第3章　関数

A, B を集合とする．集合 A の各々の要素に対して集合 B の要素を対応させる仕方を関数 (function)，あるいは，写像 (map または mapping) という．

より正確にいうと，集合 A から集合 B への関数は，集合 A の全ての要素を集合 B の要素に対応付け，かつ，集合 A の全ての要素に対して，それに対応付けられる集合 B の要素は唯 1 つとなるものである．

例 40 関数と関数でないものの例.

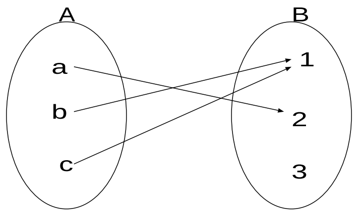

図 3.1: 関数の例 (集合 $\{a, b, c\}$ から集合 $\{1, 2, 3\}$ への関数)

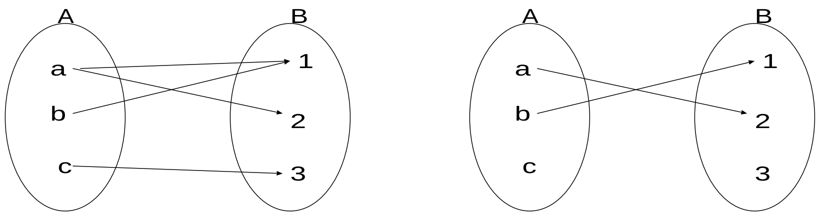

図 3.2: 関数でない例 (左: a に対応する値が 2 つある．右: c に対応する値がない．)

3.1　定義域と値域

f が集合 A から集合 B への関数であることを $f : A \to B$ と表す．このとき，A を f の定義域 (始域, ソースとも言う, domain), B を f のコドメイン (終域, ターゲット とも言う, codomain)[1] と呼ぶ．また，f によって $x \in A$ が $y \in B$ に対応付けられる (写される) ことを $f(x) = y$ と書く．x を f の引数 (ひきすう，argument)，y を f による x の値 (あたい，value) という．

[1] 高校までで習った「値域 (range)」と，コドメインは異なるものである．たとえば，実数から実数への関数 f が $f(x) = 0$ で定義されるとき f のコドメインは実数の集合だが，f の値域は $\{0\}$ である．ややこしいことに，昔は,「値域」という言葉をコドメインの意味にも使っていたので，用語が混乱していることがある．このテキストも去年までは「昔」の用語を使っていた．

3.2 複数の引数をもつ関数

関数の定義域を集合の直積とすることにより，引数を 2 個以上もつ関数を表すことができる．

二引数関数 (二項関数, binary function): $f : (A \times B) \to C$

なお，通常は，$A \times B \to C$ のように，かっこを省略する．$x \in A, y \in B$ に対して $f(\langle x, y \rangle)$ のことを $f(x, y)$ とも書く．

例 41 二つの整数の和を求める関数を考える．$plus : \mathcal{N} \times \mathcal{N} \to \mathcal{N}$．$plus(\langle x, y \rangle) = x$ と y の和. あるいは, $plus(x, y)$ と書く．

二引数関数と同様に多引数関数 (多項関数) も定義できる．

$f : A_1 \times A_2 \times \cdots \times A_n \to B$

例 42 n 個の自然数の最大値を求める関数 max は，$max : \mathcal{N}^n \to \mathcal{N}$ と書ける．

コンピュータ科学においては，引数の個数 n が不定の場合を扱うこともある．たとえば，いくつかの (何個かわからない) 自然数を与えられて，その中の最大値を返す関数を考えることもある．本講義資料の範囲内では，そのようなものは関数とは考えない．

3.3 像 (image)

関数 $f : A \to B$ と 集合 $C \subset A$ に対して，f による C の像 $f(C)$ は以下で定義される．

$$f(C) = \{f(x) \in B \mid x \in C\}$$

図 3.3 参照.

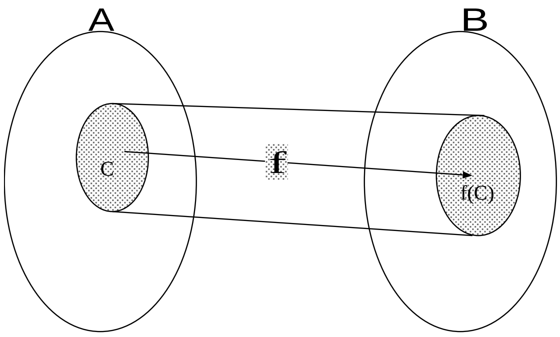

図 3.3: 関数 f による集合 C の像 $f(C)$.

