Complete Guide to Command Prompt, Bash, and PowerShell

Table of Contents

---

1. CORE CONCEPTS THAT COMPRISE ALL SHELLS

1.1 The Shell Hierarchy

A shell is a command interpreter that acts as an intermediary between the user and the operating system kernel.

Bash (Bourne Again Shell) - Standard shell on Linux/Unix systems - Superset of the original Bourne shell (sh) - Incorporated features from Korn shell (ksh) and C shell (csh) - Interactive and scriptable

Command Prompt (CMD) - Native Windows command-line interpreter - Limited scripting capability - Batch file (.bat, .cmd) support - Part of Windows for 30+ years

PowerShell - Modern Windows scripting language - Object-oriented pipeline architecture - Cmdlet-based (verb-noun structure) - More powerful than CMD

1.2 Shell Execution Flow

All shells follow this basic process:

1. **Parsing**: Shell reads input and splits into tokens
2. **Expansion**: Variables, wildcards, and special characters are expanded
3. **Redirection**: Input/output streams are configured
4. **Execution**: Commands are located and executed
5. **Exit Status**: Return code (0 = success, 1-255 = error)

1.3 Command Types

All shells recognize these command categories:

Built-in Commands - Part of the shell itself - No separate process created - Examples: cd, echo, exit, set

External Commands - Separate executable files - Located in PATH directories - New process created (forking)

Functions/Scripts - User-defined command sequences - Can be built-in or external

1.4 Standard Streams

All shells manage three standard streams:

Stream	Descriptor	Purpose
stdin	0	Standard input (keyboard)
stdout	1	Standard output (display)
stderr	2	Error messages

1.5 Variables and Environment

All shells support variables for storing data:

Bash Variable Assignment

```
VARNAME="value"                # Simple variable
export VARNAME="value"         # Environment variable
readonly CONST="value"         # Read-only variable
declare -i NUM=42             # Integer variable
```

CMD Variable Assignment

```
set VARNAME=value             :: Simple variable
set /p VARNAME=Prompt text    :: User input into variable
set /a RESULT=10+5            :: Arithmetic operation
```

PowerShell Variable Assignment

```
$VARNAME = "value"            # Simple variable
[int]$NUM = 42                # Type-declared variable
$env:VARNAME = "value"        # Environment variable
```

1.6 Quoting and Escaping

All shells handle special characters through quoting:

Concept	Bash	CMD	PowerShell
Single quotes	Literal (no expansion)	Not used	Literal
Double quotes	Allow expansion	Not used	Allow expansion
Backticks	Command substitution	Command substitution	Command substitution
Backslash	Escape character	Escape character	Escape character

1.7 Wildcards and Globbing

All shells support pattern matching:

Pattern	Bash	CMD	PowerShell
*	Match any characters	Match any characters	Match any characters
?	Match single character	Match single character	Match single character

[abc]	Match one of set	Match one of set	Match one of set
{a,b,c}	Brace expansion	Not supported	Not standard

---

2. BASH COMMANDS AND SYNTAX

2.1 File and Directory Navigation

pwd - Print working directory

```
Syntax: pwd [options]
Options: -L (logical), -P (physical)
Example: pwd
Output: /home/user/documents
```

cd - Change directory

```
Syntax: cd [directory]
Examples:
  cd /home/user          # Absolute path
  cd ..                  # Parent directory
  cd ~                   # Home directory
  cd -                   # Previous directory
  cd                     # Home directory (no args)
```

ls - List directory contents

```
Syntax: ls [options] [file(s)]
Options:
  -a          All files (including hidden .files)
  -l          Long format with details
  -h          Human-readable file sizes
  -R          Recursive (all subdirectories)
  -S          Sort by file size
  -t          Sort by modification time
  -r          Reverse sort order
Examples:
  ls                    # Current directory
  ls -la /home          # Long format, all files
  ls -lhS *.txt        # Text files, largest first
```

mkdir - Make directory

```
Syntax: mkdir [options] directory_name(s)
Options:
  -p          Create parent directories as needed
  -m          Set permission mode
Examples:
  mkdir mydir
  mkdir -p path/to/deep/directory
```

