**ФЕДЕРАЛЬНОЕ АГЕНСТВО ПО ОБРАЗОВАНИЮ**

**ОРЛОВСКИЙ ГОСУДАРСТВЕННЫЙ**

**ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**ФАКУЛЬТЕТ ДИСТАНЦИОННОГО ОБУЧЕНИЯ**

Допустить к защите «\_\_\_»\_\_\_\_\_\_\_\_2009г.

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Курсовая работа

по дисциплине «Вычислительные системы, сети и телекоммуникации»

тема: «Работа с каталогами и подкаталогами. Работа с файлами»

|  |  |
| --- | --- |
| Работу выполнил студент  **Осипов А.С.**  Группа **3-1И**  ЦДО **г.Орел** | ***Замечания по работе*** |
| Работу проверила  **Костенко Т.П.** |

Курсовая работа защищена с оценкой \_\_\_\_\_\_\_\_\_\_ «\_\_\_»\_\_\_\_\_\_\_\_\_\_200\_\_г.

Подпись преподавателя \_\_\_\_\_\_\_\_\_\_\_\_\_

**г. Орел, 2009г.**

**Содержание**

Введение..................................................................................................................3

I. Теоретическая часть............................................................................................5

1.1 Работа с каталогами диска.......................................................................5

1.2 Чтение/изменение корневого каталога..................................................8

1.3 Создание/удаление подкаталога............................................................11

1.4 Чтение/изменение подкаталога..............................................................12

II. Практическая часть ..........................................................................................15

Заключение.............................................................................................................17

Библиографический список использованной литературы……………….…18

Приложение А. Листинг программы...................................................................19

Приложение Б. Листинг программы....................................................................25

**Введение**

Компьютер понимает только один язык - язык машинных команд. Конечно, для общения и работы с компьютером необязательно знать этот язык, но практически любой профессиональный программист рано или поздно сталкивается с необходимостью его изучения. К счастью, при этом человеку не нужно пытаться постичь значение различных комбинаций двоичных чисел, так как еще в 50-е г.г. программисты стали использовать для программирования символический аналог машинного языка, который назвали языком ассемблера. Этот язык точно отражает все особенности машинного языка. В отличие от языков высокого уровня, язык ассемблера для каждого типа компьютеров свой.

Самую эффективную программу можно написать только на ассемблере, так как этот язык является «родным» для компьютера. Это очень трудоемкий и требующий большого внимания и практического опыта процесс. Поэтому реально на ассемблере пишут в основном программы, которые должны обеспечить эффективную работу с аппаратной частью компьютера. Иногда на ассемблере пишутся критичные ко времени выполнения или расходованию памяти фрагменты программы. Впоследствии они оформляются в виде подпрограмм и совмещаются с кодом на языке высокого уровня.

В курсовой работе будет рассмотрены некоторые задачи работы с каталогами диска.

Каждый диск имеет один корневой каталог, с которого начинается поиск всех остальных каталогов. Корневой каталог может содержать элементы, указывающие на подкаталоги, которые в свою очередь могут содержать ссылки на другие подкаталоги, образуя древовидную структуру каталогов. Корневой каталог всегда расположен в определенных секторах диска; подкаталоги хранятся как обычные дисковые файлы, поэтому они могут быть расположены в любом месте на диске. Отметим, что фиксированный диск может содержать до четырех корневых каталогов, если он разбит на разделы, хотя MS DOS «видит» только один корневой каталог. Каталоги могут иметь различные размеры, в зависимости от размера диска и его разбиения на разделы.

