컬렉션

- 다수의 데이터를 처리할 수 있는 자료구조로 반복 가능한 데이터의 집합이다.

> 반복 가능하다 : 반복문 내에서 순환할 수 있다.

1. List
   - 공식 API : https://api.dart.dev/stable/3.5.1/dart-core/List-class.html
   - 데이터 순서가 있고 중복 허용
   - 기본

```
// 값이 존재하는 List 객체 생성
List<String> colors = ['Red', 'Orange', 'Yellow'];

// 값이 없는 List 객체 생성
List<String> colorList = <String>[];
// List 클래스가 가지고 있는 메서드를 통해 값을 추가
colorList.add('Red');
colorList.add('Orange');
colorList.add('Yellow');
```

   - List의 Properties

프로퍼티	설명
first	컬렉션 첫번째 요소
last	컬렉션 마지막 요소
isEmpty	컬렉션이 비어있지 않으면 false, 비었으면 true
isNotEmpty	컬렉션이 비어있지 않으면 true, 비었으면 false
reversed	컬렉션 역순
single	컬렉션 에 단 1개의 요소만 있다면 해당 요소 리턴

   - List의 Methods

메소드	설명
indexOf(요소)	요소의 인덱스 값
add(데이터)	데이터 추가
addAll([데이터1, 데이터2])	여러 데이터 추가
remove(요소)	요소 삭제
removeAt(인덱스)	지정한 인덱스 삭제
contains(요소)	요소가 포함되어 있으면 true, 아니면 false
clear()	컬렉션 요소 전체 삭제
sort()	컬렉션 요소 정렬

2. Set
   - 공식 API : https://api.dart.dev/stable/3.5.1/dart-core/Set-class.html
   - 데이터 순서가 없고 중복 허용하지 않음
   - 기본

```
// 값이 존재하는 Set 객체 생성
Set<String> colors = {'RED', 'Orange', 'Yellow'};

// 값이 없는 Set 객체 생성
```

```
Set<String> colorList = <String>{};
// Set 클래스가 가지고 있는 메서드를 통해 값을 추가
colorList.add('Red');
colorList.add('Orange');
colorList.add('Yellow');
```

- Set의 Properties
  - 순서와 관련된 프로퍼티는 존재하지 않음

프로퍼티	설명
first	컬렉션 첫번째 요소
last	컬렉션 마지막 요소
isEmpty	컬렉션이 비어있지 않으면 false, 비었으면 true
isNotEmpty	컬렉션이 비어있지 않으면 true, 비었으면 false
single	컬렉션에 단 1개의 요소만 있다면 해당 요소 리턴

- Set의 Methods
  - 인덱스와 관련된 메서드는 존재하지 않음

메소드	설명
add(데이터)	데이터 추가
addAll([데이터1, 데이터2])	여러 데이터 추가
remove(요소)	요소 삭제
contains(요소)	요소가 포함되어 있으면 true, 아니면 false
clear()	컬렉션 요소 전체 삭제
sort()	컬렉션 요소 정렬

3. Map
   - 공식 API : https://api.dart.dev/stable/3.5.1/dart-core/Map-class.html
   - 키(key)와 값(value)으로 구성되며 키는 중복되지 않고 값은 중복 가능
   - 기본

```
main() {
  // 값이 존재하는 Map 객체 생성
  Map<int, String> testMap = {1: 'Red', 2: 'Orange', 3: 'Yellow'};
  testMap.forEach((key, value) {
    print('$key : $value');
  });

  // 값이 없는 Map 객체 생성
  Map<String, int> testMaps = Map();
  // 객체에 필드를 추가하는 방식으로 키와 해당 값을 추가
  testMaps['Red'] = 1;
  testMaps['Orange'] = 2;
  testMaps['Yellow'] = 3;

  testMaps.forEach((key, value) {
    print('$key : $value');
  });

  // 값을 수정하는 경우
  testMaps['Red'] = 10;
  print('${testMaps['Red']}');

  // Map 클래스가 가지고 있는 메서드를 활용
```

```
  // update(key, 이미 등록되어 있는 경우, 등록되지 않은 경우);
  testMaps.update('Blue', (value) => 20, ifAbsent: () => 0);
  print('${testMaps['Blue']}');
}
```

- Map의 Properties

프로퍼티	설명
isEmpty	컬렉션이 비어있지 않으면 false, 비었으면 true
isNotEmpty	컬렉션이 비어있지 않으면 true, 비었으면 false
keys	컬렉션이 가지고 있는 모든 키(key)를 반환
values	컬렉션이 가지고 있는 모든 값(value)을 반환

- Map의 Methods

메소드	설명
containsKey(키)	키가 포함되어 있으면 true, 아니면 false
containsValue(Value)	값이 포함되어 있으면 true, 아니면 false
remove(키)	키에 해당하는 Entry(key,value) 삭제
removeWhere(익명함수)	특정 조건을 기준으로 Entry(key,value) 삭제
update(키, 수정할 값[, 추가할 값])	해당 키가 있을 경우 두번째 인자로 수정 해당 키가 없을 경우 세번째 인자로 등록