SANTA CLARA UNIVERSITY
Electrical and Computer Engineering Department

Real-Time Embedded Systems - ECEN 121
Lab 8
DYLAN THORNBURG & KAI HOSHIDE

USB Flash Drive

Andrew Wolfe

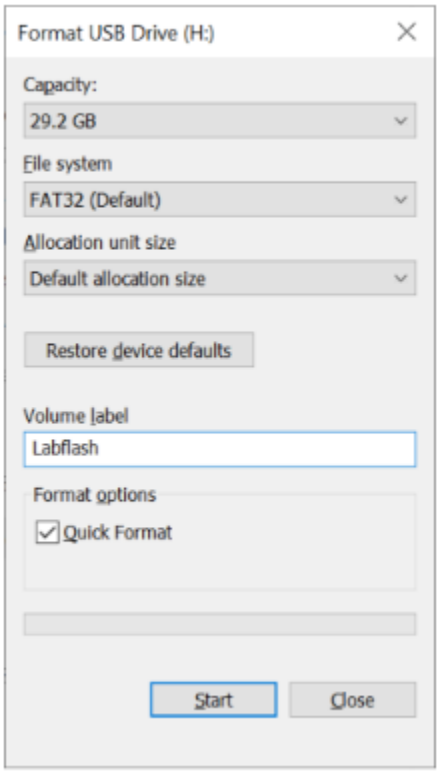
Pre Lab:

1. Review fatgen103.pdf.
2. Review STM32CUBE FAT manual.pdf.
3. Look at the main page at FatFs - Generic FAT Filesystem Module.
4. In your prelab, describe the difference between a long file name and a short file name.
Long file names are limited to 255 characters (260 for full paths) and short file names are limited to 8+3 or 11 characters (80 for full path names).
5. Is πman53.txt a long file name or a short file name? Explain why.
It's a long file name because pi is not read correctly.
6. What parameters are required for fread() and what is the data type of each parameter?
Fread has four parameters (fread(P1,P2,P3,P4). P1 is the buffer or where you'll be putting the read data. P2 is the size of each element in bytes. P3 is the count of elements to read. P4 is the pointer to the file stream you'll read from. P1 and P4 are pointers (void * and FILE *) and P2 and P3 are unsigned integers.

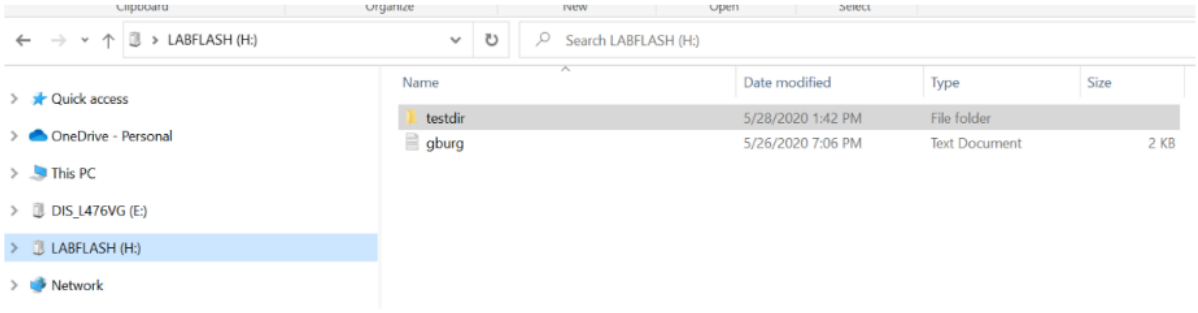
NOTE: For this prelab we were instructed to look at the FatFs website and in that website it describes a function f_read() that switches the order of the parameters when compared to fread(). IF this question was intended to ask us about f_read, then just switch the parameter order so f_read(P4, P1, P3, P2). Also P2 will be a pointer.
7. How many total bytes are used to represent the parameters of fread()?
On a 64-bit system, it is 4*64/8=32 bytes. On a 32-bit system, it is 4*32/8=16 bytes.