**I. Теоретическая часть.**

**1.1 Работа с каталогами диска.**

Каждый диск имеет один корневой каталог, с которого начинается поиск всех остальных каталогов. Корневой каталог может содержать элементы, указывающие на подкаталоги, которые в свою очередь могут содержать ссылки на другие подкаталоги, образуя древовидную структуру каталогов. Корневой каталог всегда расположен в определенных секторах диска; подкаталоги хранятся как обычные дисковые файлы, поэтому они могут быть расположены в любом месте диска. Отметим, что фиксированный диск может содержать до четырех корневых каталогов, если он разбит на разделы, хотя MS DOS "видит" только один корневой каталог. Каталоги могут иметь различные размеры, в зависимости от размера диска и его разбиения на разделы. В следующей таблице приведены размеры и позиции корневых каталогов для разных типов дисков:

|  |  |  |  |
| --- | --- | --- | --- |
| Тип диска | Размер каталога | Число элементов | Начальный сектор |
| дискета 160К | 4 сектора | 64 | 9 |
| дискета 180К | 4 сектора | 64 | 9 |
| дискета 320К | 7 секторов | 112 | 15 |
| дискета 360К | 7 секторов | 112 | 15 |
| дискета 1.2М | 14 секторов | 224 | 29 |
| жесткий 10М | ----------переменные------------ | | |
| жесткий 20М | ----------переменные------------ | | |

В зависимости от разбиения на разделы фиксированный диск может иметь различные размеры каталога и номер начального сектора. Если весь диск отведен для MS DOS, то на XT и AT под корневой каталог отводится 32 сектора, что позволяет иметь в нем 512 элементов.

Как корневой каталог, так и подкаталоги, используют 32 байта для хранения информации об одном файле, независимо от типа диска. Таким образом в каждом секторе может храниться информация о 16-ти элементах каталога. Каждое 32-байтное поле разбито следующим образом:

|  |  |
| --- | --- |
| Байты |  |
| 0-7 | Имя файла |
| 8-10 | Расширение файла |
| 11 | Атрибут файла |
| 12-21 | Зарезервировано |
| 22-23 | Время последнего доступа к файлу |
| 24-25 | Дата последнего доступа к файлу |
| 26-27 | Начальный кластер |
| 28-31 | Размер файла |

Точка между именем файла и его 3-байтным расширением не хранится. Все поля выровнены на левую границу, а пустые байты заполняются пробелами (код ASCII 32). Атрибут файла определяет является ли файл спрятанным, защищенным от записи и т.д. [5.2.6]. Он определяет также специальные элементы каталога, такие как подкаталоги или метка тома. Информация о времени и дате упакована, поэтому для чтения этих значений требуются битовые операции [5.2.5].

Начальный кластер указывает на позицию в таблице размещения файлов (FAT), которая обсуждалась в [5.1.1]. FAT хранит информацию о свободном пространстве на диске, а также отводит сектора при записи файла. FAT отводит дисковое пространство порциями, большими чем 1 сектор, которые называются кластерами. Файл расположен в цепочке кластеров и FAT содержит соответствующую цепочку элементов, указывающих, где эти кластеры расположены на диске. Каталог должен указывать на начальное звено цепочки элементов файла в FAT, и эта информация содержится в поле начальный номер кластера. Поскольку файл обычно занимает последний отведенный ему кластер не целиком, то поле размер файла хранит точную длину файла в байтах.

**1.2 Чтение/изменение корневого каталога.**

Каталоги диска подразделяются на корневой каталог (обсуждаемый здесь) и подкаталоги (обсуждаемые в [5.2.3]). Когда пользователь программы вводит имя какого-либо файла для работы, то бывает полезным проверить, имеется ли этот файл на самом деле. Обычно изменения в корневом каталоге производятся в ходе обычных файловых операций или с помощью специальных функций DOS. Однако можно работать с каталогом напрямую. Большая нужда в таком подходе возникает при работе на языках высокого уровня, где утилиты DOS по большей части недоступны.

Корневой каталог читается и изменяется загрузкой его в память с использованием подхода, показанного в [5.4.2], когда читаются абсолютные сектора диска. Эти операции не оставляют места между секторами, когда они загружаются в память. Буфер, содержащий данные сектора, может рассматриваться как набор 32-байтных полей и пара указателей, которые могут использоваться для движения по каталогу. Один указатель всегда кратен 32 и указывает на начало элемента каталога. Второй указатель добавляется к первому и указывает на одно из полей в 32-байтном элементе. Данные в памяти могут быть изменены требуемым образом, а затем весь буфер записывается обратно на диск.

