\*);    //проверка наличия файла

void addElement(const char\* fname);

void header\_1();

void header\_2();

void viewBase(const char\* fname);

void viewSearchedDB(const char\* fname);

void searchBySpouse(const char\* fname1, const char\* fname2);

void sortByName(const char\* fname);

void sortByBirthday(const char\* fname);

const char PR\_R[] = "rb";    //признак открытия бинарного файла на чтение

const char PR\_S[] = "r+b";    //признак открытия файла на чтение и запись

const char PR\_W[] = "wb";    //признак открытия файла на запись

const char PR\_A[] = "ab";    //признак открытия файла на добавление

int main() {

  int var;

  char otv;

  char fname\_i[20] = FNAME\_I;

  char fname\_r[20] = FNAME\_R;

  setlocale(LC\_ALL, "Russian");

  for (;;) {//Выбор вида действия

    printf("**\n**    Вид действия:**\n**");

    printf("  1 - создание базы данных**\n**");

    printf("  2 - добавление новых записей**\n**");

    printf("  3 - сортировка по ФИО друга**\n**");

    printf("  4 - сортировка друзей по дате рождения**\n**");

    printf("  5 - поиск друзей по имени супруга**\n**");

    printf("  6 - просмотр базы данных**\n**");

    printf("  7 - просмотр базы данных поиска друзей**\n**");

    printf("  8 - завершение задачи**\n**");

    printf("  Введите вид действия ->");

    scanf("%i", &var);

    switch (var) {

      default:break;

      case 1:

        if (doesFileExist(fname\_i, PR\_R)) {

          printf(" Файл базы данных друзей с именем ", fname\_i);

          printf(" был создан раньше.**\n**");

          printf(" Создаём файл с новым именем? [Y/N] ");

          while ((otv = getchar()) == '**\n**');

          if (otv == 'Y' || otv == 'y') {

            printf(" Задайте имя создаваемого файла: ");

            scanf("%s", fname\_i);

          } else break;

        }

        if (!doesFileExist(fname\_i, PR\_W)) {

          printf("**\n** Ошибка открытия файла для записи**\n**");

          break;

        }

        printf(" Создаем базу ", fname\_i, '**\n**');

        getchar();

        addElement(fname\_i);

        printf("**\n** Создание файла закончено.**\n**");

        if (doesFileExist(fname\_i, PR\_R))

          printf(" База данных готова к работе**\n**");

        else printf("**\n** база не создана**\n**");

        break;

      case 2:

        if (doesFileExist(fname\_i, PR\_R)) {

          printf(" Файл базы данных друзей с именем ", fname\_i);

          printf(" был создан раньше.**\n**");

          printf(" Будем добавлять новые записи в него? [Y/N] ");

          while ((otv = getchar()) == '**\n**');

          if (otv == 'N' || otv == 'n') {

            printf(" Задайте имя файла другой базы: ");

            scanf("%s", fname\_i);

            if (!doesFileExist(fname\_i, PR\_R)) {

              printf(" Такая база данных не создавалась**\n**");

              break;

            }

          }

        }

        printf(" Добавляем записи в файл ", fname\_i, '**\n**');

        addElement(fname\_i);

        printf("**\n** Изменение файла закончено.");

        break;

      case 3:

        if (!doesFileExist(fname\_i, PR\_S)) {

          printf("**\n** Ошибка открытия файла для чтения и записи**\n**");

          break;

        }

        sortByName(fname\_i);

        printf("**\n** Сортировка по имени друзей закончена.");

        break;

      case 4:

        if (!doesFileExist(fname\_i, PR\_S)) {

          printf("**\n** Ошибка открытия файла для чтения и записи**\n**");

          break;

        }

        sortByBirthday(fname\_i);

        printf("**\n** Сортировка по дате рождения друзей закончена.");

        break;

      case 5:

        if (!doesFileExist(fname\_i, PR\_R)) {

          printf("**\n** Ошибка открытия файла для чтения**\n**");

          break;

        }

        if (!doesFileExist(fname\_r, PR\_W)) {

          printf("**\n** Ошибка открытия файла для записи**\n**");

          break;