Preparation:

Connect the flash drive that I supplied to the lab computer and find it in the File Explorer. Right click and select format. Choose the FAT32 file system, default allocation size, and Quick Format. Pick a volume label if you want. Hit Start. Close out of the format dialog when finished.



Now copy the file Lab8usbfiles.zip from the Lab Handouts directory. Unzip that file and copy the contents of that file to your USB drive. It should look like this when you are done.



Now you can eject the flash drive and connect it to your board using the USB OTG adapter. **Lab Project 1:**

You are going to read the contents of the file gburg.txt from the USB drive to a buffer in memory on the DISCO board. You will need to use the debugger to see the contents of this buffer.

Documentation of the various FatFs library calls is in the STM32CUBE FAT manual in the Readings directory and at http://elm-chan.org/fsw/ff/00index_e.html

Our FAT library may not support everything at the web site. Refer to the local manual as well.

Your program should:

1. open the file gburg.txt
2. Read the contents of that file into the buffer rtext[]
3. Close the file
4. Turn on the green LED if successful
5. Turn on the red LED is an error is detected

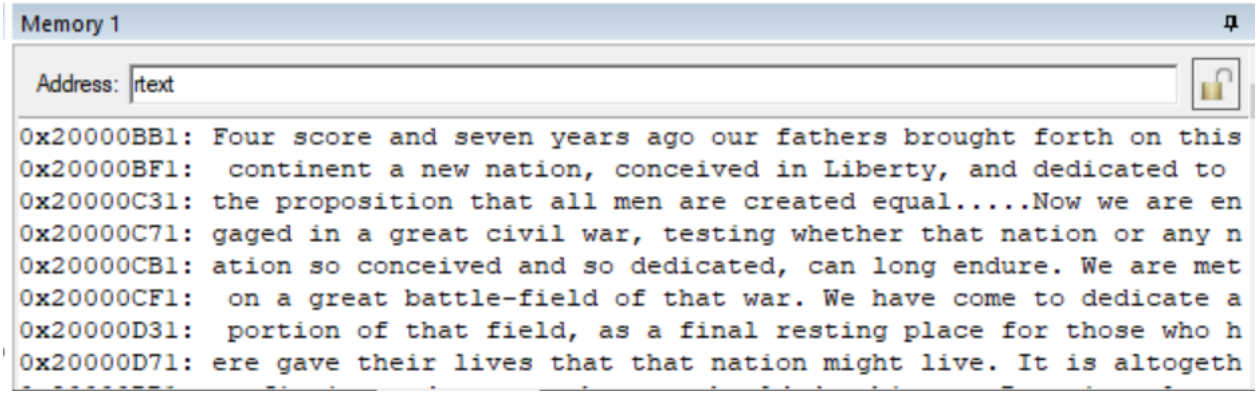
I actually wrote a complete program that did this correctly. Unfortunately, you can't have it. If you want, you can download the project Lab8p1.zip from the Lab Handouts directory and use that as a starting point. It is 99% of the code you need, but some jerk went in and typed "Oh No This is Wrong" in place of some of the code. If you fix it, it will read the file into a buffer. Otherwise, you can start from scratch. Like Lab 7, you will need to open it in CubeMX and generate code to match the library paths to your machine.

To demo – show the program running in the debugger and displaying the contents of rtext[] in ASCII. Turn in your main.c file (or at least enough of it to show all the changes) with the report.

