Project Progress

Team Grey

Review of Weekly Progress

Weekly Progress

Progress Category: **Communication** / **Setting** / **Design** / **Implementation**

Week1	**Communication** - Github repository 생성 - Documentation 공유 목적의 Notion 생성
Week2	**Design** - Project에서 요구하는 기능의 전반적 흐름 이해 - 각 단계별로 Master와 Worker 사이에서 수행되어야 하는 일과 module 간의 관계 이해 - 여러 Shuffling methods에 대한 논의 - 이를 바탕으로 프로젝트 전반의 milestone 구체화

Weekly Progress

Progress Category: **Communication** / **Setting** / **Design** / **Implementation**

Week3	**Communication** - 용어 통일: Master, Worker - github commit convention 결정 - scala coding style convention 결정 **Setting** - 개발 환경 통일: Scala 2.13.8 - Test 환경으로 Docker 사용 결정 **Design** - Sampling method 검증 및 전체적인 flow에서 sampling 순서 결정 - 전체적인 flow 도식화 **Implementation** - gRPC의 전반적인 작동 방식 및 demo 진행 상황 공유

Weekly Progress

Progress Category: **Communication** / **Setting** / **Design** / **Implementation**

Week4	**Setting** - 가상환경 세팅: Docker **Design** - Flow chart 정리 **Implementation** - gensort를 통한 dataset generation - gRPC를 통한 Master, Worker 간 connection 구현 - Java library를 이용하여 external merge sort 구현 - partitioning 구현

Logistics

Communication

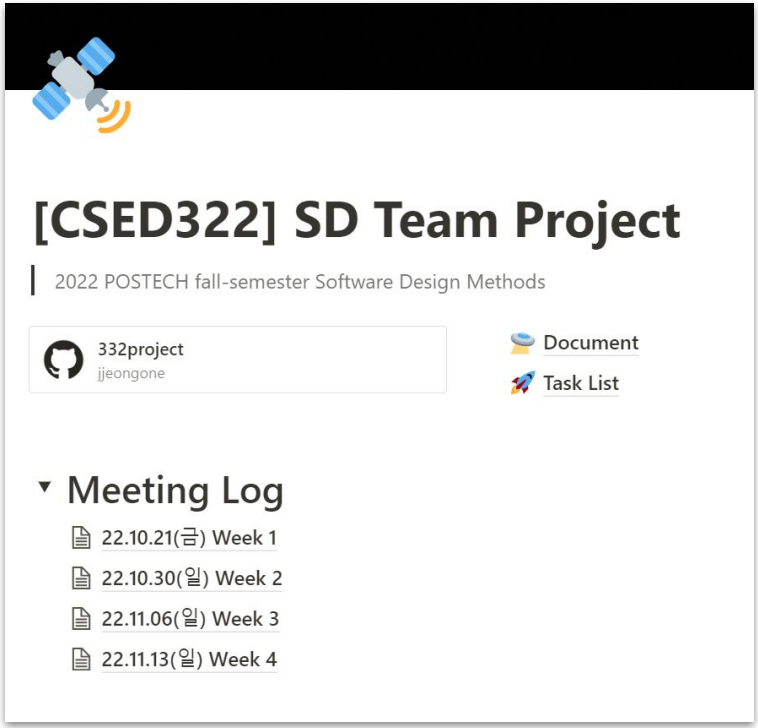
- 매주 일요일 오후, GSR에서 2시간 이상 대면 미팅 진행
- 미팅에서 각자의 진행 상황 공유
- 공유된 진행 상황을 바탕으로 프로젝트 관련 중요 사안 결정
  - 팀원 모두의 동의를 기반으로 함
- 다음 미팅까지 진행할 각자의 Milestone 설정
- 한 명이 미팅 내용 기반으로 github README.md에 Weekly progress 업로드

순번	분관	유형	공간	동반이용자수	사용시간	상태
1	Library	GSR	4F 화상회의실-B	2명	2022/11/13 14:00 ~ 18:00	사용종료
7	Library	GSR	4F 화상회의실-B	2명	2022/11/06 16:00 ~ 19:00	사용종료
9	Library	GSR	4F 화상회의실-B	2명	2022/10/30 14:30 ~ 16:30	사용종료

Documentation

Notion

- 각 주차의 상세 Meeting Log
- Milestone, Scala Coding Convention을 비롯한 Documents



Details

Development Environment

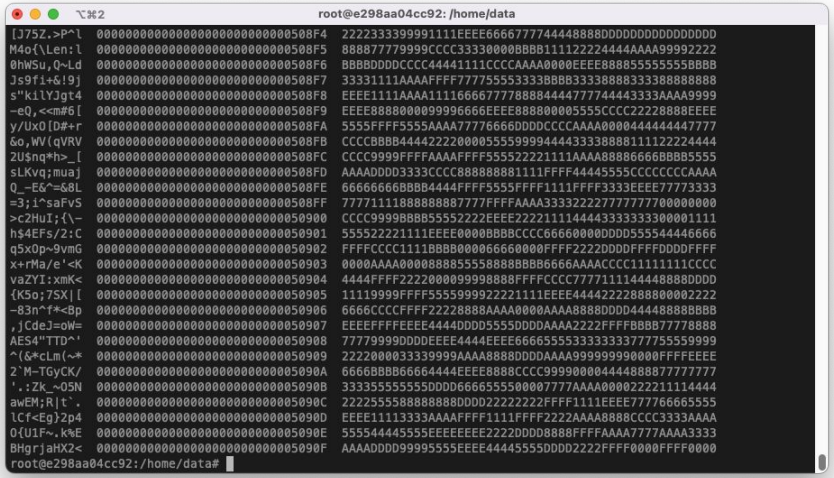
Programming Environment

- OS: Ubuntu 22.04
- Java: 1.8.0(openjdk-8)
- Scala: 2.13.8
- testing environment: docker

