**Лабораторна робота №7**

Таблиці рішень

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Умови | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 1 | Файл не знайдений? | д | н | н | н | н | н | н | - | - |
| 2 | Найменування спеціальності містить не тільки букви? | - | д | н | н | н | н | н | - | - |
| 3 | Табельний номер містить не тільки цифри? | - | н | д | н | н | н | н | - | - |
| 4 | ФІО лікаря містить не тільки букви? | - | н | н | д | н | н | н | - | - |
| 5 | У файлі зустрічаються декілька однакових табельних номерів? | - | н | н | н | д | н | н | - | - |
| 6 | У файлі зустрічаються декілька однакових ФІО лікаря? | - | н | н | н | н | д | н | - | - |
| 7 | Проглянуті всі записи? | н | н | н | н | н | н | н | д | д |
| 8 | Виявлена некоректність F1? | - | - | - | - | - | - | - | д | н |
|  | Дії |  |  |  |  |  |  |  |  |  |
| 1 | Повідомлення: «Файл <ім'я> не знайдений!» | 1 |  |  |  |  |  |  |  |  |
| 2 | Повідомлення: «Найменування <ім'я> спеціальності небуквене » |  | 1 |  |  |  |  |  |  |  |
| 3 | Повідомлення: «Табельний номер <N> лікаря нечислової» |  |  | 1 |  |  |  |  |  |  |
| 4 | Повідомлення: «ФІО лікаря <ім'я> небуквені» |  |  |  | 1 |  |  |  |  |  |
| 5 | Повідомлення: «Табельний номер <N> неунікальний» |  |  |  |  | 1 |  |  |  |  |
| 6 | Повідомлення: «ФІО лікаря <ім'я> неунікальні» |  |  |  |  |  | 1 |  |  |  |
| 7 | Встановити ознаку некоректності | 2 | 2 | 2 | 2 | 2 | 2 |  |  |  |
| 8 | Занести запис з файлу в таблицю t1 |  |  |  |  |  |  | 1 |  |  |
| 9 | Продовжити обробку F1 | 3 | 3 | 3 | 3 | 3 | 3 | 2 |  |  |
| 10 | Повідомлення: «Внесіть виправлення до F1» |  |  |  |  |  |  |  | 1 |  |
| 11 | Припинити обробку |  |  |  |  |  |  |  | 2 |  |
| 12 | Перейти до наступної функції |  |  |  |  |  |  |  |  | 1 |

Схеми програм

Перевірка на існування файлу, введення даних з файлу



Проверка на корректность файла





Листинг программы

#include<iostream.h>

#include<fstream.h>

#include<string.h>

#include<stdlib.h>

#include<conio.h>

#include<iomanip.h>

#include<windows.h>

const int l=31;

struct fio

{ char fam[l];

char im[l];

char ot[l];

};

struct doctor

{ fio d;

char spec[l];

char nt[l];

};

struct patient

{ fio p;

char nt[l];

char np[l];

};

struct tablF3

{ fio d;

fio p;

char spec[l];

char np[l];

int nb;

};

const int N=100;

class massiv

{private:

doctor d[N];

patient p[N];

int m;

int k;

int n;

tablF3 f[N];

public:

void inputfileF1();

void outputF1();

void mistakeF1();

};

void massiv::inputfileF1()

{ ifstream fin;

char file[l];

cout<<"imya faila:";

cin>>file;

fin.open(file);

if(fin==NULL)

{ cout<<"fail ne naiden";

getch();

exit(1);

}

m=0;

fin>>d[m].nt>>d[m].d.fam>>d[m].d.im>>d[m].d.ot>>d[m].spec;

while (fin.good())

{ m++;

fin>>d[m].nt>>d[m].d.fam>>d[m].d.im>>d[m].d.ot>>d[m].spec;

}

}

void massiv:: outputF1()

{ int i;

char buf1[1024], buf2[1024], buf3[1024], buf4[1024], buf5[1024];

for(i=0;i<m;i++)

{ CharToOem(d[i].nt, buf1);

CharToOem(d[i].d.fam, buf2);

CharToOem(d[i].d.im, buf3);

CharToOem(d[i].d.ot, buf4);

CharToOem(d[i].spec, buf5);

cout<<setw(4)<<setiosflags(ios::left)<<buf1

<<setw(14)<<setiosflags(ios::left)<<buf2

<<setw(10)<<setiosflags(ios::left)<<buf3

<<setw(15)<<setiosflags(ios::left)<<buf4

<<setw(5)<<setiosflags(ios::left)<<buf5<<endl;

if((i+1)%10==0) getch();