CODE:

```
static void FS_FileOperations(void) {
        FRESULT res;                    /* FatFs function common result code */
 uint32_t bytesread;              /* File write/read counts */

         /* Register the file system object to the FatFs module */
 if(f_mount(&USBHFatFS, (TCHAR const*)USBHPath, 0) == FR_OK) {
                 /* Open the text file object with read access */
                 if(f_open(&MyFile, rfilename, FA_READ) == FR_OK) {
                         /* Read data from the text file */
                         res = f_read(&MyFile, rtext, sizeof(rtext)-1, &bytesread);
                         if((bytesread > 0) && (res == FR_OK)) {
                                 /* Close the open text file */
                                 f_close(&MyFile);
                                 /* Compare read data with the expected data */
                          HAL_GPIO_WritePin(LD_G_GPIO_Port, LD_G_Pin, GPIO_PIN_SET);
                          return;
                         }
                 }
         }
         HAL_GPIO_WritePin(LD_R_GPIO_Port, LD_R_Pin, GPIO_PIN_SET);
}
```



Lab Project 2:

For part 2 of this lab, your program will catalog all of the directories on the USB drive and print a list of the directories out to a file on the USB drive called DIRLIST.TXT. The complete path of each file is listed once. The contents of the DIRLIST.TXT file after the program runs will look something like this (only longer)

/DIRLIST.TXT
/SYSTEM~1/WPSETT~1.DAT
/SYSTEM~1/INDEXE~1
/GBURG.TXT
/TESTDIR/DIR1/F3.TXT
/TESTDIR/DIR1/DIR1A/F1.TXT
/TESTDIR/DIR1/DIR1A/F2.TXT
/TESTDIR/DIR2/F4.TXT
/TESTDIR/DIR2/F5.TXT
/TESTDIR/DIR2/F6.RTF
/TESTDIR/DIR3/PPT1~1.PPT
/TESTDIR/DIR4/DIRX~1/DIRY~1/DIR3/PPT1~1.PPT
/TESTDIR/DIR4/DIRX~1/DIRY~1/DIR1/F3.TXT
/TESTDIR/DIR4/DIRX~1/DIRY~1/DIR1/DIR1A/F1.TXT
/TESTDIR/DIR4/DIRX~1/DIRY~1/DIR1/DIR1A/F2.TXT

Your program should do the following:

- Mount the USB drive
- Scan the directory and each subdirectory for files, printing the path names into the character

string dirlist[].

- Open a file in the root directory called dirlist.txt
- Write out the data in dirlist[] to this file
- Close the file
- Turn on the green LED if successful
- Turn on the red LED is an error is detected

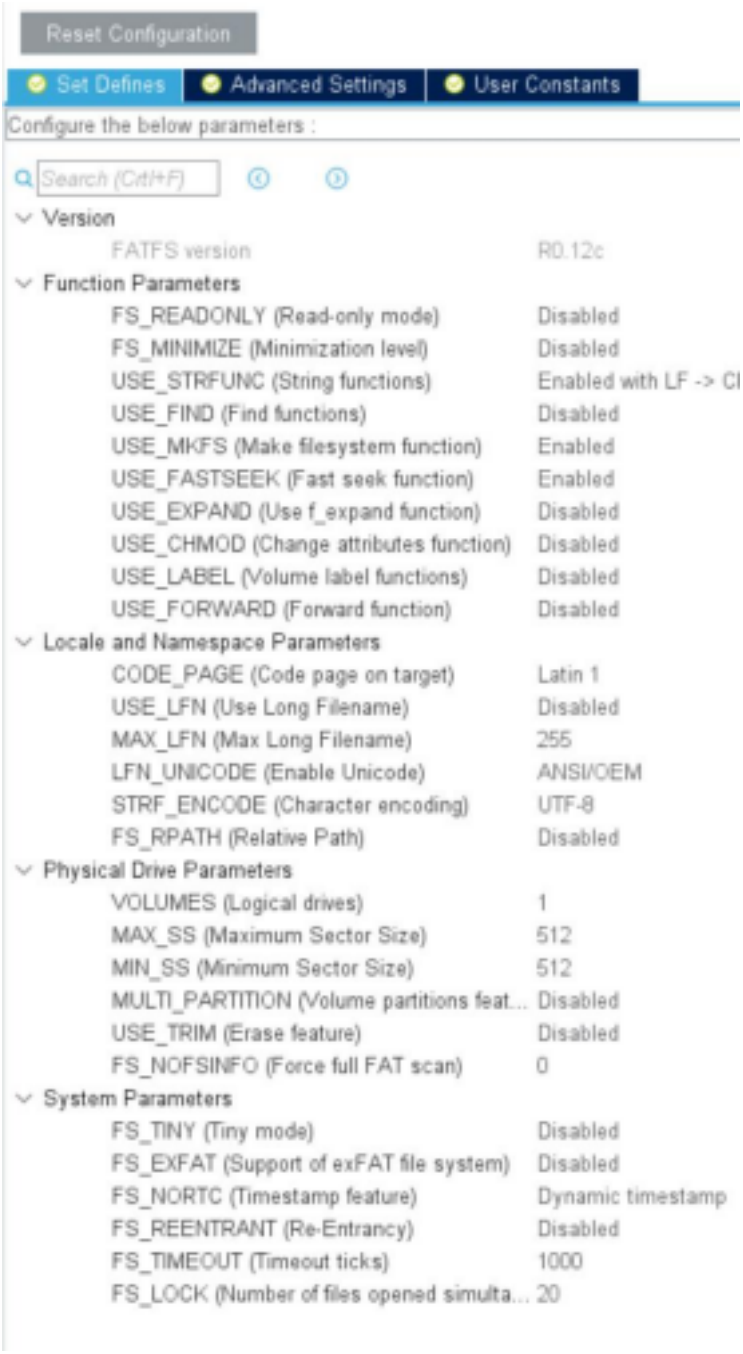
Now, you say – how do I scan directories?

