Министерство образования Республики Беларусь

Учреждение образования

«Брестский Государственный технический университет»

Кафедра ИИТ

**Лабораторная работа №12**

По дисциплине «Основы алгоритмизации и программирования»

Тема: «Индексные файлы»

**Выполнил:**

Студент 1 курса

Группы ИИ-21

Карагодин Д.Л.

**Проверил:**

Гирель Т.Н

Брест 2022

**Цель работы:** Изучить принципы программирования с использованием индексных файлов.

**Вариант №1**

**Текст задания**

**В программу, разработанную в лабораторной работе 12 внести следующие изменения и дополнения:**

**Сортировка должна выполняться по двум полям на выбор при помощи создания индексных файлов. Содержимое индексного файла переписывается в случае изменения значения ключевого поля (поля, по которому выполняется сортировка) или в случае удаления, добавления записей.**

**Текст программы**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <string.h>

#include <malloc.h>

#pragma pack(push, 1)

struct data {

char name[15];

char surname[15];

char middle\_name[15];

char local\_street[20];

short discount;

};

#pragma pop

short\* index\_massive;

short razmer = 0;

struct data\* people;

FILE\* file;

FILE\* index\_file;

char typefile[3];

char name[30];

short type\_of\_sort=0;

void indexWrite() {

short count = 0;

index\_file = fopen("index.txt", "w+t");

fprintf(index\_file, "%hd\n",type\_of\_sort);

while (count < razmer) {

if (count != razmer - 1) fprintf(index\_file, "%hd\n", index\_massive[count]);

else fprintf(index\_file, "%hd", index\_massive[count]);

count++;}

fclose(index\_file);}

void indexread() {

if(fopen("index.txt","rt+")!= NULL){

index\_file = fopen("index.txt", "rt+");

fscanf(index\_file, "%hd\n", &type\_of\_sort);

index\_massive = (short\*)realloc(index\_massive, sizeof(short) \* razmer);

short count = 0;

while (count < razmer) {fscanf(index\_file, "%hd", &index\_massive[count]);count++;}

fclose(index\_file);}}

void sort(short silent=1) {

if (silent == 0) type\_of\_sort = 0;

short choise=0, maxmin, id;

char\* sort\_name=(char\*)calloc(15,sizeof(char));

struct data buffer; short index\_buffer;

struct data \*buffer\_people = (struct data\*)calloc(razmer,sizeof(struct data));

for (short i = 0; i < razmer; i++) { buffer\_people[i] = people[i]; index\_massive[i] = i; }

if (type\_of\_sort == 0) {

printf("Хотите сортировать\nПо Скидке -> 1\nИли По Имени -> 2 ?\nВаш выбор: ");

scanf("%hd", &choise);}

if (choise == 1 || type\_of\_sort==1 || type\_of\_sort==2) {

if (type\_of\_sort == 0) {

printf("Хотите сортировать\nПо возрастанию -> 1\nИли убыванию -> 2 ?\nВаш выбор: ");

scanf("%hd", &choise);}

if (choise == 1||type\_of\_sort==1) {

for (short i = 0; i < razmer; i++) {

maxmin = 100; id = 0;

for (short j = i; j < razmer; j++) {

if (buffer\_people[j].discount <= maxmin) { maxmin = buffer\_people[j].discount; id = j; }}

buffer = buffer\_people[id]; index\_buffer = index\_massive[id];

buffer\_people[id] = buffer\_people[i]; index\_massive[id] = index\_massive[i];

buffer\_people[i] = buffer; index\_massive[i] = index\_buffer;}

type\_of\_sort = 1;}

else {

for (short i = 0; i < razmer; i++) {

maxmin = 0; id = 0;

for (short j = i; j < razmer; j++) {

if (buffer\_people[j].discount >= maxmin) { maxmin = buffer\_people[j].discount; id = j; }}

buffer = buffer\_people[id]; index\_buffer = index\_massive[id];

buffer\_people[id] = buffer\_people[i]; index\_massive[id] = index\_massive[i];

buffer\_people[i] = buffer; index\_massive[i] = index\_buffer;}

type\_of\_sort = 2;}}

else {

if (type\_of\_sort == 0) {

printf("Хотите сортировать\nПо A-Z -> 1\nИли Z-A -> 2 ?\nВаш выбор: ");

