Hadoop Streaming

Hadoop Streaming

- Hadoop streaming: Hadoop utility distribution
- It allows you to create and run map/reduce jobs with ***any executable*** or script as mapper/reducer:
  - C, Python, Java, Ruby, C#, perl, shell commands
- Map and Reduce classes can even be written in different languages

Using Streaming Utility

```
> hadoop jar <dir>/hadoop-
*streaming*.jar \
      -file /path/to/mapper.py \
      -mapper /path/to/mapper.py \
      -file /path/to/reducer.py \
      -reducer /path/to/reducer.py \
      -input /user/hduser/books/* \
      -output /user/hduser/books-output
```

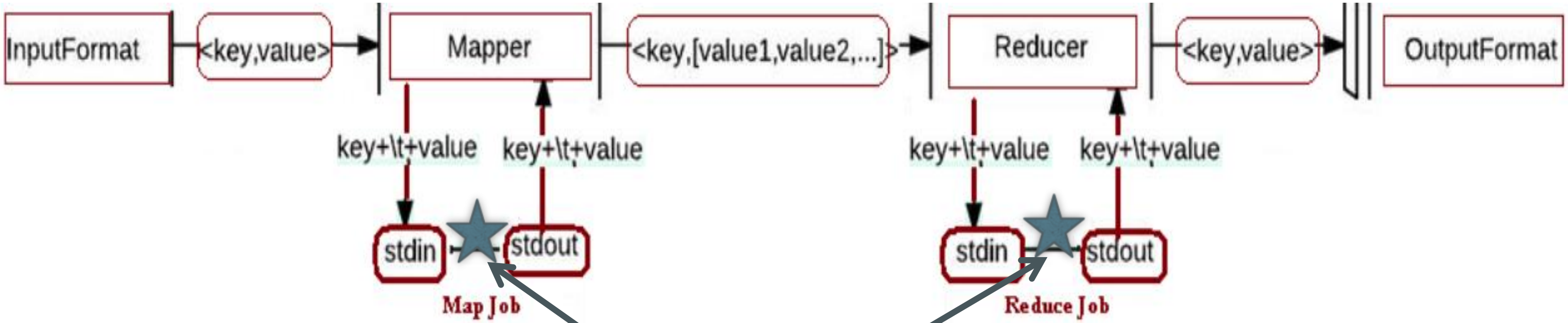
Path to the streaming jar library

Location of mapper file, and define it as mapper

Location of reducer file, and define it as reducer

Input and output locations

Execution Flow



Your Code

Hadoop Streaming: Basic Concept

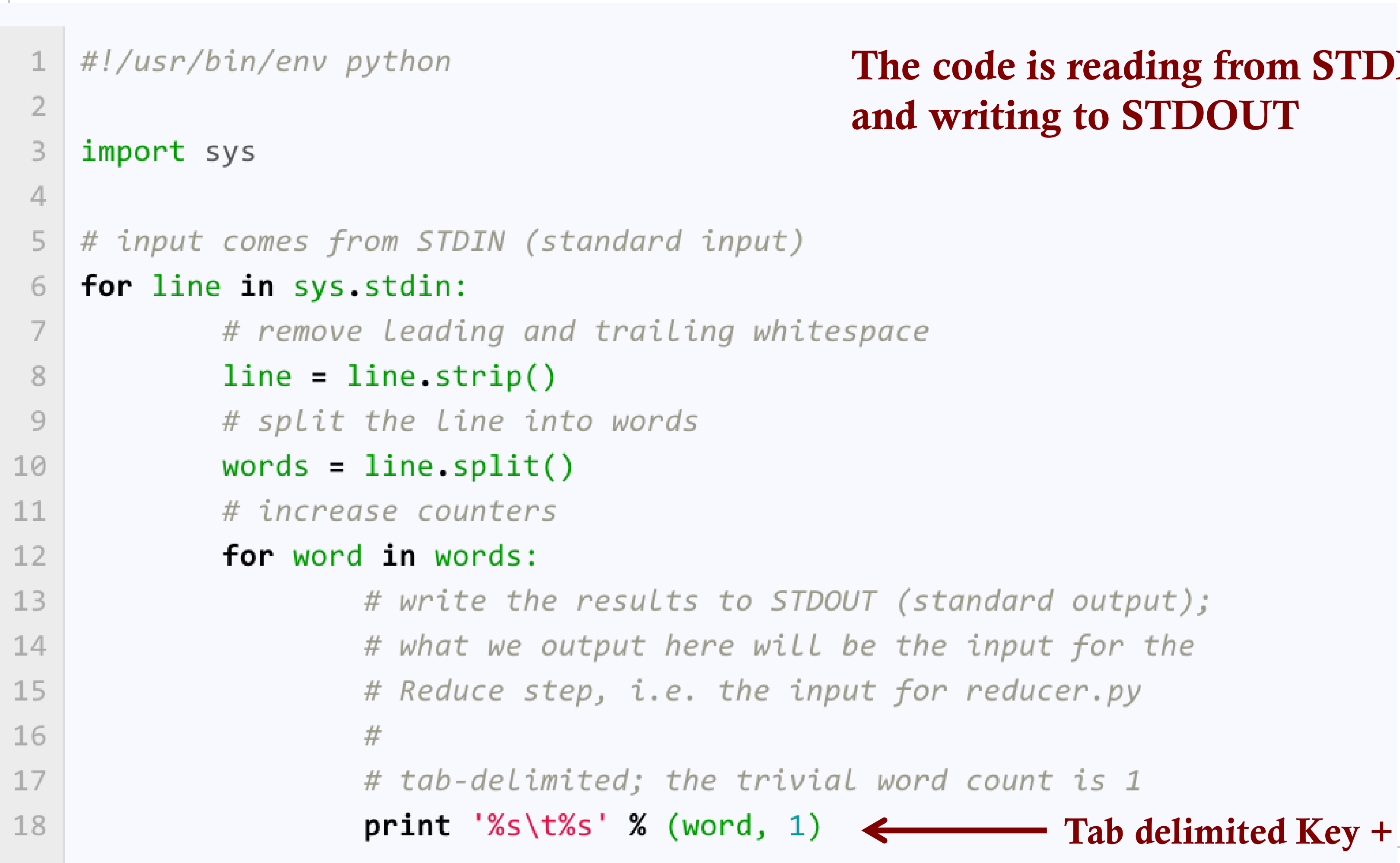
- Map and reduce functions read their input from STDIN and produce their output to STDOUT

- **Map**
  - Hadoop streaming reads the input data line by line
  - Pass it to the map function through the STDIN
  - ***Do your code (any language)***
  - ***Produce output to STDOUT***
    - ***Key + \t + value***

  User's code

  - Hadoop streaming reads output from STDOUT
    - Performs shuffling and sorting based on Key part

WordCount: Mapper.py



```
#!/usr/bin/env python

import sys

# input comes from STDIN (standard input)
for line in sys.stdin:
        # remove leading and trailing whitespace
        line = line.strip()
        # split the line into words
        words = line.split()
        # increase counters
        for word in words:
                # write the results to STDOUT (standard output);
                # what we output here will be the input for the
                # Reduce step, i.e. the input for reducer.py
                #
                # tab-delimited; the trivial word count is 1
                print '%s\t%s' % (word, 1)
```

