Data Analysis

(Data modelling, collecting, and analyses 4)

Fall, 2020

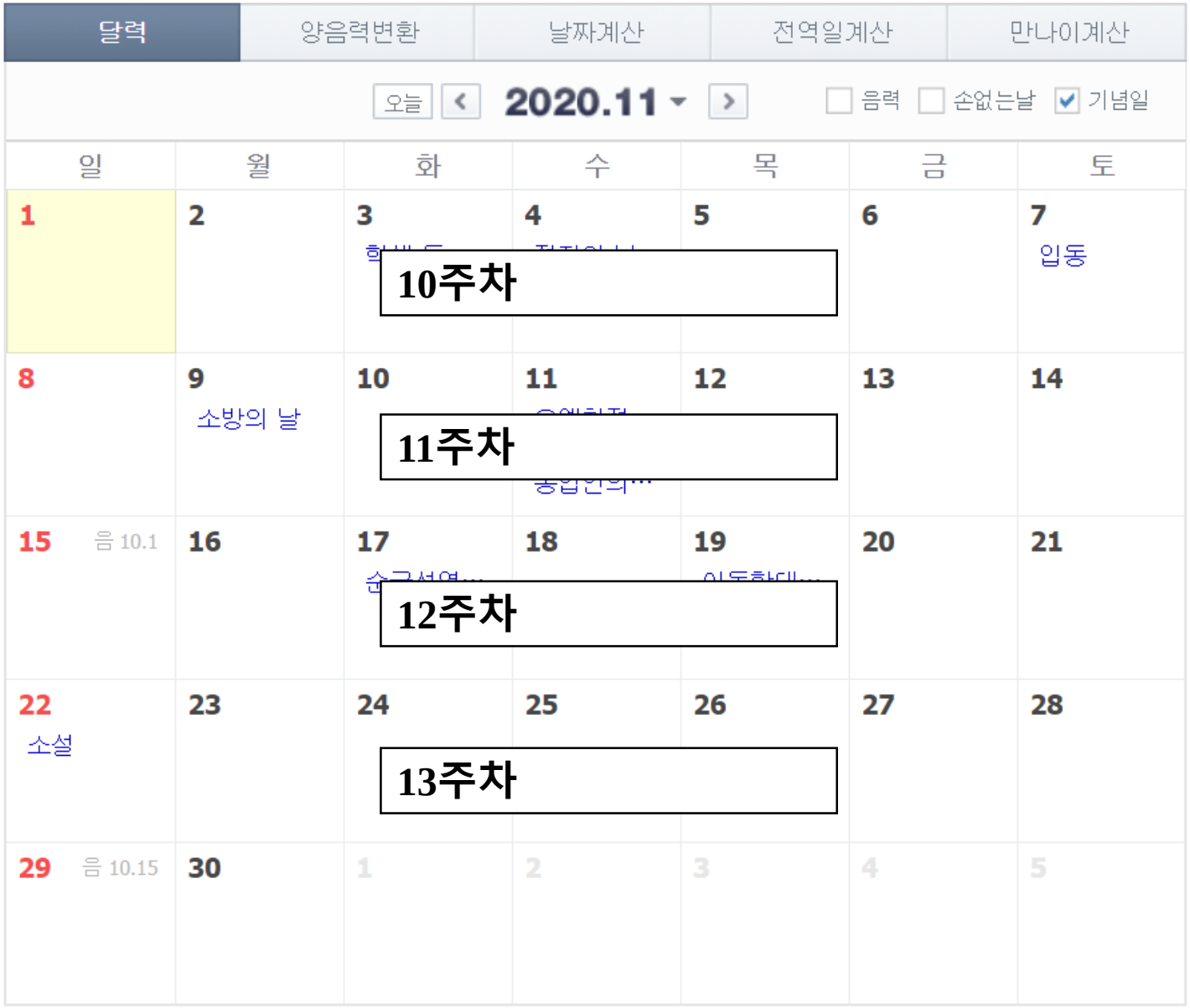
Calendar


달력
양음력변환
날짜계산
전역일계산
만나이계산
오늘
2020.09
음력
손없는날
기념일
일
월
화
수
목
금
토
30
31
1
소개
2
음 7.15
3
환경
세팅
4
지식재산…
5
6
7
백로
8
복습
1
9
10
9. 1
복습
2
11
12
13
14
15
16
17
음 8.1
18
19
청년의 날
3주차
20
21
치매극복…
22
23
24
25
26
4주차
27
28
29
30
1
2
3
5

