# Pipeline Architecture

CSSE6400

Brae Webb

February 28, 2022

Question

# Can you name a *pipeline archiecture*?

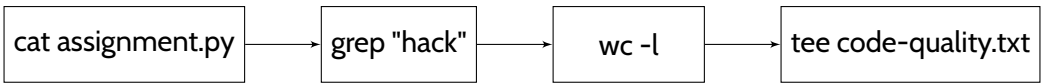
Question

Can you name a *pipeline archiecture*?

Answer

How about *bash*?

```
>> cat assignment.py | grep "hack" | wc -l \
    | tee code-quality.txt
```


cat assignment.py
grep "hack"
wc -l
tee code-quality.txt


Filter
Pipe
Filter
Pipe
Filter


Filters

# Modular software components


Filters

# Modular software components

Pipes

# The flow of data between filters

Producers

# Source of data.

Producers

# Source of data.

Transformers

# Transform data.

Producers

Source of data.

Transformers

Transform data.

Testers

Filter data.

Producers

Source of data.

Transformers

Transform data.

Testers

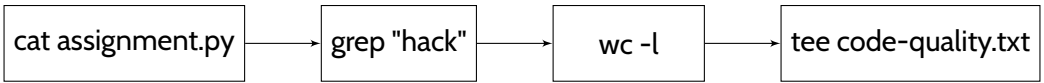
Filter data.

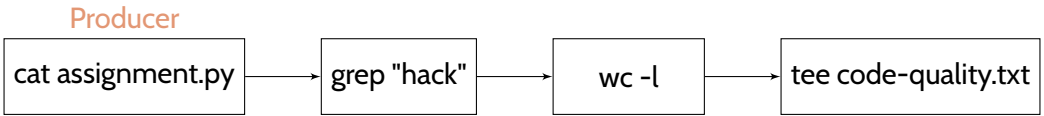
Consumers

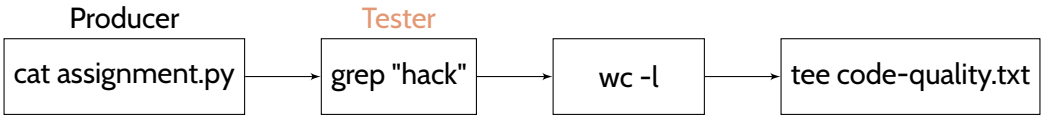
Target for results.

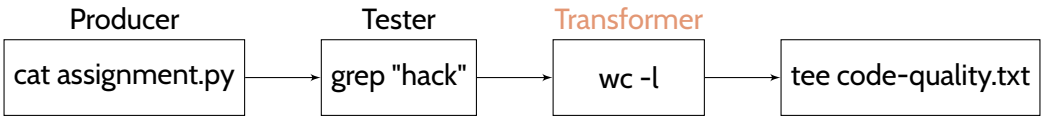
Exercise

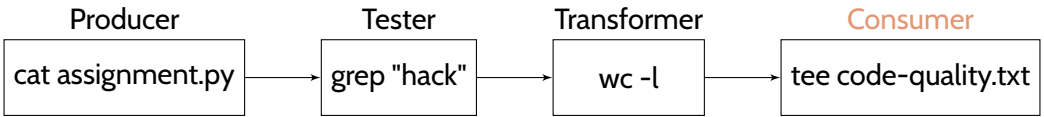
# Label the bash pipeline.


cat assignment.py
grep "hack"
wc -l
tee code-quality.txt


Producer
cat assignment.py
grep "hack"
wc -l
tee code-quality.txt


Producer
Tester
cat assignment.py
grep "hack"
wc -l
tee code-quality.txt


Producer
cat assignment.py
Tester
grep "hack"
Transformer
wc -l
tee code-quality.txt


Producer
cat assignment.py
Tester
grep "hack"
Transformer
wc -l
Consumer
tee code-quality.txt

Question

# Does this seem familiar?

Poll

# Who has done *functional programming*?

```
let sum = reduce
            (λ total value → total + value)
            (map (λ seq → size seq) xs)
            0
```

```
1 let sum = reduce + (map size xs) 0
```

### Definition 1. *map*

```
1 map : (τ1 → τ2) → τ1Seq → τ2Seq
2 map f xs
```

Definition 2. *reduce*

```
1 reduce : (τ1 → τ1 → τ1) → τ1Seq → τ1 → τ1Seq
2 reduce f xs initial
```

Question

# What's the advantage of the map reduce pattern?

Question

# What's the advantage of the map reduce pattern?

Answer

Parallelism [1]

Question

# Using pipeline terminology, what filters do the *map* and *reduce* operators correspond to?

## References

[1] Jeffrey Dean and Sanjay Ghemawat.
Mapreduce: Simplified data processing on large clusters.
In *OSDI'04: Sixth Symposium on Operating System Design and Implementation*, pages 137–150, San Francisco, CA, 2004.