МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования

Белорусский Государственный Университет Транспорта

Кафедра "Информационные технологии"

Курсовая работа

по предмету

"Информатика и информационные технологии"

Выполнил: студент группы Эм-21 Крупенков Е.А.

Проверил: преподаватель Балащенко Д. В.

Гомель 2010

Содержание

Введение

Листинг программы

Заключение

Список использованной литературы

# Введение

Язык программирования Си был разработан в лабораториях Bell Labs в период с 1969 по 1973 годы. Существует несколько легенд, касающихся причин разработки Си и его отношения к операционной системе UNIX, включая следующие:

1) Разработка Си стала результатом того, что его будущие авторы любили компьютерную игру, подобную популярной игре Asteroids (Астероиды). Они уже давно играли в неё на главном сервере компании, который был недостаточно мощным и должен был обслуживать около ста пользователей. Томпсон и Ритчи посчитали, что им не хватает контроля над космическим кораблём для того, чтобы избегать столкновений с некоторыми камнями. Поэтому они решили перенести игру на свободный PDP-7, стоящий в офисе. Однако этот компьютер не имел операционной системы, что заставило их её написать. В конце концов, они решили перенести эту операционную систему ещё и на офисный PDP-11, что было очень тяжело, потому что её код был целиком написан на ассемблере. Было вынесено предложение использовать какой-нибудь высокоуровневый портативный язык, чтобы можно было легко переносить ОС с одного компьютера на другой. Язык Си, который они хотели сначала задействовать для этого, оказался лишён функциональности, способной использовать новые возможности PDP-11. Поэтому они и остановились на разработке языка Си.

2) Самый первый компьютер, для которого была первоначально написана UNIX, предназначался для создания системы автоматического заполнения документов. Первая версия UNIX была написана на ассемблере. Позднее для того, чтобы переписать эту операционную систему, был разработан язык Си.

К 1973 году язык Си стал достаточно силён, и большая часть ядра UNIX, первоначально написанная на ассемблере PDP-11/20, была переписана на Си.

Язык С++ возник в начале 1980-х годов, когда сотрудник фирмы Bell Laboratories Бьёрн Страуструп придумал ряд усовершенствований к языку Си под собственные нужды.

Название "Си++" было придумано Риком Масситти (Rick Mascitti) и впервые было использовано в декабре 1983 года. Ранее, на этапе разработки, новый язык назывался "Си с классами".

Нововведениями С++ в сравнении с С являются:

* поддержка объектно-ориентированного программирования;
* поддержка обобщённого программирования через шаблоны;
* дополнительные типы данных;
* исключения;
* пространства имён;
* встраиваемые функции;
* перегрузка операторов;
* перегрузка имён функций;
* ссылки и операторы управления свободно распределяемой памятью;
* дополнения к стандартной библиотеке.

Язык С++ во многом является надмножеством языка С.

Целью данной курсовой работы является усовершенствование навыков работы с языками С и С++. Мне предстоит разработать и реализовать информационно-поисковый справочник "Блок питания". В программе будет реализовано динамическое распределение памяти, работа с файлами и две несложные сортировки. Решение этих задач позволит глубже понять возможности языка и получить практические навыки в написании программ.

# Листинг программы

**“my. h”**

Заголовочный файл, в нем подключены все необходимые библиотеки, а также объявлены используемые в программе функции и структура данных.

#include<stdio. h>

#include<stdlib. h>

#include<string. h>

#include<iostream>

#include<iomanip>

#include<fstream>

#include<istream>

struct e{

char year [5];

char name [40];

int a;

short p;

short db;

int number;

struct e \*next;

struct e \*pre;

struct e \*I;

};

int vvod (struct e \*k, int n);

void vyvodAll (struct e \*k);

void vyvodOne (struct e \*k, int n);

void max3 (struct e \*k);

void min4 (struct e \*k);

void buble (struct e \*\*head, int n);

void insertion (struct e \*\*head);

void f2s (struct e \*f, struct e \*s, struct e \*\*head);

void writeFileB (struct e \*k, int argc, char \*\*argv);

void readFileB (struct e \*\*head, int \*n, int argc, char \*\*argv);

void nameSort (char \*fileName);

void del (struct e \*\*head, int n);

void readName (char\* name);

void writeFileBC (struct e \*k, char \*fileName);

void readFileBC (struct e \*\*head, char \*fileName, int \*n);

int vvodC (struct e \*k, int n);

**“2. cpp”**

Файл, содержащий главною функцию main (). Она представляет собой меню программы.