Имеется два метода чтения абсолютных секторов диска и в обоих случаях только одно число отличает случаи чтения и записи. Поскольку ошибка при записи на диск может легко повредить все содержимое диска, то надо действовать аккуратно. Сначала убедитесь, что операция чтения сектора выполнена верно во всех отношениях. После этого можно попробовать записать на диск, взяв точную копию кода, использованного для чтения и заменив только номер функции.

MS DOS обеспечивает две пары функций для поиска файлов, одна для файлов, открытых методом управляющих блоков файла, а другая - для файлов, открытых методом дескриптора файлов. Одна из функций каждой пары ищет первое появление имени файла в каталоге, а другая ищет последующие появления, когда в имени файла содержатся джокеры. Только метод, использующий дескриптор файла позволяет искать подкаталоги.

**Метод FCB:**

Функция 11Н прерывания 21H ищет первое появление файла. Установите DS:DX на неоткрытый FCB и выполните функцию. При возврате AL будет содержать 0, если файл найден, и FF - если нет. DTA заполняется информацией из каталога. Для обычных FCB первый байт DTA содержит номер накопителя (1=А и т.д.), а следующие 32 байта содержат элемент каталога. Для расширенного FCB первые 7 байтов файла копируются в первые 7 байтов расширенного FCB, восьмой байт указывает на накопитель, а следующие 32 байта - элемент каталога.

;---в сегменте данных

FCB DB l,NEWDATABAK',25DUP(0)

;---ищем файл

MOV АH,11H ;функция поиска в каталоге

LEA DX,FCB ;указываем на FCB

INT 21Н ;ищем

CMP AL,0 ;успешно?

JNE NO\_FILE ;если нет, то процедура обработки ошибки

LEA BX,DTA ;теперь DS:BX указывает на элемент каталога

После использования функции 11Н можно использовать функцию 12Н для поиска следующих подходящих элементов, когда имя файла содержит джокеры. В данном случае в имени файла допустим только символ "?", но не "\*". Эта функция работает в точности так же, как и первая, и если найден второй файл, то информация о первом файле в DTA будет уничтожена повторной записью.

**Метод дескриптора файлов:**

Функция 4ЕН прерывания 21Н ищет файл с данным именем. DS:DX должны указывать на строку, дающую путь файла. Например, B:\EURO-PE\FRANCE\PARIS указывает на файл PARIS. Строка может содержать до 63 символов и завершаться символом ASCII 0. Имя файла может содержать джокеры, включая как "?", так и "\*". Поместите атрибут файла в СХ; если он обычный то 0, в противном случае проконсультируйтесь в [5.2.6] относительно значений атрибута.

При возврате устанавливается флаг переноса, если файл не найден. Если файл найден, то функция заполняет DTA информацией о файле. Отметим частный случай использования DTA методом дескриптора файлов - обычно, DTA используется функциями MS DOS для работы через FCB. Первые 21 байт DTA зарезервированы DOS для поиска следующих совпадающих файлов. Двадцать второй байт дает атрибут файла, за ним следуют два байта, содержащие время и еще два байта содержащие дату. Следующие 4 байта содержат размер файла (младшее слово сначала). И, наконец, дается имя файла в виде строки переменной длины, заканчивающейся байтом ASCII 0. Точка (ASCII 46) разделяет имя и расширение и не один из этих элементов не заполнен пробелами.

;—в сегменте данных

PATH DB B:FRANCE\PARIS\4EME',0 ;—ищем файл

MOV АН,4ЕН ;номер функции

LEA DX,PATH ;DS:DX указывают на путь

MOV СХ,0 ;обычный атрибут файла

INT 21Н ;ищем файл

JC NO\_FILE ;уход, если не найден

LEA BX,DTA ;DS:BX указывают на DTA

MOV AL,[BX]+21 ;теперь атрибут файла в AL

Следующее появление имени файла (когда используются джокеры) ищется с помощью функции 4FH прерывания 21Н. Она готовится в точности так же, как и функция 4ЕН, при этом указатель DTA не должен меняться. Когда других совпадений не найдено, то устанавливается флаг переноса, а в АХ появляется 18.