Calendar


달력
양음력변환
날짜계산
전역일계산
만나이계산
오늘
2020.10
음력
손없는날
기념일
일 월 화 수 목 금 토
27 28 29 30 1 음 8.15 추석 국군의 날 2 노인의 날 3 개천절
4 5 세계 한… 6 7 8 9 한글날 10
11 12 13 14 15 16 부마민주… 17 음 9.1 문화의 날
18 19 20 21 22 23 상강 24 국제연합일
25 독도의날 중양절 26 27 28 29 지방자치… 30 31 음 9.15
6주차
7주차
8주차: 중간고사
9주차

Calendar


달력
양음력변환
날짜계산
전역일계산
만나이계산
오늘
2020.11
음력
손없는날
기념일
일
월
화
수
목
금
토
1
2
3
4
5
6
7
입동
10주차
8
9
소방의 날
10
11
12
13
14
11주차
15
음 10.1
16
17
18
19
20
21
12주차
22
소설
23
24
25
26
27
28
13주차
29
음 10.15
30
1
2
3
4
5

Calendar


달력
양음력변환
날짜계산
전역일계산
만나이계산
오늘
2020.12
음력
손없는날
기념일
일 월 화 수 목 금 토
29 30 1 2 3 4 5
무역의 날
14주차
6 7 8 9 10 11 12
대설
15주차
13 14 15 음 11.1 16 17 18 19
16주차: 기말고사 주간
20 21 22 23 24 25 26
동지
성탄절
27 28 29 음 11.15 30 31 1 2
원자력의…

Table of Contents

Collection Framework: HashSet

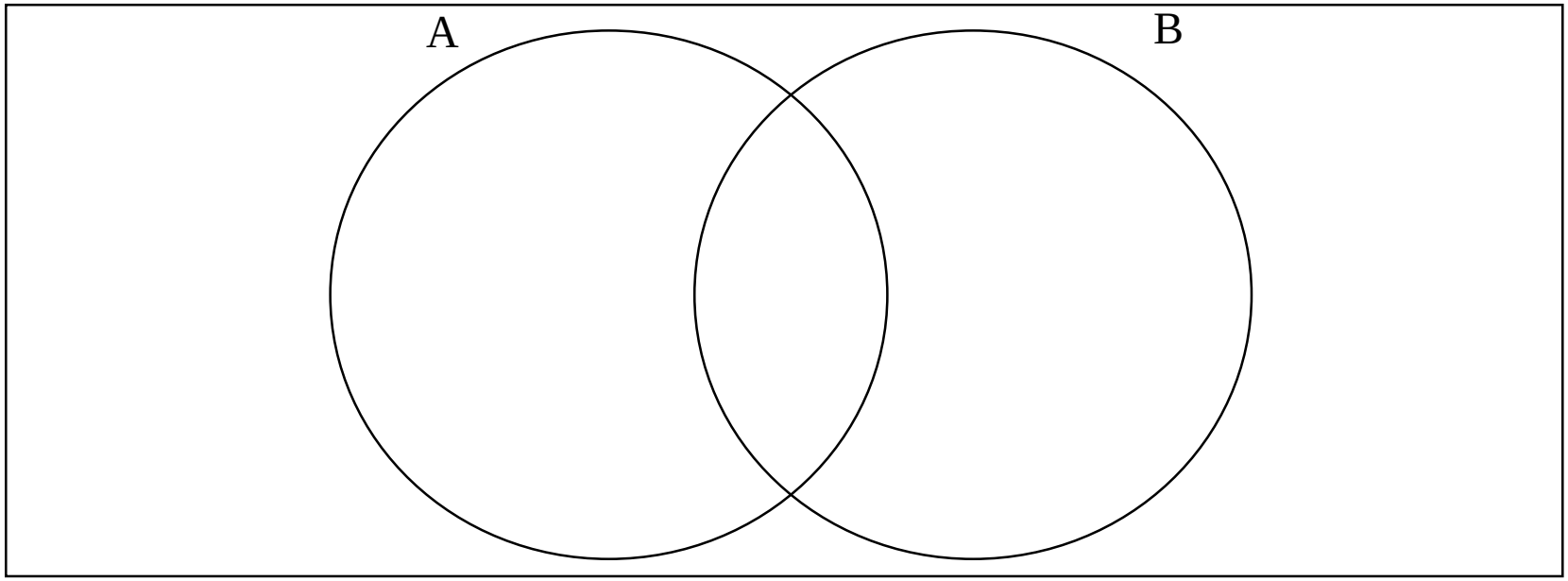
- Test with Email
  - Check whether contains() works well

