THE UNIVERSITY COLLEGE
OF THE CARIBOO

COMPUTING 253

Small Computer Systems: Organisation and Architecture

Workshop #1 –*"Having a little dance with DOS"*

No required submission

Introduction

The Command Line Interface (CLI) is the most traditional method of interacting with an operating system. The Graphical User Interface (GUI) is just a pretty face that sits atop the underlying CLI that actually makes everything work.

As part of Microsoft-based GUIs like Win9x/Me/NT/2000/XP (not including WinCE, Pocket PC, and WinXPEmbedded), the command line interface (DOS) still exists; although hidden from the general user by a bright and attractive start-up screen. Linux and UNIX are similar with their command line shells, supporting the X-Windows library and a window manager GUI.

The purpose of the first workshop is to experience the wonder, fascination, and *confusion* of having to remember and understand every command you type. You will learn the power of what a simple sequence of keystrokes can accomplish…compared to an endless collection of mouse clicks and window-menu selections.

Topics of this workshop,

1. The booting procedure, and required presence, of the *command interpreter*.
2. Separation of internal and external commands, and where each is located.
3. Purpose and syntax of common commands, and access to built-in syntax help.
4. Directories: creation, destruction, and navigation; discussion of the "root."
5. Paths and the differences between the command/execution path and directory paths.
6. "Boot-up" (initial) configuration files: **config.sys** and **autoexec.bat**

You are encouraged to further research the following two topics, either through resources (online, books) or just continuing to use DOS more,

7. Filenames and wildcards.
8. Drive and directory presence, when accessing local and far file locations within commands.
9. DOS-level programming with the "batch" command language (similar to VAX's digital command language (DCL) and UNIX/Linux's *scripting* languages)

Procedure

Unlike subsequent workshops, this consists of a team of one: yourself.

Select one of the computers from any of the clusters (server or workstation, it does not matter). Make sure you have a 3.5" floppy disk that you are willing to have all the data deleted. Although the computer's hard disk already has an operating system installed (such as Win98, 2000, or Linux), you will "boot" using a **DOS/Win98 System Boot Disk** provided by your instructor.

In later workshops, you are expected to use a selection of DOS commands, and their effects, with minimal assistance from your instructor, so highlight and make whatever notes you think are necessary.

As a resource, the lab has some DOS Reference Manuals, but do not forget about the built-in syntax help with each command.

Also, keep in mind that letter case does not matter with DOS commands; therefore, the command "DIR" is the same as "dir" and "Dir".

Documentation

As far as formal documentation goes, there is none. The only notes you make are for yourself.

Finishing

When you have finished the observations and tasks, make sure that the computer and cluster you used are how you found them. Return the DOS/Win98 System Boot Disk.

If you have questions on any part of the workshop, ask your instructor or note them and find the answers later.

Observations and Tasks

Part 1: BOOTING UP! *(…but before turning on the computer!!)*

1. Initially, as a cold machine, the computer has nothing loaded in memory (RAM). Once powered, the required operating system (OS) must be loaded from a floppy or hard disk and copied to memory.
2. Place the **DOS/Win98 System Boot Disk** into the floppy disk drive and press the power button.
3. The "bootstrap loader" program, in the computer's hardware BIOS, attempts to start the computer by locating the operating system (OS) on disk *(the lab's systems' boot sequence is: CDROM, floppy, then hard disk)*
4. The OS being loaded is the DOS of Windows 98 (called MS-DOS v7.10). Also loaded are some drivers, such as extended screen control, keyboard extras, and enabling the CD-ROM drive.
5. Once "booted," the computer displays the DOS prompt (A:\>), which indicates the focus on drive (A:) and the current directory (\ - the root).
6. The Windows98 GUI is not available, nor loaded, so all input and output must is performed at the DOS prompt via the console (together, the *keyboard* and *monitor* are called the console).

Part 2: Where's the Operating System?