**1.3 Создание/удаление подкаталога.**

Программа может создавать или удалять подкаталоги, при выполнении некоторых условий. Для создания подкаталога необходимо, чтобы было по крайней мере одно пустое место в корневом каталоге. Для удаления подкаталога необходимо, чтобы он не содержал файлов или ссылок на другие подкаталоги. Кроме того, Вы не можете удалить подкаталог, который является Вашим текущим каталогом (тот, с которым по умолчанию выполняются все операции над каталогами). Отметим также, что невозможно удалить корневой каталог.

Поскольку управляющие блоки файлов обслуживают только корневой каталог, то для создания или удаления подкаталога надо использовать дескрипторы файлов.

**Создание подкаталога:**

DS:DX должны указывать на строку, дающую накопитель и путь к каталогу, в котором должен быть создан подкаталог. Строка должна завершаться байтом ASCII 0. Для открытия подкаталога с именем PRIMATES в корневом каталоге накопителя А: надо записать строку в виде "A:\PRIMATES". Для открытия подкаталога в другом подкаталоге с именем MAMMALS напишите "A:\MAMMALS\PRIMATES". Имя накопителя А: может быть опущено если Вы работаете с накопителем, используемым по умолчанию, и путь может начинаться с текущего каталога. Поместите в АН 39Н и выполните прерывание 21Н; если указан правильный путь, то будет создан новый каталог. В противном случае будет установлен флаг переноса, а АХ будет содержать код ошибки 3 (путь неверен) или 5 (нет доступа). В примере создается подкаталог PRIMATES:

;---в сегменте данных

PATH DB 'A:MAMMALS\PRIMATES',0

;---создаем подкаталог с именем PRIMATES

LEA DX,PATH ;DS:DX должны указывать на путь

MOV АН,39Н ;номер функции

INT 21Н ;создаем подкаталог

JC ERROR\_ROUT ;обработка ошибок

**Удаление подкаталога:**

Для удаления подкаталога надо сформировать строку, в точностью совпадающую с той, которую Вы указывали при создании каталога. Затем поместите в АН ЗАН и выполните прерывание 21Н. Опять при невыполнении функции в АХ будут возвращены коды 3 или 5 (код 5 может указывать, что каталог непустой).

**1.4 Чтение/изменение подкаталога.**

Подкаталоги во многом подобны корневому каталогу, за исключением того, что они хранятся как обычные файлы, а не в заранее предопределенных секторах. Подкаталоги невозможно спутать с обычными файлами, поскольку объект каталога, относящийся к подкаталогу, имеет специальный байт атрибутов (с установленным битом 5 - см. [5.2.6]). Подкаталоги начинаются с двух специальных 32-байтных объектов, первый из которых имеет имя точка, а второй - две точки. Они ориентируют подкаталог среди окружающих каталогов. Ссылки на подкаталоги нижнего уровня записываются как обычные ссылки на файлы.

Предполагается, что подкаталог может быть прочитан как любой другой файл, поэтому вроде бы не составляет труда загрузить его в память. Но, к сожалению, создатели MS DOS поместили 0 в поле длины файла для элементов, относящихся к подкаталогам. В результате DOS считает, что этот файл имеет нулевую длину и отказывается читать его. Нет простого способа преодолеть эту проблему.

Функции работы через дескрипторы файлов, которые использовались для доступа к корневому каталогу [5.2.1] могут так же просто обращаться к любому подкаталогу. Чтобы вывести все содержимое каталога надо просто использовать функцию 4ЕН для поиска файлов \*.\*, а затем повторять поиск, используя функцию 4FH. Когда больше не будет файлов, то будет установлен флаг переноса, a AL будет содержать 18. Каждый раз, когда будет обнаружен очередной элемент, в DTA будет записана информация о файле, включая полный его путь (отмечаем использование DTA в функциях, использующих дескриптор файла). Следующий пример выводит полные пути всех обычных файлов подкаталога.

;---в сегменте данных

PATH DB 'A:MAMMALS\\*.\*',0

DTAH DB 256DUP(?)

