Yandex

MapReduce

Streaming

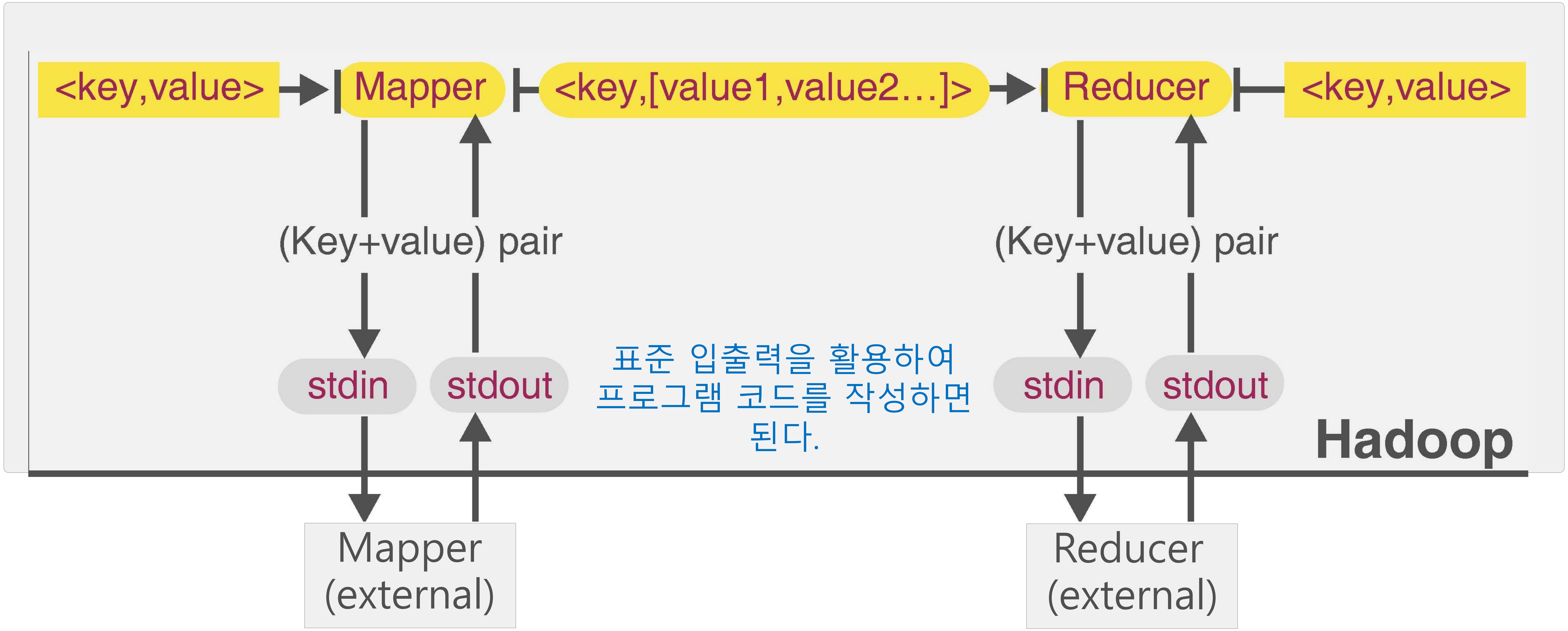
MapReduce in Python

2가지 방법이 있다.

Jython

Scipy와 Numpy같은 C로 만든 라이브러리와 호환이 불가하다.

하둡에서 다양한 언어를 지원



표준 입출력을 활용하여
프로그램 코드를 작성하면
된다.

코드 내부에서
처리할 일

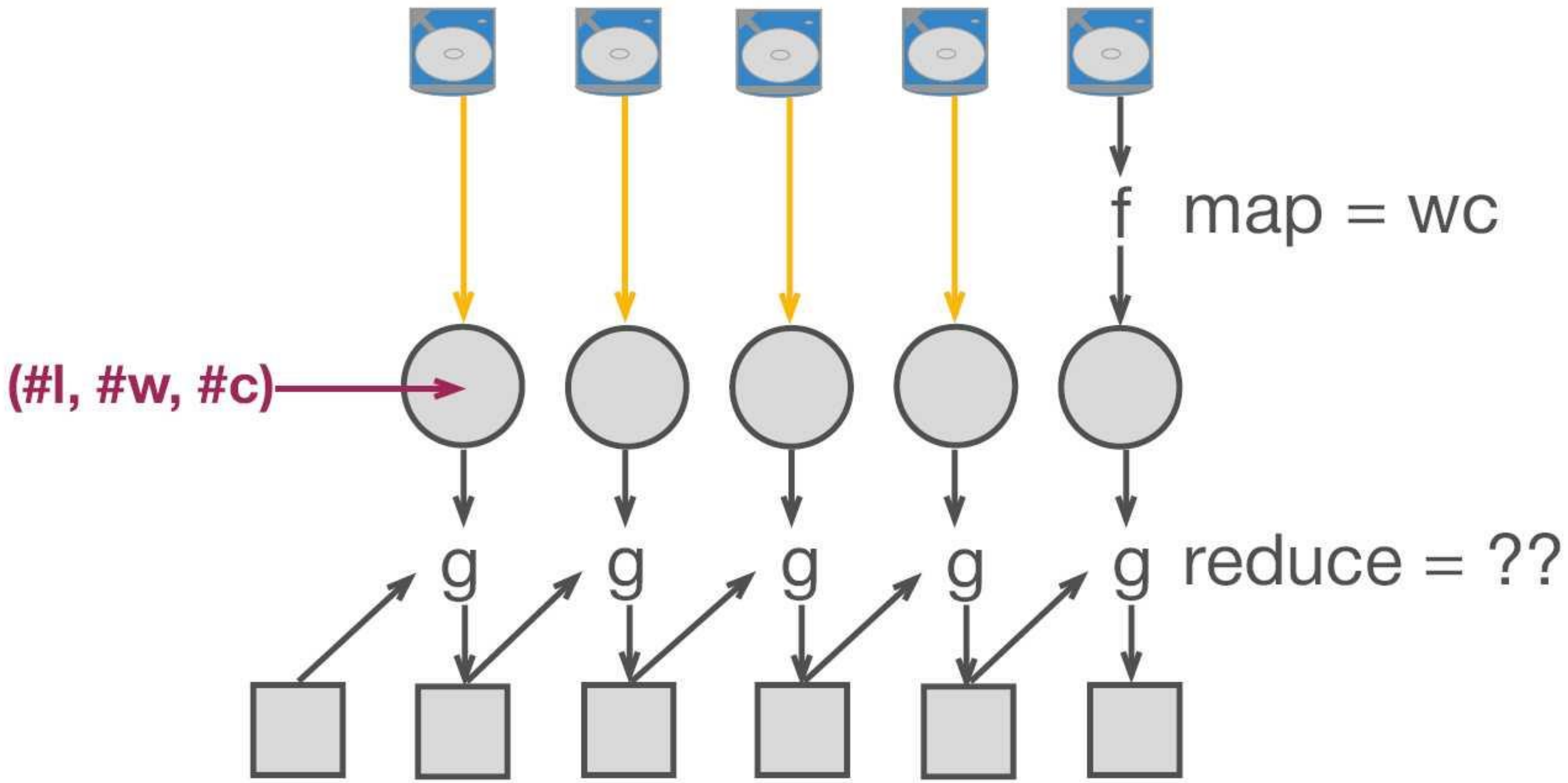
define input format
process data
define output format