- Network: gRPC, ScalaPB
- Logger: java.util.logger

Input Dataset

```
root@8c34382543f9:/home/64# ./gensort -a 330000 partition1
root@8c34382543f9:/home/64# ls -l -h
total 32M
-rwxrwxr-x 1  500  500 138K Mar 17  2013 gensort
-rwxrwxr-x 1 root root  32M Nov 12 17:08 partition1
-rwxrwxr-x 1  500  500 132K Mar 17  2013 valsort
```



Library for External Merge Sort

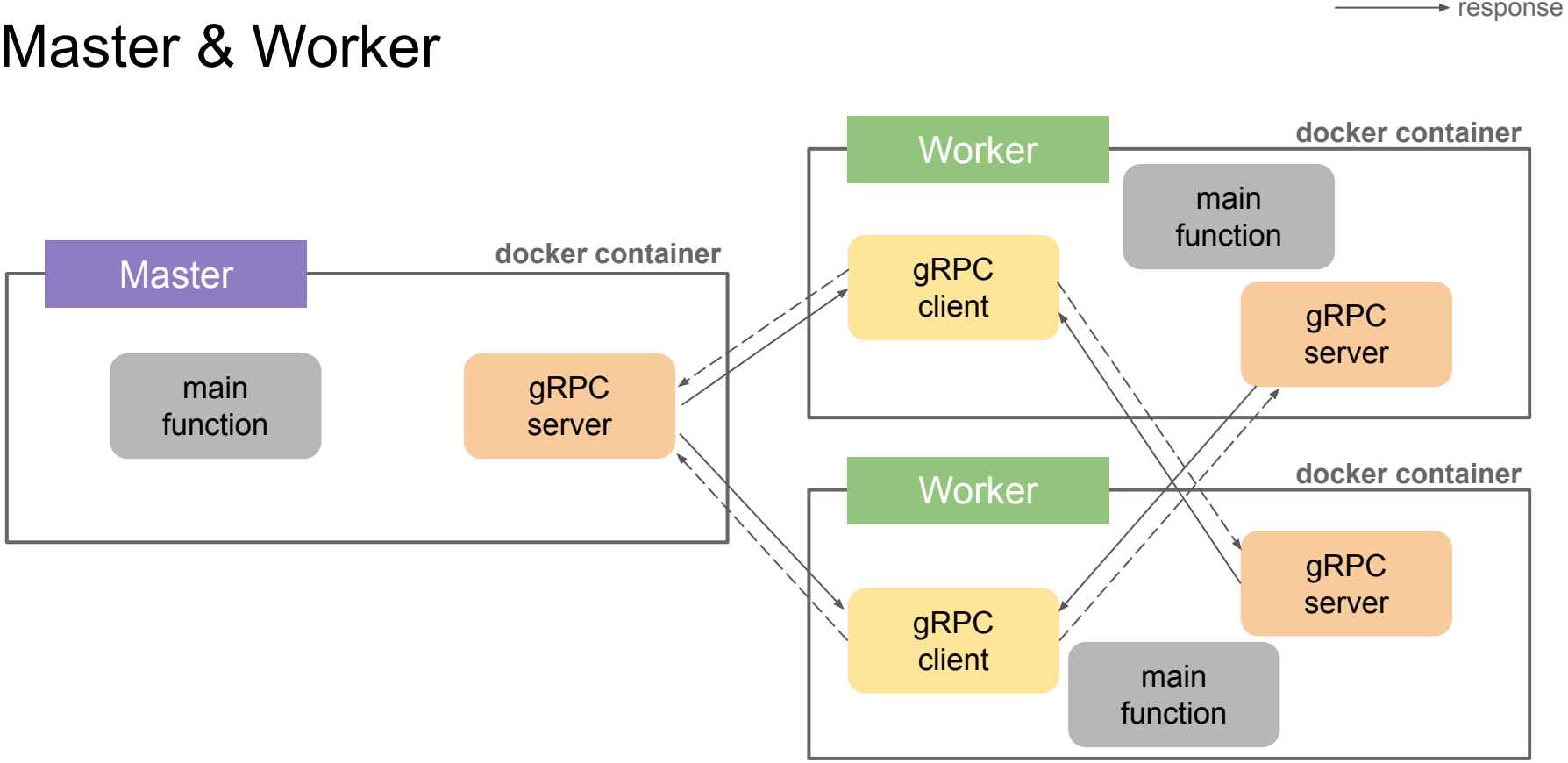
externalsortinginjava 0.6.1 API

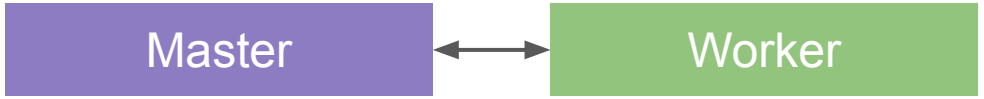
- API Document
- Dependency added on build.sbt
- Useful Methods

Sorting	**ExternalSort.sortInBatch**(BufferReader fbr, long datalength, Comparator<String> cmp, int maxtmpfiles, long maxMemory, Charset cs, File tmpdirectory, boolean distinct, int numHeader, boolean usegzip, boolean parallel)
Sorting & Merging	**ExternalSort.mergeSortedFiles**(List<File> file, File outputfile)