Question #1

- Compute the number of non-redundant identifiers who only sending email(s)
- Compute the number of non-redundant identifiers who only receiving email(s)
- Compute the number of non-redundant identifiers who both sending and receiving email(s)
- Compute the number of non-redundant identifiers who attends an email network

Collection Framework: HashSet

- HashSet



메서드	설 명
boolean addAll(Collection c)	지정된 Collection(c)의 객체들을 Collection에 추가한다.(합집합)
boolean containsAll(Collection c)	지정된 Collection의 객체들이 Collection에 포함되어 있는지 확인한다.(부분집합)
boolean removeAll(Collection c)	지정된 Collection에 포함된 객체들을 삭제한다.(차집합)
boolean retainAll(Collection c)	지정된 Collection에 포함된 객체만을 남기고 나머지는 Collection에서 삭제한다.(교집합)

In-class assignment 2

- Implement your own MyHashSet implementing Set<E>
  - Implementing all the methods
  - Data Abstraction (example)
    - MyBucket<E>[]
  - The maximum size of hash set never changes for you since initialization
  - Solve Question #1 and #2 with your collection

```
public class MyHashSet<E> implements Set<E>{

private MyBucket<E>[] bucketChain = null;

@SuppressWarnings("unchecked")
public MyHashSet(int capacity) {
bucketChain = new MyBucket[capacity];
}

@SuppressWarnings("rawtypes")
@Override
public int size() {
int cnt = 0;
for(MyBucket b: bucketChain) {
if(b != null)
cnt++;
}
return cnt;
}
```

```
public class MyBucket<E> {
int hashCode;
ArrayList<E> bucketList;
}
```

Question #2

- Compute the identifier where its occurrence is maximum in the dataset.
  - e.g., Dataset
    - 1 2
    - 1 4
    - 1 8
    - 2 3
    - 5 8