}

getch();

}

bool IsChar(char \*x)

{ int i=0, count=0;

while(\*(x+i)!='\0')

{ if((\*(x+i)<='я' && \*(x+i)>='А') ||

(\*(x+i)<='z' && \*(x+i)>='A') || \*(x+i)=='\_'|| \*(x+i)=='-')

count++;

i++;

}

if(count == i) return true;

else return false;

}

bool IsInt(char \*x)

{ int i=0, count=0;

while(\*(x+i)!='\0')

{ if((\*(x+i)<='9' && \*(x+i)>='0')) count++;

i++;

}

if(count == i) return true;

else return false;

}

void massiv::mistakeF1()

{ int i,j,proverka,proverka1,proverka2,proverka3;

char buf[2048];

char \*mis="";

proverka=0;

for(i=0;i<m;i++)

{if (!IsChar(d[i].d.fam)|| !IsChar(d[i].d.im) || !IsChar(d[i].d.ot))

{ mis=strcat(strcat(strcat(strcat(d[i].d.fam," "),d[i].d.im)," "),d[i].d.ot);

CharToOem(mis, buf);

cout<<"Mistake: FIO vracha '"<<buf<<"' ne bukvennie!"<<endl;

getch();

proverka++;

}

if (!IsInt(d[i].nt))

{ mis=d[i].nt;

CharToOem(mis, buf);

cout<<"Mistake: Tabelniy nomer '"<<buf<<"' vracha ne chislovoy!"<<endl;

getch();

proverka++;

}

if (!IsChar(d[i].spec))

{ mis=strcat(d[i].spec," ");

CharToOem(mis, buf);

cout<<"Mistake: naimenovanie spetsialnosti '"<<buf<<"' ne bukvennoe!"<<endl;

getch();

proverka++;

}

}

//----------------------------------------------------------------

for(i=0;i<m;i++)

{ proverka1=0;

proverka2=0;

for(j=0;j<m;j++)

{ if((strcmp(d[i].d.fam,d[j].d.fam)==0)&&

(strcmp(d[i].d.im,d[j].d.im)==0)&&

(strcmp(d[i].d.ot,d[j].d.ot)==0))

proverka1++;

if(strcmp(d[i].nt,d[j].nt)==0) proverka2++;

}

if(proverka1>1)

{ mis=strcat(strcat(strcat(strcat(d[i].d.fam," "),d[i].d.im)," "),d[i].d.ot);

CharToOem(mis, buf);

cout<<"FIO vracha '"<<buf<<"'ne unikalni!"<<endl;

getch();

proverka++;

break;

}

if(proverka2>1)

{ strcpy(mis,d[i].nt);

CharToOem(mis, buf);

cout<<"Tabelniy nomer vracha '"<<buf<<"'ne unikalniy!"<<endl;

getch();

proverka++;

break;

}

}

if (proverka>0)

{ cout<<"Obnaruzhena nekorrektnost faila F1! Vnesite ispravleniya."<<endl;

getch();

exit(1);}

}

void main()

{ massiv h;

int p;

while(1)

{ clrscr();

cout<<"1.vvod dannih iz faila F1 i vivod na ekran.\n";

cout<<"2.proverka na korrektnost faila F1.\n";

cout<<"3.vihod iz programmi.\n";

cout<<endl<<endl;

cout<<"vash vibor (1%11):";

cin>>p;

switch(p)

{

case 1:{h.inputfileF1();h.outputF1();break;}

case 2:{h.mistakeF1();break;}

case 3: cout<<"konets raboti"; getch(); return;

default:cout<<"net punkta"; getch(); break;

}

}

}

Результати тестування програми

1. Перевірка на коректність файлу F1

1) Мета тесту: перевірити існування файлу F1

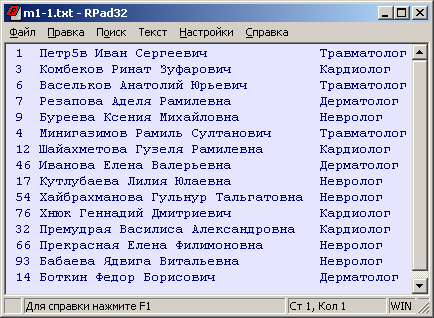
Зміст тесту: файл «р1.txt»

Очікуваний результат: повідомлення: «fail ne naiden»

Результат тестування: неуспіх

2) Мета тесту: перевірити функціонування програми, якщо ФІО лікаря містить не тільки букви

Зміст тесту: файл «m1-1.txt»

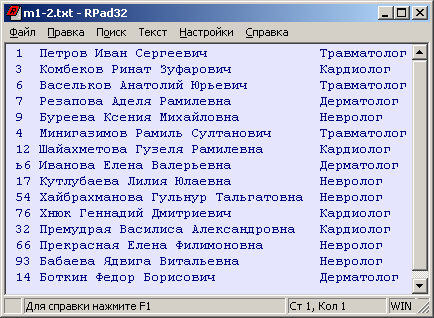


Очікуваний результат: повідомлення: «Mistake: FIO vracha ‘Петр5в Іван Сергійович’ ne bukvennie!»

