Piping and redirecting output: Takeaways

Syntax

- Redirecting standard output to overwrite a file:

```
echo "Dataquest is best!" > best.txt
```

- Redirecting standard output to add text to a file:

```
echo "Dataquest is awesome!" >> awesome.txt
```

- Sorting the lines of a file in alphabetical order:

```
sort < beer.txt
```

- Sorting the lines of a file in reverse alphabetical order:

```
sort -r < beer.txt
```

- Searching through contents of a file to find a specific instance of text:

```
grep "pass" beer.txt
```

- Using a wildcard character to represent a single character when searching:

```
grep "beer" beer?.txt
```

- Using a wildcard character to represent any number of characters:

```
grep "beer" *.txt
```

- Redirecting standard input to standard output:

```
tail -n 10 logs.txt | grep "Error"
```

- Printing the contents of a file:

```
cat beer.txt
```

- Running two commands sequentially:

```
echo "All the beers are gone" >> beer.txt && cat beer.txt
```

- Using a backslash escape character to add a quote to a file:

```
echo "\"Get out of here,\" said Neil Armstrong to the moon people." >> famous_quotes.txt
```

Concepts

- The `?` wildcard character is used to represent a single, unknown character.
- The `*` wildcard character is used to represent any number of characters.
- We can use the pipe character (`|`) to send the standard output of one command to the standard output of another command.
- Escape characters tell the shell to treat the character coming directly after it as a plain character.

Resources

- Escape Characters
- Wildcard Characters