;---установка DTA

LEA DX,DTA ;DS:DX указывают на DTA

MOV AH,1АН ;функция установки DTA

INT 21H ;устанавливаем DTA

;---ищем первый файл

MOV AH,4EH ;номер функции

LEA DX,PATH ;указываем на строку пути

MOV СХ,0 ;только нормальные атрибуты

INT 21Н ;ищем \*.\*

JC ERROR ;обработка ошибок

;---выводим имя файла

NEXT\_LINE: LEA BX,DTA ;ВХ указывает на DTA

ADD ВХ,30 ;смещение для имени файла

NEXT\_CHAR: MOV DL,[BX] ;получаем символ из имени

CMP DL,0 ;проверка на конец строки

JE ENDSTR ;уход, если конец

MOV АН,2 ;иначе, выводим символ

INT 21H ;

INC BX ;увеличиваем указатель

JMP SHORT NEXT\_CHAR ;следующий символ

;--- возврат каретки/перевод строки в конце строки

END\_STR: MOV АН,2 ;функция вывода символа

MOV DL,13 ;код возврата каретки

INT 21H ;выводим

MOV DL,10 ;код перевода строки

INT 21Н ;выводим

;---ищем следующий файл

LEA DX,PATH ;указываем на строку пути

MOV AH,4FH ;номер функции

INT 21Н ;ищем следующий файл

JC FINISHED ;если нет, то выход

JMP SHORT NEXT\_LINE ;иначе выводим имя файла

FINISHED:

**II. Практическая часть**

По условию нам необходимо написать программу, производящую создание, удаление, чтение, изменение каталогов.

В начале нашей программы, мы устанавливаем текущий каталог при помощи функции ЗВН 21-ого прерывания. Для выполнения данной функции нам необходимо указать адрес строки из сегмента данных со значением пути к необходимому каталогу в виде строки ASCII, в конце код 0. Также ставим условие обработки ошибок (если ошибка (CF=1), то в АХ - код ошибки).

Затем, с помощью функции 47Н 21-ого прерывания, читаем полный путь к текущему каталогу. Для входных параметров нам понадобиться в DL указать необходимый дисковод, а также поставим SI указывать на 64 байтовую переменную CURRENT\_DIR, которая, собственно, и будет хранить путь к текущему каталогу. Далее эту переменную мы выведем на экран. Следует заметить, что функция возвращает строку без имени накопителя.

Теперь перейдем к созданию подкаталога. Для этого нам понадобиться 39-ая функция 21-ого прерывания. Здесь нам всего лишь необходимо чтобы DX указывал на нашу строку с путем (именем) каталога, в конец пишем код 0.

Далее для демонстрации функции вывода содержимого подкаталога, создадим в созданном нами каталоге 3 файла функцией ЗсН 21-ого прерывания. Затем перейдем к процедуре вывода содержимого подкаталога на экран.

Для начала определим адрес области обмена с диском (DTA) для последующих операций с блоками управления файлами. С помощью функции 1АН 21-ого прерывания установим DTA как область памяти из 256 байт с неопределенным значением.

Чтобы вывести все содержимое каталога на экран, необходимо использовать функцию 4ЕН 21-ого прерывания для поиска файлов \*.\*, а затем повторять поиск, используя функцию 4FH. Когда больше не будет файлов, то будет установлен флаг переноса, a AL будет содержать 18. Каждый раз, когда будет обнаружен очередной элемент, в DTA будет записана информация о файле, включая полное его имя.

**Заключение**

В данной курсовой работе была рассмотрена задача создание, удаление, чтение, изменение каталогов. Мы вывели на экран все содержимое каталога, теперь можно его удалить. Для удаления каталога мы сформировали такую же строку пути, которые мы указывали при создании каталога. По функции 3АН 21-ого прерывания наш каталог был удален, заметим, что каталог не будет удален, если он будет содержать какие-либо файлы, поэтому прежде удалили оттуда все файлы и выполнили функцию.

**Библиографический список использованной литературы**

1. Пирогов В.Ю. Ассемблер. Учебный курс. - 2-е изд., перераб. и доп. СПБ.: БХВ-Петербург, 2003.

2. Журден Р. Справочник программиста на персональном компьютере фирмы IBM PC.