Design

Master & Worker

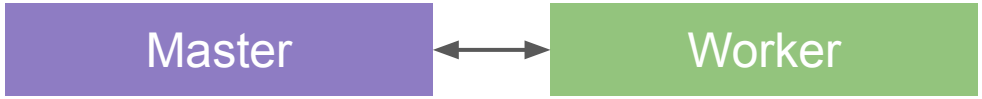




Message

def workerRegistration (address: String): Boolean

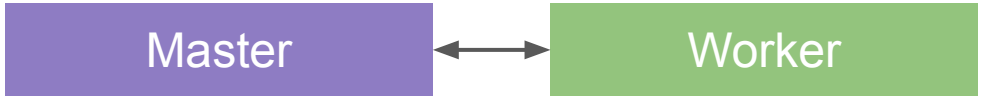
When	worker 자신의 IP address를 파악, 현재 master와 registration이 되어있는지 여부 확인 후
Request	worker IP address
Response	success/fail
Functionality	- 현재 master에 등록된 worker 개수와 프로그램을 실행함에 있어 필요한 총 worker 개수를 비교 - 필요한 worker 개수보다 적을 경우, master의 worker list에 추가하고 True를 반환 - 이미 필요한 worker가 모두 모인 경우 False를 반환 - 이 때 master에 등록하는 순서대로 worker가 ordering 됨



Message

def getWorkerPivots (address: String, sample: File): List[Worker]

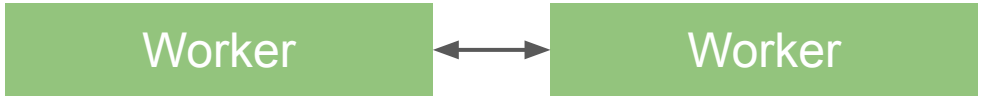
When	master와 connection이 완료되고 worker 내부적인 external merge sort가 완료되었을 때
Request	worker에서 sampling을 수행한 file
Response	각 worker의 IP address, pivot 정보
Functionality	- 각 worker별로 getSample 함수를 통해 sample file을 생성 - 각 worker별로 IP address와 sample file을 받음 - 전체 worker에서 sample을 받았으면 setPivot 함수를 통해 각 worker에 해당하는 pivot을 계산 - 전체 worker에 worker information이 담긴 list를 반환



Message

def getPartitionInfo (address: String, partition: Map[Worker, Int]): Map[Worker, Int]

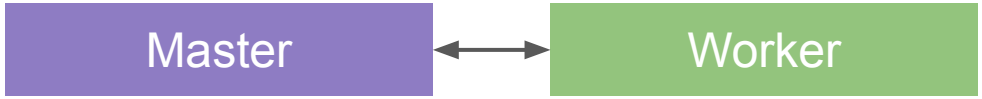
When	partitionByPivot이 완료된 후
Request	해당 worker에서 가지고 있는 pivot(worker)에 따른 partition file 개수에 대한 정보
Response	다른 worker로부터 받아야 하는 file 개수
Functionality	- getWorkerPivot에서 받은 worker 정보를 기반으로 external merge sort가 수행된 file에 대한 partitioning을 수행 - pivot에 따른 partition file의 개수를 저장



Message

def shuffling (address: String, count: Int): List[File]

When	getPartitionInfo message response를 받은 후
Request	받아야 하는 file의 개수를 보냄
Response	request를 보낸 worker의 pivot range에 해당하는 file list를 보냄
Functionality	- request를 보낸 worker에서 요구한 file의 개수와 실제 machine에 존재하는 file의 개수가 맞는지 확인 후 file을 반환



Message

def mergeDone (address: String, range:[Long, Long]): Boolean

When	merge가 다 되었을 때 이를 master에 알림
Request	worker 내부의 file 정보와 함께 master에게 merge 완료를 알림
Response	worker의 request를 잘 받았음을 반환
Functionality	- mergeIntoSortedFile을 통해 worker 내부에서 sorting을 수행 - 해당 worker에 저장된 file의 min key값과 max key 값을 master에 전송

Worker

Functions

def externalMergeSort (files: List[File]): File

Purpose	worker 내부적인 sorting을 수행할 때
Input	sorting을 수행해야 하는 file list
Output	sorting이 완료된 하나의 file
Functionality	- external merge sort를 통해 worker 내부적인 sorting을 수행 - externalsortinginjava library

Worker

Functions

def sampling (file: File): File

Purpose	worker 내부적으로 sampling을 수행
Input	sorting이 완료된 file
Output	sampling된 값들을 저장하는 하나의 file
Functionality	- 주어진 sampling factor를 통해 sorting된 file에 대한 sampling을 수행

Master

Functions

def setPivot (files: List[File]): List[Worker]

Purpose	worker로부터 받은 sample을 이용하여 pivot을 계산할 때
Input	worker로부터 받은 sample file list
Output	계산된 pivot값을 가지는 worker list
Functionality	- worker로부터 받은 sample을 이용하여 각 worker에 해당하는 pivot 값을 계산 - master에 저장되어 있는 worker information에 pivot 정보 저장

Worker

Functions

def partitionByPivot (file: File, workers: List[Worker]):Map[Worker, List[File]]

Purpose	worker 내부에서 pivot에 따른 partitioning 수행
Input	- partitioning을 수행할 파일 - 각 worker에 대한 정보
Output	각 worker에 따른 partitioned file list
Functionality	- 각 worker에 따른 pivot을 이용하여 worker 내부 file에 대한 partitioning을 수행