    1 is seen 3 times

    2,8 … 2 times

    3,4,5 … 1 time

    So 1 is the answer

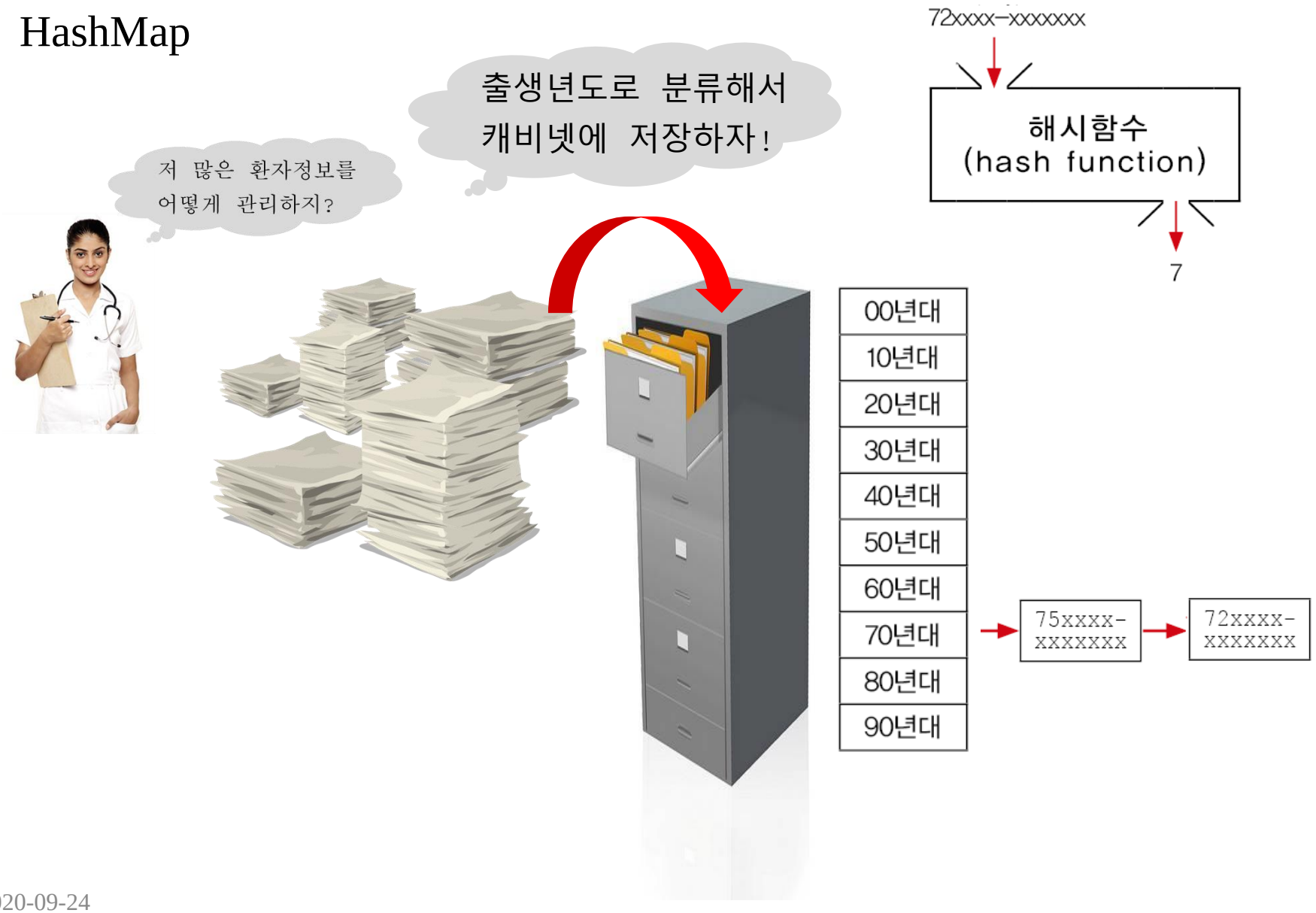
Collection Framework: HashMap

- HashMap
  - Key-value pairs of
  - Unordered, Non-redundant elements E
  - Based on Hashing
  - CRUD by hash
  - HashSet is just a special case of HashMap

```
public class HashSet<E> implements Set<E> {
        …
        private HashMap<E,Object> map;
        …
}
```