rmdir - Remove empty directory

```
Syntax: rmdir [options] directory
Options:
  -p          Remove parent directories if empty
Example: rmdir mydir
```

pushd - Save current directory and change to another

```
Syntax: pushd [directory]
Example: pushd /var/log
```

popd - Return to directory saved by pushd

```
Syntax: popd
```

dirs - Display directory stack

```
Syntax: dirs [options]
Example: dirs
```

2.2 File Operations

touch - Create empty file or update timestamp

```
Syntax: touch [options] file_name(s)
Options:
  -a          Change access time only
  -m          Change modification time only
Examples:
  touch newfile.txt
  touch -d "2024-01-15 10:30" file.txt
```

cp - Copy files or directories

```
Syntax: cp [options] source destination
Options:
  -r          Recursive copy (directories)
  -i          Interactive (prompt before overwrite)
  -n          No clobber (don't overwrite)
  -v          Verbose
Examples:
  cp file.txt file_copy.txt
  cp -r /source/dir /destination/dir
```

mv - Move or rename files

```
Syntax: mv [options] source destination
Options:
  -i          Interactive (prompt before overwrite)
  -n          No clobber (don't overwrite)
  -v          Verbose
Examples:
  mv oldname.txt newname.txt
  mv file.txt /path/to/destination/
```

rm - Remove files or directories

```
Syntax: rm [options] file|directory
Options:
  -r (or -R) Recursive (remove directories)
  -f          Force (no confirmation)
  -i          Interactive (confirm each deletion)
  -v          Verbose
Examples:
  rm file.txt
```

```
  rm -rf directory/
  rm *.tmp
```

cat - Display or concatenate files

```
Syntax: cat [options] [file(s)]
Options:
  -n          Number all output lines
  -b          Number non-empty lines only
Examples:
  cat file.txt
  cat file1.txt file2.txt > combined.txt
  cat -n file.txt
```

head - Display first lines of file

```
Syntax: head [options] [file]
Options:
  -n NUM      Display first NUM lines (default 10)
Examples:
  head file.txt
  head -20 file.txt
```

tail - Display last lines of file

```
Syntax: tail [options] [file]
Options:
  -n NUM      Display last NUM lines
  -f          Follow file (watch for changes)
Examples:
  tail file.txt
  tail -20 file.txt
  tail -f /var/log/syslog
```

less - Page through file content

```
Syntax: less [options] file
Examples:
  less largefile.txt
  cat file.txt | less
```

wc - Word/line/character count

```
Syntax: wc [options] [file]
Options:
  -l          Count lines only
  -w          Count words only
  -c          Count bytes only
Examples:
  wc file.txt
  wc -l *.txt
```

2.3 Text Processing and Search

grep - Search for patterns in files

```
Syntax: grep [options] pattern [file(s)]
Options:
  -i          Case-insensitive search
```

```
  -n          Show line numbers
  -c          Count matching lines
  -v          Invert match (non-matching lines)
  -r          Recursive search in directories
Examples:
  grep "error" logfile.txt
  grep -i "ERROR" file.txt
  grep -r "pattern" /path/to/search/
```

sed - Stream editor for text transformation

```
Syntax: sed [options] 'commands' [file]
Basic Commands:
  s/old/new/     Substitute first occurrence per line
  s/old/new/g    Substitute all occurrences
  /pattern/d     Delete lines matching pattern
Examples:
  sed 's/foo/bar/' file.txt
  sed 's/foo/bar/g' file.txt
  sed '/^$/d' file.txt
```

awk - Text processing language

```
Syntax: awk [options] 'program' [file]
Special Variables:
  $0     Entire line
  $1, $2, ...  Fields (columns)
  NR     Record number (line number)
Examples:
  awk '{print $1}' file.txt
  awk -F: '{print $1}' /etc/passwd
```

sort - Sort lines in file

```
Syntax: sort [options] [file]
Options:
  -r          Reverse order
  -n          Numeric sort
  -k NUM      Sort by column NUM
Examples:
  sort file.txt
  sort -r file.txt
```

grep - Search for patterns

```
Syntax: grep [options] pattern [file(s)]
Options:
  -i          Case-insensitive
  -n          Show line numbers
  -c          Count matching lines
  -v          Invert match
  -r          Recursive
Examples:
  grep "error" logfile.txt
  grep -r "pattern" /path/
```

