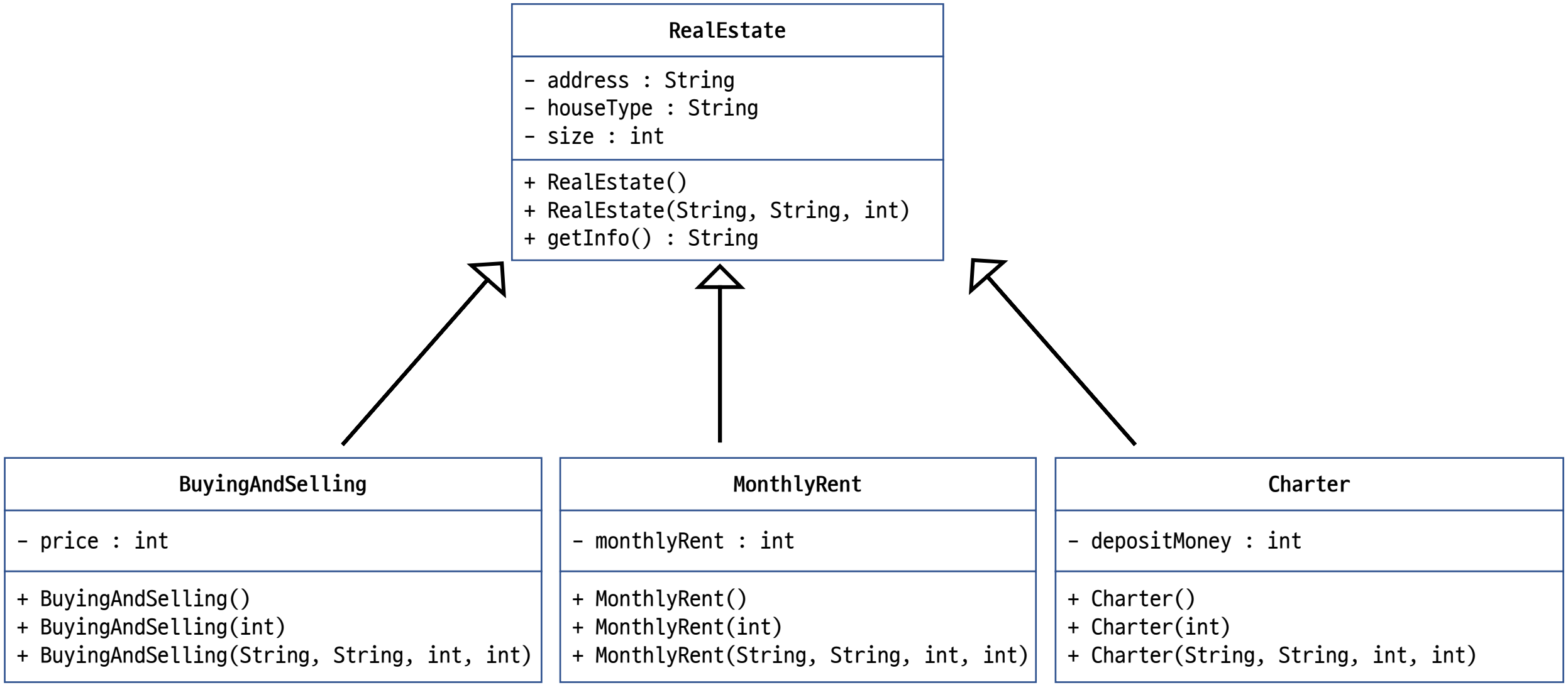
Java Fundamental

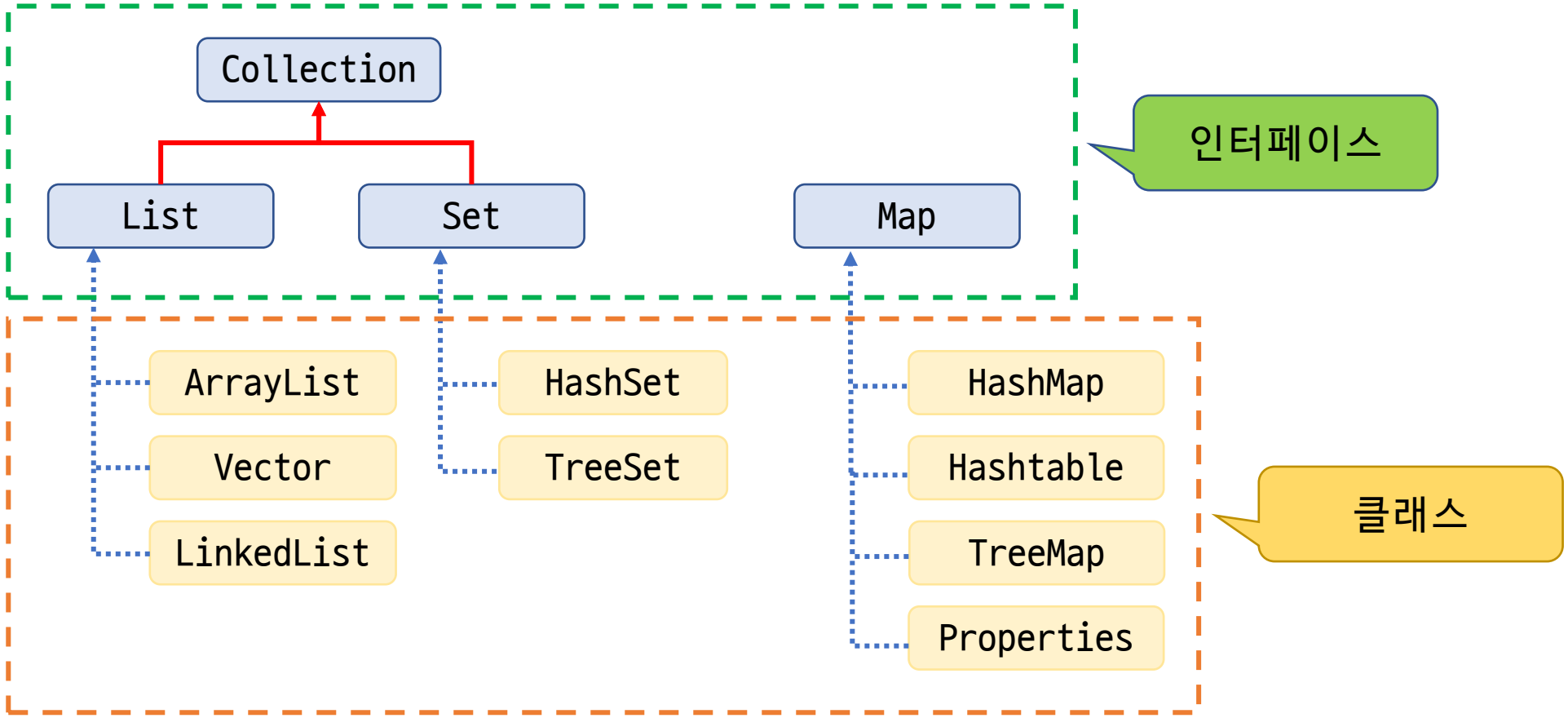
[연습-2] 클래스 다이어그램의 상속관계



컬렉션(Collection)

자바는 **자료구조(Data Structure)**를 사용해서 객체들을 효율적으로 **추가, 삭제, 검색**할 수 있도록 인터페이스와 구현 클래스를 java.util 패키지에서 제공합니다. 이들을 컬렉션 이라고 부른다.

컬렉션의 계층도



List

List는 배열과 비슷하게 인덱스로 관리한다. 배열과의 차이점은 크기를 조절할 수 있다는 점이다.
또한 추가,삭제,검색을 위한 다양한 메소드들이 제공된다.

List의 주요 메소드

기능	메소드	설명
객체 추가	boolean add(E e)	주어진 객체를 맨 끝에 추가
	void add(int index, E element)	주어진 인덱스에 객체를 추가
	E set(int index, E element)	주어진 인덱스에 저장된 객체를 주어진 객체로 변경
객체 검색	boolean contains(Object o)	주어진 객체가 저장되어 있는지 조사
	E get(int index)	주어진 인덱스에 저장된 객체를 리턴
	boolean isEmpty()	컬렉션이 비어 있는지 조사
	int size()	저장되어 있는 전체 객체 수를 리턴
객체 삭제	void clear()	저장된 모든 객체를 삭제
	E remove(int index)	주어진 인덱스에 저장된 객체를 삭제
	boolean remove(Object o)	주어진 객체를 삭제