The code is reading from STDIN and writing to STDOUT

Tab delimited Key + value

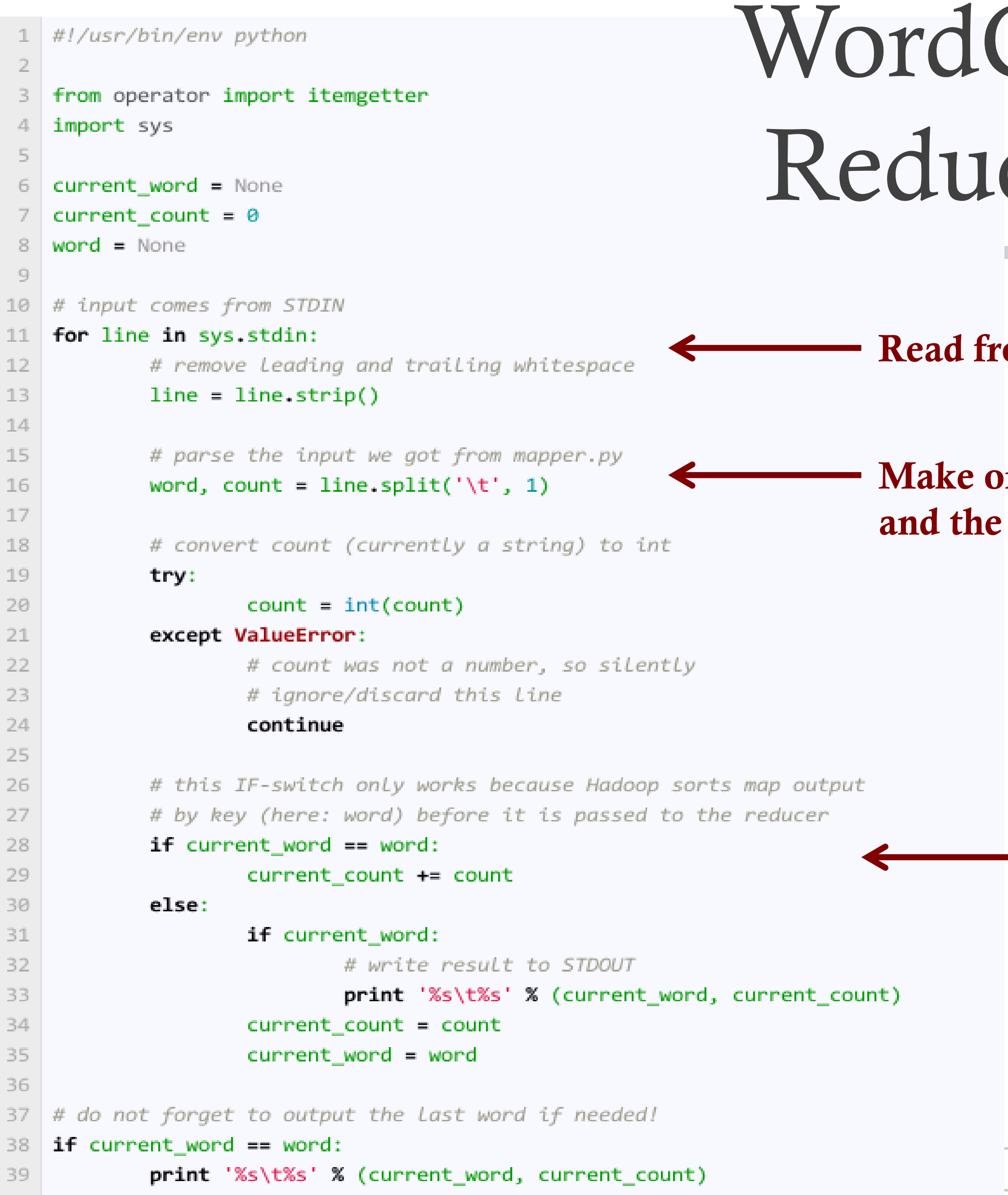
Hadoop Streaming

- **Reducer**
  - Hadoop streaming shuffles and sorts map outputs based on Key
  - Passes one record at a time to reduce function through STDIN

  - ***Do your code (any language)***
  - ***Produce output to STDOUT***
    - ***Key + \t + value***

  - Hadoop streaming reads the output from STDOUT
    - Writes to the output file

WordCount: Reducer.py



```python
#!/usr/bin/env python

from operator import itemgetter
import sys

current_word = None
current_count = 0
word = None

# input comes from STDIN
for line in sys.stdin:
        # remove leading and trailing whitespace
        line = line.strip()

        # parse the input we got from mapper.py
        word, count = line.split('\t', 1)

        # convert count (currently a string) to int
        try:
                count = int(count)
        except ValueError:
                # count was not a number, so silently
                # ignore/discard this line
                continue

        # this IF-switch only works because Hadoop sorts map output
        # by key (here: word) before it is passed to the reducer
        if current_word == word:
                current_count += count
        else:
                if current_word:
                        # write result to STDOUT
                        print '%s\t%s' % (current_word, current_count)
                current_count = count
                current_word = word

# do not forget to output the last word if needed!
if current_word == word:
        print '%s\t%s' % (current_word, current_count)
```

Read from STDIN

Make one split to get the word and the count

If it is like the previous word, then increment. Otherwise, report.

More Information

- http://www.michael-noll.com/tutorials/writing-an-hadoop-mapreduce-program-in-python/