像に関して, 以下の性質が成り立つ.

$$y \in f(C) \Leftrightarrow \exists x \in C \; y = f(x)$$

3.4 逆像 (inverse image) (†)

$f : A \to B$ と $D \subset B$ に対して，f による D の逆像 $f^{-1}(D)$ は以下のように定義される．

$$f^{-1}(D) = \{x \in A \mid f(x) \in D\}$$

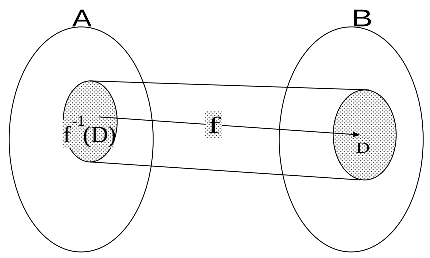

図 3.4: $f^{-1}(D)$.

逆像に関して, 以下の性質が成り立つ.

$$x \in f^{-1}(D) \Leftrightarrow f(x) \in D$$

例 43 図 3.5 に対して，関数 f の定義域は $A = \{a, b, c\}$, f の値域は $B = \{1, 2, 3\}$, $f(\{a, c\}) = \{1, 2\}$, $f(A) = \{1, 2\}$, $f(\{a\}) = \{1\}$, $f(\{a, b\}) = \{1\}$, $f^{-1}(\{1, 2\}) = \{a, b, c\}$, $f^{-1}(\{1, 3\}) = \{a, b\}$, $f^{-1}(\{3\}) = \phi$, $f^{-1}(B) = \{a, b, c\} = A$.

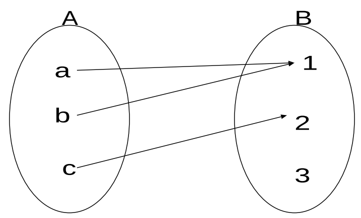

図 3.5: 関数 f.

3.5 関数の等しさ

定義域と値域が同じ 2 つの関数 $f : A \to B$, $g : A \to B$ が等しいとは，以下が成立することである.

$$\forall x \in A \quad f(x) = g(x)$$

このとき $f = g$ と書く. 関数の等しさとして他の定義を考えることもあるので，この等しさの定義を特に外延的な等しさ (extensional equality) という.

例 44 $f(x) = 2x$, $g(x) = x + x$ とすると $f = g$ である.

3.6 関数の合成 (composition)

2 つの関数 $f : A \to B$, $g : B \to C$ があるとき，f と g の合成を定義することができる．すなわち，$g \circ f : A \to C$ は，$(g \circ f)(x) = g(f(x))$ で定義される関数をあらわす.

ここで，$g \circ f$ における f, g の順番に注意する必要がある．f を先に適用して次に g を適用した合成関数を表すときに $g \circ f$ と表記する．これは，直感に反するが，$(g \circ f)(x) = g(f(x))$ とい

う自然な定義に対応付けるためのものである．一方，第 4 章では，関係の合成 $R \circ T$ を定義するが，この時は R を先に適用して次に T を適用する．すなわち，関数の合成と関係の合成は同じ $\circ$ という記号を使うが，逆の順番であることに注意されたい．関係と関数で合成の順番を変える絶対的な理由はなく，過去の慣習によるものである．

合成 $\circ$ は，以下の法則 (結合法則) を満たす:

$$(h \circ (g \circ f))(x) = h((g \circ f)(x)) = h(g(f(x))) = (h \circ g)(f(x)) = ((h \circ g) \circ f)(x)$$

しかし，$f \circ g = g \circ f$ とは限らない．

例 45 $f(x) = x^2$, $g(x) = x + 1$. $(f \circ g)(x) = f(x+1) = (x+1)^2$. $(g \circ f)(x) = g(x^2) = x^2 + 1$.

3.7 恒等関数 (identity function)

集合 A 上の恒等関数 $id_A : A \to A$ とは，$id_A(x) = x$ で定義される関数である．
$f : A \to B$ のとき, $f \circ id_A = f = id_B \circ f$.

3.8 単射 (一対一写像, one to one, injection)

関数 $f : A \to B$ が，以下の条件を満たすとき，単射という．

$$\forall x \in A\ \forall y \in A\ (f(x) = f(y) \Rightarrow x = y)$$

この条件は対偶を取って $\forall x \in A\ \forall y \in A\ (x \neq y \Rightarrow f(x) \neq f(y))$ とも書くことができる．つまり，A の異なる要素が B の異なる要素に写されるとき，単射という．

例 46 (図 3.6)

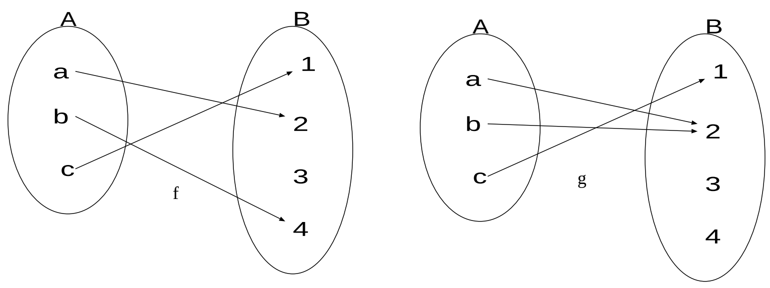

図 3.6: 単射 $f : A \to B$ と単射でない関数 $g : A \to B$.