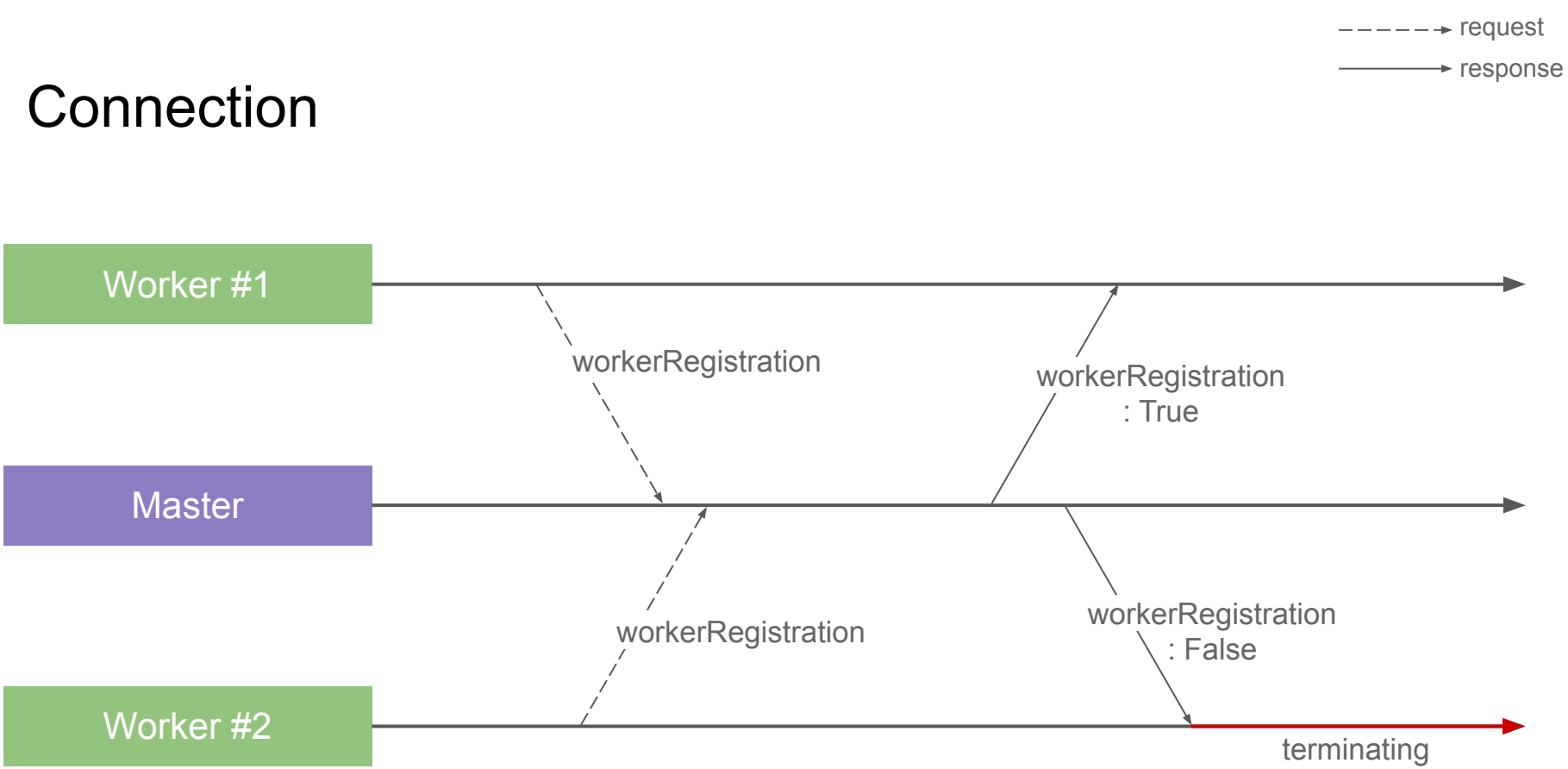
Worker

Functions

def mergeIntoSortedFile(files: List[File]):File

Purpose	shuffling이 완료된 후 worker 내부에서 merging 수행
Input	merge해야 하는 file list
Output	merging이 완료된 하나의 file
Functionality	- worker들 간 shuffling이 모두 완료되었을 때 실행 - externalsortinginjava library: ExternalSort.mergeSortedFiles