2.4 File Permissions and Ownership

chmod - Change file mode/permissions

```
Syntax: chmod [options] mode file
Modes:
  u=owner, g=group, o=others, a=all
  r=read(4), w=write(2), x=execute(1)
Examples:
  chmod 755 script.sh
  chmod u+x script.sh
  chmod -R 755 directory/
```

chown - Change file owner

```
Syntax: chown [options] owner[:group] file
Examples:
  chown user file.txt
  chown user:group file.txt
  chown -R user:group directory/
```

2.5 Process Management

ps - List processes

```
Syntax: ps [options]
Options:
  -e          All processes
  -f          Full format
  -u USER     Processes for USER
Examples:
  ps
  ps -ef
  ps aux | grep process_name
```

top - Interactive process monitor

```
Syntax: top [options]
Examples:
  top
  top -u username
```

kill - Terminate processes

```
Syntax: kill [options] pid
Signals:
  -9 (SIGKILL)    Force kill
  -15 (SIGTERM)   Graceful termination
Examples:
  kill 1234
  kill -9 1234
```

jobs - List background jobs

```
Syntax: jobs [options]
Examples:
  jobs
  jobs -l
```

2.6 System Information

uname - System information

```
Syntax: uname [options]
Options:
  -a          All information
  -s          Kernel name
  -r          Kernel release
Examples:
  uname
  uname -a
```

whoami - Current user

```
Syntax: whoami
Example: whoami
```

date - Display/set system date and time

```
Syntax: date [options] [+FORMAT]
Examples:
  date
  date '+%Y-%m-%d %H:%M:%S'
```

df - Disk space usage

```
Syntax: df [options] [file]
Options:
  -h          Human-readable sizes
Examples:
  df
  df -h
```

du - Directory size usage

```
Syntax: du [options] [directory]
Options:
  -h          Human-readable
  -s          Summary only
Examples:
  du -sh directory/
  du -sh *
```

2.7 Redirection and Pipes

Redirection Operators

```
>         Redirect stdout (overwrite)
>>        Redirect stdout (append)
<         Redirect stdin from file
2>        Redirect stderr (overwrite)
2>>       Redirect stderr (append)
2>&1      Redirect stderr to stdout
&>        Redirect both stdout and stderr
|         Pipe stdout to next command
|&        Pipe both stdout and stderr (Bash 4.0+)
```

Examples

```
ls > output.txt
ls >> output.txt
grep "error" < input.txt
command 2> errors.log
ls | grep "txt"
cat file.txt | sort | uniq
```

2.8 Conditional Execution

if...then...fi

```
if [ condition ]; then
  commands
elif [ condition ]; then
  commands
else
  commands
fi
```

```
Test Operators:
  -f FILE       File exists and is regular file
  -d DIR        Directory exists
  -z STRING     String is empty
  -n STRING     String is not empty
  STRING1 = STRING2      String equal
  NUM1 -eq NUM2          Numeric equal
  NUM1 -lt NUM2          Numeric less than
```

case...esac

```
case $variable in
  pattern1) commands ;;
  pattern2) commands ;;
  *) default commands ;;
esac
```

2.9 Loops

for...do...done

```
for variable in list; do
  commands
done
```

```
Examples:
for file in *.txt; do echo "Processing $file"; done
for i in {1..10}; do echo $i; done
```

while...do...done

```
while [ condition ]; do
  commands
done
```

until...do...done

```
until [ condition ]; do
  commands
done
```

2.10 Functions

Function Definition and Call

```
function_name() {
  commands
  return status
}

Example:
greet() {
  echo "Hello, $1!"
  return 0
}

greet "Alice"
```

2.11 Special Variables

Variable	Meaning
$0	Script/program name
$1, $2, ...	Positional parameters
$@	All positional parameters (as separate words)
$*	All positional parameters (as single string)
$#	Number of positional parameters
$?	Exit status of last command
$$	Process ID of shell
$!	Process ID of last background command