ArrayList

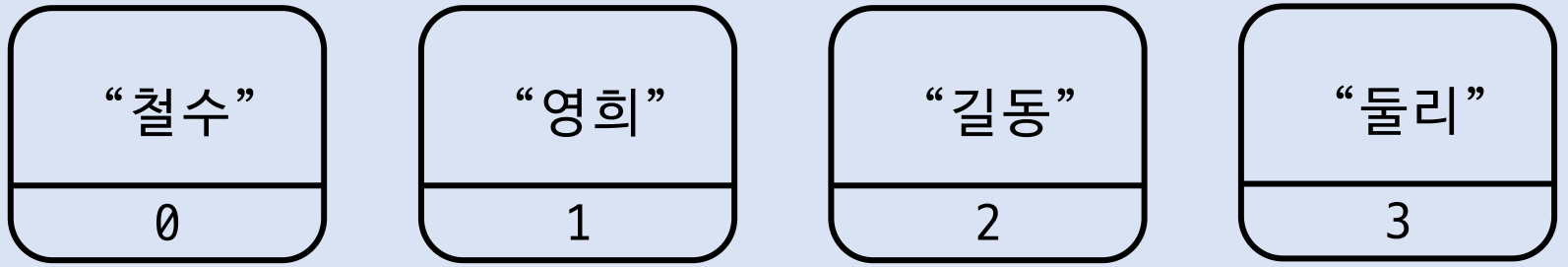
ArrayList는 List인터페이스의 대표적인 구현 클래스이다.

ArrayList 객체 생성 방법

```
ArrayList<저장할 객체타입> 변수명 = new ArrayList<>();
ArrayList<String> strList = new ArrayList<>(); // (예 : 문자열 ArrayList )
```

ArrayList 객체에 데이터 추가하기 1

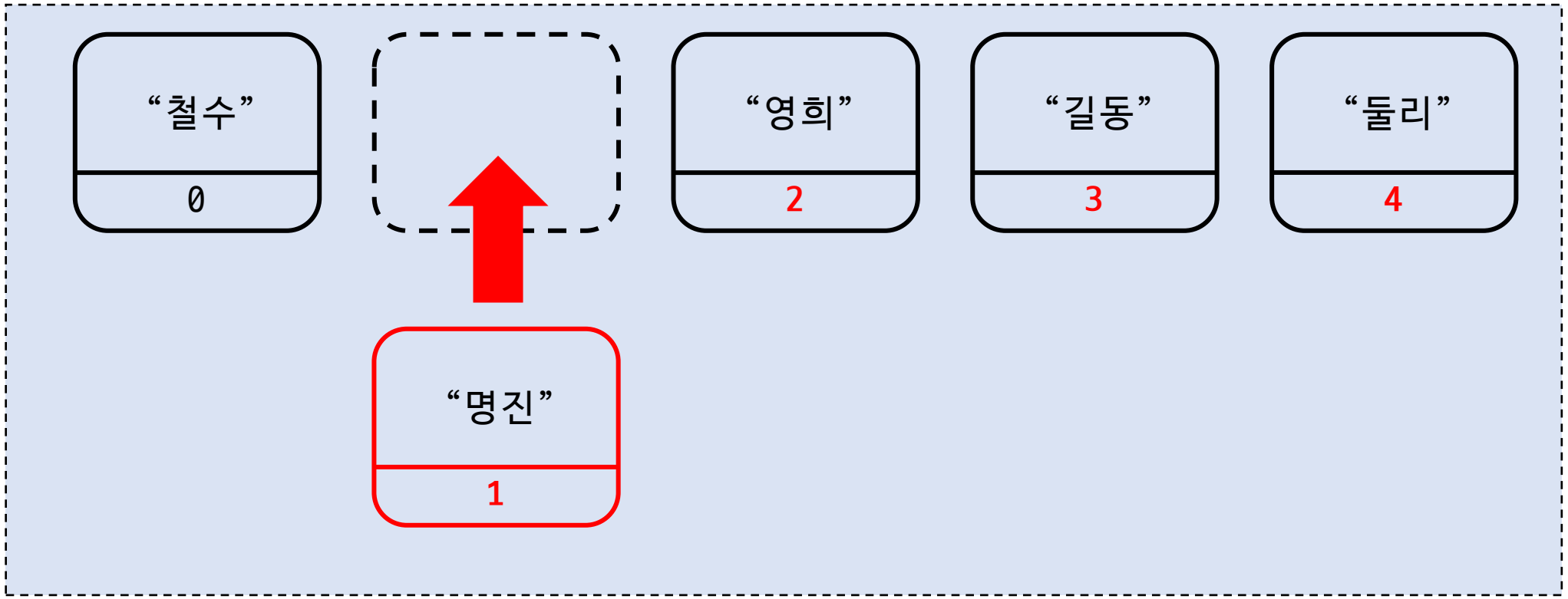
```
strList.add(“철수”);
strList.add(“영희”);
strList.add(“길동”);
strList.add(“둘리”);
```