#include "my. h"

struct e \*head;

int main (int argc, char \*argv []) {

int n=0,l;

head= (struct e\*) malloc (sizeof (struct e));

head->pre=NULL;

system ("clear");

while (l! =0) {

printf ("\nМеню: \n1-Ввод (111 - (С)) \n2-Вывод всех\n3-Вывод одной\n4-Max разъёмов\n5-Min мощности\n6-Сортировка пузырьком для количества разъёмов (по возростанию) \n7-Сортировка вставками для мощности (по убыванию) \n8-В файл (888 - (С)) \n9-Из файла (999 - (С)) \n10-Сортировка сохранённого файла\n11-Удаление записи\n0-Выход\n");

if (! scanf ("%d",&l)) {

l=31337;

while (getchar ()! ='\n')

continue;

}

switch (l) {

case 1:

system ("clear");

n=vvod (head,n);

break;

case 2:

system ("clear");

if (n! =0)

vyvodAll (head);

else

printf ("Введите что-нибудь\n");

break;

case 3:

system ("clear");

if (n! =0)

vyvodOne (head,n);

else

printf ("Введите что-нибудь\n");

break;

case 4:

system ("clear");

if (n! =0)

max3 (head);

else

printf ("Введите что-нибудь\n");

break;

case 5:

system ("clear");

if (n! =0)

min4 (head);

else

printf ("Введите что-нибудь\n");

break;

case 6:

system ("clear");

if (n==0) {

printf ("Введите что-нибудь\n");

break;

}

if (n==1) {

printf ("Всего одна запись. Нечего сортировать");

break;

}

buble (&head, n);

break;

case 7:

system ("clear");

if (n==0) {

printf ("Введите что-нибудь");

break;

}

if (n==1) {

printf ("Всего одна запись. Нечего сортировать");

break;

}

insertion (&head);

break;

case 8:

system ("clear");

if (n==0) {

printf ("Введите что-нибудь");

break;

}

else

writeFileB (head, argc, argv);

break;

case 9:

system ("clear");

readFileB (&head, &n, argc, argv);

break;

case 10:

system ("clear");

nameSort (argv [1]);

break;

case 11:

system ("clear");

if (n! =0) {

del (&head, n);

n--;

}

else

printf ("Введите что-нибудь");

break;

case 0:

break;

case 111:

system ("clear");

n=vvodC (head,n);

break;

case 888:

system ("clear");

if (n==0) {

printf ("Введите что-нибудь");

break;

}

else

writeFileBC (head, argv [2]);

break;

case 999:

system ("clear");

readFileBC (&head, argv [2], &n);

break;

default:

system ("clear");

printf ("Нет такого пункта меню");

continue;

}

}

return 0;

}

**“vvod. cpp”**

Функция vvod () считывает с клавиатуры данные, вводимые пользователем. Она динамически выделяет память и формирует в памяти связный список. Так же есть возможность добавлять элементы в существующий список.

Функция написана с использованием потоковых классов С++

Принимаемые значения: количество уже считанных элементов, указатель на первый элемент

Возвращаемые значения: количество считанных элементов

#include "my. h"

using namespace std;

int vvod (struct e \*k, int n) {

struct e \*q;

q=k;

int i=0,c;

char y;

if (n! =0) {

while (k->next! =NULL)

k=k->next;

k->next= (struct e\*) malloc (sizeof (struct e));

k=k->next;

k->pre=q;

}

for (; y! ='Y'; n++) {

while (cin. get ()! ='\n')

continue;

cout<<"Введите год БП №"<<n+1<<endl;

for (i=0;; i++) {

k->year [i] =cin. get ();

if (k->year [i] =='\n') {

k->year [i] ='\0';

break;

