

# WTF is a Command Line?

nearly always synonymous with the “shell”

- OK, so WTF is the “shell” then?
- The “shell”: keyboard → directly to the OS
  - GUI access to the shell is through a terminal emulator
  - Popular Linux terminals:
    - KDE – `konsole`
    - Gnome - `gnome-terminal`
    - Check out more:
      - [https://en.wikipedia.org/wiki/Command-line\\_interface](https://en.wikipedia.org/wiki/Command-line_interface)
      - [https://en.wikipedia.org/wiki/List\\_of\\_terminal\\_emulators](https://en.wikipedia.org/wiki/List_of_terminal_emulators)
      - [https://en.wikipedia.org/wiki/Computer\\_terminal](https://en.wikipedia.org/wiki/Computer_terminal)
      - [https://en.wikipedia.org/wiki/Comparison\\_of\\_command\\_shells](https://en.wikipedia.org/wiki/Comparison_of_command_shells)
      - [https://en.wikipedia.org/wiki/List\\_of\\_command-line\\_interpreters](https://en.wikipedia.org/wiki/List_of_command-line_interpreters)

# 2 Types of Unix Shell

(there's also: ksh, tcsh, zsh and others)

- Bourne Shell

```
#!/bin/sh
```

```
i=2
```

```
j=1
```

```
while [ $j -le 10 ]
```

```
do
```

```
    echo '2 **' $j = $i
```

```
    i=`expr $i '*' 2`
```

```
    j=`expr $j + 1`
```

```
done
```

- C Shell

```
#!/bin/csh
```

```
set i = 2
```

```
set j = 1
```

```
while ( $j <= 10 )
```

```
    echo '2 **' $j = $i
```