---

3. COMMAND PROMPT (CMD) COMMANDS AND SYNTAX

3.1 File and Directory Navigation

cd - Change directory

```
Syntax: cd [drive:][path]
Examples:
  cd C:\Users\Documents
  cd ..
  cd \
```

md or mkdir - Create directory

```
Syntax: md [drive:][path]
Examples:
  md NewFolder
  mkdir C:\Users\Public\Shared
```

rd or rmdir - Remove directory

```
Syntax: rd [drive:][path] [/s] [/q]
Options:
  /s     Remove directory and all contents
  /q     Quiet mode (no prompts)
Examples:
  rd EmptyFolder
  rd /s /q FolderWithContents
```

dir - List directory contents

```
Syntax: dir [drive:][path] [options]
Options:
  /s     Recursive subdirectories
  /b     Bare format (names only)
  /p     Pause after each screen
  /o:S   Sort by size
Examples:
  dir
  dir /s /b *.txt
```

tree - Display folder structure

```
Syntax: tree [drive:][path] [/f] [/a]
Options:
  /f     Show files
Examples:
  tree
  tree /f
```

pushd - Save and change directory

```
Syntax: pushd [path]
Example: pushd C:\temp
```

popd - Return to saved directory

```
Syntax: popd
Example: popd
```

3.2 File Operations

copy - Copy files

```
Syntax: copy [/Y | /-Y] [/V] source destination
Options:
  /Y     Suppress prompt to confirm
  /V     Verify copy
Examples:
  copy file1.txt file2.txt
  copy C:\source\file.txt C:\dest\
```

xcopy - Extended copy with options

```
Syntax: xcopy source destination [options]
Options:
  /S     Copy subdirectories
  /E     Copy empty subdirectories
  /Y     Suppress prompts
Examples:
  xcopy C:\source C:\dest /S /Y
```

del or erase - Delete files

```
Syntax: del [drive:][path]filename [options]
Options:
  /P     Confirm before delete
  /F     Force delete of read-only files
  /S     Delete from subdirectories
Examples:
  del file.txt
  del *.tmp
```

type - Display file contents

```
Syntax: type [drive:][path]filename
Examples:
  type file.txt
  type C:\config\settings.ini
```

rename or ren - Rename file

```
Syntax: rename [drive:][path]filename1 filename2
Examples:
  rename oldfile.txt newfile.txt
  ren *.txt *.bak
```

move - Move or rename files

```
Syntax: move [source] [destination]
Examples:
  move file.txt C:\destination\
  move oldname.txt newname.txt
```

attrib - Display/change file attributes

```
Syntax: attrib [+R | -R] [+A | -A] [+S | -S] [+H | -H] file
Options:
  +R or -R   Set/clear read-only
  +H or -H   Set/clear hidden
Examples:
  attrib +H file.txt
  attrib -R file.txt
```

fc - Compare files

```
Syntax: fc [options] file1 file2
Options:
  /b         Binary comparison
Examples:
  fc file1.txt file2.txt
```

3.3 Text Display and Search

find - Search for files

```
Syntax: find [/V] [/C] [/N] [/I] "string" [[drive:][path]filename]
Options:
  /V       Display lines that do not contain string
  /C       Count matching lines
  /N       Show line numbers
  /I       Case-insensitive search
Examples:
  find "error" logfile.txt
  find /i "warning" *.log
```

findstr - Enhanced find/search

```
Syntax: findstr [options] strings [filename]
Options:
  /I       Case-insensitive
  /N       Show line numbers
  /S       Recursive subdirectories
Examples:
  findstr "error" logfile.txt
  findstr /I /S "TODO" *.txt
```

more - Page through file

```
Syntax: more [options] [filename]
Examples:
  more largefile.txt
  dir | more
```