        }

        searchBySpouse(fname\_i, fname\_r);

        printf("**\n** Поиск по названию страны пребывания закончен.");

        break;

      case 6:

        if (!doesFileExist(fname\_i, PR\_R)) {

          printf("**\n** Ошибка открытия файла для чтения**\n**");

          break;

        }

        viewBase(fname\_i);

        break;

      case 7:

        if (!doesFileExist(fname\_r, PR\_R)) {

          printf("**\n** Ошибка открытия файла для чтения**\n**");

          break;

        }

        viewSearchedDB(fname\_r);

        break;

      case 8: return 0;

    }

  }

}

//Добавление новых элементов в базу данных

void addElement(const char\* fname) {

  char ans;

  friendInfo friendCard;

  FILE\* base;

  base = fopen(fname, PR\_A);

  rewind(base);

  do {

    printf("**\n** ФИО друга? ");

    scanf("%s", &friendCard.name);

    printf(" Адрес ? ");

    scanf("%s", &friendCard.address);

    printf(" Номер телефона ? ");

    scanf("%s", &friendCard.phoneNumber);

    printf(" День рождения друга ? ");

    scanf("%s", &friendCard.dateOfBirth.day);

    printf(" Месяц рождения друга ? ");

    scanf("%s", &friendCard.dateOfBirth.month);

    printf(" Год рождения друга ? ");

    scanf("%s", &friendCard.dateOfBirth.year);

    printf(" Имя супруга друга ? ");

    scanf("%s", &friendCard.nameOfSpouse);

    fwrite(&friendCard, sizeof(friendCard), 1, base);

    printf("**\n** Продолжать?[Y/N]");

    while ((ans = getchar()) == '**\n**');

  } while (ans == 'Y' || ans == 'y');

  fclose(base);

}

//Вывод заголовка при просмотре исходного файла

void header\_1() {

  int i;

  printf("**\n**");

  for (i = 1; i <= 69; i++)

    printf("-");

  printf("**\n**");

  printf("|%13s|%13s|%13s|%10s|%10s|**\n**", "ФИО Друга   ", "Адрес       ",

         "Телефон     ", "Дата рождения  ", "ФИО супруга ");

  for (i = 1; i <= 69; i++)

    printf("-");

}

//Вывод заголовка при просмотре файла поиска тура

void header\_2() {

  int i;

  printf("**\n**");

  for (i = 1; i <= 69; i++)

    printf("-");

  printf("**\n**");

  printf("|%13s|%13s|%13s|%10s|%10s|**\n**", "ФИО Друга   ", "Адрес       ",

         "Телефон     ", "Дата рождения  ", "ФИО супруга ");

  for (i = 1; i <= 69; i++)

    printf("-");

}

//Просмотр базы данных друзей

void viewBase(const char\* fname) {

  int i;

  string date;

  friendInfo friendCard;

  FILE\* baza;

  baza = fopen(fname, PR\_R);

  printf("**\n**     База данных друзей");

  header\_1();

  while (fread(&friendCard, sizeof(friendCard), 1, baza) > 0) {

    printf("**\n**|%11s |%11s |%11s |%3s %3s %6s |%11s |", friendCard.name, friendCard.address,

           friendCard.phoneNumber, friendCard.dateOfBirth.day, friendCard.dateOfBirth.month,

           friendCard.dateOfBirth.year, friendCard.nameOfSpouse);

  }

  printf("**\n**");

  for (i = 1; i <= 69; i++)

    printf("-");

  fclose(baza);

}

//Просмотр базы данных поиска Друзей по имени супруга

void viewSearchedDB(const char\* fname) {

  int i;

  friendInfo friendCard;

  FILE\* baza;

  baza = fopen(fname, PR\_R);    //открываем файл на чтение

  printf("**\n**  База данных поиска друзей по имени супруга");

  header\_2();

  while (fread(&friendCard, sizeof(friendCard), 1, baza) > 0) {

    printf("**\n**|%11s |%11s |%11s |%3s %3s %6s |%11s |", friendCard.name, friendCard.address,

           friendCard.phoneNumber, friendCard.dateOfBirth.day, friendCard.dateOfBirth.month,

           friendCard.dateOfBirth.year, friendCard.nameOfSpouse);