}

if (k->year [i] <48 || k->year [i] >57 || i>3) {

k->year [i] ='\0';

while (cin. get ()! ='\n')

continue;

break;

}

}

cout<<"Введите название БП №"<<n+1<<endl;

for (i=0;; i++) {

k->name [i] =cin. get ();

if (k->name [i] =='\n') {

k->name [i] ='\0';

break;

}

if (k->name [i] <31 || k->name [i] >126 || i>38) {

k->name [i] ='\0';

while (cin. get ()! ='\n')

continue;

break;

}

}

cout<<"Введите количество разьёмов БП №"<<n+1<<endl;

cin>>k->a;

while (cin. fail ()) {

cout<<"Недопустимое число. Попробуйте ещё раз"<<endl;

cin. clear ();

while (cin. get ()! ='\n')

continue;

cin>>k->a;

}

while (cin. get ()! ='\n')

continue;

cout<<"Введите мощность БП №"<<n+1<<endl;

cin>>k->p;

while (cin. fail ()) {

cout<<"Недопустимое число. Попробуйте ещё раз"<<endl;

cin. clear ();

while (cin. get ()! ='\n')

continue;

cin>>k->p;

}

cout<<"Закончить (y/any key)"<<endl;

while (cin. get ()! ='\n')

continue;

y=cin. get ();

y=toupper (y);

k->I=k;

k->number=n+1;

if (y=='Y') {

k->next=NULL;

}

else{

k->next= (struct e\*) malloc (sizeof (struct e));

k->next->pre=k;

k=k->next;

}

}

system ("clear");

printf ("Ввод закончен");

return n;

}

**“vyvodAll. cpp”**

Функция vyvodAll () выводит все записи на экран в виде отформатированной таблицы.

Принимаемые значения: указатель на первый элемент

Возвращаемые значения: нет

#include "my. h"

void vyvodAll (struct e \*k) {

int j;

system ("clear");

printf (". | Год | Название | Разъёмы | Шум | Мощность |\n");

for (j=0; j<77; j++)

printf ("=");

while (1) {

printf ("\n% -2d|% -5s|% -40s|% -9d|% -5hd|% -9hd|\n",k->number,k->year,k->name,k->a,k->db,k->p);

for (j=0; j<77; j++)

printf ("-");

k=k->next;

if (k==NULL)

break;

}

return;

}

**“vyvodOne. cpp”**

Функция vyvodOne () выводит одну запись на экран по её номеру.

Принимаемые значения: количество считанных элементов, указатель на первый элемент. Возвращаемые значения: нет

#include "my. h"

void vyvodOne (struct e \*k, int n) {

int j, i;

system ("clear");

printf ("Какую запись вывести? \n");

while (! scanf ("%d",&i)) {

printf ("Неверное число\n");

while (getchar ()! ='\n')

continue;

}

if (i>n || i<1)

printf ("Нет такой записи\n");

else{

while (k->number! =i) {

k=k->next;

}

printf (". | Год | Название | Разъёмы | Шум | Мощность |\n");

for (j=0; j<77; j++)

printf ("=");

printf ("\n% -2d|% -5s|% -40s|% -9d|% -5hd|% -9hd|\n",k->number,k->year,k->name,k->a,k->db,k->p);

for (j=0; j<77; j++)

printf ("-");

}

return;

}

**“max3. cpp”**

Функция max3 () - поиск БП с максимальным количеством разъёмов

Принимаемые значения: указатель на первый элемент

Возвращаемые значения: нет

#include"my. h"

void max3 (struct e \*k) {

struct e \*q;

q=k;

while (1) {

if (k->a>q->a)

q=k;

if (k->next==NULL)

break;

k=k->next;

}

printf ("Максимальным количеством разъёмов обладает БП #%d: %d",q->number,q->a);

return;

}

**“min4. cpp”**

функция min4 () - поиск БП минимальной мощности

Принимаемые значения: указатель на первый элемент

Возвращаемые значения: нет

#include"my. h"

void min4 (struct e \*k) {

struct e \*q;

q=k;

while (1) {

if (k->p<q->p)

q=k;

if (k->next==NULL)

break;

k=k->next;

}

printf ("Минимальным количеством разъёмов обладает БП #%d: %hhd",q->number,q->p);

return;