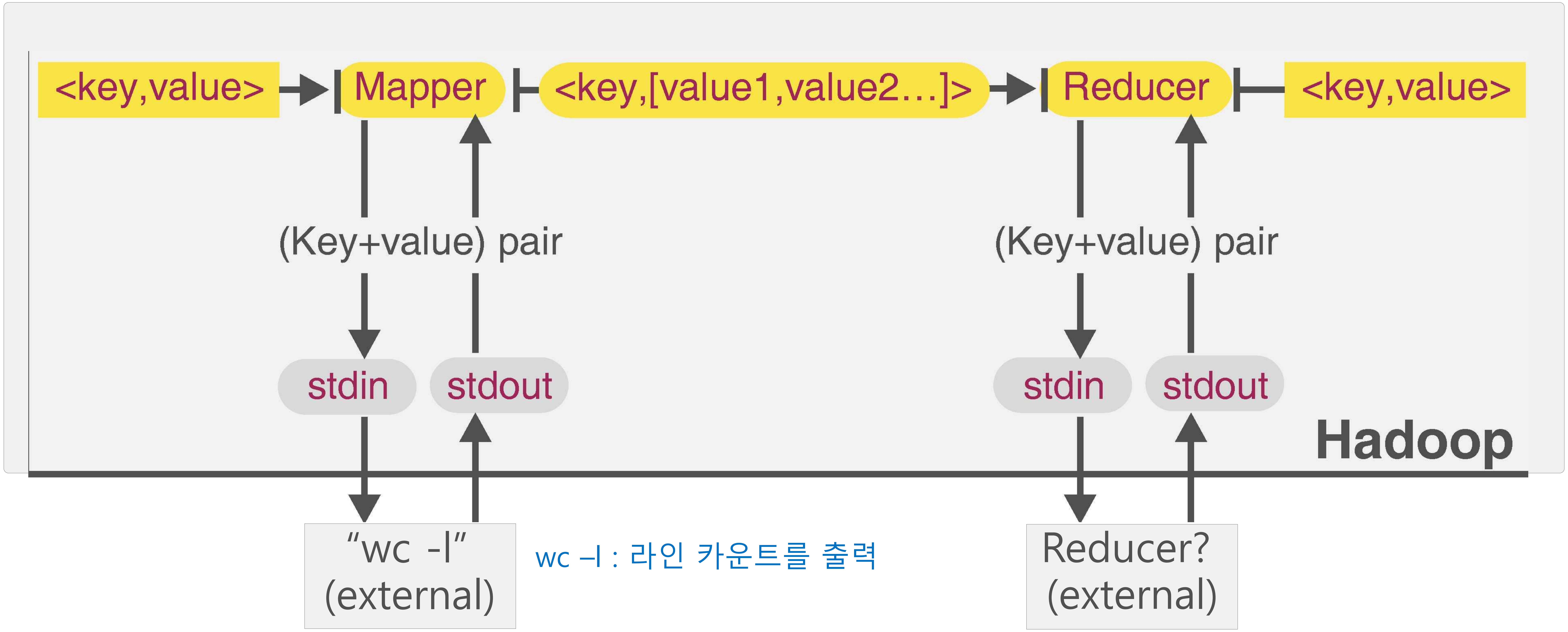
define input format
aggregate sorted data by key
process data
define output format

Distributed Shell: **wc**

파일의 총 라인 수를 파악하는 문제 with Shell Programming

wc는 UNIX의 wordcount 명령어





wc –l : 라인 카운트를 출력

이 문제는 각 mapper에서 line수를 구한 결과를 단순히 더하면 되기 때문에 reducer가 필요없다.

yarn 실행을 위해서 Hadoop-straming.jar 파일의 위치를 알아야 한다.
(locate 활용가능)

/opt/cloudera/parcels/CDH-5.9.0-1.cdh5.9.0.p0.23/lib/hadoop-mapreduce/hadoop-streaming.jar

```
HADOOP_STREAMING_JAR="/path/to/hadoop-streaming.jar"
yarn jar $HADOOP_STREAMING_JAR \
                -mapper 'wc -l' \
                -numReduceTasks 0 \
                -input /data/wiki/en_articles \
                -output wc_mr  (폴더명)
```

reducer를 실행하지 않기 위해 0으로 지정

```
HADOOP_STREAMING_JAR="/path/to/hadoop-streaming.jar"
yarn jar $HADOOP_STREAMING_JAR \
                   -mapper 'wc -l' \
                   -numReduceTasks 0 \
                   -input /data/wiki/en_articles \
                   -output wc_mr
```

```
ERROR streaming.StreamJob: Error Launching job : Output directory
hdfs://virtual-master.atp-fivt.org:8020/user/adral/wc_mr already exists
Streaming Command Failed!
```