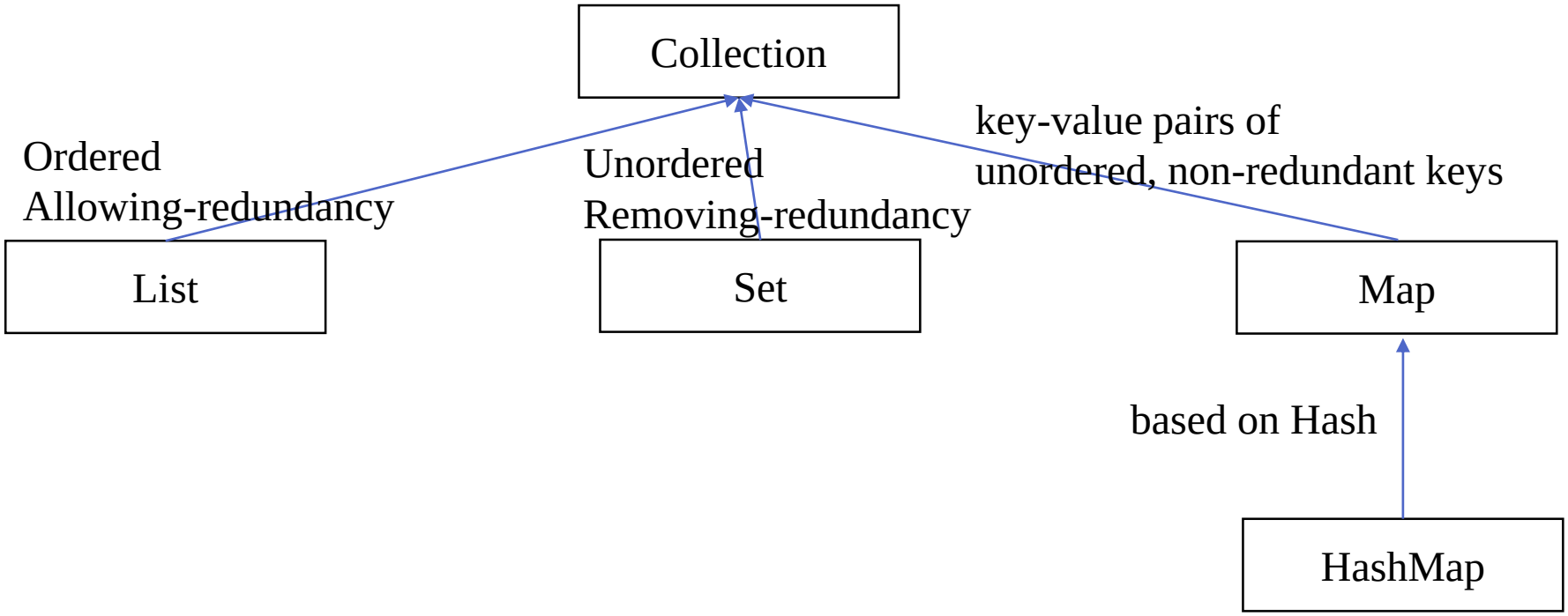
Collection Framework: HashMap

- HashMap



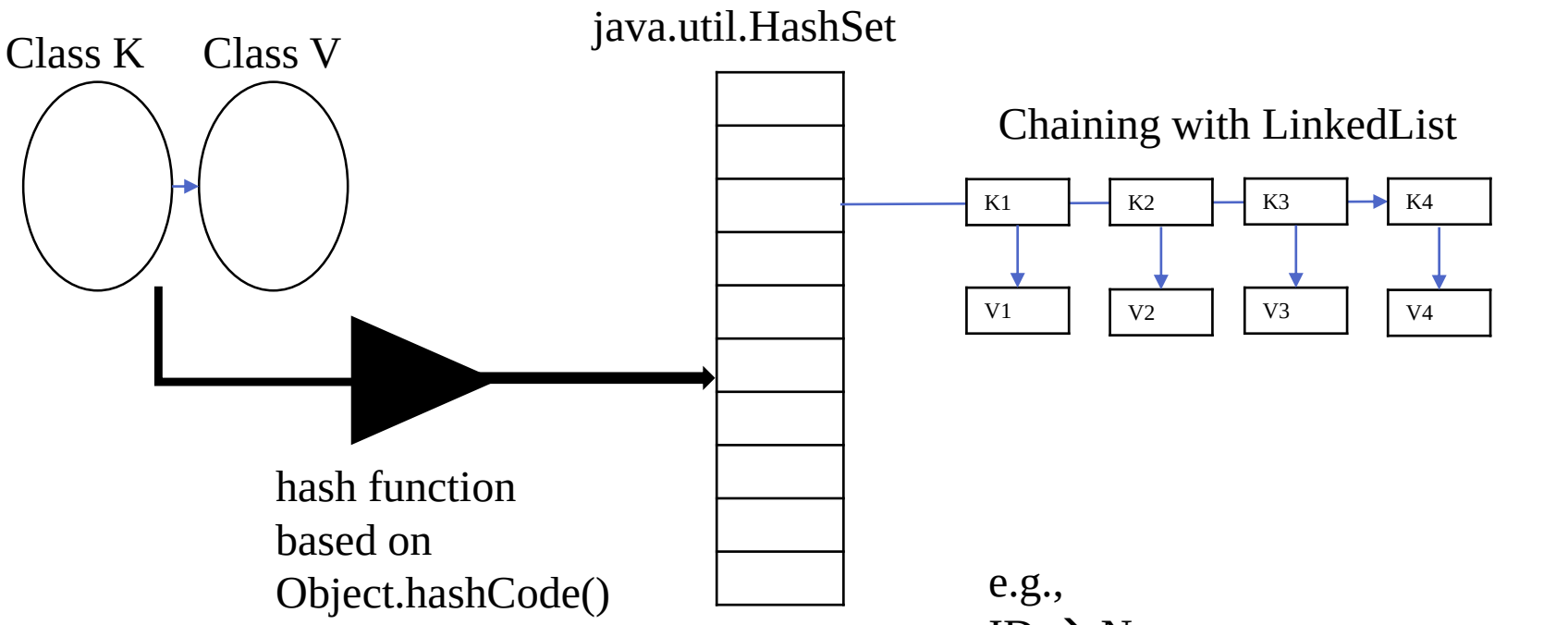
Collection Framework: HashMap

- HashSet consists of unordered, non-redundant elements



http://www.thecshandbook.com/Hash_Set

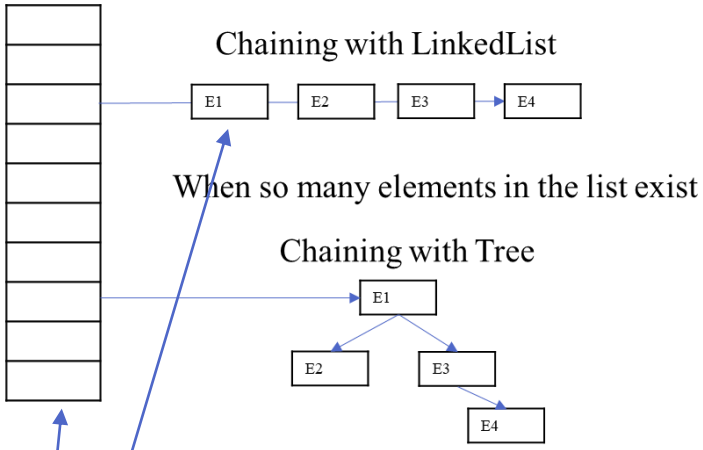
Collection Framework: HashMap



e.g.,
ID → Name
Phone Number → Device Name

Collection Framework: HashMap

- HashMap
  - Unordered, Non-redundant elements E
  - Based on Hashing
  - CRUD by hash




```
public class HashMap<K,V> extends AbstractMap<K,V>
    implements Map<K,V>, Cloneable, Serializable {

    transient Node<K,V>[] table;
     static class Node<K,V> implements
    Map.Entry<K,V> {
            final int hash;
            final K key;
            V value;
            Node<K,V> next;
              …
    }

}
```

Collection Framework: HashMap

- HashMap
  - implementing Map

메서드	설 명
void clear( )	Map의 모든 객체를 삭제한다.
boolean containsKey(Object key)	지정된 key객체와 일치하는 Map의 key객체가 있는지 확인한다.
boolean containsValue(Object value)	지정된 value객체와 일치하는 Map의 value객체가 있는지 확인한다.
Set entrySet( )	Map에 저장되어 있는 key-value쌍을 Map.Entry타입의 객체로 저장한 Set으로 반환한다.
boolean equals(Object o)	동일한 Map인지 비교한다.
Object get(Object key)	지정한 key객체에 대응하는 value객체를 찾아서 반환한다.
int hashCode( )	해시코드를 반환한다.
boolean isEmpty( )	Map이 비어있는지 확인한다.
Set keySet( )	Map에 저장된 모든 key객체를 반환한다.
Object put(Object key, Object value)	Map에 value객체를 key객체에 연결(mapping)하여 저장한다.
void putAll(Map t)	지정된 Map의 모든 key-value쌍을 추가한다.
Object remove(Object key)	지정한 key객체와 일치하는 key-value객체를 삭제한다.
int size( )	Map에 저장된 key-value쌍의 개수를 반환한다.
Collection values( )	Map에 저장된 모든 value객체를 반환한다.

Collection Framework: HashMap

- HashMap
  - Practice HashMap
    - CRUD
    - Generics
