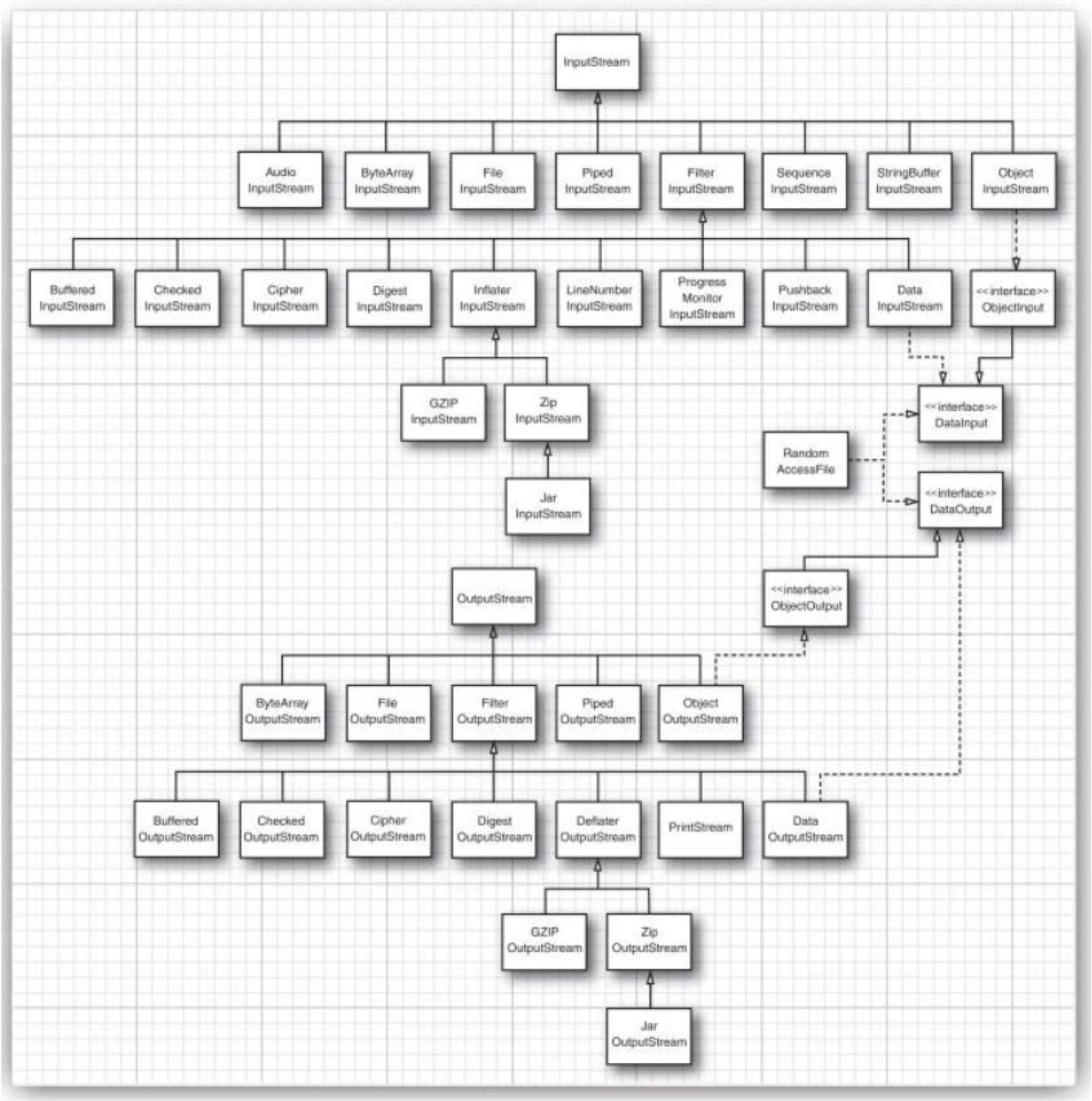
Application Design Using Java

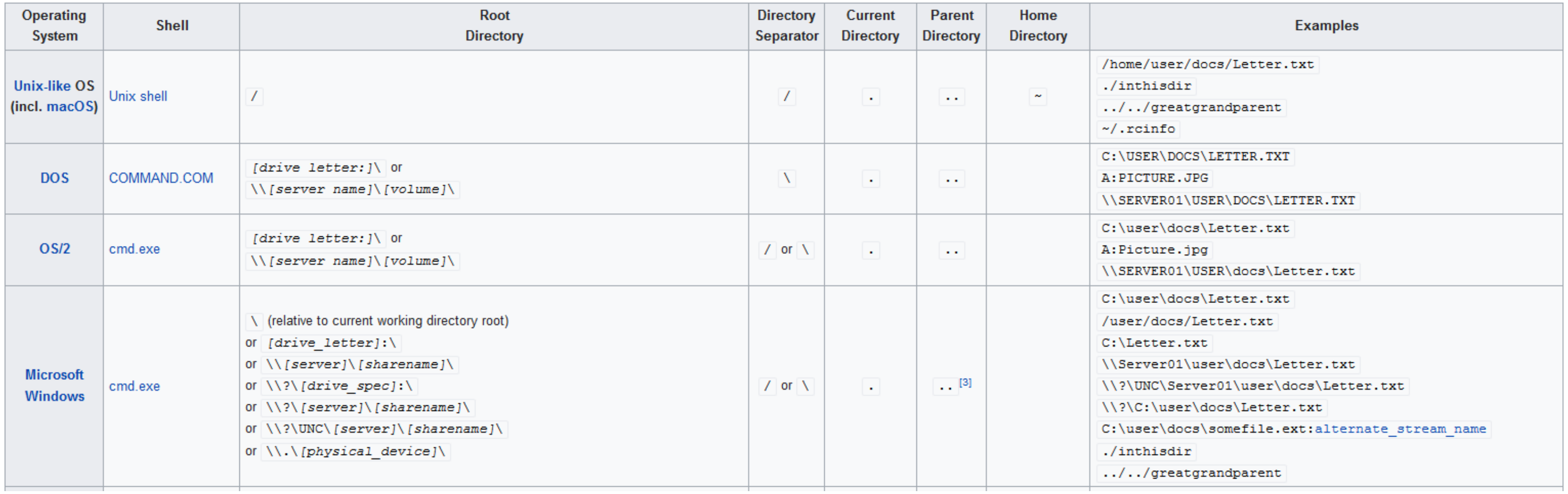
Lecture 05

File I/O

- Packages
  - java.io
  - java.nio
- File/Path name abstraction
  - File
  - Files
  - Path
  - Paths
- Actual I/O
  - Streams
  - RandomAccessFile