}

**“** **buble. cpp”**

Функция buble () - сортировка методом пузырька для количества разъёмов (по возростанию)

Принимаемые значения: количество считанных элементов, указатель на указатель на первый элемент

Возвращаемые значения: нет

#include "my. h"

void buble (struct e \*\*head, int n) {

struct e \*k;

k=\*head;

int f,j;

for (j=0; j< (n-1); j++) {

for (f=0; f< (n-1); f++) {

if ( (k->a) > (k->next->a)) {

f2s (k,k->next,head);

k=k->pre;

k->number=k->number+k->next->number;

k->next->number=k->number-k->next->number;

k->number=k->number-k->next->number;

}

k=k->next;

}

k=\*head;

}

printf ("Отсортировано");

return;

}

**“** **insertion. cpp”**

Функция insertion () - сортировка вставками для мощности (по убыванию)

Принимаемые значения: указатель на указатель на первый элемент

Возвращаемые значения: нет

#include "my. h"

void insertion (struct e \*\*head) {

struct e \*q;

q= (\*head) - >next;

while (q->next! =NULL) {

for (q= (\*head) - >next; q->next! =NULL && q->p<=q->pre->p;)

q=q->next;

if (q->p>q->pre->p) {

while (q->p>q->pre->p) {

f2s (q->pre, q, head);

q->number=q->number+q->next->number;

q->next->number=q->number-q->next->number;

q->number=q->number-q->next->number;

if (q->pre==NULL)

break;

}

}

}

printf ("Отсортировано");

return;

}

**“** **writeFileB ”**

Функция writeFileB () - функция записи данных из памяти в бинарный файл. Если программа была запущена с параметрами, то в качестве имени файла выступит аргумент из командной строки.

Функция написана с использованием потоковых классов С++

Принимаемые значения: количество аргументов, принятых функцией main () из командной строки, указатель на массив указателей на строки, считанные как аргумент из командной строки

Возвращаемые значения: нет

#include "my. h"

using namespace std;

void writeFileB (struct e \*k, int argc, char \*\*argv) {

int i;

char name [11];

if (argc<2) {

while (cin. get ()! ='\n')

continue;

readName (name);

if (name [0] =='\0')

return;

}

else

strcpy (name, \* (argv+1));

ofstream f (name, ios:: binary);

if (f. fail ()) {

cout<<"Невозможно создать файл c именем"<<name<<endl;

return;

}

system ("clear");

cout<<"Файл с именем '"<<name;

while (1) {

f. write ( (char\*) k->year,5);

f. write ( (char\*) k->name,40);

f. write ( (char\*) &k->a,4);

f. write ( (char\*) &k->db,2);

f. write ( (char\*) &k->p,2);

if (k->next==NULL)

break;

else

k=k->next;

}

if (f. fail ())

cout<<"' не создан";

if (! f. fail ())

cout<<"' создан";

f. close ();

return;

}

**“** **readFileB ”**

Функция readFileB () - функция для чтения данных из файла.

Если программа была запущена с параметрами, то в качестве имени файла выступит аргумент из командной строки.

Функция написана с использованием потоковых классов С++

Принимаемые значения: количество аргументов, принятых функцией main () из командной строки; указатель на массив указателей на строки, считанные как аргумент из командной строки; указатель на указатель на первый элемент; указатель на количество считанных элементов

Возвращаемые значения: нет

#include "my. h"

using namespace std;

void readFileB (struct e \*\*head, int \*n, int argc, char \*\*argv) {

char name [10];

struct e \*k;

\*head= (struct e\*) malloc (sizeof (struct e));

k=\*head;

if (argc>2)

strcpy (name, \* (argv+1));

else{

while (cin. get ()! ='\n')

continue;

readName (name);

if (name [0] =='\0')

return;

}

ifstream f (name, ios:: binary);

if (f. fail ()) {

cout<<"Невозможно открыть файл c именем "<<name<<endl;

return;