  }

  printf("**\n**");

  for (i = 1; i <= 69; i++)

    printf("-");

  fclose(baza);

}

//Поиск туров по названию страны пребывания

void searchBySpouse(const char\* fname1, const char\* fname2) {

  char nameOfSpouse[15];

  friendInfo friendCard;

  FILE\* in,                    //исходный файл

  \* out;                //файл результатов поиска

  in = fopen(fname1, PR\_R);        //открываем файл на чтение

  out = fopen(fname2, PR\_W);        //открываем файл на запись

  printf("**\n** имя супруга для поиска? ");

  scanf("%s", &nameOfSpouse);

  while (fread(&friendCard, sizeof(friendCard), 1, in) > 0) {

    if ((strncmp(friendCard.nameOfSpouse, nameOfSpouse, 15) == 0)) {

      fwrite(&friendCard, sizeof(friendCard), 1, out);

    }

  }

  fclose(out);

  fclose(in);

}

//Сортировка по ФИО друзей

void sortByName(const char\* fname) {

  int i;

  int fl;

  friendInfo friendCard\_2, friendCard\_1;

  FILE\* base;

  base = fopen(fname, PR\_S);    //открываем файл на чтение и запись

  do {

    rewind(base);

    fl = 0;

    for (i = 0; fread(&friendCard\_1, sizeof(friendCard\_1), 1, base) > 0; i += sizeof(friendCard\_1),

        fseek(base, i, SEEK\_SET))    //позиция i от НАЧАЛА файла

    {

      if (fread(&friendCard\_2, sizeof(friendCard\_1), 1, base) > 0) {

        if (strncmp(friendCard\_1.name, friendCard\_2.name, 15) > 0) {

          fseek(base, i, SEEK\_SET);    //позиция i от НАЧАЛА файла

          fwrite(&friendCard\_2, sizeof(friendCard\_1), 1, base);

          fwrite(&friendCard\_1, sizeof(friendCard\_1), 1, base);

          fl = 1;

        }

      }

    }

  } while (fl);

  fclose(base);

}

//Сортировка по дате рождения

void sortByBirthday(const char\* fname) {

  int i;

  int fl;

  friendInfo friendCard\_2, friendCard\_1;

  FILE\* base;

  base = fopen(fname, PR\_S);    //открываем файл на чтение и запись

  do {

    rewind(base);

    fl = 0;

    for (i = 0; fread(&friendCard\_1, sizeof(friendCard\_1), 1, base) > 0; i += sizeof(friendCard\_1),

        fseek(base, i, SEEK\_SET)) {

      if (fread(&friendCard\_2, sizeof(friendCard\_1), 1, base) > 0) {

        if (strcmp(friendCard\_1.dateOfBirth.year, friendCard\_2.dateOfBirth.year) > 0) {

          fseek(base, i, SEEK\_SET);    //позиция i от НАЧАЛА файла

          fwrite(&friendCard\_2, sizeof(friendCard\_1), 1, base);

          fwrite(&friendCard\_1, sizeof(friendCard\_1), 1, base);

          fl = 1;

        }

      }

    }

  } while (fl);

  fclose(base);