존재하는 폴더에는 작업을 수행할 수 없다.

```
$ hdfs dfs -rm -r wc_mr
```

```
HADOOP_STREAMING_JAR="/path/to/hadoop-streaming.jar"
yarn jar $HADOOP_STREAMING_JAR \
                   -mapper 'wc -l' \
                   -numReduceTasks 0 \
                   -input /data/wiki/en_articles \
                   -output wc_mr
```

```
$ hdfs dfs -ls wc_mr
Found 3 items
-rw-r--r--   3 adral adral          0 2017-03-21 14:48 wc_mr/_SUCCESS
-rw-r--r--   3 adral adral          6 2017-03-21 14:48 wc_mr/part-00000
-rw-r--r--   3 adral adral          6 2017-03-21 14:48 wc_mr/part-00001
```

자동으로 mapper가 2개 실행되었다.

```
$ hdfs dfs -text wc_mr/*
1986
2114
```

1968 + 2114 = 4100

```
HADOOP_STREAMING_JAR="/path/to/hadoop-streaming.jar"
yarn jar $HADOOP_STREAMING_JAR \
                    -mapper 'wc -l' \
                    -reducer "awk '{line_count += \$1} END { print line_count
                    -numReduceTasks 1 \
                    -input /data/wiki/en_articles \
                    -output wc_mr
```

awk : 쉘 프로그래밍 언어

awk를 활용한 reducer를 추가한 예제

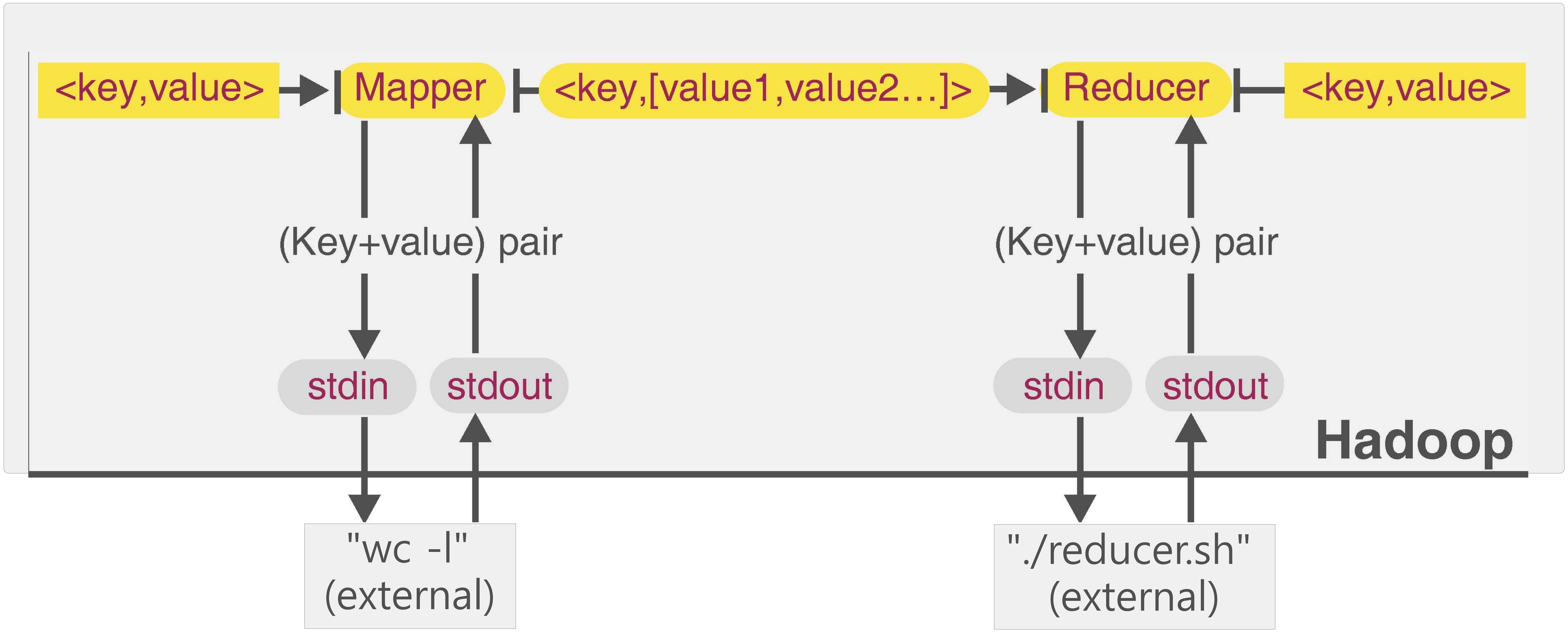
```
$ hdfs dfs -ls wc_mr_with_reducer
Found 2 items
-rw-r--r--   3 adral adral          wc_mr_with_reducer/_SUCCESS
-rw-r--r--   3 adral adral          wc_mr_with_reducer/part-00000

$ hdfs dfs -text wc_mr_with_reducer/*
4100
```

reducer.sh (sh 파일을 직접 만들어 node에 업로드 후 사용하기)

```
#!/usr/bin/env bash
awk '{line_count += $1} END { print line_count }'
```

```
HADOOP_STREAMING_JAR="/path/to/hadoop-streaming.jar"
yarn jar $HADOOP_STREAMING_JAR \
                -mapper 'wc -l' \
                -reducer './reducer.sh' \
                -file reducer.sh \
                -numReduceTasks 1 \
                -input /data/wiki/en_articles \
                -output wc_mr_with_reducer
```