ArrayList

ArrayList 객체에 데이터 추가하기 2

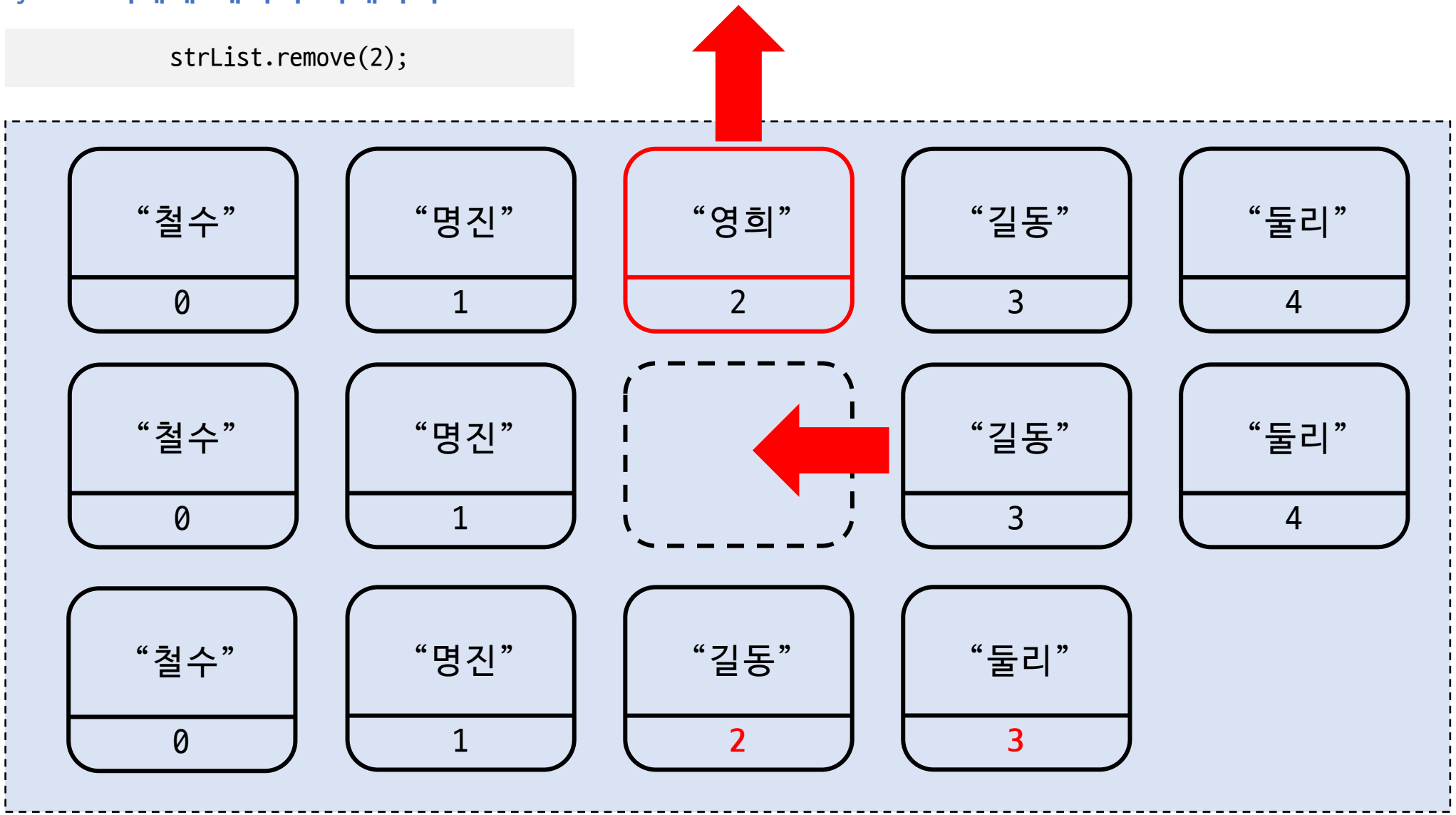
```
strList.add(1,“명진”);
```



ArrayList

ArrayList 객체에 데이터 삭제하기

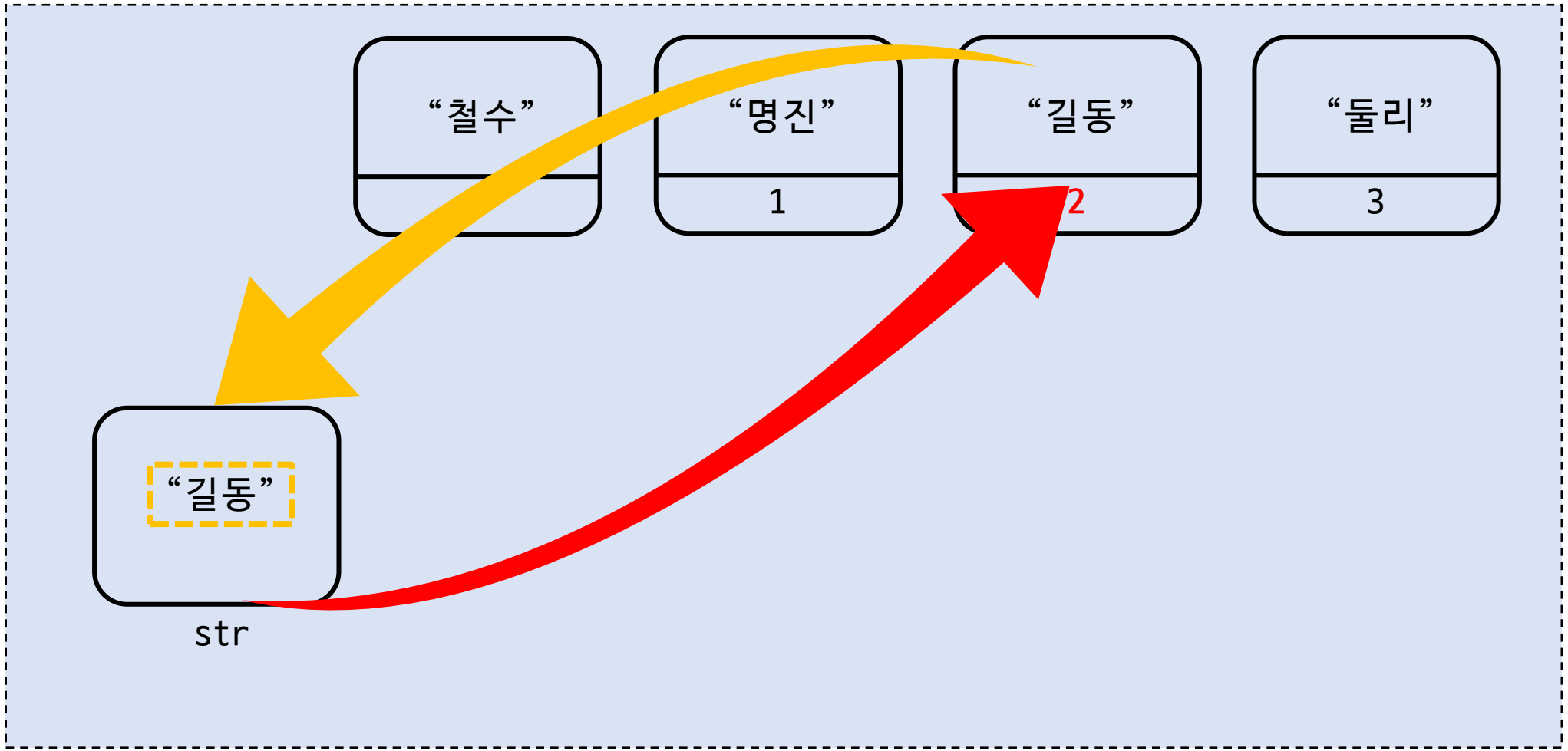
```
strList.remove(2);
```



ArrayList

ArrayList 객체에 데이터 가져오기

```
String str = strList.get(2);
```



Set

Set은 객체의 저장 순서가 유지되지 않고 중복해서 저장할 수 없는 특징을 가지고 있다.
Set은 순서가 없기 때문에 인덱스가 없고 따라서 인덱스로 객체를 검색해서 가져오는 메소드도 없다.