    - Iterator
    - hashCode
    - equals

Question #3

- Compute the identifier where its occurrence is maximum in the dataset.

- Keep all the key-value pairs of ID and its occurrence

Question #4

- Compute the number of people who received email(s) from ?.

Question #5

- Compute the number of people who received email(s) from ?.

HashMap<K,V>: K, V could be any class such as

HashMap<String, Double>

HashMap<String, HashSet<String>>

HashMap<String, HashMap<Double, Integer>>

HashMap<Integer, HashMap<Integer, HashSet<String>>>

Question #6

- Compute the number of people who send email(s) to ?.
- Compute the number of people who received email(s) from people who received email(s) from ?.
- Compute the number of people who send email(s) to people who send email(s) to ?.
- Compute the number of people [who received email(s) from people]* ?.
- Compute the number of people [who send email(s) to people]* ?.

Second Dataset

- Super User temporal network
  - http://snap.stanford.edu/data/sx-superuser.html
  - Syntax
    - Source Destination UNIX_EPOCH
  - A temporal network of interactions on the stack exchange
    - A source sends a message to a destination at a specific time

Question #7

- Compute the first unix epoch time
- Compute the second unix epoch time
- Compute the last unix epoch time
- Compute the second last unix epoch time

Summary

- Some Practice!
- Collecting, modelling, and analyses based on Hash-based collection

- Next Week
  - Collecting, modelling, and analyses based on Tree-based collection

Summary

- Some Practice!
- Collecting, modelling, and analyses based on Hash-based collection

- Next Week
  - Collecting, modelling, and analyses based on Hash-based collection (Cont.)