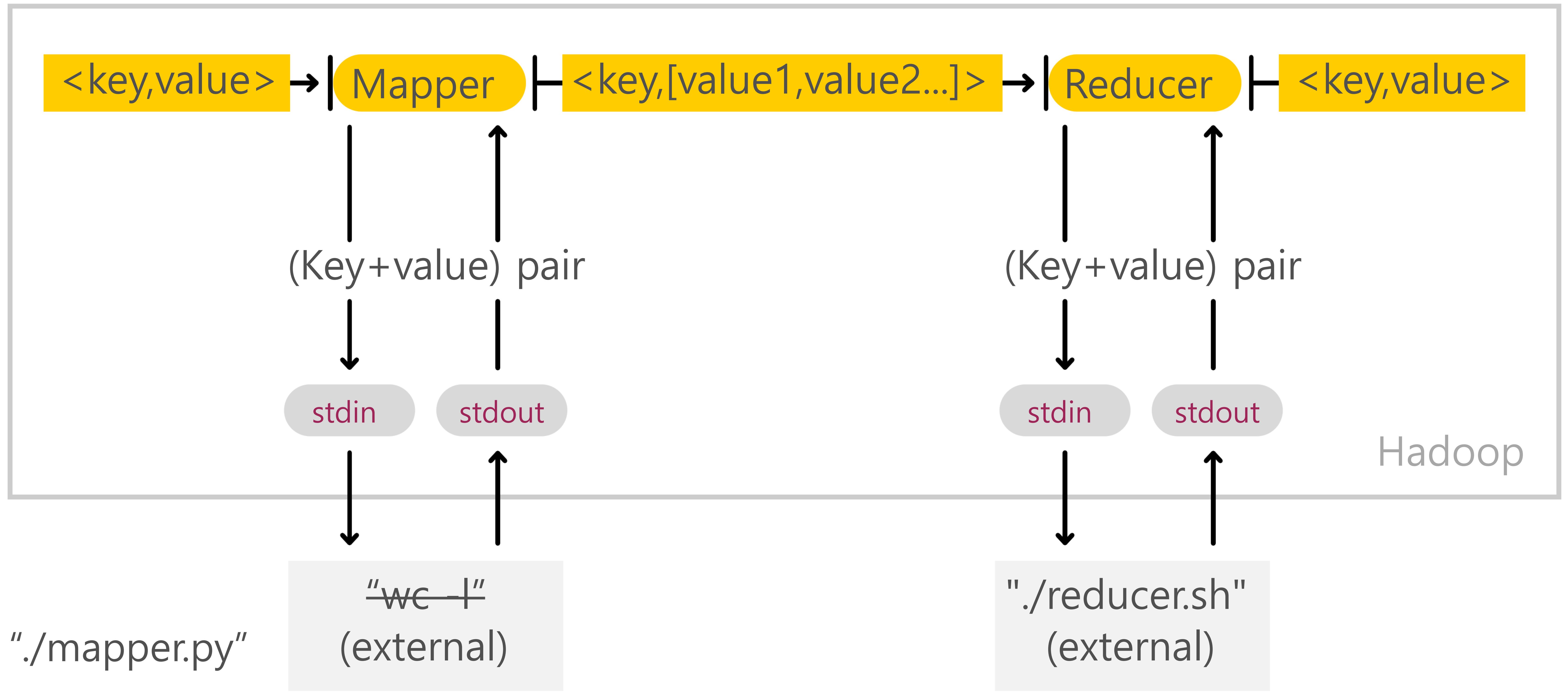
<key,value>
Mapper
<key,[value1,value2…]>
Reducer
<key,value>
(Key+value) pair
(Key+value) pair
stdin
stdout
stdin
stdout
Hadoop
"wc -l" (external)
"./reducer.sh" (external)

MapReduce

Streaming in Python

line 수를 구하는 문제 풀이

py 파일을 활용



"./mapper.py"

stdin

sys.stdin을 통해 데이터를 라인별로 받을 수 있다.

Mapper (Python): mapper.py

```python
from __future__ import print_function
import sys

line_count = 0
for line in sys.stdin:
    pass_count += 1

print(line_count)
```