Результат тестування: неуспіх

3) Мета тесту: перевірити функціонування програми, якщо табельний номер лікаря містить не тільки цифри

Зміст тесту: файл «m1-2.txt»

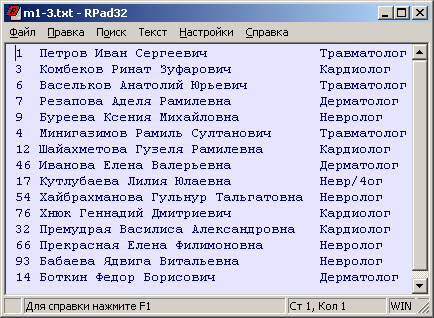


Очікуваний результат: повідомлення: «Mistake: Tabelniy nomer ‘ь6’ vracha ne chislovoy!»

Результат тестування: неуспіх

4) Мета тесту: перевірити функціонування програми, якщо спеціальність містить не тільки букви

Зміст тесту: файл «m1-3.txt»

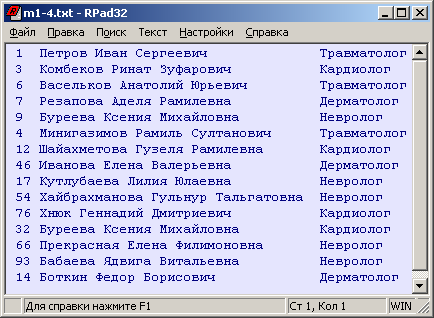


Очікуваний результат: повідомлення: «Mistake: naimenovanie spetsialnosti ‘Невр/4ог’ ne bukvennoe!»

Результат тестування: неуспіх

5) Мета тесту: перевірити функціонування програми при неунікальних ФІО лікаря

Зміст тесту: файл «m1-4.txt»

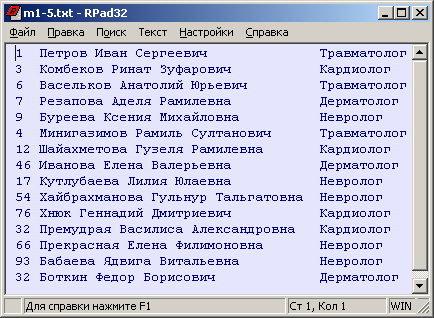


Очікуваний результат: повідомлення: «FIO vracha ‘Бурєєва Ксенія Михайлівна’ ne unikalni!»

Результат тестування: неуспіх

6) Мета тесту: перевірити функціонування програми при неунікальних табельних номерах лікарів

Зміст тесту: файл «m1-5.txt»



Очікуваний результат: повідомлення: «Tabelniy nomer vracha ‘32’ne unikalniy!»

Результат тестування: неуспіх

7) Мета тесту: перевірити функціонування програми при виявленні некоректності файлу F1

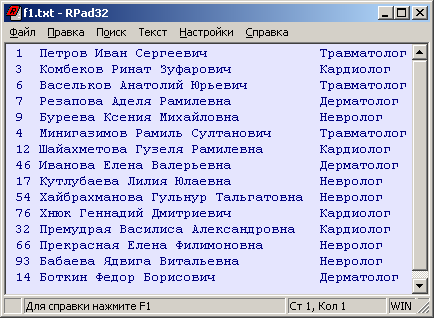
Зміст тесту: файл «m1-1.txt», «m1-2.txt», «m1-3.txt», «m1-4.txt», «m1-5.txt».

Очікуваний результат: повідомлення: «Obnaruzhena nekorrektnost faila F1! Vnesite ispravleniya»

Результат тестування: неуспіх

8) Мета тесту: перевірити функціонування програми при коректності файлу F1

Зміст тесту: файл «f1.txt».



Очікуваний результат: перехід до наступної функції.

Результат тестування: успішний

Висновок: В результаті роботи ми засвоїли прийоми програмування роботи з файлами і обробки файлової інформації.