scanf("%hd", &choise);}

if (choise == 1|| type\_of\_sort == 3) {

for (short i = 0; i < razmer; i++) {

strcpy(sort\_name,buffer\_people[i].name); id = 0;

for (short j = i; j < razmer; j++) {

if (strcmp(buffer\_people[j].name,sort\_name) <= 0) { strcpy(sort\_name, buffer\_people[j].name); id = j; }}

buffer = buffer\_people[id]; index\_buffer = index\_massive[id];

buffer\_people[id] = buffer\_people[i]; index\_massive[id] = index\_massive[i];

buffer\_people[i] = buffer; index\_massive[i] = index\_buffer;}

type\_of\_sort = 3;}

else {

for (short i = 0; i < razmer; i++) {

strcpy(sort\_name, buffer\_people[i].name); id = 0;

for (short j = i; j < razmer; j++) {

if (strcmp(buffer\_people[j].name, sort\_name) >= 0) { strcpy(sort\_name, buffer\_people[j].name); id = j; }}

buffer = buffer\_people[id]; index\_buffer = index\_massive[id];

buffer\_people[id] = buffer\_people[i]; index\_massive[id] = index\_massive[i];

buffer\_people[i] = buffer; index\_massive[i] = index\_buffer;}

type\_of\_sort = 4;}}

indexWrite();

printf("\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n");}

void output\_index() {

indexread();

printf("%-5s %-15s %-15s %-15s %-20s %-15s\n", "Номер", "Имя", "Фамилия", "Отчество", "Улица проживания", "Размер скидки");

for (short i = 0; i < razmer; i++) {

printf(" %-3d", i + 1);

printf(" %-15s", people[index\_massive[i]].name);

printf(" %-15s", people[index\_massive[i]].surname);

printf(" %-15s", people[index\_massive[i]].middle\_name);

printf(" %-20s", people[index\_massive[i]].local\_street);

printf(" %-d%%\n", people[index\_massive[i]].discount);}

printf("\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n");}

void output\_struct() {

printf("%-5s %-15s %-15s %-15s %-20s %-15s\n", "Номер", "Имя", "Фамилия", "Отчество", "Улица проживания", "Размер скидки");

for (short i = 0; i < razmer; i++) {

printf(" %-3d", i + 1);

printf(" %-15s", people[i].name);

printf(" %-15s", people[i].surname);

printf(" %-15s", people[i].middle\_name);

printf(" %-20s", people[i].local\_street);

printf(" %-d%%\n", people[i].discount);}

printf("\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n");}

void input\_struct() {

++razmer;

people = (struct data\*)realloc(people, razmer \* sizeof(struct data));

index\_massive = (short\*)realloc(index\_massive, razmer \* sizeof(short));

printf("Введите данные:\n");

printf("Имя: ");

scanf("%s", people[razmer - 1].name);

printf("Фамилия: ");

scanf("%s", people[razmer - 1].surname);

printf("Отчество: ");

scanf("%s", people[razmer - 1].middle\_name);

printf("Улица проживания: ");

scanf("%\*c");

scanf("%[^\n]", people[razmer - 1].local\_street);

printf("Размер скидки в процентах: ");

scanf("%hd", &people[razmer - 1].discount);

sort();

printf("\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n");}

void procent\_5() {

for (short i = 0; i < razmer; i++) {

printf("%-5s %-15s %-15s %-15s %-20s %-15s\n", "Номер", "Имя", "Фамилия", "Отчество", "Улица проживания", "Размер скидки");

if (people[i].discount == 5) {

printf(" %-3d", i + 1);

printf(" %-15s", people[i].name);

printf(" %-15s", people[i].surname);

printf(" %-15s", people[i].middle\_name);

printf(" %-20s", people[i].local\_street);

printf(" %-d%%\n", people[i].discount);}}

printf("\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n");}

void delete\_element() {

short id;

output\_struct();

printf("Введите id элемента который нужно удалить: ");

scanf("%hd", &id);

struct data buffer;

short index\_buffer;

for (short i = 0; i < razmer - id; i++) {

buffer = people[i + id];