There is an example at http://elm-chan.org/fsw/ff/doc/readdir.html

This code will not work as-is but can guide you on what to do.

You can solve this however you want but downloading the project Lab8p2.zip from the Lab Handouts directory will probably give you a big head start. Like Lab 7, you will need to open it in CubeMX and generate code to fix the library paths.

If you do start from scratch, you need to increase the limit on open files for FatFS (FS_LOCK):



If you start from my project, that has been done.

Demo by showing the DIRLIST.TXT file contents on your USB drive (by plugging it back into your PC). Turn in your main.c file (or at least enough of it to show all the changes) and your DIRLIST.TXT file with the report.

CODE:

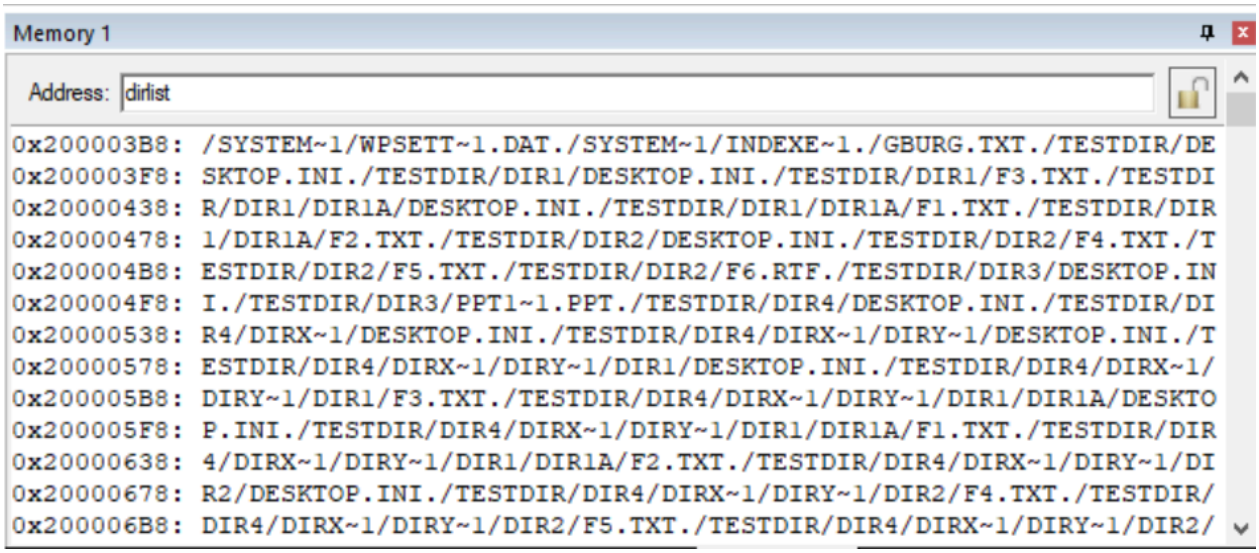
```
FRESULT scan_files( char* path) {
        FRESULT res;
        DIR dir;
        int i;
```