**Приложение А. Листинг программы**

Title Catalog\_Work

stk segment stack

dw 90 dup(?)

stk ends

Data segment

CURRENT\_DIR DB 64 DUP(?),'$'

path\_cur db 'e:\bp\bin',0,13,10,'$'

path db 'd:\catalog',0

path2 db'd:\catalog\\*.\*',0

file1 db 'd:\catalog\file 1.txt',0

file2 db 'd:\catalog\file2.txt',0

file3 db 'd:\catalog\file3.txt',0

cat\_crt db 'Catalog create> ','$'

wr\_msg db 'Napishem informaciuy w catalog',13,10,'Sama informacia: ',13,10;$'

err1\_msg db 'Catalog sushestwuiet ili newozmogno sozdat ',13,10,'$'

msgl db 'Pered udaleniem kataloga sotrite iz nego wsiu informaciu i posle nagmite enter', 13,10,'$'

msg\_cat\_setup db 'Ustanofka tekushego katologa',13,10,'$'

msg\_cur\_cat db 'Tekushii catalog> ','$'

razd db 13,10,'$'

dta db 256 dup(?)

Data ends

Code segment

assume cs:code, ds:data, ss:stk

begin:

mov ax,data

mov ds,ax

mov ah,9h

lea dx,msg\_cat\_setup

int 21h

;---Делаем E:\BP\BIN текущем каталогом

MOV AH,3BH ;Номер функции

LEA DX,PATH\_cur ;DS:DX должны указать на путь

INT 21Н ;устанавливаем текущий каталог

;---Получаем текущий каталог

MOV АН,47Н ;Номер функции

LEA SI,CURRENT\_DIR ;Указываем на область данных

MOV DL,5 ;Указываем на диск Е:\

INT 21Н ;Помещаем строку по адресу DS:SI

mov ah,9h

lea dx,msg\_cur\_cat

int 21h

mov ah,9h

lea dx,current\_dir

int 21h

;---Создаем новый подкаталог по адресу в PATH

LEA DX,path ;Указываем путь и имя подкаталога

MOV АН,39Н ;Сама функция

INT 21Н ;Создаем

JC ERROR1 ;Если ошибка то переходим на ошибку

mov ah,9h

lea dx,cat\_crt

int 21h

mov ah,9h

lea dx,razd

int 21h

mov ah,9h

lea dx,wr\_msg

int 21h

call key

;---Создаем в созданном каталоге 3 файла с расширением .txt

mov ah,3ch

mov al,2

mov cx,0

lea dx,filel

int 21h

mov ah,3ch

mov al,2

mov cx,0

lea dx,file2

int 21h

mov ah,3ch

mov al,2

mov cx,0

lea dx,file3

int 21h

;-------------

jmp a

error 1:

mov ah,9h

lea dx,errl\_msg

int 21h

jmp exit

a:

;---Устанавливаем DTA

LEA DX,DTA ;DS:DX указывает на DTA

MOV AH,1AH ;Функция установки DTA

INT 21H ;Устанавливаем DTA

;---Ищем первый файл

MOV AH,4EH ;Сама функция

LEA DX,path2 ;Указываем на строку пути

MOV СХ,0 ;Только нормальные атрибуты

INT 21Н ;Ищем \*.\*

JC ERROR\_rout ;Ошибки обрабатываем

;---Выводим имя файла

NEXT\_LINE: LEA BX,DTA ;ВХ указывает на DTA

ADD ВХ,30 ;Смещение для имени файла

NEXT\_CHAR: MOV DL,[BX] ;Получаем символ из имени

CMP DL,0 ;Проверка на конец строки

JE END\_STR ;Уходим, если конец строке

MOV АН,2 ;Иначе выводим символ

INT 21Н ;