people[i + id] = people[i + id - 1];

people[i + id - 1] = buffer; }

razmer--;

people = (struct data\*)realloc(people, razmer \* sizeof(struct data) );

index\_massive= (short\*)realloc(index\_massive, razmer \* sizeof(short));

sort();

printf("\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n");}

void fread() {

if (strcmp(typefile, "bin") != 0) {

while (!feof(file)) {

++razmer;

people = (struct data\*)realloc(people, razmer \* sizeof(struct data));

fscanf(file,"%15s", people[razmer - 1].name);

fscanf(file,"%15s", people[razmer - 1].surname);

fscanf(file,"%15s", people[razmer - 1].middle\_name);

fscanf(file, "%\*c");

fscanf(file,"%[^\n]", people[razmer - 1].local\_street);

fscanf(file,"%hd", &people[razmer - 1].discount);}}

else {

int a,count=0; fread(&a,sizeof(int),1,file);

while (count<a) {

++razmer;

people = (struct data\*)realloc(people, razmer \* sizeof(struct data));

fread(people[razmer - 1].name,sizeof(char), 15,file);

printf("%d ", ftell(file));

fread(people[razmer - 1].surname, sizeof(char), 15,file);

printf("%d ", ftell(file));

fread(people[razmer - 1].middle\_name, sizeof(char), 15,file);

printf("%d ", ftell(file));

fread(people[razmer - 1].local\_street, sizeof(char), 20,file);

printf("%d ", ftell(file));

fread(&people[razmer - 1].discount,sizeof(short),1,file);

printf("%d \n", ftell(file));

count += sizeof(struct data);}}}

void fwrite() {

rewind(file);

short count = 0;

if (strcmp(typefile, "bin") != 0) {

file = fopen(name, "w+t");

while (count<razmer) {

fprintf(file, "%s\n", people[count].name);

fprintf(file, "%s\n", people[count].surname);

fprintf(file, "%s\n", people[count].middle\_name);

fprintf(file, "%s\n", people[count].local\_street);

if(count!=razmer-1) fprintf(file, "%hd\n",people[count].discount);

else fprintf(file, "%hd", people[count].discount);

count++;}}

else {

file = fopen(name, "wb+");

int a = sizeof(struct data) \* razmer;

fwrite(&a, sizeof(int), 1, file);

while (count < razmer) {

fwrite(people[count].name, sizeof(char), 15, file);

printf("%d ", ftell(file));

fwrite(people[count].surname, sizeof(char), 15, file);

printf("%d ", ftell(file));

fwrite(people[count].middle\_name, sizeof(char), 15, file);

printf("%d ", ftell(file));

fwrite(people[count].local\_street, sizeof(char), 20, file);

printf("%d ", ftell(file));

fwrite(&people[count].discount, sizeof(short), 1, file);

printf("%d \n", ftell(file));

count++;}}}

void init() {

people = (struct data\*)calloc(sizeof(struct data), razmer);

index\_massive = (short\*)calloc(sizeof(short), razmer);

printf("Введите имя файла(с txt или bin): ");

scanf("%s", name);

char name\_buffer[30];

strcpy(name\_buffer, name);

char \*typefl=strtok(name\_buffer,".");

typefl = strtok(NULL, ".");

typefile[0] = typefl[0];

typefile[1] = typefl[1];

typefile[2] = typefl[2];

if ((file = fopen(name, "r")) == NULL) {

printf("Так как файла не было,мы его создали\n");

if (strcmp(typefile, "bin") != 0) {file = fopen(name, "wt+"); }

else { file = fopen(name, "wb+"); }}

else{

if (strcmp(typefile, "bin") != 0) { file = fopen(name, "rt+"); fread();}

else { file = fopen(name, "rb+"); fread(); }}}

int main() {

init();

indexread();

short Exit = 0;

short choise = 0;

while (Exit == 0) {

printf("Что вы собираетесь сделать?\n1.Добавить новый элемент структуры\n2.Вывести всю таблицу\n3.Вывести всех с скидкой 5%\n4.Сортировка(c индекным файлом)\n5.Удалить элемент структуры\n6.Выйти из программы\n7.Вывод по индексному файлу\nВаш выбор: ");

scanf("%hd", &choise);

printf("\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n");

switch (choise) {

case 1:

input\_struct();

break;

case 2:

output\_struct();

break;

case 3:

procent\_5();

break;

case 4:

sort(0);

break;

case 5:

delete\_element();

break;

case 6:

Exit = 1;

break;

case 7:

output\_index();

break;

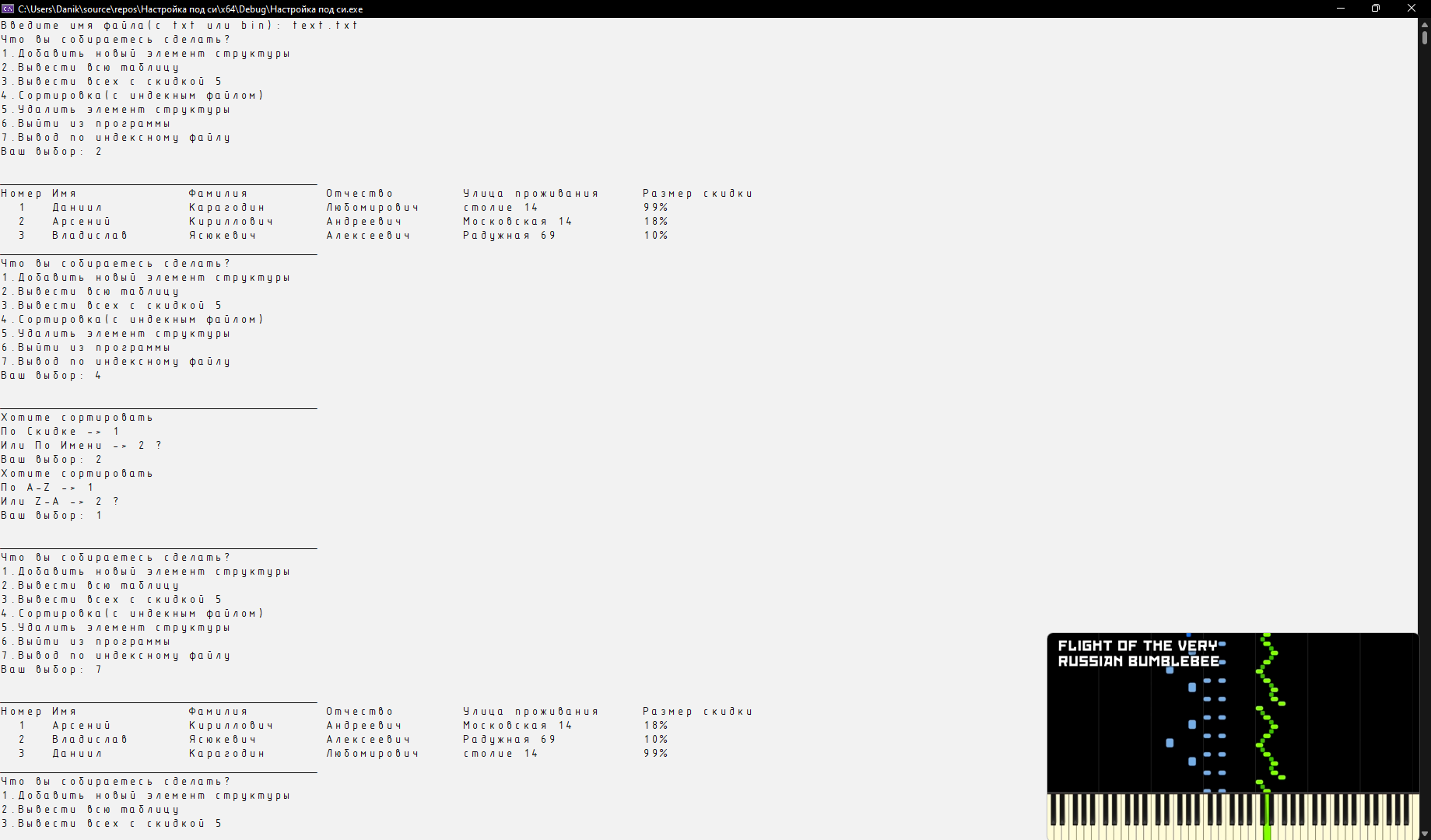
default:

printf("Вы не угадали с номером");}}fwrite();\_fcloseall();