1. To list the files on the disk, apply the most used DOS command, DIR, to display a wide listing of all the files in the current directory (\ - the root), type: DIR \ /W /A
2. The operating system is not predefined in the hardware of the computer, and must be found and loaded during boot time. Once the OS is located and loaded, the DOS environment is defined.
   To examine the variables and settings of the environment, type: SET
3. Find the line identifying the variable COMSPEC. This indicates where on the drive COMMAND.COM is located (it is required when the OS requires internal commands or application interfacing. *A very bad thing would be blank this variable, voiding any DOS commands afterwards.*
4. Use the DIR command again, and find the three (3) fundamental DOS files: IO.SYS, MSDOS.SYS, and COMMAND.COM files.
   (Hint: You may have to use the switch /A *; why?)*
5. To see which programs are currently loaded in memory, and the amount of RAM each occupies, use the MEM command with content display and a pipe (|), for screen page control (the | is above \ ), type: MEM /C | M
6. Examine how much memory COMMAND.COM occupies. Check the size of the program on disk, with: DIR COMMAND.COM

   The difference in byte size is because not all of COMMAND.COM is retained in memory after the program executes. Once booting is finished, only the internal commands are retained (these are the DOS commands inside (or *internal* to) COMMAND.COM).

Part 3: Internal and External Commands?

1. There is no strict rule as to why a particular command is *Internal* or *External*, but a good assumption is that the smallest and most used commands are internal, and largest and least used are external.
2. So far you have seen two internal commands (SET and DIR) and two external commands (MEM and MORE).
3. Internal commands are within COMMAND.COM and so stay in memory. Externals are application programs, that are loaded when necessary.
4. How it works:
   1) when an command is typed, COMMAND.COM parses the statement and attempts to separate the command from any options and switches.
   2) to perform the command, it first looks within itself, checking for a matching internal command.
   3) if not found, COMMAND.COM searches the disk for an external program to execute, starting in the current directory, then searching the execution path (see later with the PATH variable and command).
   4) finally, if the command can not be found, the message "Bad Command or Filename" is displayed.

   In searching the current directory and command/execution path, COMMAND.COM searches only for .COM, .EXE, then .BAT files.
5. Now display the current command/execution path. Use either the SET or PATH command (both SET and PATH are internal commands).
6. Again, an interesting note is the execution priority sequence. Assume there are three (3) program files in the current directory: PROG.COM, PROG.EXE, PROG.BAT, and only " PROG " was typed as a command.

   COMMAND.COM would find and execute PROG.COM first, ignoring the others. If PROG.COM did not exist, PROG.EXE would be executed. Similarly for PROG.BAT, if PROG.EXE did not exist. Much confusion has resulted from this in the days when all three programs were used: batch files (.BAT), command files (.COM), and executable files (.EXE).

7. What is the difference between the 3 program types: .COM, .EXE, and .BAT? *(Consult a DOS manual, and/or the Internet).*

Part 4: External Commands: FORMAT and DISKCOPY

1. Obtain a 3.5" floppy disk from your instructor. The following steps guide you through *formatting* it (preparing it for use and completely erasing a used disk) and *diskcopying* the System Boot Disk to this disk.
2. To disregard existing content and perform an unconditional (non-quick) format, and place a copy of the OS on the disk, keep the DOS/System Boot Disk in the drive, and type: FORMAT A: /U /S
3. When prompted, remove the System Boot Disk and insert the new disk. Let the command finish and electronically label your disk anything you wish, but tell it that you do not want to format another. At this point, the new disk is blank except for the fundamental OS files.
   What are they? Can you display them using DIR?
4. Once formatting is finished, replace the original System Boot Disk.
5. The newly formatted disk is blank, except for the fundamental files, but what about making it bootable and copying all the files?
6. Using the System Boot Disk as the SOURCE and the newly formatted disk as the TARGET, make a copy of the System Boot Disk, keep the original disk in the drive, and type: DISKCOPY A: A:
7. The disk copy process makes an exact copy of the Source disk, which means the Target's existing data is destroyed (don't worry, it was just formatted and is, essentially, empty).
8. Once finished, both disks are identical. Return the original disk to your instructor and continue using the copy.