3.4 System Information

systeminfo - Display computer info

```
Syntax: systeminfo [/S computer] [/U domain\username]
Example: systeminfo
```

tasklist - List running processes

```
Syntax: tasklist [/V] [/FI filter] [/FO format]
Options:
  /V       Verbose format
  /FI      Filter by criteria
Examples:
  tasklist
  tasklist /V
```

taskkill - Terminate process

```
Syntax: taskkill [/PID pid | /IM imagename] [/F] [/T]
Options:
  /PID     Process ID
  /IM      Image name (process name)
  /F       Force termination
Examples:
  taskkill /IM notepad.exe
  taskkill /PID 1234 /F
```

date - Display/set date

```
Syntax: date /T
Example: date /T
```

time - Display/set time

```
Syntax: time /T
Example: time /T
```

ver - Display Windows version

```
Syntax: ver
Example: ver
```

whoami - Display current username

```
Syntax: whoami [/UPNFQDN | /UPN]
Example: whoami
```

ipconfig - Display IP configuration

```
Syntax: ipconfig [/all] [/release] [/renew]
Options:
  /all      Show full information
Examples:
  ipconfig
  ipconfig /all
```

ping - Test network connectivity

```
Syntax: ping [-t] [-n count] target
Options:
  -t        Ping continuously
  -n count  Number of ping attempts
Examples:
  ping google.com
  ping -n 5 192.168.1.1
```

tracert - Trace route to target

```
Syntax: tracert [-h hops] target
Example: tracert google.com
```

3.5 Variables and Environment

set - Display or set environment variables

```
Syntax: set [variable=[value]]
Examples:
  set
  set MYVAR=Hello
  echo %MYVAR%
```

setlocal - Start local variable scope

```
Syntax: setlocal [ENABLEDELAYEDEXPANSION]
Example: setlocal ENABLEDELAYEDEXPANSION
```

endlocal - End local variable scope

```
Syntax: endlocal
Example: endlocal
```

3.6 Conditional Logic

if...else

```
if condition (
  commands
) else (
  commands
)

Conditions:
  ==        String comparison
  EQU       Numeric equal
  NEQ       Numeric not equal
  LSS       Numeric less than
  GTR       Numeric greater than
  EXIST     File/directory exists
  DEFINED   Variable is defined
```

3.7 Loops

for...in...do

```
for %%variable in (set) do commands
  or
for /F %%variable in (file) do commands
  or
for /L %%variable in (start,step,end) do commands

Examples:
for %%F in (*.txt) do echo %%F
for /L %%i in (1,1,10) do echo %%i
```

3.8 Labels and Goto

goto - Jump to label

```
:label
  commands
goto label

Example:
:option1
echo You chose option 1
goto end
```

3.9 Batch File Execution

call - Call another batch file or subroutine

```
Syntax: call [drive:][path]filename [arguments]
Examples:
  call other_script.bat
  call :subroutine arg1
```

exit - Exit batch file

```
Syntax: exit [/b] [exit_code]
Options:
  /b       Exit batch file only
Examples:
  exit
  exit /b 0
```

3.10 Input and Output

echo - Display text

```
Syntax: echo [text]
Examples:
  echo Hello World
  echo.
  @echo off
```

pause - Pause execution

```
Syntax: pause [message]
Example: pause
```

title - Set window title

```
Syntax: title [text]
Example: title My Command Prompt
```

color - Set console colors

```
Syntax: color [background][foreground]
Examples:
  color 0A
  color F0
```

cls - Clear screen

```
Syntax: cls
Example: cls
```

---

4. POWERSHELL COMMANDS AND SYNTAX

4.1 Cmdlet Structure and Syntax

PowerShell uses a Verb-Noun naming convention:

Get-Command, Get-ChildItem, Set-Location, New-Item, Remove-Item, Start-Process, Stop-Service

4.2 Basic Navigation and Directory Commands

Get-Location - Show current directory

```
Syntax: Get-Location [-PSProvider type]
Aliases: pwd, gl
Examples:
  Get-Location
  pwd
```

Set-Location - Change directory

```
Syntax: Set-Location [-Path] path [-PassThru]
Aliases: cd, sl
Examples:
  Set-Location C:\Users\Documents
  cd ..
```

Get-ChildItem - List directory contents

```
Syntax: Get-ChildItem [[-Path] path] [-Filter pattern] [-Recurse] [-Force]
Aliases: dir, ls, gci
Examples:
  Get-ChildItem
  ls -Path C:\Users
  dir -Recurse -Filter "*.txt"
```