}

else

cout<<"Файл с именем '"<<name<<"' прочитан";

f. clear ();

k->pre=NULL;

for (\*n=1;; (\*n) ++) {

f. read ( (char\*) k->year,5);

f. read ( (char\*) k->name,40);

f. read ( (char\*) & (k->a),4);

f. read ( (char\*) & (k->db),2);

f. read ( (char\*) & (k->p),2);

k->I=k;

k->number=\*n;

if (f. peek ()! =EOF) {

k->next= (struct e\*) malloc (sizeof (struct e));

k->next->pre=k;

k=k->next;

}

else{

k->next=NULL;

break;

}

}

f. close ();

return;

}

**“** **nameSort ”**

Функция nameSort () - функция для сортировки дискового файла по по полю с именем БП. Сортировка происходит методом пузырька. Если программа была запущена с параметрами, то в качестве имени файла выступит аргумент из командной строки.

Принимаемые значения: Указатель на строку, содержащую имя файла

Возвращаемые значения: нет

#include "my. h"

void nameSort (char \*fileName) {

FILE \*f;

int n=1, i,l,k=0;

char a [51];

char b [51];

char name [10];

if (! (f=fopen (fileName,"rb+"))) {

while (getchar ()! ='\n')

continue;

do{

readName (name);

if (name [0] =='\0')

return;

printf ("Невозможно открыть файл c именем '%s'. \n",name);

}while (! (f=fopen (name,"rb+")));

system ("clear");

printf ("Файл с именем '%s' отсортирован", name);

}

else

printf ("Файл с именем '%s' отсортирован", fileName);

while (n==1) {

n=0;

fseek (f,0,SEEK\_SET);

while (! feof (f)) {

fread (a,1,51,f);

fread (b,1,51,f);

if (feof (f))

break;

if ( (strcmp ( (a+5), (b+5))) >0) {

fseek (f,-102,SEEK\_CUR);

fwrite (b,51,1,f);

fwrite (a,51,1,f);

n=1;

}

fseek (f,-51,SEEK\_CUR);

}

}

fclose (f);

return;

}

**“** **del. cpp ”**

Функция del () - удаляет одну запись по её номеру

Принимаемые значения: Указатель на указатель на первый элемент, количество считанных элементов

Возвращаемые значения: нет

#include "my. h"

void del (struct e \*\*head, int n) {

struct e \*k;

k=\*head;

int d;

if (n==1) {

printf ("Запись удалена");

return;

}

printf ("Введите номер удаляемой записи\n");

while (! scanf ("%d",&d)) {

printf ("Неверное число\n");

while (getchar ()! ='\n')

continue;

}

if ( (d>n) || (n<=0)) {

printf ("Такой записи нет");

return;

}

while (k->number! =d)

k=k->next;

if (d==1) {

\*head=k->next;

}

else

k->pre->next=k->next;

if (d! =n)

k->next->pre=k->pre;

free (k);

while (k->next! =NULL) {

k=k->next;

k->number--;

}

printf ("Запись удалена");

return;

}

**“** **readName. cpp ”**

Функция readName () - вспомогательная функция. Она используется в функциях работающих с файлами. Считывает имя файла с клавиатуры и проверяет его корректность.

Принимаемые значения: Указатель на строку, в которую будут записаны считанные данные

Возвращаемые значения: нет

#include"my. h"

void readName (char\* name) {char i, c;

printf ("Введите имя файла (не более 10 символов) \nЕсли вы передумали, просто нажмите клавишу 'Ввод'\n");

c=1;

while (c==1) {

c=0;

fgets (name,11,stdin);

for (i=0; i<10 && \* (name+i)! ='\n'; i++)

if (\* (name+i) >126 || \* (name+i) <32)

c=1;

if (c==1)

printf ("Название содержит недопустимые символы. Попробуйте ещё раз\n");

if (i==10)

while ( (getchar ())! ='\n')

continue;

else

\* (name+i) ='\0';

}

return;

}

**“** **f2s. cpp ”**

Функция f2s () - вспомогательная функция. Она используется в функциях сортировки. Предназначена для обмена местами двух соседних элементов в связном списке.