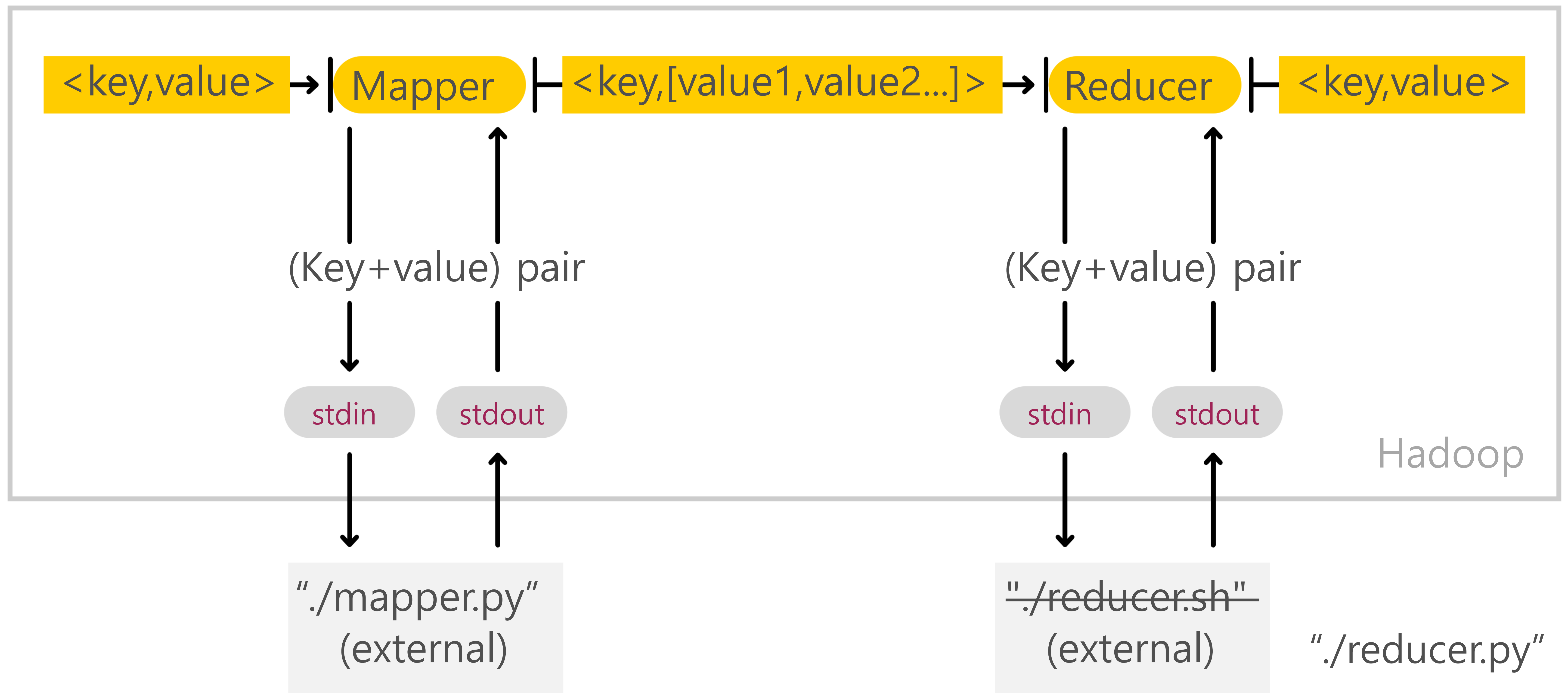
python2 와 python3 호환을 위함

stdout

출력결과는 print를 통해서 내보낸다.

```
HADOOP_STREAMING_JAR="/path/to/hadoop-streaming.jar"
yarn jar $HADOOP_STREAMING_JAR \
                             -files mapper.py, reducer.sh \
                             -mapper 'python mapper.py' \
                             -reducer './reducer.sh' \
                             -numReduceTasks 1 \
                             -input /data/wiki/en_articles \
                             -output wc_mr_with_reducer
```

```
The general command line syntax is
bin/hadoop command [genericOptions] [commandOptions]
-conf <configuration file>
-D <property=value>
-fs <local|namenode:port>
-jt <local|resourcemanager:port>
-files <comma separated list of files>
-libjars <comma separated list of jars>
-archives <comma separated list of archives>
```



<key,value>
Mapper
<key,[value1,value2...]>
Reducer
<key,value>
(Key+value) pair
(Key+value) pair
stdin
stdout
stdin
stdout
Hadoop
"./mapper.py" (external)
~~"./reducer.sh"~~ (external)
"./reducer.py"

reducer.py

stdin

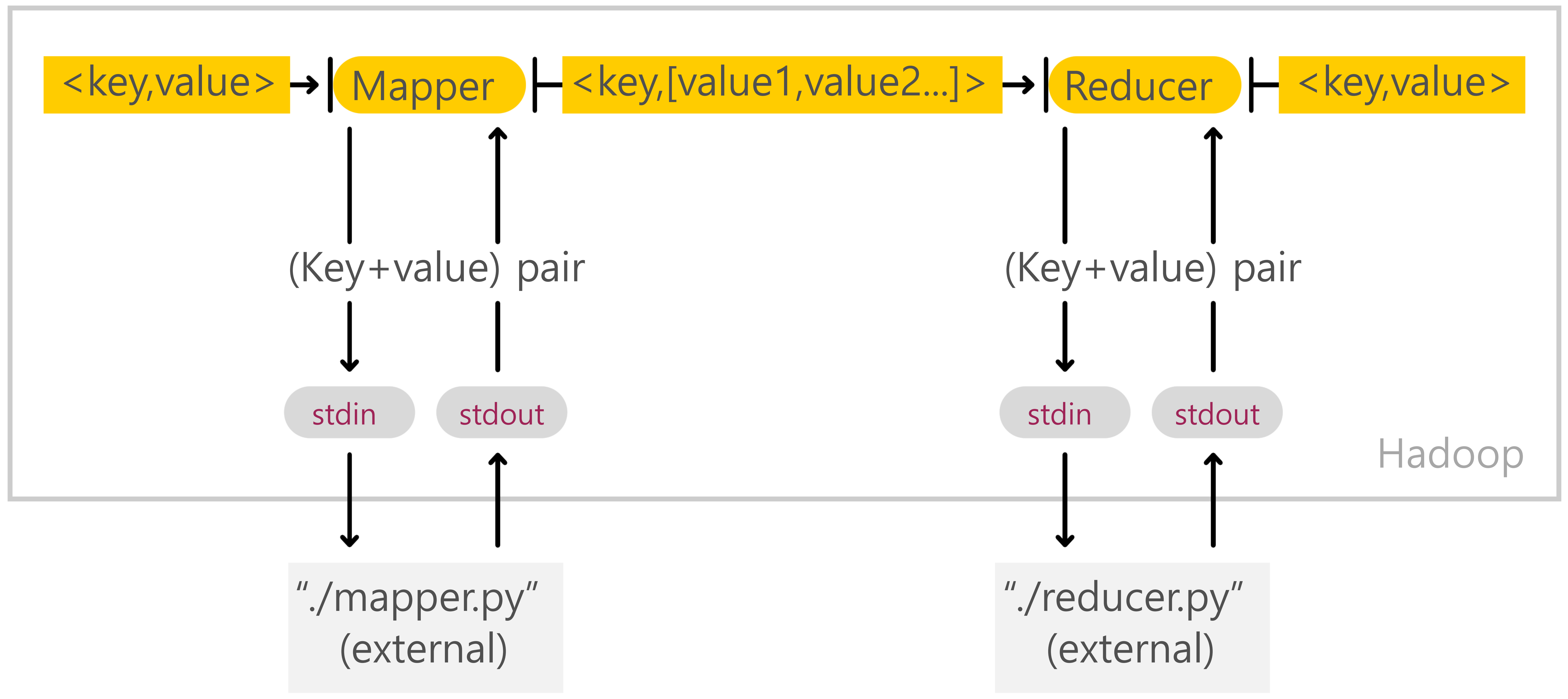
```
from __future__ import print_function
import sys

line_count = sum(
        int(value) for value in sys.stdin
)

print(line_count)
```

stdout

mapper에서 온
line 수 결과를 합한다.