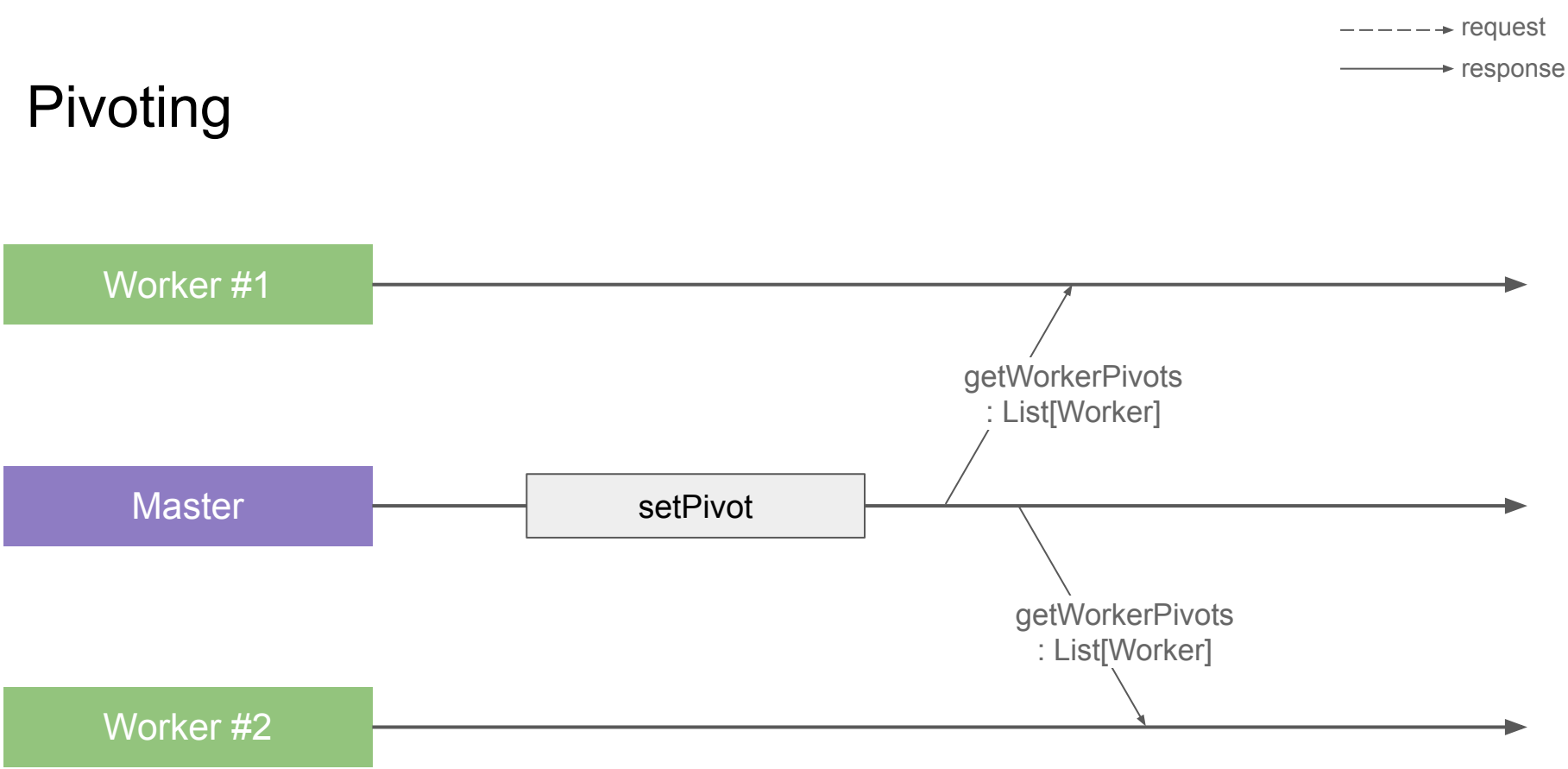
Connection


request
response
Worker #1
Master
Worker #2
workerRegistration
workerRegistration
: True
workerRegistration
workerRegistration
: False
terminating

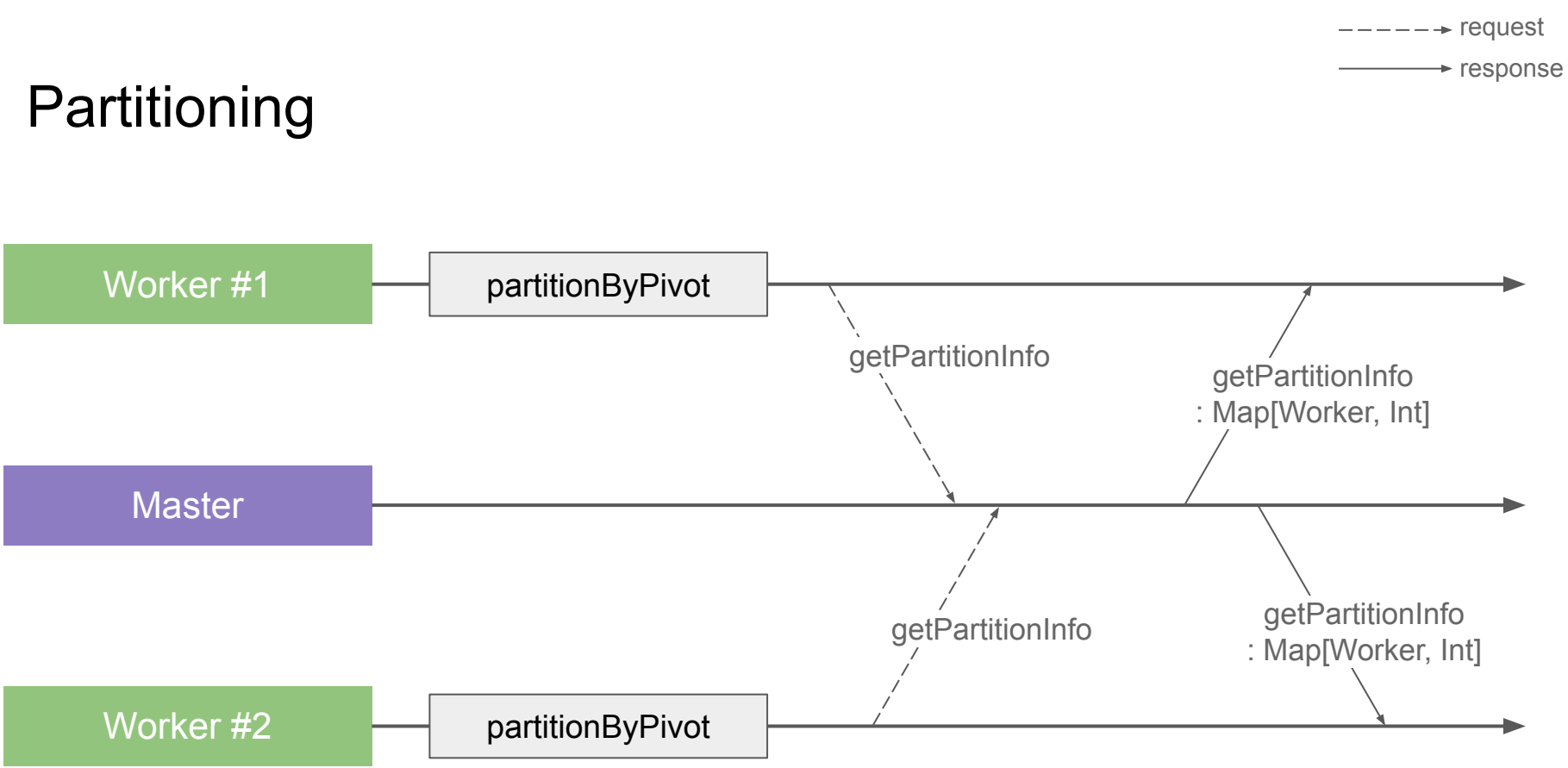
Sampling


request
response
Worker #1
externalMergeSort
sampling
getWorkerPivots
Master
getWorkerPivots
Worker #2
externalMergeSort
sampling