例 47 $f(x) = ax + b$ (ただし，$a \neq 0$ とする) で定義される関数 $f : \mathcal{R} \to \mathcal{R}$ は単射である．
$g(x) = ax^2 + bx + c$ (ただし，$a \neq 0$ とする) で定義される関数 g は単射でない．

例 48 $A = \{6k + 4 \mid k \in \mathcal{N}\}$, $B = \{3k + 4 \mid k \in \mathcal{N}\}$, $f : A \to B$, $f(x) = x + 3$ と定義する．f は単射．なぜなら，$x \neq y \Rightarrow x + 3 \neq y + 3$.

3.9　全射 (上への写像，onto, surjection)

関数 $f : A \to B$ が，以下の条件を満たすとき，全射という．

$$\forall y \in B\ \exists x \in A\ f(x) = y$$

この条件は，$f(A) = B$ とも書くことができる．つまり，f による A の像が値域 B 全体になるとき，全射である．

例 49 (図 3.7)

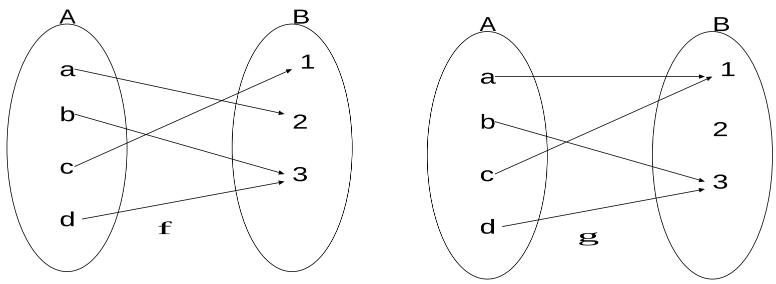

図 3.7: 全射 $f : A \to B$ と全射でない関数 $g : A \to B$.

例 50 $f : A \times B \to A, f(x, y) = x$ となる関数 f は単射でないが全射である．

例 51 $g : A \to A \times A, g(x) = \langle x, x\rangle$ となる関数 g は単射であるが全射でない．

3.10　全単射 (bijection) と逆関数 (inverse function)

単射かつ全射となる関数を全単射という．

$f : A \to B$ が全単射のとき，以下を満たす関数 $g : B \to A$ が存在する．g のことを f の逆関数という．

$$\forall x \in A\ \forall y \in B\ (f(x) = y \Leftrightarrow x = g(y))$$

このとき，$g \circ f = id_A$, $f \circ g = id_B$ が成立する．

例 52 E, O を，それぞれ偶数の集合と奇数の集合とする．$f : O \to E$, $f(x) = x - 1$，$g : E \to O$, $g(x) = x + 1$ とする．

f, g は全単射である．g は f の逆関数で，f は g の逆関数となる．

3.11　部分関数 (partial function)

関数は，定義域の要素すべてに対して，対応する値が唯一に定まるものであった．この条件を緩めて，集合 A のすべての要素に対して，集合 B の要素がたかだか 1 つ[2]対応付けられる場合，この対応付けを部分関数という．f が集合 A から集合 B への部分関数であることを，$f ; A \to B$ と書く．

[2]「たかだか 1 つ」というのは，0 個または 1 個という意味である．

例 53 div を実数上の割り算をあらすとすると，div は $\mathcal{R} \times \mathcal{R}$ から $\mathcal{R}$ への部分関数である．すなわち，$div; R \times R \to R$ である．

割算 $div(x, 0)$ (ただし $x \in R$) に対して，対応する値がない．これを「未定義の値」という．

関数は部分関数であるが，部分関数は関数とは限らない．関数を部分関数と明示的に区別したいときに，特に，全域関数 (total function) と呼ぶことがある．

コンピュータのプログラムは，いくつかの入力を与えて，出力を返すものと考えられる．このとき，プログラムは，関数とは限らない．なぜなら，プログラムの実行は，入力の値によっては停止しない (無限に計算を続ける) ため，出力が得られないことがあるからである．このように，コンピュータのプログラムは，関数ではなく部分関数としてモデル化することができる．

この章の他の節で述べた定義・性質等は関数についてのものであったが，多くのものは，部分関数に対しても拡張可能である．たとえば，部分関数同士の等しさ，合成関数，像，逆像などを，関数に対するものと同様に定義することができる．