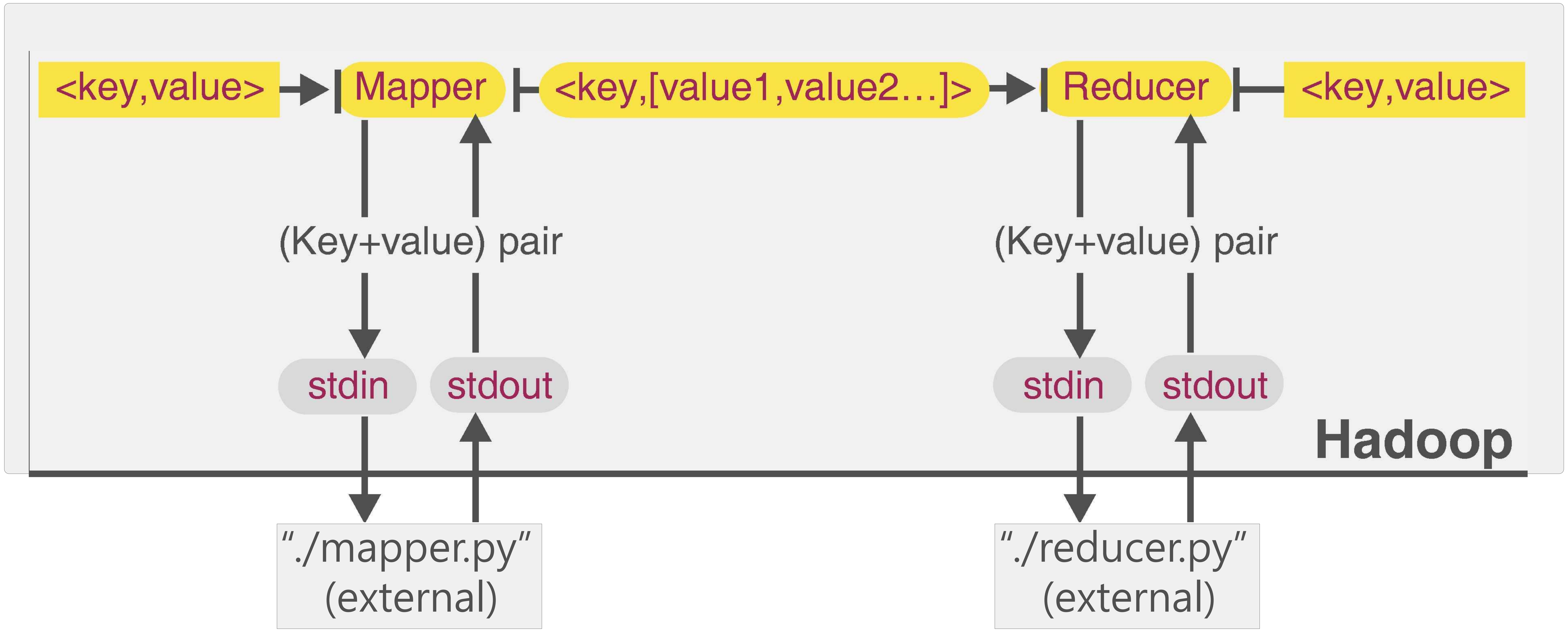
<key,value>
Mapper
<key,[value1,value2...]>
Reducer
<key,value>
(Key+value) pair
(Key+value) pair
stdin
stdout
stdin
stdout
Hadoop
"./mapper.py" (external)
"./reducer.py" (external)

MapReduce

WordCount in Python

Input 파일에서 tab이
key,value를 구분하는 기준

$\underbrace{\texttt{<article id>}}_{\texttt{key}}$ `<tab>` $\underbrace{\texttt{<article content>}}_{\texttt{value}}$



$$\underbrace{\texttt{<article id>}}_{\text{key}}\ \texttt{<tab>}\ \underbrace{\texttt{<article content>}}_{\text{value}}$$

```
from __future__ import print_function
import sys

for line in sys.stdin:
article_id, content = line.split("\t", 1)
    words = content.split()
    for word in words:
        print(word, 1, sep="\t")
```

key, value만 분리하기 위한 1

단어별로 나눠서 1을 출력
e.g.) bag 1
game 1

```
yarn jar $HADOOP_STREAMING_JAR \
                   -files mapper.py \
                   -mapper 'python mapper.py' \
                   -numReduceTasks 0 \
                   -input /data/wiki/en_articles \
                   -output word_count
```

```
$ hdfs dfs -text /data/wiki/en_articles/* | head -c 80
12 <tab> Anarchism        Anarchism is often defined as a political
philosophy which …
```

```
$ hdfs dfs -ls -h word_count
Found 3 items
-rw-r--r-- 3 adral adral 0 2017-03-22 11:40 word_count/_SUCCESS
-rw-r--r-- 3 adral adral 47.8 M 2017-03-22 11:40 word_count/part-00000
-rw-r--r-- 3 adral adral 47.9 M 2017-03-22 11:40 word_count/part-00001
```

```
yarn jar $HADOOP_STREAMING_JAR \
                  -files mapper.py \
                  -mapper 'python mapper.py' \
                  -numReduceTasks 0 \
                  -input /data/wiki/en_articles \
                  -output word_count
```

```
$ hdfs dfs -text /data/wiki/en_articles/*
| head -c 80
~~12~~ <tab> Anarchism        Anarchism is
often defined as a political philosophy
which …
```

```
$ hdfs dfs -text
    word_count/part-… | head -5
```

part-00000	part-00001
Basel 1	Anarchism 1
Basel 1	Anarchism 1
(1	is 1
) 1	often 1
or 1	defined 1
…	…

```
yarn jar $HADOOP_STREAMING_JAR \
                  -files mapper.py \
                  -mapper 'python mapper.py' \
                  -numReduceTasks 1 \
                  -input /data/wiki/en_articles \
                  -output word_count
```

```
$ hdfs dfs -text word_count/part-00000 | head
!    1
!    1
!    1
!    1
!    1
…
```

mapper만 실행해도 numReduceTasks를 1이상으로 선언하면 Shuffle&sort가 수행된다.