Принимаемые значения: указатель на первый элемент, указатель на второй элемент, указатель на указатель на первый элемент

Возвращаемые значения: нет

#include "my. h"

void f2s (struct e \*f, struct e \*s, struct e \*\*head) {

int n;

if (f! =\*head)

f->pre->next=s;

else

\*head=s;

if (s->next! =NULL)

s->next->pre=f;

s->pre=f->pre;

f->pre=s;

f->next=s->next;

s->next=f;

return;

}

**“Makefile”**

Файл необходимый для работы утилиты make

2: my. h vvod. o vyvodAll. o vyvodOne. o 2. cpp max3. o min4. o buble. o insertion. o f2s. o readFileB. o writeFileB. o nameSort. o del. o readName. o readFileBC. o writeFileBC. o vvodC. o

g++ - o 2 2. cpp vvod. o vyvodAll. o vyvodOne. o max3. o min4. o insertion. o buble. o f2s. o writeFileB. o readFileB. o nameSort. o del. o readName. o readFileBC. o writeFileBC. o vvodC. o

vvod. o: my. h vvod. cpp

g++ - c vvod. cpp

vvodC. o: my. h vvodC. cpp

g++ - c vvodC. cpp

vyvodAll. o: my. h vyvodAll. cpp

g++ - c vyvodAll. cpp

vyvodOne. o: my. h vyvodOne. cpp

g++ - c vyvodOne. cpp

max3. o: my. h max3. cpp

g++ - c max3. cpp

min4. o: my. h min4. cpp

g++ - c min4. cpp

buble. o: my. h buble. cpp f2s. cpp

g++ - c buble. cpp

insertion. o: my. h insertion. cpp f2s. cpp

g++ - c insertion. cpp

f2s. o: my. h f2s. cpp

g++ - c f2s. cpp

readFileB. o: my. h readFileB. cpp

g++ - c readFileB. cpp

writeFileB. o: my. h writeFileB. cpp

g++ - c writeFileB. cpp

readFileBC. o: my. h readFileBC. cpp

g++ - c readFileBC. cpp

writeFileBC. o: my. h writeFileBC. cpp

g++ - c writeFileBC. cpp

nameSort. o: my. h nameSort. cpp

g++ - c nameSort. cpp

del. o: my. h del. cpp

g++ - c del. cpp

readName. o: my. h readName. cpp

g++ - c readName. cpp

clean:

rm - rf \*. o 2

В процессе написания программы часть функций была переведена на язык С++. Ниже представлены листинги этих функций написанные на языке С. Свойства этих функций такие же как и у их аналогов, поэтому я не буду делать к ним описание, а просто приведу их исходный код.

**“readFileBC. с”**

#include "my. h"

void readFileBC (struct e \*\*head, char \*fileName, int \*n) {

int i;

char name [10];

struct e \*k;

\*head= (struct e\*) malloc (sizeof (struct e));

k=\*head;

FILE \*f;

if (! (f=fopen (fileName,"rb"))) {

while (getchar ()! ='\n')

continue;

do{

readName (name);

if (name [0] =='\0')

return;

printf ("Невозможно открыть файл c именем '%s'. \n",name);

}while (! (f=fopen (name,"rb")));

system ("clear");

printf ("Файл с именем '%s' прочитан", name);

}

else

printf ("Файл с именем '%s' прочитан", fileName);

k->pre=NULL;

(\*n) =0;

while (1) {

fread (k->year,1,5,f);

fread (k->name,1,40,f);

fread (& (k->a),4,1,f);

fread (& (k->db),2,1,f);

fread (& (k->p),2,1,f);

(\*n) ++;

k->I=k;

k->number=\*n;

if ( (fread (&i,4,1,f))) {

k->next= (struct e\*) malloc (sizeof (struct e));

k->next->pre=k;

k=k->next;

}

else{

k->next=NULL;

break;

}

fseek (f,-4,SEEK\_CUR);

}

fclose (f);

return;

}

**“vvodC. с”**

#include "my. h"

int vvodC (struct e \*k, int n) {

struct e \*q;

q=k;

int i=0,c;

char y;

while ( (getchar ())! ='\n')

continue;

if (n! =0) {

while (k->next! =NULL)

k=k->next;

k->next=malloc (sizeof (struct e));

k=k->next;

k->pre=q;