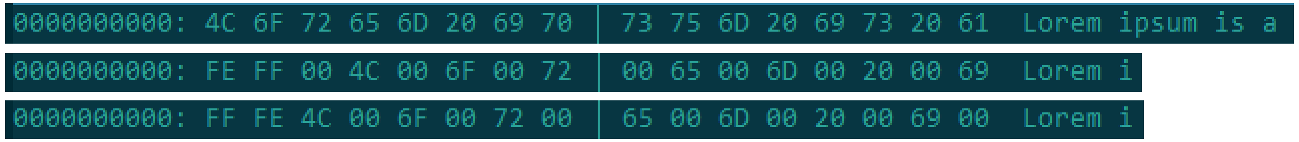


File I/O

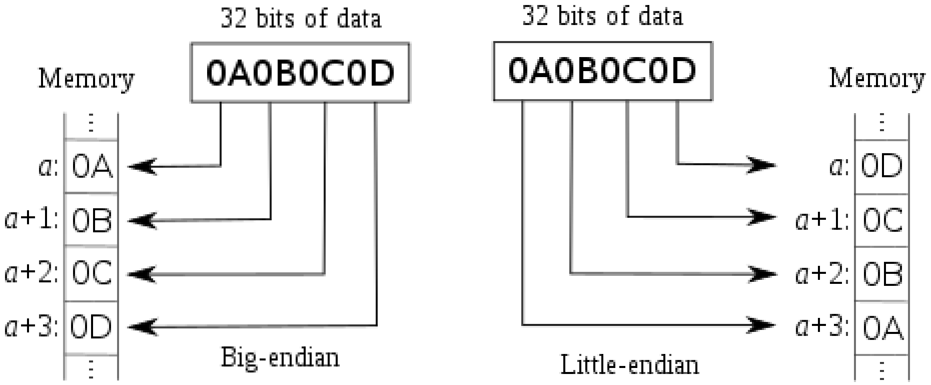


Operating System	Shell	Root Directory	Directory Separator	Current Directory	Parent Directory	Home Directory	Examples
Unix-like OS (incl. macOS)	Unix shell	/	/	.	..	~	/home/user/docs/Letter.txt ./inthisdir ../../greatgrandparent ~/.rcinfo
DOS	COMMAND.COM	[drive letter:]\ or \\[server name]\[volume]\	\	.	..		C:\USER\DOCS\LETTER.TXT A:PICTURE.JPG \\SERVER01\USER\DOCS\LETTER.TXT
OS/2	cmd.exe	[drive letter:]\ or \\[server name]\[volume]\	/ or \	.	..		C:\user\docs\Letter.txt A:Picture.jpg \\SERVER01\USER\docs\Letter.txt
Microsoft Windows	cmd.exe	\ (relative to current working directory root) or [drive_letter]:\ or \\[server]\[sharename]\ or \\?\[drive_spec]:\ or \\?\[server]\[sharename]\ or \\?\UNC\[server]\[sharename]\ or \\.\[physical_device]\	/ or \	.	.. [3]		C:\user\docs\Letter.txt /user/docs/Letter.txt C:\Letter.txt \\Server01\user\docs\Letter.txt \\?\UNC\Server01\user\docs\Letter.txt \\?\C:\user\docs\Letter.txt C:\user\docs\somefile.ext:alternate_stream_name ./inthisdir ../../greatgrandparent

- Path separator
  - File.separator
  - File.pathSeparator
- Endianess problem
  - Big-endian, little-endian, bi-endian
  - Byte order mark (BOM)
    - UTF-8: byte sequence 0xEF,0xBB,0xBF
    - UTF-16, UTF-32: U+FEFF

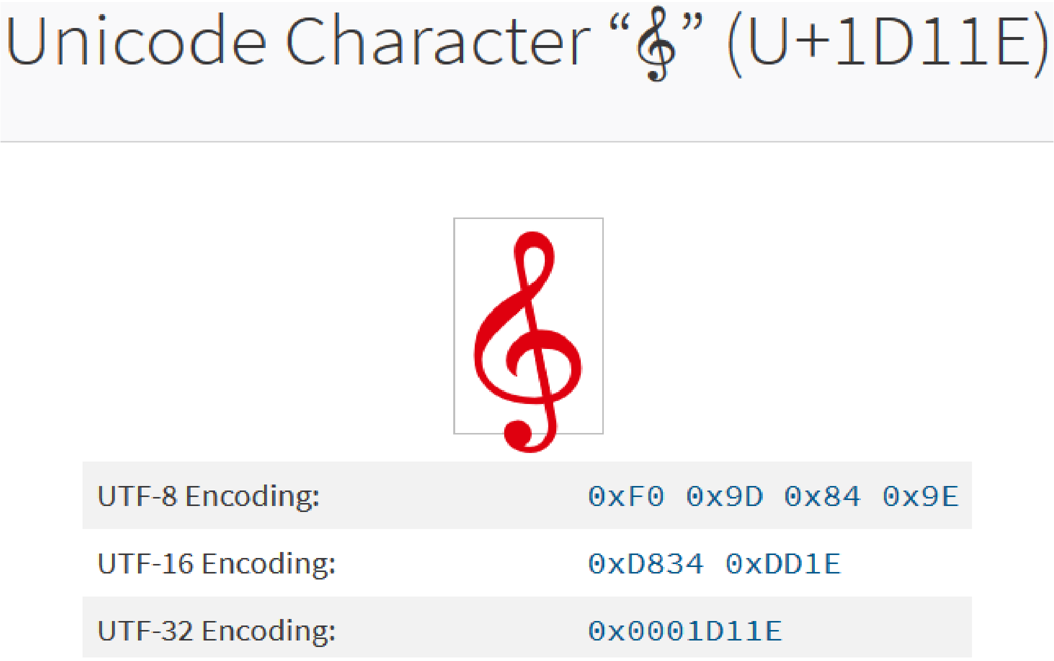


```
0000000000: 4C 6F 72 65 6D 20 69 70 | 73 75 6D 20 69 73 20 61  Lorem ipsum is a
0000000000: FE FF 00 4C 00 6F 00 72 | 00 65 00 6D 00 20 00 69  Lorem i
0000000000: FF FE 4C 00 6F 00 72 00 | 65 00 6D 00 20 00 69 00  Lorem i
```