Part 5: Command Syntax and Built-in Help

1. Although each command has its own syntax and parameter requirements, the following format describes how all commands are structured:

   command {switches} {parameters} {other switches}

   For example, **DIR /W *.EXE** shows a wide listing of all .EXE files in the *current* directory.
2. But since each command has its own purpose, few commands share switch and parameter descriptions. To examine the options for a command (or purpose of the command)— *get a book*!

   Failing this, of course, you can use the help switch: **/?**
   For example, " SET /? " displays help on the SET command.
3. This works for both internal and external commands, but keep in mind that the external commands must be available since **/?** is a switch to the external command program after the program loads into memory.
4. For practice, examine some commands with /?, such as:
   DIR, MEM, PATH, FORMAT, DISKCOPY, COPY, and XCOPY.
5. Can you tell which of the above commands are internal or external?
   (Hint: Where are the various commands located or stored?)

Part 6: Drives and Directories

DRIVES

1. This topic is either the easiest, or the most difficult, to grasp, but because it deals with file organisation, it is very important to understand.
   ("Directories" in CLI ≡ "Folders" in GUI.)
2. A drive is a device for storing programs and data files. In DOS, a drive is defined by a letter and a colon, such as: A: or C:. On DOS/Win systems, A: and B: are the floppy drives, and C: through Z: are hard disks, CD-ROM, or network drives. The limit on drive letters is the alphabet.

   ***Note**: All DOS devices end with a colon (:), such as: CON:, LPT1:, COM2:, and drives (A: to Z:). Look up CON:, LPTn:, COMn:.*
3. To "jump" from one drive to another, only the drive letter is used.
   Jump to the C: drive, type: C:
4. Jump back to A:, type: A:
5. Try some other letters to see which drives are available, but come back to A:\ before continuing.

DIRECTORIES

6. From formatting, all drives start with one major directory: the root, denoted by a single backslash \ . All directories are "below" the root directory (like branches off a main tree trunk).
   This is why drive paths are sometimes describes with a trailing \, as in: A:\ or C:\ (or on network drives as: \\ServerA\ )
7. A directory is like a container, or "box." Boxes can hold things like data files and programs, or even other directories.
8. Like a box in a box, a directory can reside within another directory. These "inner" directories are called *subdirectories*, but since all directories are "inside" (or below) the root directory, they must all be *subdirectories*.
   ***Note:** The terms, "directory" and "subdirectory" are equivalent.*
9. Directories can contain any number of files, of any file size. A directory is just an address and a name with a list of the files and directories it contains. The limit is total disk space for all files in all directories.

10. To make a directory on a disk, use the MKDIR (or MD) command.
    Make a directory on your disk called MYDIR; type: MD A:\MYDIR
11. Notice in the above command that the drive and root directory were used along with the new directory name. Since the current directory is the root, only the command, " MD MYDIR " is required, but the full form is valid and explicitly clear.
12. Now make a directory below MYDIR called DIR2,
    type: MD A:\MYDIR\DIR2
13. Notice again how the whole *directory path* is needed. The reason is the current drive and path is A:\ and DOS needs to be told that the new directory is <u>within</u> MYDIR.
14. Navigation through directories is done via the CHDIR (or CD) command. To navigate into MYDIR, type: CD MYDIR
15. The command prompt will change to reflect the move. Use the DIR command to see what is inside the current directory.
16. Notice 3 directories (as indicated on the right by **<DIR>**).
    "." identifies the current directory,
    ".." the parent (one above) directory, and
    "DIR2" the directory below.
17. "." and ".." are very useful. To move to *where you are*,
    type: CD **.** [← don't forget the **.**]
18. To move back one directory, placing yourself in the root, type: CD **..**
    (at a DOS-Prompt window, CD **...** *can back up to a grandparent!)*
19. Now move back down into MYDIR, type: CD MYDIR
20. Removing a directory is done via the RMDIR (or RD) command. To remove DIR2, type: RD DIR2
21. Notice how the root (\) and MYDIR were not used. Since the current directory is MYDIR, specifying the parent directories is not necessary, although the command also works if the full directory path is provided.
22. As an exercise, perform the following:
    a) create two directories below the root, called **Dir1** and **Dir2**
    b) create another directory below Dir2, called **SubDir3**
    c) change into directory **Dir2**
    d) change into directory **SubDir3**
    e) remove all directories: Dir1, Dir2, and SubDir3
    (note: you can not remove a directory if you are <u>in it</u>)
23. Make sure to finish in A:\ , so the prompt is **A:\>**