}

for (; y! ='y'; n++) {

printf ("Введите год БП №%d\n",n+1);

c=1;

while (c==1) {

c=0;

fgets (k->year,5,stdin);

for (i=0; i<4 && k->year [i]! ='\n'; i++)

if (k->year [i] >57 || k->year [i] <48)

c=1;

if (c==1)

printf ("Недопустимый символ\n");

if (i==4)

while ( (getchar ())! ='\n')

continue;

else

k->year [i] ='\0';

}

printf ("Введите название БП №%d\n",n+1);

c=1;

while (c==1) {

c=0;

fgets (k->name,40,stdin);

for (i=0; i<39 && k->name [i]! ='\n'; i++)

if (k->name [i] >126 || k->name [i] <31)

c=1;

if (c==1)

printf ("Недопустимый символ\n");

if (i==39)

while ( (getchar ())! ='\n')

continue;

else

k->name [i] ='\0';

}

printf ("Введите количество разьёмов БП №%d\n",n+1);

while (! scanf ("%d",&k->a))

while (getchar ()! ='\n')

continue;

printf ("Введите мощность БП №%d\n",n+1);

while (! scanf ("%hd",&k->p))

while (getchar ()! ='\n')

continue;

while ( (getchar ())! ='\n')

continue;

printf ("Закончить (y/any key) \n");

scanf ("%c",&y);

while ( (getchar ())! ='\n')

continue;

k->this=k;

k->number=n+1;

if (y=='y') {

k->next=NULL;

}

else{

k->next=malloc (sizeof (struct e));

k->next->pre=k;

k=k->next;

}

}

system ("clear");

printf ("Ввод закончен");

return n;

}

**“writeFileBC. с”**

#include "my. h"

void writeFileBC (struct e \*k, char \*fileName) {

FILE \*f;

int i;

char name [11];

if (! (f=fopen (fileName,"wb"))) {

while (getchar ()! ='\n')

continue;

do{

readName (name);

if (name [0] =='\0')

return;

printf ("Невозможно создать файл c именем '%s'. \n",name);

}while (! (f=fopen (name,"wb")));

system ("clear");

printf ("Файл с именем '%s' создан", name);

}

else

printf ("Файл с именем '%s' создан", fileName);

while (1) {

fwrite (& (k->year),5,1,f);

fwrite (& (k->name),40,1,f);

fwrite (& (k->a),sizeof (int),1,f);

fwrite (& (k->p),sizeof (short),1,f);

if (k->next==NULL)

break;

else

k=k->next;

}

fclose (f);

return;

}

Вместе с исходным кодом программы прилагается файл такого содержания

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| год | название | разъёмы | шум | мощность |
| 2009 | Thermaltake EVO\_Blue 650W (W0307RU) | 24 | 22 | 650 |
| 2008 | ETG ESP-550X-S 550W | 24 | 30 | 550 |
| 2009 | FSP Group ATX-600PNR 600W | 20 | 27 | 600 |
| 2009 | Antec HCG-620 620W | 20 | 22 | 620 |
| 2010 | Thermaltake TR2 RX 750W | 24 | 25 | 750 |

# Заключение

Результатом выполнения данной работы является информационно-поисковый справочник "Блок питания". В нём были реализованы следующие функции:

1. Создание справочника в памяти
2. Вывод справочника на экран
3. Удаление записи
4. Два способа сортировки
5. Сортировка дискового файла
6. Сохранение справочника в файле
7. Открытие справочника из файла

Я пришел к выводу, что связные списки хоть и усложняют обработку данных, но при этом дают программе гибкость в вопросе выделения памяти. Так же такой способ хранения данных позволяет экономить ресурсы компьютера, ведь место выделяется лишь по необходимости, а не резервируется при запуске программы.

От себя могу сказать, что я не только углубил имеющиеся знания, но и получил новые.

# Список использованной литературы

1. Герберт Шилдт. Полный справочник по С++. Ossborne. 2003
2. Стивен Прата. /Язык программирования С. - М. Издательство "Диасофт", 2002. - 896 с.
3. П. Дейтел, X. Дейтел. / Как программировать на С++. - М, Издательство "Бином" 2001 - 1152 стр.