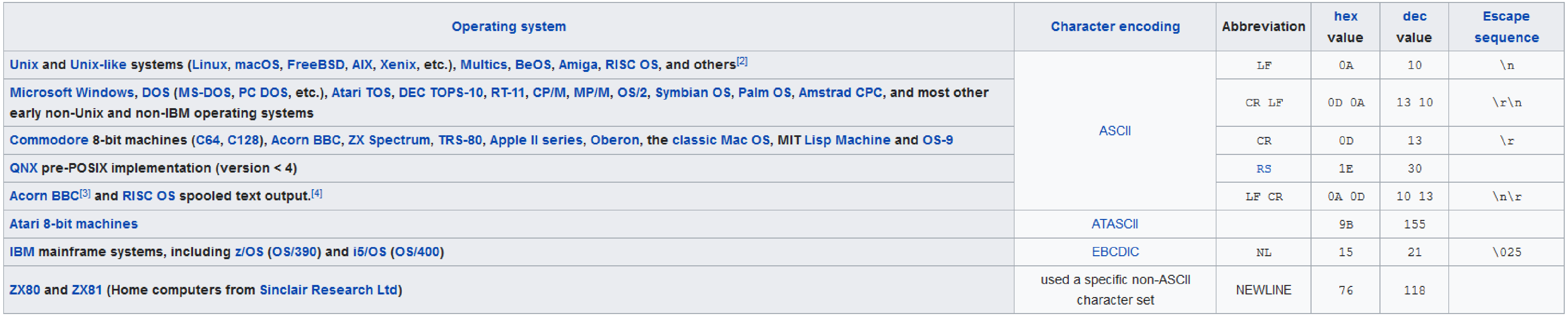
File I/O

- Binary
- Text
  - Encoding problem
  - Newline problem



Unicode Character “𝄞” (U+1D11E)

UTF-8 Encoding:	`0xF0 0x9D 0x84 0x9E`
UTF-16 Encoding:	`0xD834 0xDD1E`
UTF-32 Encoding:	`0x0001D11E`



Operating system	Character encoding	Abbreviation	hex value	dec value	Escape sequence
Unix and Unix-like systems (Linux, macOS, FreeBSD, AIX, Xenix, etc.), Multics, BeOS, Amiga, RISC OS, and others[2]	ASCII	LF	0A	10	\n
Microsoft Windows, DOS (MS-DOS, PC DOS, etc.), Atari TOS, DEC TOPS-10, RT-11, CP/M, MP/M, OS/2, Symbian OS, Palm OS, Amstrad CPC, and most other early non-Unix and non-IBM operating systems		CR LF	0D 0A	13 10	\r\n
Commodore 8-bit machines (C64, C128), Acorn BBC, ZX Spectrum, TRS-80, Apple II series, Oberon, the classic Mac OS, MIT Lisp Machine and OS-9		CR	0D	13	\r
QNX pre-POSIX implementation (version < 4)		RS	1E	30	
Acorn BBC[3] and RISC OS spooled text output.[4]		LF CR	0A 0D	10 13	\n\r
Atari 8-bit machines	ATASCII		9B	155	
IBM mainframe systems, including z/OS (OS/390) and i5/OS (OS/400)	EBCDIC	NL	15	21	\025
ZX80 and ZX81 (Home computers from Sinclair Research Ltd)	used a specific non-ASCII character set	NEWLINE	76	118	

//TODO before next lecture:

- Challenge problem:
  - Write a Java expression that swaps the contents of two variables without using a temporary.

    To be clear, it has to be a single expression, i.e., of the form “expr;”. You may assume variables are whole numbers, and they have already been defined and initialized.
- Java puzzler (posted on Submitty Forum)