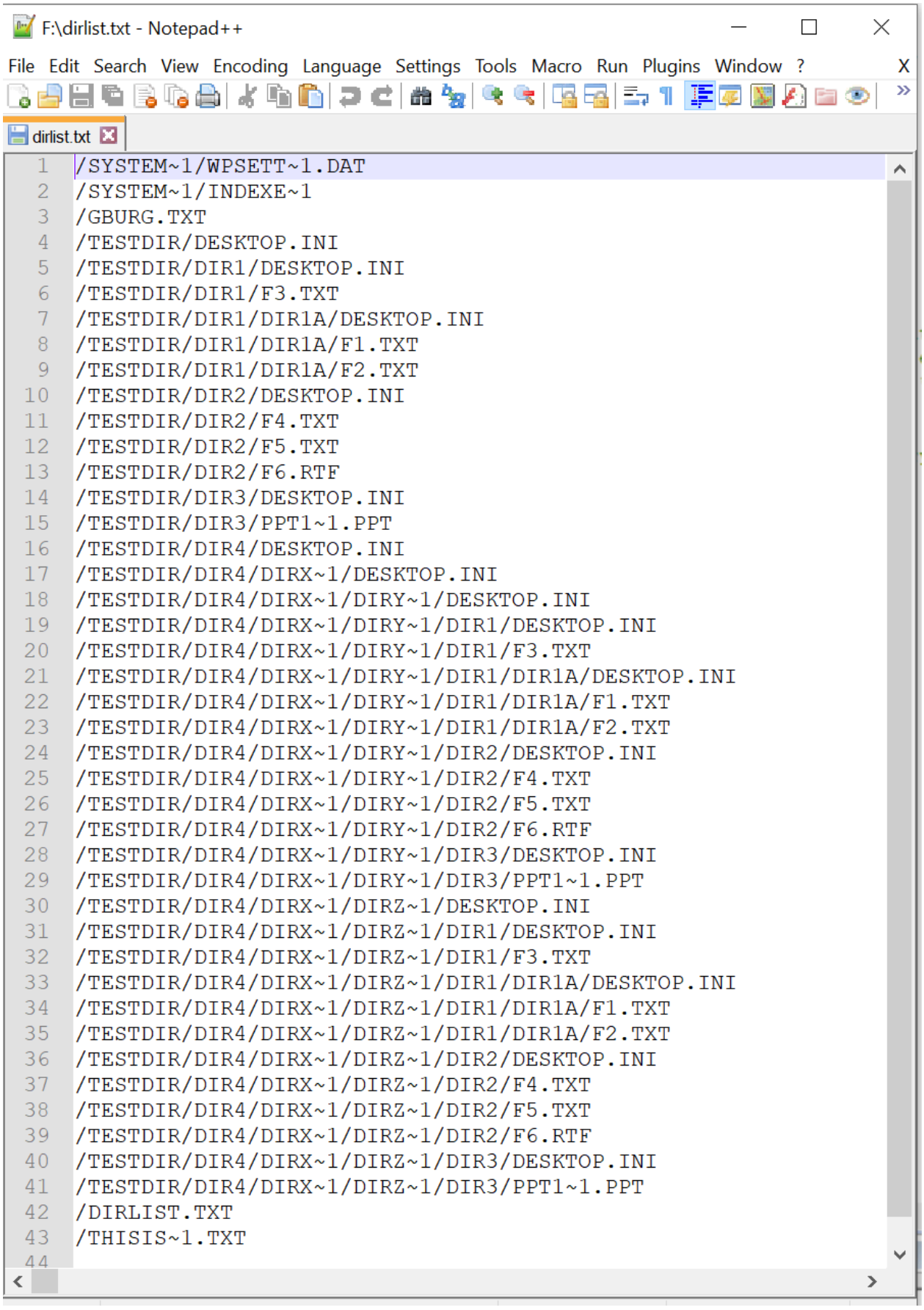
```
        static FILINFO fno;
        res = f_opendir(&dir, path);                    /* Open the directory */
  if (res == FR_OK) {
                                  while (1) {
      res = f_readdir(&dir, &fno);                  /* Read a directory item */
      if (res != FR_OK || fno.fname[0] == 0) break;  /* Break on error or end of dir */
      if (fno.fattrib & AM_DIR) {                   /* It is a directory */
        i = strlen(path);
        sprintf(&path[i], "/%s", fno.fname);
        res = scan_files(path);                     /* Enter the directory */
        if (res != FR_OK) break;
        path[i] = 0;
      }
                                                                  else
                                                                        {                               /* It is a file. */
        sprintf(dirlist + strlen(dirlist), "%s/%s\n", path, fno.fname);
      }
                                  }
    f_closedir(&dir);
  }
 return res;
}

FRESULT write_dirlist(char* path, char* list) {
        FIL OutFile;
        FRESULT res;
        uint32_t byteswritten;

        if (f_open(&OutFile, path, FA_WRITE | FA_CREATE_ALWAYS) == FR_OK)
        {
                res = f_write(&OutFile, list, strlen(list), &byteswritten);
                if (res == FR_OK)
                {
                        f_close(&OutFile);
                }
        }
 return res;
}


/**
 * @brief  Main routine for Mass Storage Class
 * @param  None
 * @retval None
 */
```