New-Item - Create file or directory

```
Syntax: New-Item -Name name -ItemType type [-Value value]
Examples:
  New-Item -Name "newfile.txt" -ItemType File
  New-Item -Name "newfolder" -ItemType Directory
```

Remove-Item - Delete file or directory

```
Syntax: Remove-Item [-Path] path [-Recurse] [-Force]
Aliases: rm, del, rmdir
Examples:
  Remove-Item file.txt
  rm -Path C:\temp\folder -Recurse -Force
```

Copy-Item - Copy file or directory

```
Syntax: Copy-Item [-Path] source [-Destination] dest [-Recurse] [-Force]
Examples:
  Copy-Item file.txt copy.txt
  Copy-Item -Path "C:\source" -Destination "C:\dest" -Recurse
```

Move-Item - Move or rename

```
Syntax: Move-Item [-Path] source [-Destination] dest [-Force]
Aliases: mv
Examples:
  Move-Item file.txt newlocation\
  mv oldname.txt newname.txt
```

4.3 File Content Manipulation

Get-Content - Read file contents

```
Syntax: Get-Content [-Path] path [-Head lines] [-Tail lines]
Aliases: gc, cat, type
Examples:
  Get-Content file.txt
  gc file.txt -Head 10
```

Set-Content - Write to file (overwrite)

```
Syntax: Set-Content [-Path] path -Value value
Aliases: sc
Examples:
  Set-Content file.txt -Value "New content"
```

Add-Content - Append to file

```
Syntax: Add-Content [-Path] path -Value value
Aliases: ac
Examples:
  Add-Content log.txt -Value "New entry"
```

Select-String - Search file contents

```
Syntax: Select-String [-Pattern] pattern [[-Path] path]
Aliases: sls
Examples:
  Select-String "error" logfile.txt
  sls "ERROR" *.log
```

4.4 Process Management

Get-Process - List processes

```
Syntax: Get-Process [[-Name] name] [-Id pid]
Aliases: gps, ps
Examples:
  Get-Process
  gps notepad
```

Stop-Process - Terminate process

```
Syntax: Stop-Process [[-Id] pid | [-Name] name] [-Force]
Aliases: kill
Examples:
  Stop-Process -Name notepad
  Stop-Process -Id 1234 -Force
```

Start-Process - Launch process

```
Syntax: Start-Process [-FilePath] path [-ArgumentList args] [-NoNewWindow] [-Wait]
Examples:
  Start-Process -FilePath "notepad.exe"
  Start-Process notepad -ArgumentList "file.txt"
```

4.5 System Information

Get-ComputerInfo - System information

```
Syntax: Get-ComputerInfo [-Property properties]
Example: Get-ComputerInfo
```

Get-Date - Get system date/time

```
Syntax: Get-Date [-Format format]
Examples:
  Get-Date
  Get-Date -Format "yyyy-MM-dd HH:mm:ss"
```

Measure-Object - Calculate statistics

```
Syntax: Measure-Object [-Property prop] [-Sum] [-Average]
Aliases: measure
Examples:
  Get-ChildItem | Measure-Object -Property Length -Sum
```

4.6 Help and Documentation

Get-Help - Display command help

```
Syntax: Get-Help [command] [-Detailed] [-Full] [-Examples]
Aliases: help
Examples:
  Get-Help Get-ChildItem
  help Get-Content -Full
```

Get-Command - List available commands

```
Syntax: Get-Command [[-Name] name] [-CommandType type]
Aliases: gcm
Examples:
  Get-Command
  Get-Command -Name Get-*
```

4.7 Filtering and Selection

Where-Object - Filter objects by criteria

```
Syntax: Where-Object -Property property -Operator value
         Where-Object {condition}
Aliases: where, ?
Examples:
  Get-Process | Where-Object {$_.Handles -gt 1000}
  Get-ChildItem | where {$_.Length -gt 1MB}
```

Select-Object - Select/project object properties

```
Syntax: Select-Object [-Property] properties
Aliases: select
Examples:
  Get-Process | Select-Object Name, Id
  Get-ChildItem | Select-Object Name -First 5
```

