МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО

Учреждение образования Республики Беларусь

«Гомельский государственный технический

университет им. П.О.Сухого»

Кафедра «Автоматизированных и информационных систем»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 11

по дисциплине «Операционные системы»

на тему: «Управление файловой системой»

Выполнил студент гр. ИТИ-11:

Бут-Гусаим Д.Г. Принял преподаватель-стажёр: Карась О.В.

Гомель 2022

**Цель работы:** разработать модель файловой системы

**Задание:** разработать приложение, создающее виртуальный файл и позволяющее - форматировать виртуальный файл с возможностью задания размера кластера; - создавать каталоги в виртуальном файле; - производить учѐт свободного пространства; - реализовывать поиск файлов и директорий; - сохранять в виртуальный файл файлы с жѐсткого диска; - удалять файлы из виртуального файла; - записывать на жѐсткий диск файлы из виртуального файла; - создавать в виртуальном файле текстовые файлы; - предоставлять возможность редактировать текстовые файлы внутри виртуального файла. Файловую систему внутри виртуального файла выбрать согласно варианта.

|  |  |
| --- | --- |
| **Вариант** | **Дополнительная утилита** |
| 2 | Скандиск |

Работа скандиска представлено на рисунке 1.

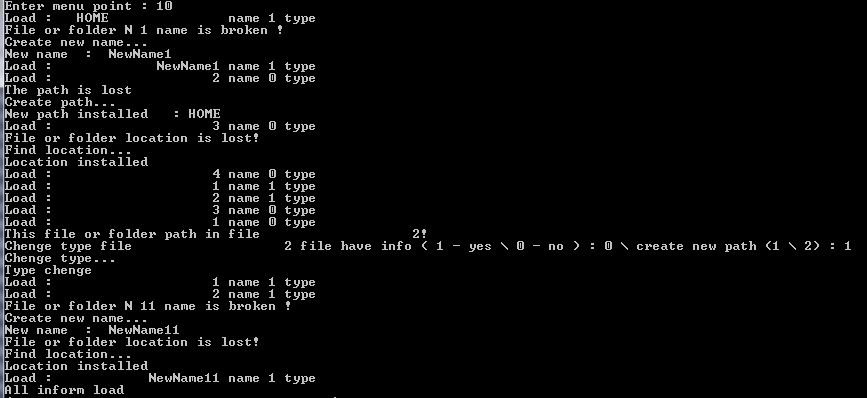


Рисунок 1 – Работа СканДиска

В ОС используется несколько методов выделения файлу дискового пространства. Для каждого из методов запись в директории, соответствующая символьному имени файла, содержит указатель, следуя которому можно найти все блоки данного файла. Основная память представляет собой упорядоченный массив однобайтовых ячеек, каждая из которых имеет свой уникальный адрес (номер). Вторичную память (это главным образом диски) также можно рассматривать как одномерное линейное адресное пространство, состоящее из последовательности байтов. В отличие от оперативной памяти, она является энергонезависимой, имеет существенно большую емкость и используется в качестве расширения основной памяти. Многоуровневую схему используют следующим образом. Информация, которая находится в памяти верхнего уровня, обычно хранится также на уровнях с большими номерами. Если процессор не обнаруживает нужную информацию на i-м уровне, он начинает искать ее на следующих уровнях. Когда нужная информация найдена, она переносится в более быстрые уровни

**Вывод:** в процессе выполнения данной лабораторной работы была разработана модель файловой системы.

**ПРИЛОЖЕНИЕ А:**

void ProvName(char \*n,int amt){

bool test = false;

for(int i = 0; i < strlen(n); i++){

if( (\*(n+i) > 47 && \*(n+i)<58) || (\*(n+i) > 64 && \*(n+i)<91) || (\*(n+i) > 96 && \*(n+i)<123) ) test = true;

}

if(!test){

printf("File or folder N %d name is broken !\n",amt);

printf("Create new name...\n");

\*(n+0) = 'N';

\*(n+1) = 'e';

\*(n+2) = 'w';

\*(n+3) = 'N';

\*(n+4) = 'a';

\*(n+5) = 'm';

\*(n+6) = 'e';

\*(n+7) = '\0';

char tmp[4];

int code;

itoa(amt,tmp,10);

tmp[3] = '\0';

strcat(n,tmp);

printf("New name : %s\n",n);

}

}

bool ProvInfo(char \*n){

bool test = false;

for(int i = 0; i < strlen(n); i++){

if( \*(n+i) != ' ' ){

test = true;

}

}

return test;

}

int ScanDisk(File \*d,float \*m){

FILE \*f = fopen("hdd.txt","rt+");

int i = 0,tmp,l;

while(!feof(f)){

fseek(f,(i\*284)+1,0);

fgets((d+i)->name,20,f);