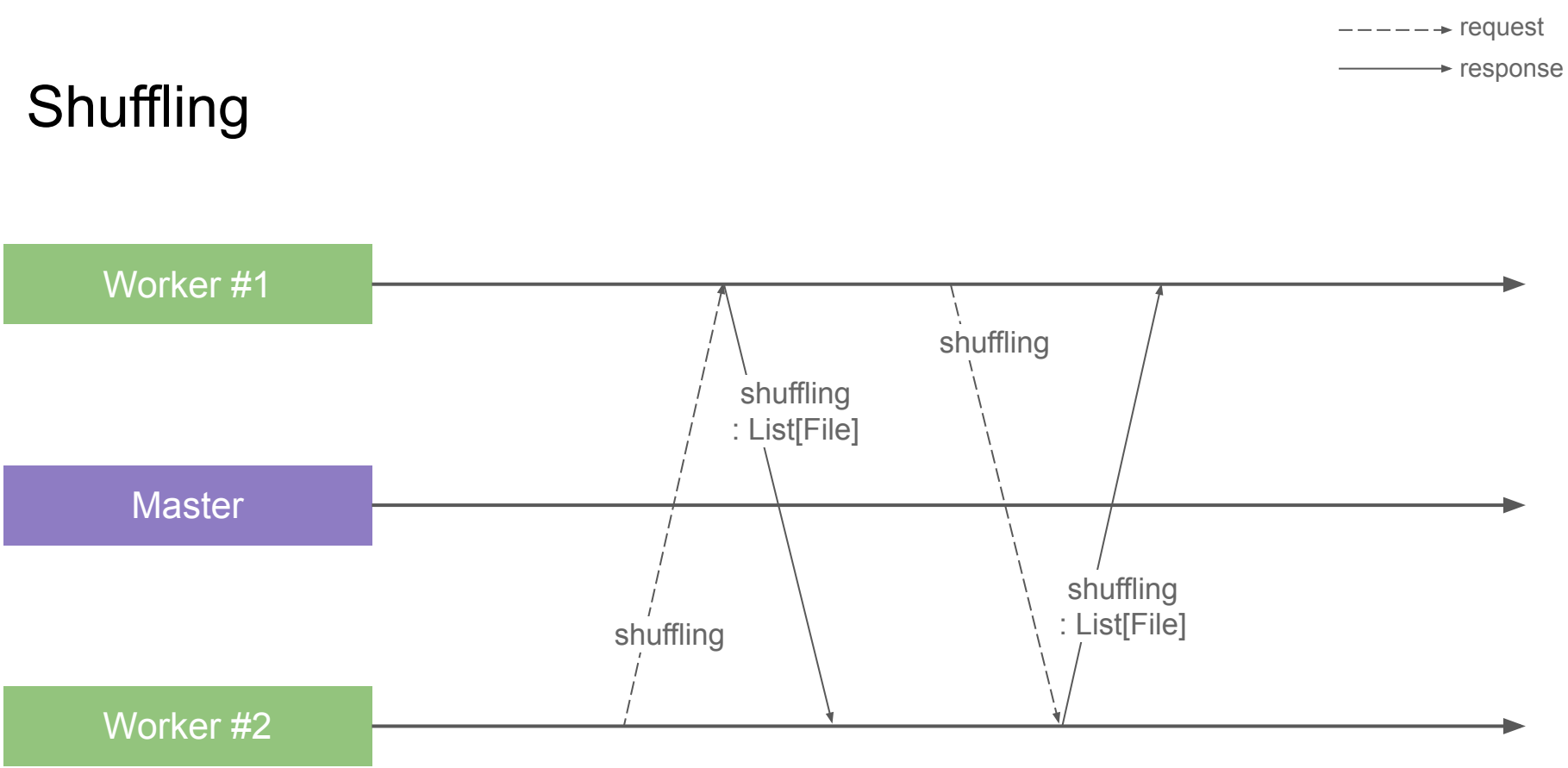
Pivoting


request
response
Worker #1
Master
Worker #2
setPivot
getWorkerPivots
: List[Worker]
getWorkerPivots
: List[Worker]