key별로 정렬이 된 것을 확인가능

$\underbrace{\text{<article id>}}_{\text{key}}$ <tab> $\underbrace{\text{<article content>}}_{\text{value}}$

mapper.py
수정버전

```
from __future__ import print_function
import re
import sys

for line in sys.stdin:
    article_id, content = line.split("\t", 1)
     words = re.split("\W+", content)
     for word in words:
         if word:
             print(word, 1, sep="\t")
```

특수문자를 지우기 위한 정규식 처리

₩W : word가 아닌 것들(!, 공백 등)

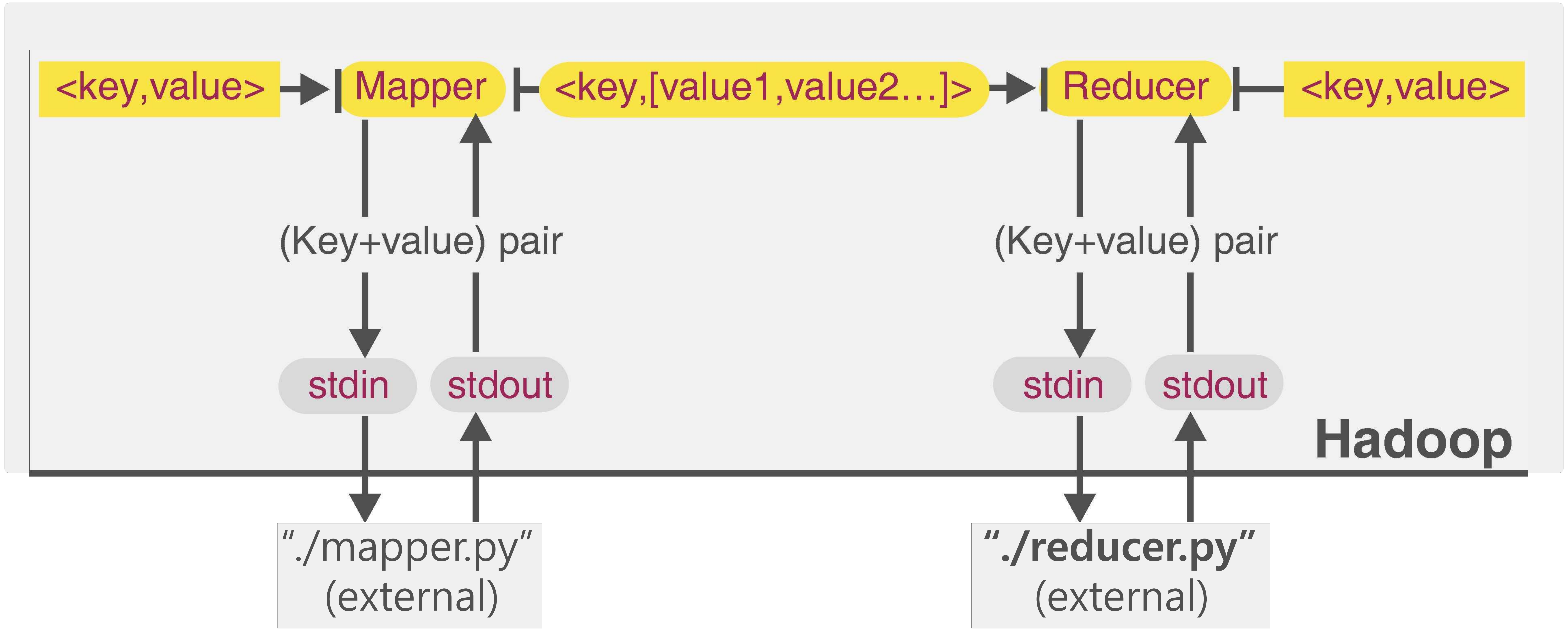
```
yarn jar $HADOOP_STREAMING_JAR \
                      -files mapper.py \
                      -mapper 'python mapper.py' \
                      -numReduceTasks 1 \
                      -input /data/wiki/en_articles \
                      -output word_count
```

특수문자가 제거됨

```
$ hdfs dfs -text word_count/part-00000 | head -4
0 1
0 1
0 1
0 1
```

```
$ hdfs dfs -tail word_count/part-00000 | tail -4
zyu1 1
zyu1 1
zz   1
zz   1
```