ProvName((d+i)->name,i);

(d+i)->name[20]='\0';

fseek(f,(i\*284)+20,0);

fscanf(f,"%d",&tmp);

(d+i)->type = tmp;

fseek(f,(i\*284)+25,0);

fgets((d+i)->info,254,f);

(d+i)->info[255]='\0';

if(ProvInfoL((d+i)->info)){

(\*m) += ProvInfoL((d+i)->info);

}

fseek(f,(i\*284)+278,0);

fscanf(f,"%d",&(d+i)->pos);

if((d+i)->pos != i){

printf("File or folder location is lost!\n");

printf("Find location...\n");

(d+i)->pos = i;

printf("Location installed\n");

}

fseek(f,(i\*284)+281,0);

fscanf(f,"%d",&(d+i)->per);

if(((d+i)->per > i)){

printf("The path is lost\n");

printf("Create path...\n");

(d+i)->per = 0;

printf("New path installed : HOME\n");

}else{

if((d+(d+i)->per)->type == 0) {

printf("This file or folder path in file %s!\n",(d+(d+i)->per)->name);

printf("Chenge type file %s file have info ( 1 - yes \\ 0 - no ) : %d \\ create new path (1 \\ 2) : ",(d+(d+i)->per)->name,ProvInfo((d+(d+i)->per)->info));

scanf("%d",&l);

if(l == 2){

printf("The path is lost\n");

printf("Create path...\n");

(d+i)->per = 0;

printf("New path installed : HOME\n");

}else{

printf("Chenge type...\n");

(d+(d+i)->per)->type = 1;

(d+(d+i)->per)->info[0] = '\0';

printf("Type chenge\n");

}

}

}

printf("Load : ");

printf("%20s name ",(d+i)->name);

printf("%d type ",(d+i)->type);

printf("%255s info ",(d+i)->info);

printf("%d %d pos per \n",(d+i)->pos,(d+i)->per);

i++;

}

fclose(f);

printf("All inform load\n");

return(i-1);

}

int main(){

File disk[100],derect;

strcpy(disk[0].name,"HOME \0");

disk[0].type = 1;

disk[0].pos = 0;

disk[0].per = 0;

disk[0].info[0] = '\0';

derect = disk[0];

int menu=0,i,sizeKlast,col=1;

float memory;

printf("Enter size of cluster : ");

sizeKlast = ReadValueInt();

memory = sizeKlast \* 4.096;

while(menu != 11){

printf("<========================================>\n");

printf("<= %-8.2f =>\n",memory);

printf("<========================================>\n");

printf("<========================================>\n");

printf("<= %-20s =>\n",derect.name);

printf("<========================================>\n");

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

if(derect.pos == disk[i].per && i!=0)

if(disk[i].type == 0) printf("<-%-32s(file)->\n",disk[i].name);

else printf("<-%-30s(folder)->\n",disk[i].name);

}

printf("<========================================>\n");

printf("<==================MENU==================>\n");

printf("<========================================>\n");

printf("<-1-Create folder------------------------>\n");

printf("<-2-Open folder-------------------------->\n");

printf("<-3-Create file-------------------------->\n");

printf("<-4-Open file---------------------------->\n");

printf("<-5-Find file or folder------------------>\n");

printf("<-6-Delete file or folder---------------->\n");

printf("<-7-Back--------------------------------->\n");

printf("<-8-Load hdd----------------------------->\n");

printf("<-9-Save hdd----------------------------->\n");

printf("<-10-CHKDSK------------------------------>\n");

printf("<-11-Exit-------------------------------->\n");

printf("Enter menu point : ");

menu = ReadValueInt();

switch(menu){

case 1:{

CreateFolder(disk,derect,&col);

printf("%d",col);

break;

}

case 2:{

ChengeDer(disk,&derect,&col);

break;

}

case 3:{

CreateFile(disk,derect,&col);

if(memory -= strlen(disk[col].info) < 0){

col--;

printf("Out of memory!\n");

}else{

memory -= strlen(disk[col].info);

}

break;

}

case 4:{

OpenFile(disk,derect,&col);

break;

}

case 5:{

Find(disk,&col);

break;

}

case 6:{

Del(disk,derect,&col,&memory);

break;

}

case 7:{

derect = disk[derect.per];

break;

}

case 8:{

col = Load(disk,&memory);

break;

}

case 9:{

Save(disk,&col);

break;

}

case 10:{

col = ScanDisk(disk,&memory);

break;

}

case 11:{

printf("Exit..............\n");

break;

}

default:{

printf("Incorrect point!\n");

break;

}

}

}

}