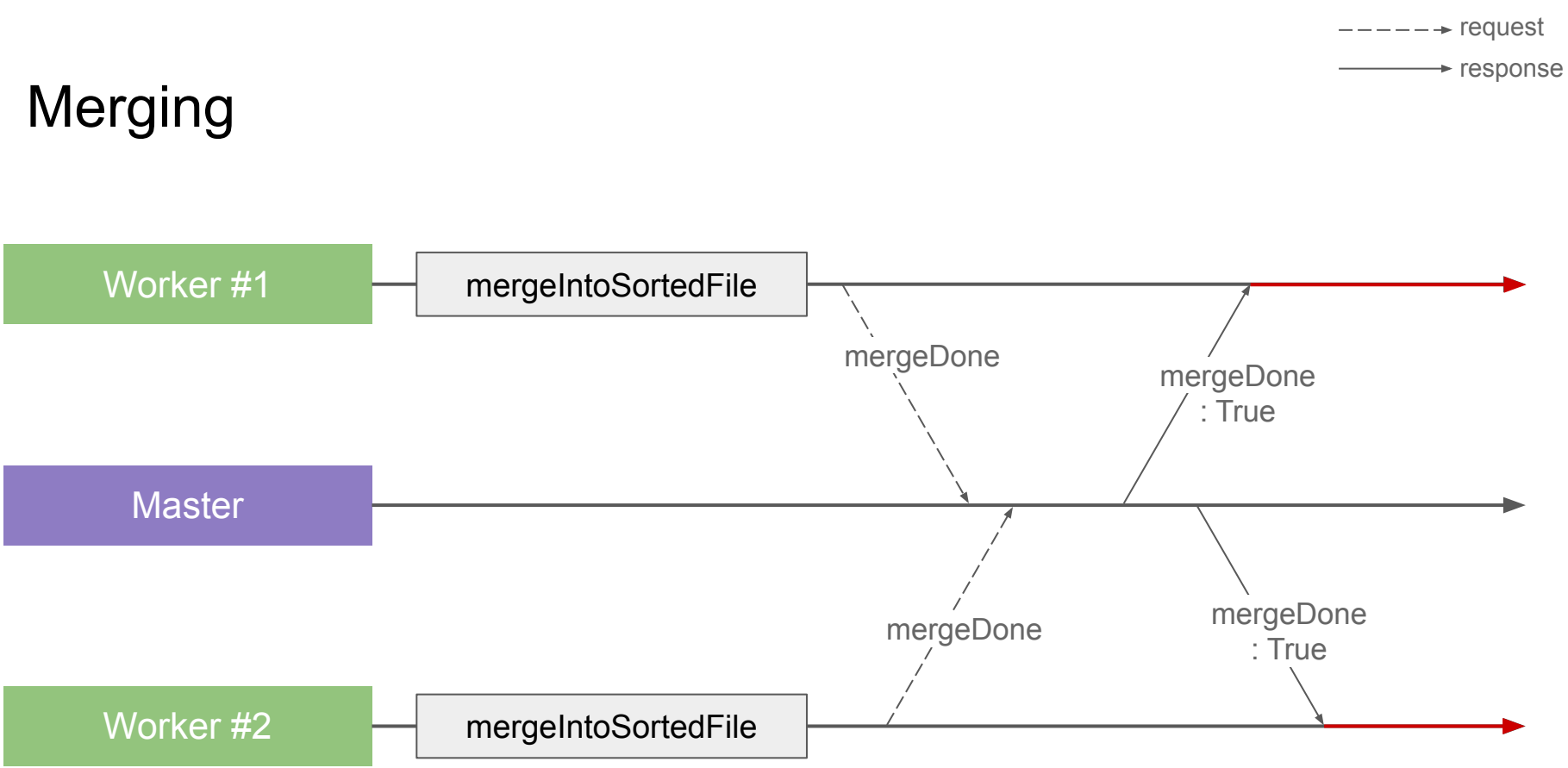
Partitioning



Shuffling


request
response
Worker #1
Master
Worker #2
shuffling
shuffling
: List[File]
shuffling
shuffling
: List[File]

Merging


request
response
Worker #1
mergeIntoSortedFile
mergeDone
mergeDone
: True
Master
mergeDone
mergeDone
: True
Worker #2
mergeIntoSortedFile

Progress

Milestone By Week

Week 1 진행 방향 및 공부할 내용 □

Week 2 Midterm □

Week 3 Body Design(1) + Testing Environment Setting (1) □

Week 4 Body Design(2) + Testing Environment Setting (2) □

Week 5 Coding 시작 + Design 수정 □

Week 6 Coding 마무리 + Unit Test 설계 ×

Week 7 Debugging + Testing □

Week 8 마무리 □

gRPC Network Progress

- gRPC and ScalaPB environment setting □
- Simple Request-Response Demo □
- Send Tuple, Array Message □
- Send File Message □
- Send List of Files □
- Register Multiple Services □