Memory 1
Address: dirlist
0x200003B8: /SYSTEM~1/WPSETT~1.DAT./SYSTEM~1/INDEXE~1./GBURG.TXT./TESTDIR/DE
0x200003F8: SKTOP.INI./TESTDIR/DIR1/DESKTOP.INI./TESTDIR/DIR1/F3.TXT./TESTDI
0x20000438: R/DIR1/DIR1A/DESKTOP.INI./TESTDIR/DIR1/DIR1A/F1.TXT./TESTDIR/DIR
0x20000478: 1/DIR1A/F2.TXT./TESTDIR/DIR2/DESKTOP.INI./TESTDIR/DIR2/F4.TXT./T
0x200004B8: ESTDIR/DIR2/F5.TXT./TESTDIR/DIR2/F6.RTF./TESTDIR/DIR3/DESKTOP.IN
0x200004F8: I./TESTDIR/DIR3/PPT1~1.PPT./TESTDIR/DIR4/DESKTOP.INI./TESTDIR/DI
0x20000538: R4/DIRX~1/DESKTOP.INI./TESTDIR/DIR4/DIRX~1/DIRY~1/DESKTOP.INI./T
0x20000578: ESTDIR/DIR4/DIRX~1/DIRY~1/DIR1/DESKTOP.INI./TESTDIR/DIR4/DIRX~1/
0x200005B8: DIRY~1/DIR1/F3.TXT./TESTDIR/DIR4/DIRX~1/DIRY~1/DIR1/DIR1A/DESKTO
0x200005F8: P.INI./TESTDIR/DIR4/DIRX~1/DIRY~1/DIR1/DIR1A/F1.TXT./TESTDIR/DIR
0x20000638: 4/DIRX~1/DIRY~1/DIR1/DIR1A/F2.TXT./TESTDIR/DIR4/DIRX~1/DIRY~1/DI
0x20000678: R2/DESKTOP.INI./TESTDIR/DIR4/DIRX~1/DIRY~1/DIR2/F4.TXT./TESTDIR/
0x200006B8: DIR4/DIRX~1/DIRY~1/DIR2/F5.TXT./TESTDIR/DIR4/DIRX~1/DIRY~1/DIR2/