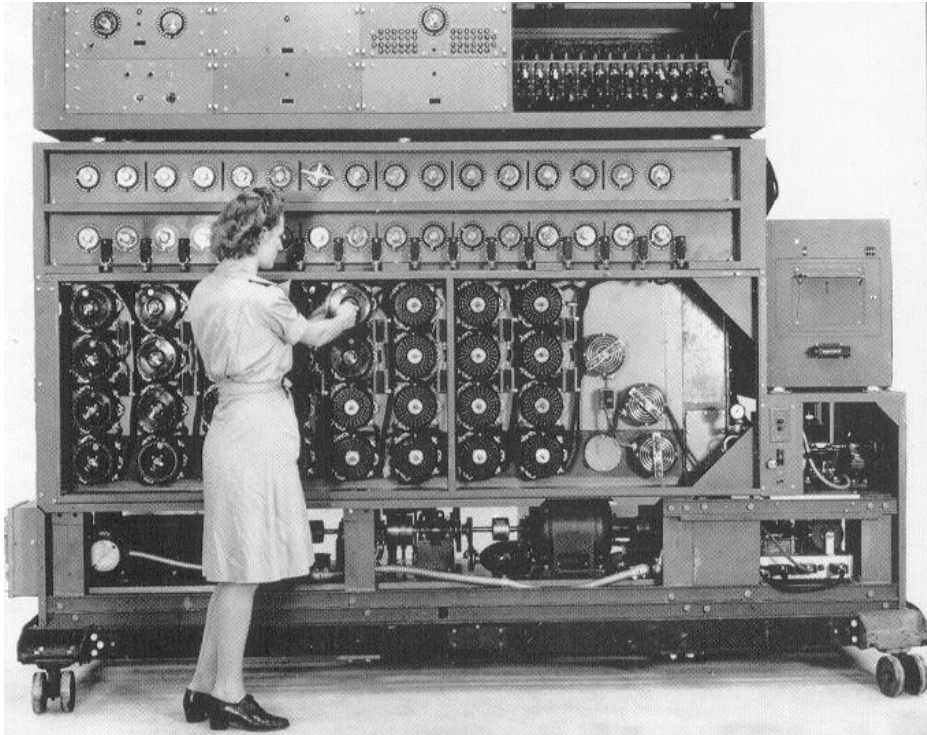
```
    @ i *= 2
```

```
    @ j++
```

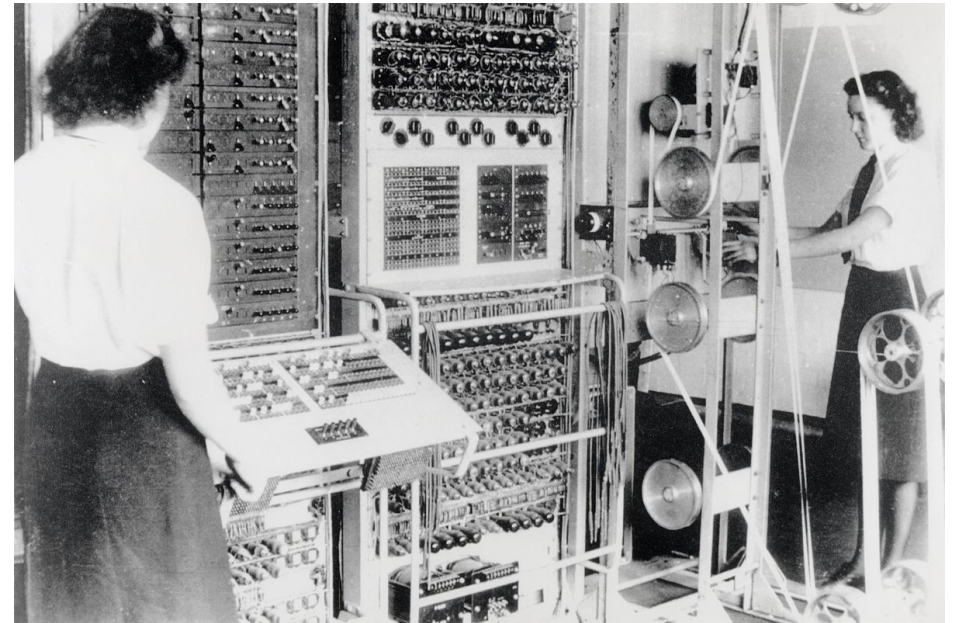
```
end
```

# Early Computer Interactions

switches, lights, no keyboards

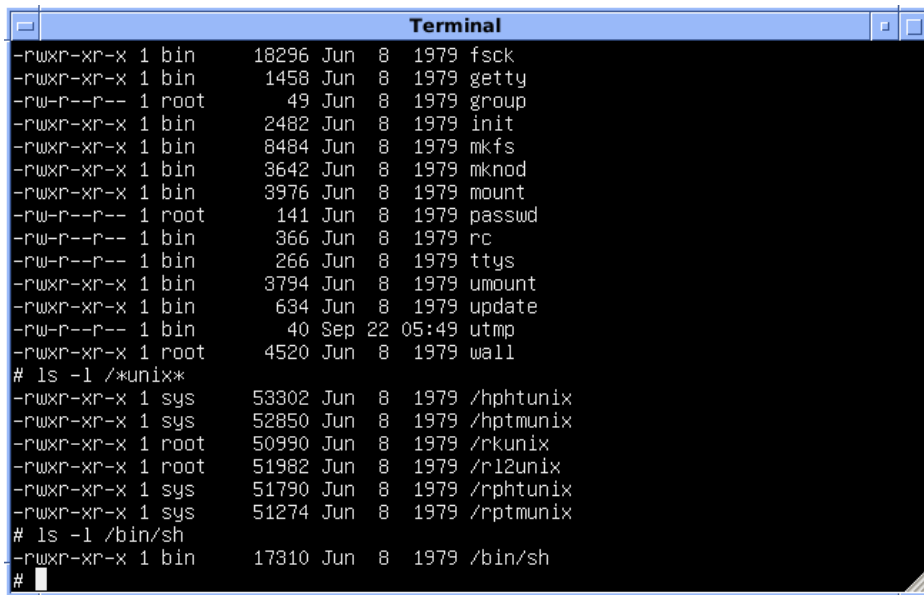


[http://cryptomuseum.com/crypto/bombe/img/us\\_bombe\\_full.jpg](http://cryptomuseum.com/crypto/bombe/img/us_bombe_full.jpg)



[Link to Colossus image](#)

# Text !!!



A terminal window titled "Terminal" showing a list of files and their permissions. The output is as follows:

```
-rwxr-xr-x 1 bin      18296 Jun  8 1979 fsck
-rwxr-xr-x 1 bin      1458 Jun  8 1979 getty
-rw-r--r-- 1 root        49 Jun  8 1979 group
-rwxr-xr-x 1 bin     2482 Jun  8 1979 init
-rwxr-xr-x 1 bin     8484 Jun  8 1979 mkfs
-rwxr-xr-x 1 bin     3642 Jun  8 1979 mknod
-rwxr-xr-x 1 bin     3976 Jun  8 1979 mount
-rw-r--r-- 1 root      141 Jun  8 1979 passwd
-rw-r--r-- 1 bin       366 Jun  8 1979 rc
-rw-r--r-- 1 bin       266 Jun  8 1979 ttys
-rwxr-xr-x 1 bin     3794 Jun  8 1979 umount
-rwxr-xr-x 1 bin       634 Jun  8 1979 update
-rw-r--r-- 1 bin        40 Sep 22 05:49 utmp
-rwxr-xr-x 1 root     4520 Jun  8 1979 wall
# ls -l /*unix*
-rwxr-xr-x 1 sys     53302 Jun  8 1979 /hphtunix
-rwxr-xr-x 1 sys     52850 Jun  8 1979 /hptmunix
-rwxr-xr-x 1 root    50990 Jun  8 1979 /rkunix
-rwxr-xr-x 1 root    51982 Jun  8 1979 /rl2unix
-rwxr-xr-x 1 sys     51790 Jun  8 1979 /rphtunix
-rwxr-xr-x 1 sys     51274 Jun  8 1979 /rptmunix
# ls -l /bin/sh
-rwxr-xr-x 1 bin     17310 Jun  8 1979 /bin/sh
#
```

Image source

- RUNCOM (rc)
  - Louis Pouzin ~1964
- Multics Shell
  - Glenda Schroder ~1965
- Thomson Shell (sh)
  - Ken Thompson at Bell Labs ~1971
- Bourne Shell (sh)
  - Stephen Bourne at Bell Labs ~1979
  - Complete rewrite of Thompson

# File System Commands

- Where are we at and what's here?
  - `$ pwd`
  - Print working directory
  - `$ ls -a`
  - List all files in this directory
- Let's make a directory to work in:
  - `$ mkdir terminal-practice`
  - Make directory
  - `$ ls -a`
  - `$ cd terminal-directory`
  - Change directory to the argument given (terminal-directory)
  - `$ ls -a`

# File System Commands

- Let's make a few files to work in:
  - `$ touch my-scripts.sh`
  - Touch updates a file's timestamp, commonly used to create a file
  - `$ ls -a`
  - `$ touch .gitignore`
  - File names with a "." prepended are "hidden"
  - `$ ls`
  - No gitignore file
  - `$ ls -a`
  - Whomp, there it is!

# What's next

- Shell Scripts
- Python Scripts
- System Commands
- C Files / Programs
- Aliases
- Networking Commands