Part 7: Directory Related topics

1. The CD command also displays the current directory path; type: CD
2. You are fortunate that the command prompt always displays the current path, this is <u>not</u> always the case.
   To clear the current prompt, type: PROMPT
3. Notice that the directory path has disappeared from the prompt, leaving only the drive letter and the "greater than" symbol (>)
   —so you must always remember which is the current directory
4. There are many Prompt settings (use PROMPT /? to examine them). Experiment with different prompts; try one that shows the <u>date and time</u>.
   (Consult a DOS manual for the special ANSI settings for colours.)
5. To restore the prompt set by **autoexec.bat** on the boot disk,
   type: PROMPT PG *(use /? to learn the meaning of* $P *and* $G*)*
6. Directory usage and moving/copying/accessing files from remote directories, is a skill that requires practise. For this reason Microsoft and Apple changed the name from "directories" to "folders" in the GUI environment: *too many bad memories from using the CLI and directories*.
7. Using directories can be extremely powerful and efficient, especially when 5 DOS commands can be written as one, *or eliminating 12 mouse clicks altogether*.
8. Consider getting a listing of <u>all</u> files on a drive, and in all subdirectories; type: DIR C:\ /A /S
9. Search for a specific file on the hard drive: DIR C:\FORMAT.COM /S
10. Use the information on the screen to jump to the C: drive and move into the directory containing the FORMAT.COM file.
    Do you remember the commands to do this?

Part 8: Command/Execution PATH and Directory PATH

1. Make sure the current drive and directory is A:\
2. From the previous discussion, you have seen that the command/execution path used by COMMAND.COM relies on files being placed into directories and the locations identified to find the correct programs.
3. To see the command/execution path, type: SET and look for the "PATH=" variable; or just type: PATH
4. On the DOS/Win98 System Boot Disk, only one path directory is specified: A:\. For a hard disk with a complete Windows OS, the path may be much longer. The AUTOEXEC.BAT file on the hard disk (or in this case, floppy disk) sets the command/execution path at boot-up.

5. The TYPE command lets you examine text files. To examine the AUTOEXEC.BAT files, type: TYPE A:\AUTOEXEC.BAT
then, type: TYPE C:\AUTOEXEC.BAT
6. It will not seriously effect the current session to change the command/execution path variable to include the MYDIR directory you created on your disk; type: PATH=A:\;A:\MYDIR
7. Notice that a semicolon (**;**) separating the two directory paths A:\ and A:\MYDIR. Also notice that the path for each is complete, so if you wished to include the removed directory DIR2, the statement would look like: PATH=A:\;A:\MYDIR;A:\MYDIR\DIR2
8. Every time the OS is restarted, the PATH must be redefined, which is why it is included in the "automatic execution" batch file, AUTOEXEC.BAT.