Sort-Object - Sort objects

```
Syntax: Sort-Object [-Property] property [-Descending]
Aliases: sort
Examples:
  Get-Process | Sort-Object -Property Memory -Descending
```

4.8 Piping and Output

Pipeline (|) - Connect cmdlet output to input

```
command1 | command2 | command3

Examples:
Get-Process | Where-Object {$_.Memory -gt 100MB} | Sort-Object Memory
```

```
Get-ChildItem | Select-Object Name | Out-File results.txt
```

Out-File - Send output to file

```
Syntax: Out-File [-FilePath] path [-Append]
Examples:
  Get-Process | Out-File -FilePath processes.txt
```

Out-GridView - Display in grid window

```
Syntax: Out-GridView [-InputObject object]
Example: Get-Process | Out-GridView
```

4.9 Conditional Statements

if...elseif...else

```
if (condition) {
  commands
} elseif (condition) {
  commands
} else {
  commands
}
```

switch - Multi-branch selection

```
switch (value) {
  pattern1 { commands }
  pattern2 { commands }
  default { commands }
}
```

4.10 Loops

foreach - Iterate over collection

```
foreach ($item in $collection) {
  commands
}
```

ForEach-Object - Pipeline iteration

```
... | ForEach-Object {commands}
Aliases: foreach, %
Examples:
  1..10 | ForEach-Object { Write-Host $_ }
```

while - Loop while condition true

```
while (condition) {
  commands
}
```

do...while - Execute then loop

```
do {
  commands
} while (condition)
```

4.11 Functions

Function Definition

```
function FunctionName {
  param(
    [type]$parameter1,
    [type]$parameter2
  )
  commands
  return value
}
```

---

5. SHARED CONCEPTS ACROSS ALL SHELLS

5.1 Command Structure

All shells follow basic command structure:

`command [options/flags] [arguments]`

5.2 Exit Status/Return Codes

All shells track command success/failure:

Exit Code	Meaning
0	Success
1-255	Error (specific number indicates error type)
127	Command not found

5.3 Comparison Operators (All Shells)

Bash - `-eq`, `-ne`, `-lt`, `-le`, `-gt`, `-ge` (numeric) - `=`, `!=`, `<`, `>` (string)

CMD - `EQU`, `NEQ`, `LSS`, `LEQ`, `GTR`, `GEQ` (numeric) - == (string equality)

PowerShell - `-eq`, `-ne`, `-lt`, `-le`, `-gt`, `-ge` - `-like`, `-notlike`, `-match`, `-notmatch`

5.4 Logical Operators (All Shells)

Bash - `&&` AND (execute if previous succeeds) - `||` OR (execute if previous fails) - `!` NOT (negate condition)

CMD - `&&` AND (execute if previous succeeds) - `||` OR (execute if previous fails)

PowerShell - `-and` AND - `-or` OR - `-not` NOT

5.5 String Concatenation

Bash

```
str1="hello"
str2="world"
combined="$str1 $str2"
```

CMD

```
set str1=hello
set str2=world
set combined=%str1% %str2%
```

PowerShell

```
$str1 = "hello"
$str2 = "world"
$combined = "$str1 $str2"
$combined = $str1 + $str2
```

5.6 Comments

Bash

```
# This is a comment
```

CMD

```
:: This is a comment
REM This is also a comment
```

PowerShell

```
# This is a comment
```

---

6. QUICK REFERENCE COMPARISON

Common Tasks Across Shells

Task	Bash	CMD	PowerShell
List files	ls	dir	Get-ChildItem
Change dir	cd	cd	Set-Location
Create file	touch	copy nul	New-Item
Copy file	cp	copy	Copy-Item
Delete file	rm	del	Remove-Item
Search text	grep	findstr	Select-String
Process list	ps	tasklist	Get-Process
Kill process	kill	taskkill	Stop-Process
Set variable	export	set	$var =
Conditional	if [ ]	if ( )	if () { }
Loop	for..in	for	foreach
Function	() { }	:label	function
Redirect output	>	>	| Out-File
Pipe	|	|	|

File Extension Conventions

- **Bash**: No extension, shebang line `#!/bin/bash`
- **CMD**: `.bat` or `.cmd`
- **PowerShell**: `.ps1`