Set의 주요 메소드

기능	메소드	설명
객체 추가	boolean add(E e)	주어진 객체를 저장. 객체가 성공적으로 저장되면 true를 리턴하고 중복 객체면 false를 리턴
객체 검색	boolean contains(Object o)	주어진 객체가 저장되어 있는지 조사
	Iterator<E> iterator()	저장된 객체를 한 번씩 가져오는 반복자를 리턴
	boolean isEmpty()	컬렉션이 비어 있는지 조사
	int size()	저장되어 있는 전체 객체 수를 리턴
객체 삭제	void clear()	저장된 모든 객체를 삭제
	boolean remove(Object o)	주어진 객체를 삭제

Iterator의 주요 메소드

리턴 타입	메소드	설명
boolean	hasNext()	가져올 객체가 있으면 true를 리턴하고 없으면 false를 리턴.
E	next()	컬렉션에서 하나의 객체를 반환.
void	remove()	컬렉션에서 객체를 제거.

Map

Map은 키(Key)와 값(Value)로 구성된 객체를 저장하는 구조를 가지고 있다.
키(Key)는 중복 저장될 수 없지만 값(Value)는 중복 저장될 수 있다. 만약 기존에 저장된 키와 동일한 키로 값을 저장하면 기존의 값은 없어지고 새로운 값으로 대체된다.

Map의 주요 메소드

기능	메소드	설명
객체 추가	V put(K key, V value)	주어진 키로 값을 저장. 새로운 키일 경우 null을 리턴하고 동일한 키가 있을 경우 값을 대체하고 이전 값을 리턴.
객체 검색	boolean containsKey(Object key)	주어진 키가 있는지 여부를 확인.
	boolean containsValue(Object value)	주어진 값이 있는지 여부를 확인.
	Set<Map.Entry<K,V>> entrySet()	키와 값의 쌍으로 구성된 모든 Map.Entry 객체를 Set에 담아서 리턴.
	V get(Object key)	주어진 키가 있는 값을 리턴.
	boolean isEmpty()	컬렉션이 비어 있는지 조사
	int size()	저장되어 있는 전체 객체 수를 리턴.
객체 삭제	void clear()	저장된 모든 객체를 삭제.
	V remove(Object key)	주어진 키와 일치하는 객체를 삭제하고 값을 리턴.

HashMap

HashMap은 Map 인터페이스를 구현한 대표적인 Map 컬렉션이다.

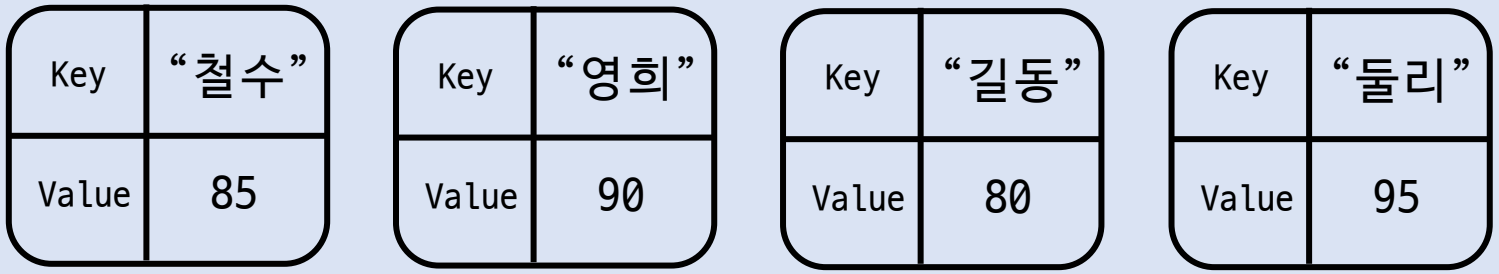
주로 Key 타입은 String을 많이 사용하고 Value 타입은 저장할 객체 타입을 사용한다.

