Basic Character Matching

- `[who]` matches any single character: w, h, or o in text.
- `[0-9]` matches any single digit.
- `[a-z]` matches any lowercase letter.

Grep: Searching for Patterns

- Find all Python files:
  `ls | grep '\.py$'`
  Matches filenames ending with ".py".
- Find lines ending with "h":
  `grep 'h$' file.txt`
  Matches lines where "h" is at the end.
- Find lines where the word "error" appears anywhere:
  `grep 'error' logfile.txt`
  Matches the string "error" in each line.

Sed: Editing Text in Place

- Replace "foo" with "bar" in a file:
  `sed 's/foo/bar/g' input.txt`
  Every instance of "foo" is changed to "bar".
- Delete lines containing digits:
  `sed '/[0-9]/d' file.txt`
  Removes lines that contain any digit.

Awk: Field Extraction

- Print lines with a number in the first field:
  `awk '$1 ~ /^[0-9]+$/' file.txt`
  First field matches one or more digits.

Advanced Patterns

- Match a four-digit year:
  `grep -E '[0-9]{4}' file.txt`
  Finds any sequence of exactly four digits.

- Validate phone numbers:
  `grep -E '^$$[0-9]{3}$$ [0-9]{3}-[0-9]{4}$' phonelist.txt`
  Matches precisely the pattern "(XXX) XXX-XXXX".
- At least two consecutive digits:
  `grep -E '[0-9]{2,}' file.txt`
  Two or more consecutive digits.

Boundaries and Special Characters

- Match lines starting with "abc":
  `grep '^abc' file.txt`
  "^" anchors the pattern at the line start.
- Match lines ending with "xyz":
  `grep 'xyz$' file.txt`
  "$" anchors the pattern at the line end.

More Practical Uses

- Delete non-numeric lines:
  `grep '^[0-9]*$' file.txt`
  Matches lines that contain only digits.
- Extract date formats (YYYY-MM-DD):
  `grep -E '[0-9]{4}-[0-9]{2}-[0-9]{2}' file.txt`
  Finds dates in the file.