   ***Note**: If left blank, the PATH refers only to the "current" directory (also called, the "local" or "focused" directory).*

Part 9: Copying, Moving, Renaming, and Deleting Files; along with discussing XCOPY and MOVE

1. Directory structures allow for the organisation of files stored on disk (floppy or hard); *a Win9x office computer with a standard software installation has 7000-10000 files and 400-800 directories)*. Therefore, knowing how to copy, move, and rename files in the same directory, and remote directories, is as important as the data in the files.
2. Change to the root (\) of drive A:
3. Create a text file on the disk using the *console copy command*, type:
COPY CON: FILE.TXT
4. Type in the sentence, "DOS is simple. I like DOS."
Close the file by pressing <CTRL - Z> (or hitting <F6>) then [ENTER].
5. Use the DIR command to verify that the file is on the disk (and its size), and display the contents of the text file, type: TYPE FILE.TXT
6. In file relocation, there are three (3) possibilities,
1) copy a file to another file with a different name (or in another, remote directory)
2) move a file to a remote directory (possibly with another name)
3) rename the name of a file in the current directory
7. Make a copy of the file to the MYDIR directory,
type: COPY FILE.TXT A:\MYDIR\FILE.TXT
8. In the previous command, the same result could have been obtained by using following commands instead. *Why?*

   a) COPY FILE.TXT MYDIR\FILE.TXT

   b) COPY FILE.TXT MYDIR
9. COPY can also change the name of the file as it copies the contents,
type: COPY FILE.TXT FILE2.TXT
10. Use DIR to see that FILE.TXT and FILE2.TXT have the same size and date…*they should be identical!*
11. There are two commands available for copying files, COPY (internal) and XCOPY (external). The key difference between the two is copying speed. *Research both commands to find another key difference (hint: memory.)*
12. Rather than copying a file, consider relocating the file to another location, type: MOVE FILE2.TXT MYDIR
13. The file is now "moved" to another directory. Moving on the same drive is faster than copying, since no data moves, only the FAT is updated *(FAT—file allocation table).*
14. The MOVE command can also be used to change the name of a file. Change into the MYDIR directory.
15. Change the name of the file in the MYDIR directory,
type: MOVE FILE2.TXT NEWFILE2.TXT
16. Before the MOVE command was introduced to DOS (in version 5.0), the RENAME (or REN) was used to change filenames, and it still exists. Change the filename back, type: REN NEWFILE2.TXT FILE2.TXT
17. So why use REN, if MOVE offers more flexibility?
Consider which is internal *and which* external.
18. Move back to the root (A:\) of the diskette.
19. To delete a file, use the DEL (or ERASE) command to erase the instances of the FILE.TXT file, type: DEL FILE.TXT
20. Type: DEL \MYDIR\FILE.TXT
21. Type: ERASE \MYDIR\FILE2.TXT
22. Finally, remove the directory A:\MYDIR.
Do you remember how?

Part 10: File and Directory Names

1. An interesting note about the names of files and directories in DOS, is that they are limited to a maximum of **8 characters**, with an extension of **3 characters** after the dot (**.**). All files and directories must have at least one character in the filename (the extension is optional for files; and rarely used with directories), but at most 8 total characters can be used.

2. This limitation, called "*8.3*," has been partially hidden in Win9x, and above, through the use of the Virtual FAT (see textbook, VFAT). But filenames must still have an 8.3 version, which is why DIR shows files with long filenames, using a *tilde* (**~**).
3. The 8.3 convention is a relic from the operating system CP/M, that Microsoft *referenced* in creating MS-BASIC and DOS (MS-DOS/PC-DOS). CP/M was originally the most common OS for home computers.
4. But why was 8.3 chosen, and not another, such as 7.4, 9.2, or 11.0? Try to think beyond the "it fits the bit count" reasoning.

   A historically accurate answer is an extra 1% on your final grade.

Part 11: Configuration Files

1. There are two "boot-up" configuration files used with DOS (and Win9x), and the user can modify both,
   - CONFIG.SYS – contains settings to load driver programs for special hardware, or establish settings for the OS environment before COMMAND.COM is loaded and run.
   - AUTOEXEC.BAT – a special *batch program* that is automatically executed by COMMAND.COM. Unlike the statements of CONFIG.SYS, the AUTOEXEC.BAT's statements are just DOS commands grouped together so that the user does not need to enter them manually on each boot.
2. Use EDIT to examine both files on the boot disk, and compare them with similar files on the hard disk (in the C:\ directory).