HashMap 객체 생성 방법

```
HashMap<저장할 Key타입,저장할 Value타입> 변수명 = new HashMap<>();
HashMap<String,Integer> testMap = new HashMap<>(); // (예 : Key 문자열, Value Integer HashMap )
```

HashMap 객체에 데이터 추가하기 1

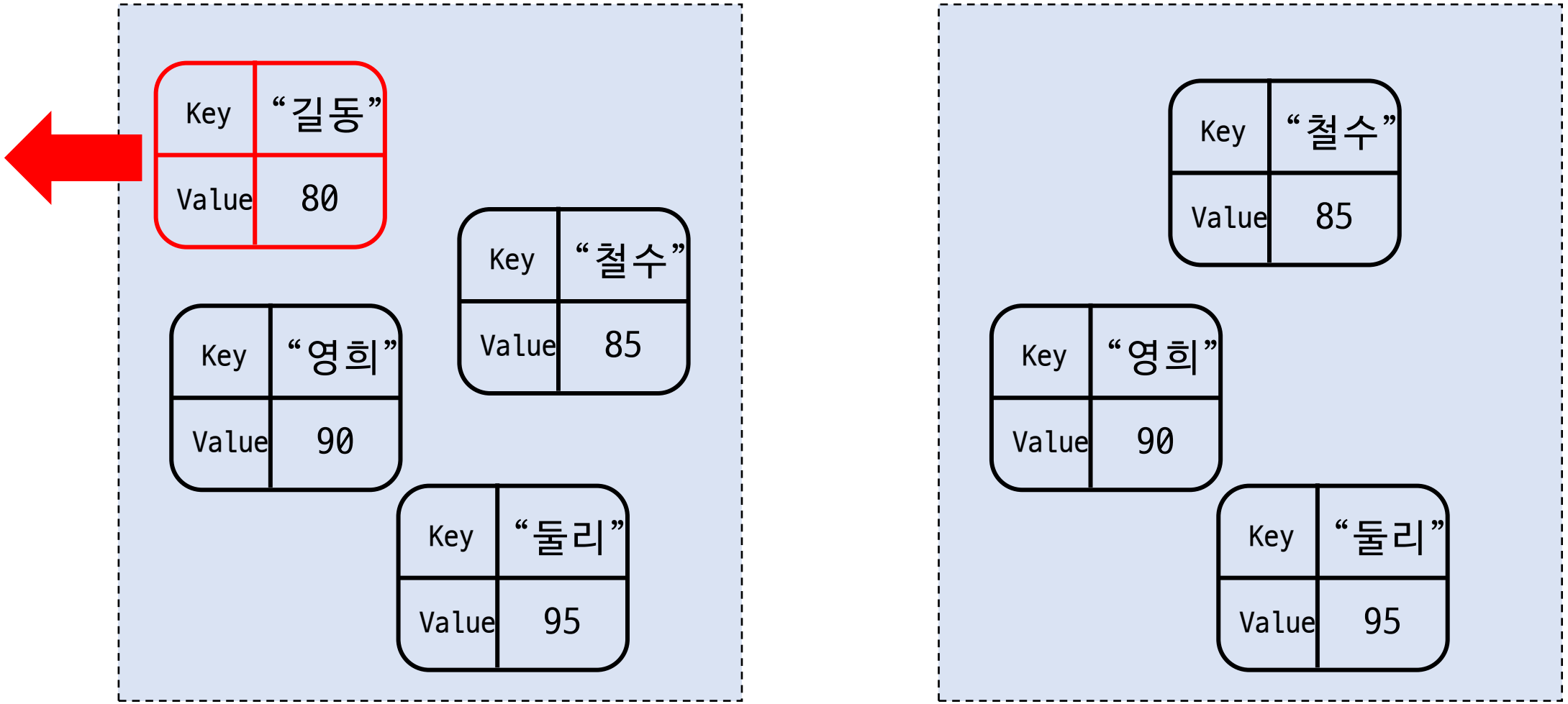
```
testMap.put("철수",85);
testMap.put("영희",90);
testMap.put("길동",80);
testMap.put("둘리",95);
```



HashMap

HasMap 객체에 데이터 삭제하기

```
testMap.remove("길동");
```



HashMap

HashMap 객체에 데이터 가져오기

```
Integer score = testMap.get("둘리");
```