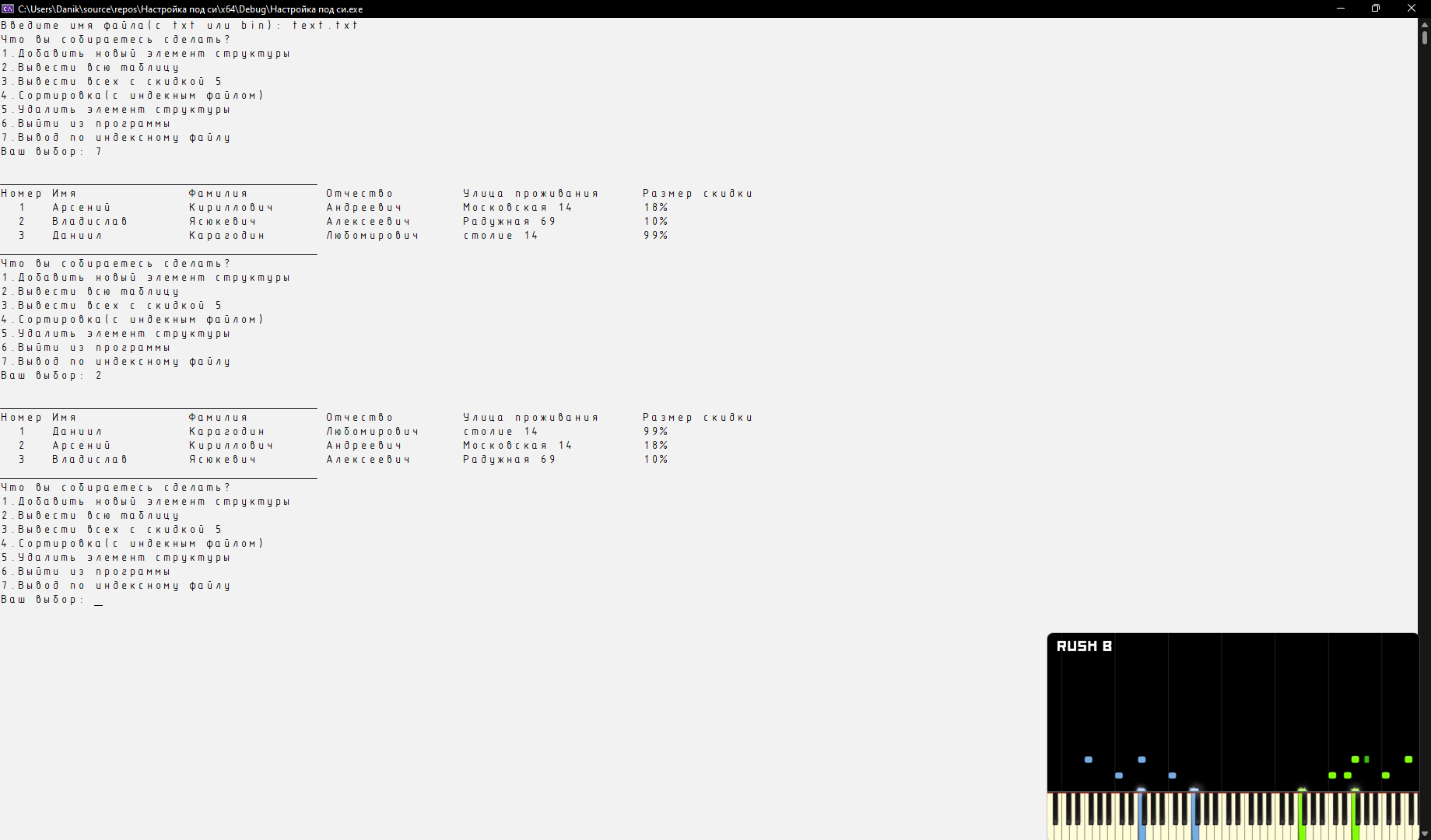
return 0;}

**Результат работы**

**#1**



**#2**



**Вывод:** Изучил принципы программирования с использованием индексных файлов.