gRPC Network Progress

Master가 Client의 Request를 모두 받고 Response를 처리하도록 하기

workerRegistration □

getWorkerPivot □

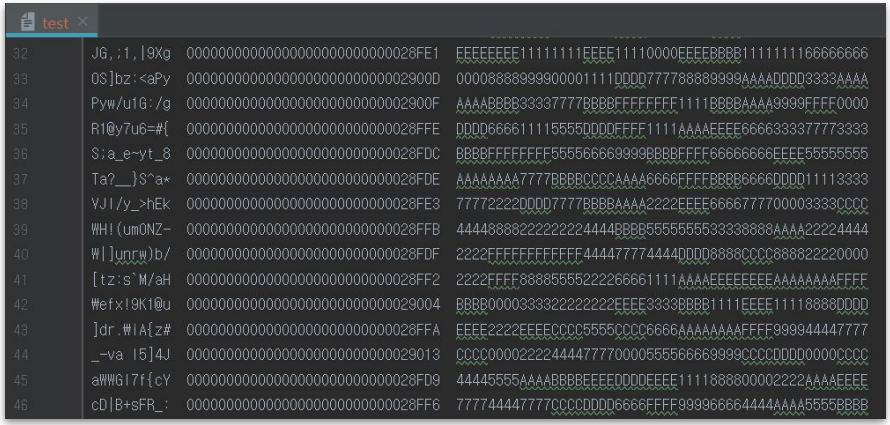
Shuffling □

mergeDone □

Partitioning Progress

partitionByPivot 구현

- gensort로 생성된 일부 데이터로 Local 테스트
- Docker 환경에서 확인 필요



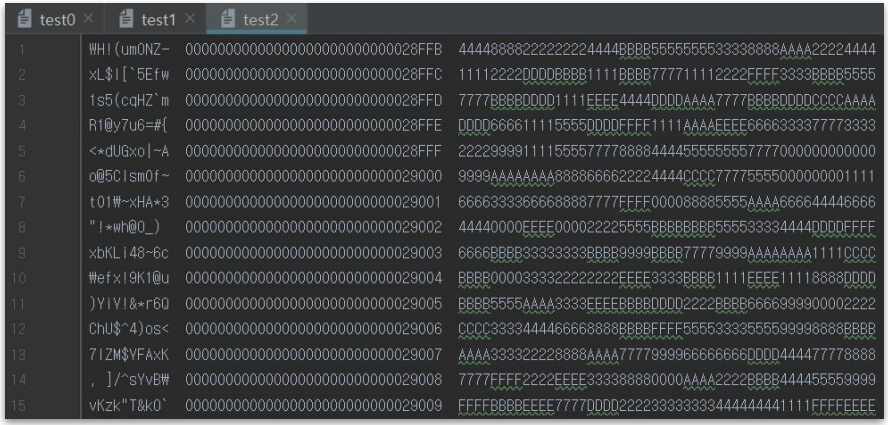
input (a File)

output (partitioned Files)

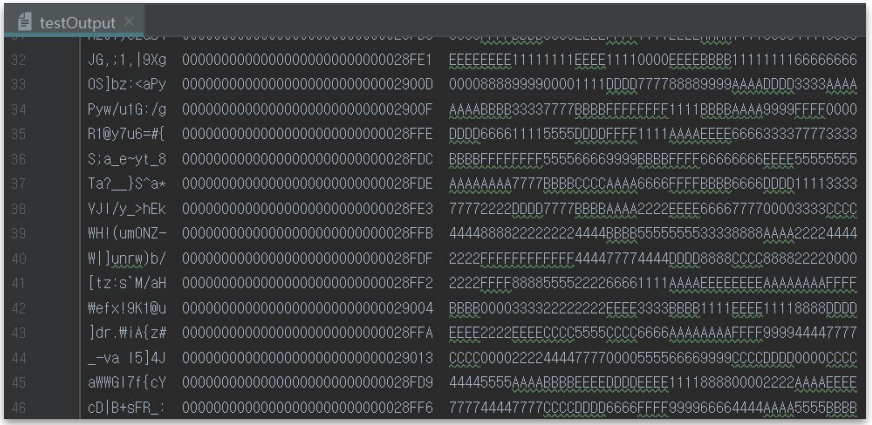
Sorting Progress

externalMergeSort 구현

- gensort로 생성된 일부 데이터로 Local 테스트
- Docker 환경에서 확인 필요



input (unsorted Files)

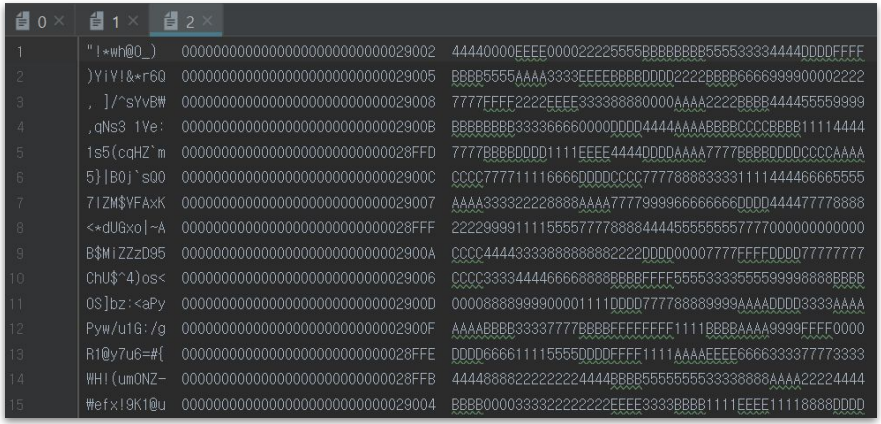


output (a sorted File)

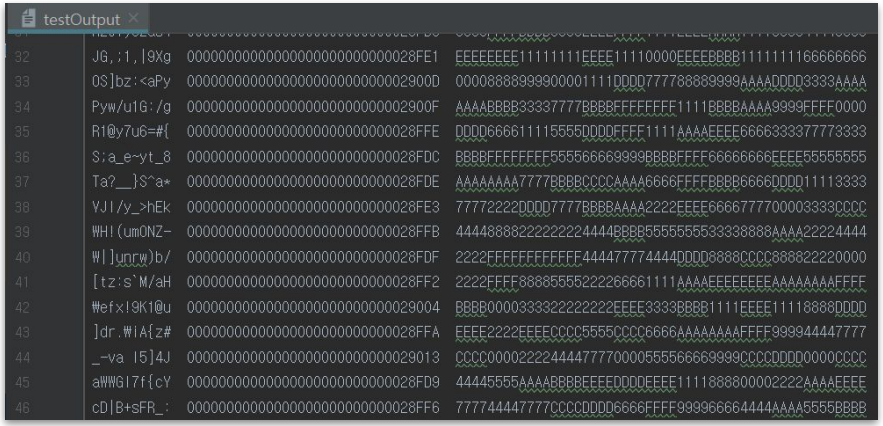
Merging Progress

mergeIntoSortedFile 구현

- gensort로 생성된 일부 데이터로 Local 테스트
- Docker 환경에서 확인 필요



input (sorted Files)



output (a sorted File)