Pre-URP 0803

집합의 구성 방법

집합의 고전적 정의: 상상할 수 있는 명백한 기준에 따라 그 구성원이 될 수 있는 것들을 모은 것

(1) 원소나열법: $\{a, b, c, d, \cdots\}$
(2) 조건제시법: $\{x \in \mathbb{N} \mid x > 10\}$

원소나열법은 유한 집합만 구성할 수 있음
조건제시법은 항상 집합을 구성할 수 있는가?

집합

- $x \in X$: true or false
- Principle of Extension: $X = Y \Leftrightarrow ((z \in X) \Leftrightarrow (z \in Y))$
- Empty set $\emptyset$ uniquely exists

함수

집합 X 와 Y 사이의 **관계**$^{\text{relation}}$란 $X \times Y$ 의 부분집합을 의미한다.
집합 X 에서 Y 로 가는 **함수**$^{\text{function}}$란 두 집합 사이의 관계 중 다음과 같은 조건을 만족하는 관계를 의미한다.

함수의 수학적 정의

$f \subset X \times Y$ 가 X 에서 Y 로 가는 함수이려면,
(1) 임의의 $x \in X$ 에 대해서 $(x, y) \in f$ 인 y가 존재한다.
(2) (1)과 같은 y 는 x 에 대해 유일하게 존재한다. 즉, $(x, y_1), (x, y_2) \in f$ 이면 $y_1 = y_2$이다.

$(x, y) \in f$ 일 때, $f(x) = y$ 라 쓴다. X 를 함수 f 의 정의역, Y 를 공역이라 하며

$$f : X \to Y$$

와 같이 표기한다.

Implementation at Go

집합

구성요소

- 원소

기능

- 원소의 포함여부
- 합집합, 차집합, 교집합, ...
- 곱집합 (카테시안): 이후 함수 정의에 필요함

Set 구조체

Field

- `elements []any`
- `elementtype reflect.Kind`

Method

- `contains(any)` // must be elementtype input
- `union(Set) Set` // $A \cup B$
- `intersection(Set)` // $A \cap B$
- `difference(Set)` // $A - B$
- `product(Set)` // $A \times B$

단점

원소들의 타입이 전부 같아야 함
슬라이스는 모든 집합을 대변할 수 없음 $(\mathbb{R}, \mathbb{N}, \mathbb{Z}, \cdots)$
float과 같은 타입도 set으로 취급하고 싶음

간소화

원소의 포함여부에 대한 함수만 있다면 집합이라고 할 수 있다고 하자
-> Set을 인터페이스로 정의

```
type Set interface {
        containChecker(any) bool
}

type Number struct {
        numberType reflect.Type
}

var Real = Number{reflect.TypeOf(1.00)}

var Integer = Number{reflect.TypeOf(1)}

func (N Number) containChecker(x any) bool {
        if reflect.TypeOf(x) == N.numberType {
                return true
        }
        return false
}
```

```
type Map interface {
        mapping(any) any
}

type Function struct {
        domain      Set
        codomain    Set
        maprelation func(any) any
}

func (F Function) mapping(input any) any {
        return F.maprelation(input)
}
```

```
func exmapping(input any) any {
        // Convert the input to float64
        x, ok := input.(float64)
        if !ok {
                // Handle the case where the input is not a float64
                // You could return an error or a default value here
                return nil
        }
        return math.Cosh(x)
}

var exfunc1 Function = Function{
        domain:      Real,
        codomain:    Real,
        maprelation: exmapping}
```

이후 해야 할 것

FiniteSet 구조체

Iterable 인터페이스

Matrix 구조체

Determinant 함수 정의 (Iterable 이용)