F:\dirlist.txt - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
dirlist.txt
/SYSTEM~1/WPSETT~1.DAT
/SYSTEM~1/INDEXE~1
/GBURG.TXT
/TESTDIR/DESKTOP.INI
/TESTDIR/DIR1/DESKTOP.INI
/TESTDIR/DIR1/F3.TXT
/TESTDIR/DIR1/DIR1A/DESKTOP.INI
/TESTDIR/DIR1/DIR1A/F1.TXT
/TESTDIR/DIR1/DIR1A/F2.TXT
/TESTDIR/DIR2/DESKTOP.INI
/TESTDIR/DIR2/F4.TXT
/TESTDIR/DIR2/F5.TXT
/TESTDIR/DIR2/F6.RTF
/TESTDIR/DIR3/DESKTOP.INI
/TESTDIR/DIR3/PPT1~1.PPT
/TESTDIR/DIR4/DESKTOP.INI
/TESTDIR/DIR4/DIRX~1/DESKTOP.INI
/TESTDIR/DIR4/DIRX~1/DIRY~1/DESKTOP.INI
/TESTDIR/DIR4/DIRX~1/DIRY~1/DIR1/DESKTOP.INI
/TESTDIR/DIR4/DIRX~1/DIRY~1/DIR1/F3.TXT
/TESTDIR/DIR4/DIRX~1/DIRY~1/DIR1/DIR1A/DESKTOP.INI
/TESTDIR/DIR4/DIRX~1/DIRY~1/DIR1/DIR1A/F1.TXT
/TESTDIR/DIR4/DIRX~1/DIRY~1/DIR1/DIR1A/F2.TXT
/TESTDIR/DIR4/DIRX~1/DIRY~1/DIR2/DESKTOP.INI
/TESTDIR/DIR4/DIRX~1/DIRY~1/DIR2/F4.TXT
/TESTDIR/DIR4/DIRX~1/DIRY~1/DIR2/F5.TXT
/TESTDIR/DIR4/DIRX~1/DIRY~1/DIR2/F6.RTF
/TESTDIR/DIR4/DIRX~1/DIRY~1/DIR3/DESKTOP.INI
/TESTDIR/DIR4/DIRX~1/DIRY~1/DIR3/PPT1~1.PPT
/TESTDIR/DIR4/DIRX~1/DIRZ~1/DESKTOP.INI
/TESTDIR/DIR4/DIRX~1/DIRZ~1/DIR1/DESKTOP.INI
/TESTDIR/DIR4/DIRX~1/DIRZ~1/DIR1/F3.TXT
/TESTDIR/DIR4/DIRX~1/DIRZ~1/DIR1/DIR1A/DESKTOP.INI
/TESTDIR/DIR4/DIRX~1/DIRZ~1/DIR1/DIR1A/F1.TXT
/TESTDIR/DIR4/DIRX~1/DIRZ~1/DIR1/DIR1A/F2.TXT
/TESTDIR/DIR4/DIRX~1/DIRZ~1/DIR2/DESKTOP.INI
/TESTDIR/DIR4/DIRX~1/DIRZ~1/DIR2/F4.TXT
/TESTDIR/DIR4/DIRX~1/DIRZ~1/DIR2/F5.TXT
/TESTDIR/DIR4/DIRX~1/DIRZ~1/DIR2/F6.RTF
/TESTDIR/DIR4/DIRX~1/DIRZ~1/DIR3/DESKTOP.INI
/TESTDIR/DIR4/DIRX~1/DIRZ~1/DIR3/PPT1~1.PPT
/DIRLIST.TXT
/THISIS~1.TXT