INC ВХ ;Увеличиваем указатель

JMP SHORT NEXT\_CHAR ;следующий символ

;---возврат каретки\перевод строки в конце строки

END\_STR: MOV АН,2 ;Функция вывода символа

MOV DL,13 ;код возврата каретки

INT 21Н ;Выводим

MOV DL,10 ;код перевода строки

INT 21Н ;выводим

;---Ищем следующий файл

LEA DX,path2 ;Указываем на строку пути

MOV AH,4FH ;Номер функции

INT 21Н ;Ищем сл. файл

JC FINISHED ;Если нет, то выходим

JMP SHORT NEXT\_LINE ;Иначе выводим имя файла

FINISHED:

mov ah,9h

lea dx,msgl

int 21h

call key

;Удаляем наш подкаталог CATALOG

LEA DX,PATH ;DS:DX Путь к подкаталогу

MOV AH,3aH ;Функция удаления

INT 21Н ;Удаляем

JC ERROR 1 ;Обработка ошибок

exit:

mov ah,4ch

int21h

error\_rout:

jmp exit

;-------------------------------------------

;Процедура, ждет нажатия любой клавиши

;-------------------------------------------

key ргос

xor ah,ah

int 16h

ret

key endp

;-------------------------------------------

code ends

end begin

**Приложение Б. Листинг программы**

Title Catalog\_Work

stk segment stack

dw 90 dup(?)

stk ends

Data segment

CURRENT\_DIR DB 64DUP(?),'$'

path\_cur db 'e:\bp\bin'A13,10,'$'

path db 'd:\catalog',0

path2 db 'd:\catalog\\*.\*',0

file 1 db 'd:\catalog\file 1.txt',0

file2 db 'd:\catalog\file2.txt',0

file3 db'd:\catalog\file3.txt',0

cat\_crt db 'Catalog create>','$'

wrmsg db ‘Napishem informaciuy w catalog',13,10,'Sama informacia: ',13,10,'$'

errl\_msg db 'Catalog sushestwuiet ili newozmogno sozdat ',13,10,'$'

msgl db 'Pered udaleniem kataloga sotrite iz nego wsiu informaciu i posle nagmite enter',13,10,'$'

msg\_cat\_setup db 'Ustanofka tekushego katologa',13,10,'$'

msg\_cur\_cat db 'Tekushii catalog> ','$'

razd db 13,10,'$'

dta db 256 dup(?)

Data ends

Code segment

assume cs:сode, ds:data, ss:stk

begin:

mov ax,data

mov ds,ax

mov ah,9h

lea dx,msg\_cat\_setup

int21h

MOV AH,3BH

LEA DX,PATH\_cur

INT 21H

MOV AH,47H

LEA SI,CURRENT\_DIR

MOV DL,5

INT 21H

mov ah,9h

lea dx,msg\_cur\_cat

int 21h

mov ah,9h

lea dx,current\_dir

int 21h

LEA DX,path

MOV AH,39H

INT 21H

JC ERROR 1

mov ah,9h

lea dx,cat\_crt

int 21h

mov ah,9h

lea dx,razd

int 21h

mov ah,9h

lea dx,wr\_msg

int21h

call key

mov ah,3ch

mov al,2

mov cx,0

lea dx,filel

int 21h

mov ah,3ch

mov al,2

mov cx,0

lea dx,file2

int21h

mov ah,3ch

mov al,2

mov cx,0

lea dx,file3

int 21h

jmp a

error 1:

mov ah,9h

lea dx,errl\_msg

int 21h

jmp exit

a:

LEA DX,DTA

MOV АН,1АН

INT 21H

MOV AH,4EH

LEA DX,path2

MOV CX,0

INT 21H

JC ERROR\_rout

NEXT\_LINE: LEA BX,DTA

ADD BX,30

NEXT\_CHAR: MOV DL,[BX]

CMP DL,0

JE END\_STR

MOV AH,2

INT 21H

INC BX

JMP SHORT NEXT\_CHAR

END\_STR: MOV AH,2

MOV DL,13

INT 21H

MOV DL,10

INT 21H

LEA DX,path2

MOV AH,4FH

INT 21H

JC FINISHED

JMP SHORT NEXT\_LINE

FINISHED:

mov ah,9h

lea dx,msgl

int21h

call key

LEA DX,PATH

MOV AH,

INT 21H

JC ERROR 1

exit:

mov ah,4ch

int 21h

error\_rout:

jmp exit

key proc

xor ah,ah

int 16h

ret

key endp

code ends

end begin