   Provide a description of each statement (its purpose) in the files on the boot diskette (see listings). Refer to a DOS manual and the Internet.

Part 12: Batch Files, and the Batch Programming Language

1. All Operating Systems have a core programming language, but these languages are not for designing end-user programs, as are languages such as C/C++, COBOL, FORTRAN, VB, or Java.
2. OS languages are intended for manipulating the resources of the operating system, or providing the user a tool for performing this manipulation.

   UNIX-like OSes incorporates its *scripting* languages, VAX/VMS includes DCL (digital command language), and all the Windows OSes have the *batch language*.
   (Windows 2003 Server contains the most feature-filled and powerful batch language—it is almost a complete programming language like C.)
3. Although batch programming is beyond the scope of this introductory workshop, create the batch file below to examine a simple program.
   (you have already examined the Autoexec.bat program)
4. For an excellent reference to DOS and batch programming, consult the web pages:
   http://home7.inet.tele.dk/batfiles/
   http://users.cybercity.dk/~bse26236/batutil/help/INDEX.HTM
5. Open the editor and create a batch file named, myprog.bat, type: EDIT MYPROG.BAT
6. Enter the following program. When finished, exit the editor and run the program by just typing the name of the program: myprog

```
@echo off
echo This program displays different files on a disk
pause

:loop
echo Which? .exe(E), .com(C), .sys(S), or exit(X)
choice /cECSX

if errorlevel 4 goto END
if errorlevel 3 goto SYS
if errorlevel 2 goto COM
if errorlevel 1 goto EXE

goto loop

:EXE
dir /w/a *.exe
goto loop

:COM
dir /w/a *.com
goto loop

:SYS
dir /w/a *.sys
goto loop

:END
exit
```

Final Thoughts about DOS

1. With the large-scale migration of operating systems towards a GUI, it would seem that DOS is finally dead. This is true only if everything you ever do is accomplished graphically, but what if the GUI does not work?
2. Many Windows users have been stranded at the DOS level because Windows refuses to start correctly. Much like being placed in a foreign city with an unfamiliar language, these users are completely lost.
   A *little* knowledge of DOS goes a *very* long way.
3. And even though operating systems like PalmOS, WinCE/Pocket PC and the Apple MacOS are completely GUI, there are others that still have an important CLI. UNIX, Linux, VAX/VMS, and Novell Netware have GUIs, but are useless without knowledge of the fundamental command line instructions.

   Note: *The latest Macintosh Operating System (MacOS 10.x) is based on UNIX, and therefore has a hidden command shell supporting the GUI.*
4. Finally, quoted as being overheard at the release party of MS Windows 95, and sarcastically repeated for each major upgrade that changes the look of the GUI, "I don't care what the clickin' mouse does now, I can still do it the old way in DOS!" (and many users still continue to do so).

Config.sys & Autoexec.bat files (similar to DOS/Win98 System Boot disk)

config.sys

```
break=on

device=a:\himem.sys /shadowram:on /testmem:off
device=a:\emm386.exe noems

dos=high,UMB
buffershigh=30,8
fileshigh=55
switches=/f
lastdrive=z:

devicehigh /l:1,9072 =a:\ansi.sys

rem -- CD-ROM ----------------------
device=a:\mtmcdai.sys /D:ide01

shell=a:\command.com a:\ /e:1204 /p /msg
```

autoexec.bat

```
@echo off
rem verify on

a:\mode.com con:rate=32 delay=1
loadhigh /L:1,6384 a:\doskey.com

a:\amouse.com

rem -- CD-ROM ----------------------
rem a:\mscdex.exe /d:ide01 /m:10 /l:q /s /v
a:\mscdex.exe /d:ide01 /m:10 /s

path=a:\

set prompt=$p$g
```

May the* command line *be with you.