}

//проверка наличия физического файла

bool doesFileExist(const char\* fname, const char\* pr) {

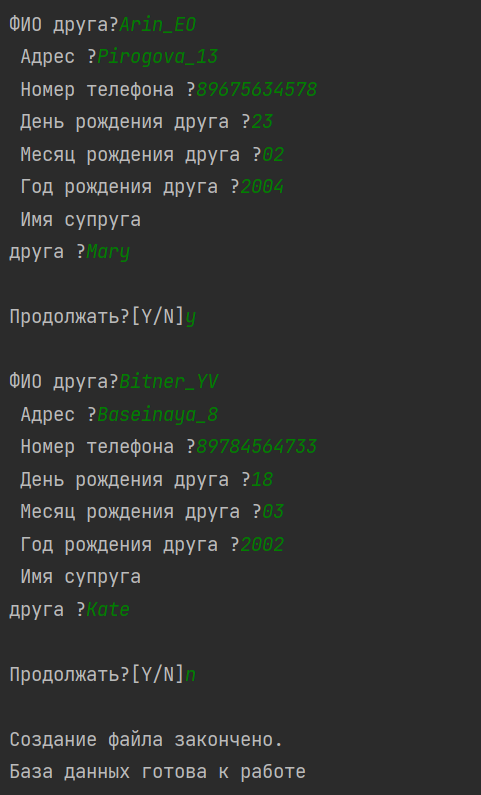
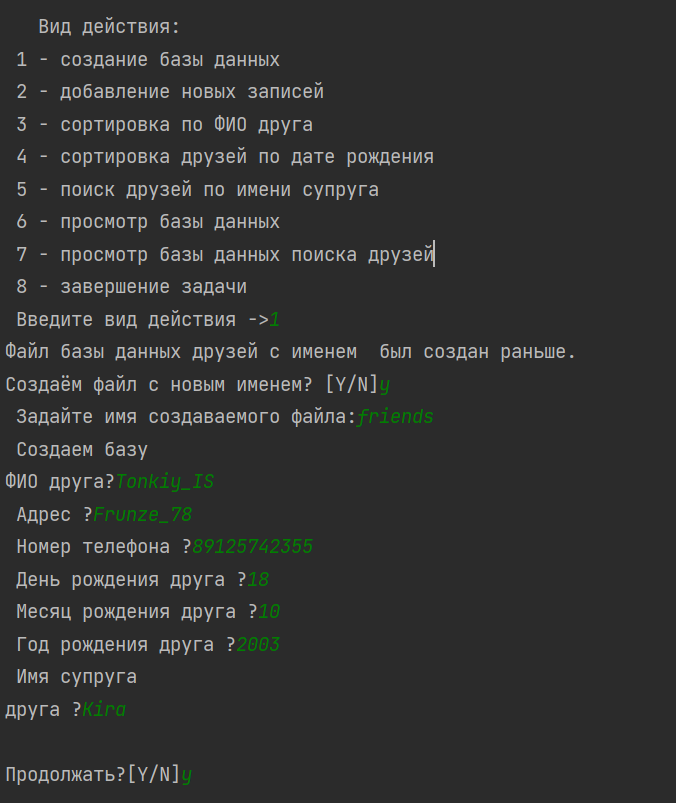
  //Открываем файл

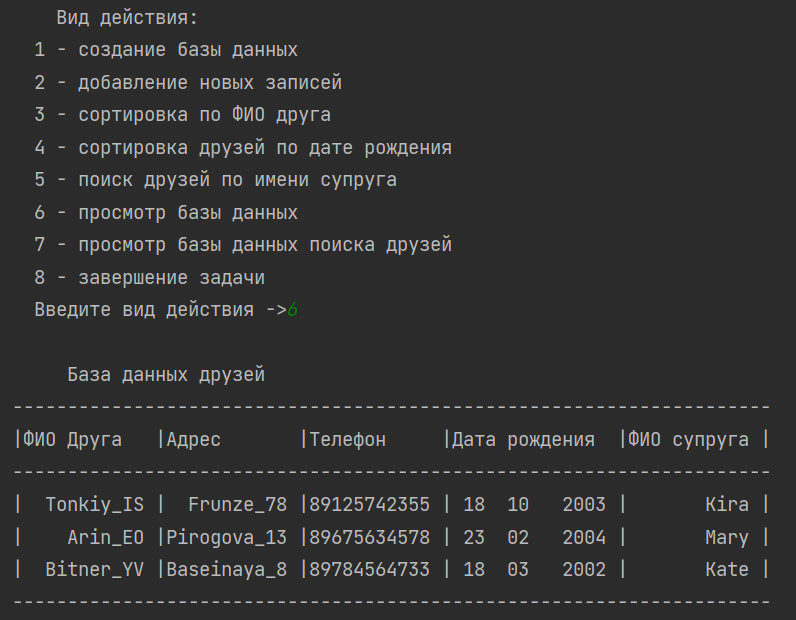
  if (!(fopen(fname, pr)))        //файл не существует

    return false;

  return true;

}

***Скриншоты результатов выполнения программы***

******