WordCount



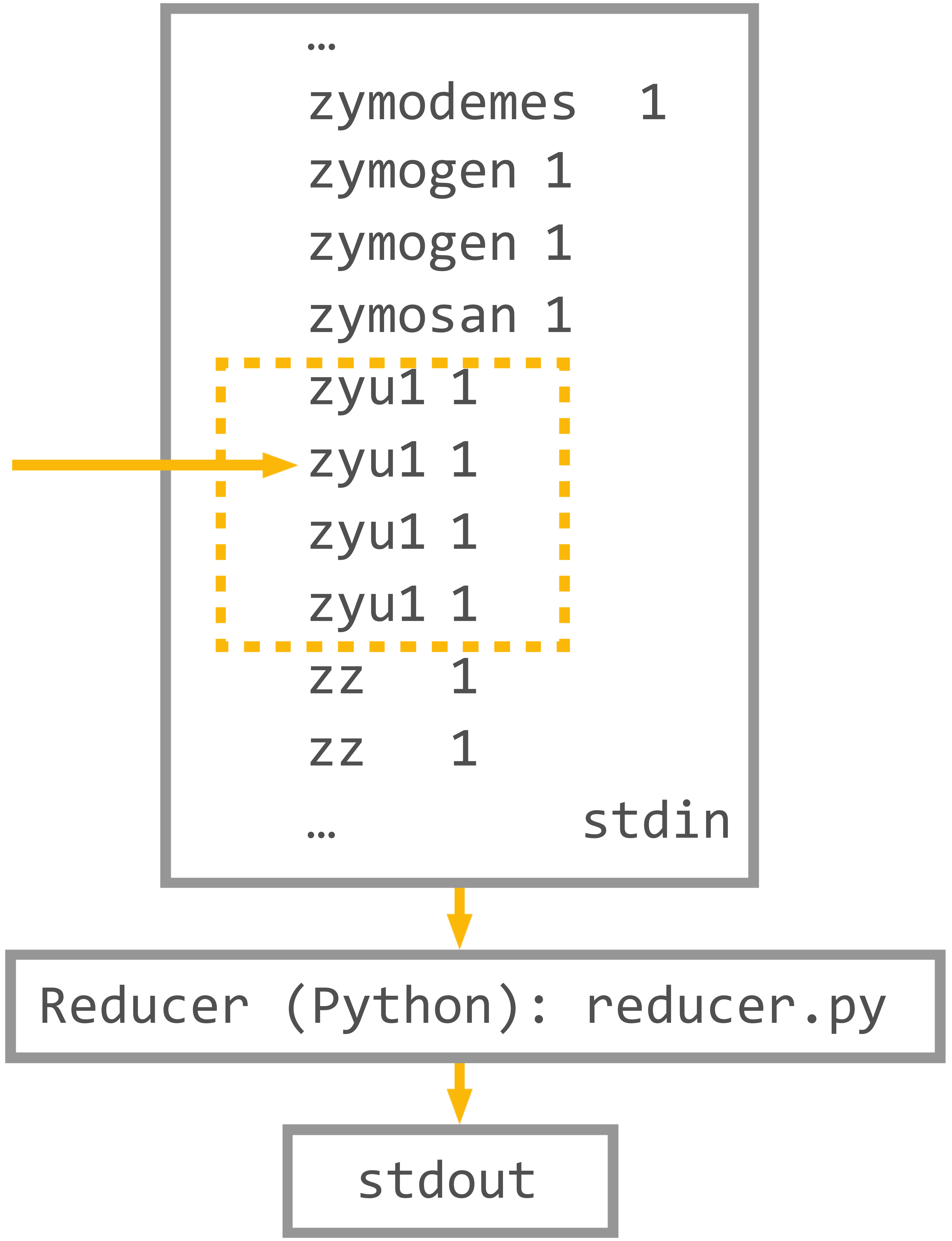
define input format

aggregate sorted data by key

process data

define output format

shuffle & sort된 결과값을
처리해야 함



```
…
zymodemes   1
zymogen 1
zymogen 1
zymosan 1
zyu1 1
zyu1 1
zyu1 1
zyu1 1
zz   1
zz   1
…           stdin
```

reducer.py

```
from __future__ import print_function
import sys

current_word = None
word_count = 0

for line in sys.stdin:
    word, counts = line.split("\t", 1)
    counts = int(counts)
    if word == current_word:
        word_count += counts
    else:
        if current_word:
            print(current_word, word_count, sep="\t")
        current_word = word
        word_count = counts

if current_word:
    print(current_word, word_count, sep="\t")
```

현재 Key를 체크하기 위함

마지막 input의 결과를 처리하기 위함

```
…
zymosan 1
zyu1 1
zyu1 1
zyu1 1
zyu1 1
zz 1
…
```

```
yarn jar $HADOOP_STREAMING_JAR \
                      -files mapper.py,reducer.py \
                      -mapper 'python mapper.py' \
                      -reducer 'python reducer.py' \
                      -numReduceTasks 1 \
                      -input /data/wiki/en_articles \
                      -output word_count
```

```
$ hdfs dfs -ls -h word_count
Found 2 items
-rw-r--r--   3 adral adral          0 2017-03-22 13:05 word_count/_SUCCESS
-rw-r--r--   3 adral adral      3.2 M 2017-03-22 13:05 word_count/part-00000
```

```
$ hdfs dfs -ls -h word_count
Found 2 items
-rw-r--r--   3 adral adral          0 2017-03-22 13:05 word_count/_SUCCESS
-rw-r--r--   3 adral adral      3.2 M 2017-03-22 13:05 word_count/part-00000
```

```
$ hdfs dfs -text word_count/part-00000
0  14905
00 844
000   8186
…
zymodemes    1
zymogen  2
zymosan  1
zyu1  4
zz 2
```

```
…
zymosan 1
zyu1 1
zyu1 1
zyu1 1
zyu1 1
zz 1
…
```

```
yarn jar $HADOOP_STREAMING_JAR \
                      -files mapper.py,reducer.py \
                      -mapper 'python mapper.py' \
                      -reducer 'python reducer.py' \
                      -input /data/wiki/en_articles \
                      -output word_count
```

numReduceTasks를 지정하지 않으면
자동으로 여러 개를 생성

```
$ hdfs dfs -ls -h word_count
Found 11 items
-rw-r--r--   3 adral adral          0 2017-03-22 13:19 word_count/_SUCCESS
-rw-r--r--   3 adral adral    331.0 K 2017-03-22 13:18 word_count/part-00000
-rw-r--r--   3 adral adral    332.1 K 2017-03-22 13:18 word_count/part-00001
-rw-r--r--   3 adral adral    331.7 K 2017-03-22 13:18 word_count/part-00002
-rw-r--r--   3 adral adral    329.8 K 2017-03-22 13:18 word_count/part-00003
-rw-r--r--   3 adral adral    326.1 K 2017-03-22 13:18 word_count/part-00004
-rw-r--r--   3 adral adral    332.2 K 2017-03-22 13:18 word_count/part-00005
-rw-r--r--   3 adral adral    332.3 K 2017-03-22 13:18 word_count/part-00006
```

```
$ hdfs dfs -tail word_count/part-… | tail -5
```

part-00000	part-00005
…	…
	zsu 1
zuang 1	
zucchini 5	
	zuchetto 1
zuerst 1	
	zure 1
…	…

see: TotalOrderPartitioner

각각의 reducer에서는 결과가 정렬되지만
globally sort는 되지 않는다.

TotalOrderPartitioner라는 옵션이 따로 존재

Summary

You know **what** MapReduce Streaming **is** and how it works

You know **how to write** MapReduce Bash and Python Streaming applications

You should be able **to solve** WordCount or similar problems in MapReduce in Python **by yourself**

BigDATAteam