プログラミング基礎 実習資料 第 01 時限

1　基礎的な数学

言語は考えを記述・具体化するものである．では，問題解決を目的として記述・具体化するべき考えとは何であろうか．扱う対象が理工学の問題であるならば数学の考え方が当然に必要になってくる．そこで，本節では，高等学校までの数学で学んだ関数，数列，集合などを復習する．しかし，高校までにこうした数学を履修していないなどのために初めてこうした概念に触れる学生やすぐに思い出すことのできない学生は，本節は簡潔に過ぎるので，効率的に学習するには他の初等的な文献に当たってほしい．

1.1　関数

変数 x の値に対応して，変数 y の値を確定する規則が与えられたと仮定するとき，y を x の **関数** という[*1]．特定の関数は

$$y = f(x) \tag{1}$$

と書く．例えば，規則「2 乗にせよ」を考える．この関数は

$$f(x) = x^2 \tag{2}$$

と表される．

問題 1 (A 問題)**.** 入力に規則「2 倍にせよ」を適用して得られた出力を考える．出力は入力の関数であるか．

解答例. 出力は入力の関数である．実際，入力を x，出力を $y = double(x)$ とすると，

$$double(x) = x + x$$

と表すことができる．（ここでは関数名に f ではなく $double$ を用いた．） □

絶対値を計算する関数 abs を考える．この関数は入力の種類によって場合分けが必要である：

$$abs(x) = \begin{cases} x & \textbf{if } x \geq 0 \\ -x & \textbf{otherwise} \end{cases} \tag{3}$$

問題 2 (B 問題)**.** 入力が正であったら 1，0 であったら 0，負であったら -1 を出力する関数 $sign$ を定義せよ．

[*1] プログラミング言語において「関数」は，別の意味で使われる．

解答例. 以下の通りである.

$$sign(x) = \begin{cases} 1 & \textbf{if } x > 0 \\ 0 & \textbf{if } x = 0 \\ -1 & \textbf{otherwise} \end{cases} \tag{4}$$

□

問題 3 (A 問題). 何を入力しても 3 を出力する関数 *three* を定義せよ.

解答例. 以下の通りである.

$$three(x) = 3 \tag{5}$$

□

このように入力によって出力が変化せず一定の関数のことを**定数関数**という.

問題 4 (B 問題). 何を入力しても 42 を出力する関数 *fourtytwo* を定義せよ.

解答例. 以下の通りである.

$$fourtytwo(x) = 42 \tag{6}$$

□

問題 5 (A 問題). 2 倍して絶対値をとる関数 *da* を *double*, *abs* を用いて定義せよ.

解答例. **合成関数**を用いると

$$da(x) = abs(double(x)) \tag{7}$$

と表せる. □

問題 6 (B 問題). 関数 *double* を用いて, 4 倍する関数を *quad* を定義せよ. ただし, 合成関数を用いなさい.

解答例. **合成関数**を用いると

$$quad(x) = double(double(x)) \tag{8}$$

と表せる. □

本節の最初に記した関数についての説明において入力と出力との対応に数式が必要とされていないことに注意してほしい. 例えば, ある集合 $R = \{(1,4),(2,9),(3,2),(4,98)\}$ を考える. R に含まれる (x,y) を考えたとき, y は x の関数である. なお, この関数は入力が $1,2,3,4$ 以外のときは, 出力が未定義である.

数学における関数は, 各入力に対して出力を関連づけるが, 出力がどのように得られるかについては何も述べていない. コンピュータ上では, 入力から出力を得る計算過程は, 各手順が明確で有限の時間に実際に行うことができる必要がある. こうした計算過程は**アルゴリズム**とよばれる. アルゴリズムは, プログラミング言語を用いて記述することができる.

問題 7 (D 問題). ある集合 $R = \{(1,4),(2,9),(3,2),(4,98)\}$ を考える. 「入力 x が与えられたときある y が

存在して R に (x, y) という組が含まれるならば y を出力する」計算過程を与えよ．

解答例. 集合 R 中の組を次にように第 1 要素の昇順[*2]に一列に並べる[*3]:

$$(1, 4), (2, 9), (3, 2), (4, 98)\,.$$

$i = 1$ として 1 から順に処理を開始する．

1. この列の先頭から i 番目の組の第 1 要素が入力 x に等しいかを確認する．もし等しい場合はその組の第 2 要素を出力して計算を終了する．
2. i に 1 を加える．
3. i が 5 であるならば何も出力せずに終了する．
4. 1 に戻る．

□

本授業では，こうしたアルゴリズムをプログラミング言語で実装できるようになる．

1.2 項と式

$42, -10, 3.14$ などの**定数**や，x, y などの**変数**が掛けあわされた式 $42x, -10, y, 3.14xy$ などを**項**という．ひとつの項からできている式を**単項式**という．項を 1 つ以上足し合わせた式を**多項式**という．

これらは排反ではないことに注意せよ．例えば，式 x は変数であり，ひとつの変数からできている項であり，ひとつの項からできている単項式であり，多項式である．

問題 8 (A 問題)**.** 3.14，xy，および $x^2 + 2x + 1$ は何であるかを論じよ．

解答例. 3.14 は定数である．定数は項である．この式は，ひとつの項からできているので単項式である．また，単項式は多項式であるので，この式は多項式でもある．

xy は，変数 x, y が掛け合わされた項である．また，単項式であり，多項式でもある．

$x^2 + 2x + 1$ は，単項式ではない多項式である． □

演算子には**優先順位**がある．例えば，関数適用，かけ算，足し算の順に優先されるので，$f(x) + 2y$ は $(f(x)) + (2y)$ である．演算子には**結合順序**がある．例えば，差を表す演算子 $-$ は左に結合し，$3 - 2 - 1$ は $(3 - 2) - 1$ を意味し，$3 - (2 - 1)$ を意味しない．

プログラミング言語でも式や優先順位，結合順序といった概念が用いられる．例えば，プログラミング言語においても，因子，項，**式**，および**文**と呼ばれる構文単位がある．プログラミング言語の学習において，これらを区別して理解することが重要である．また，演算子の優先順位や結合順序も定められていることが普通である．Haskell や C では，おおよそ高校までの数学で用いてきた演算子の優先順位と一致している．

問題 9 (D 問題)**.** 優先順位が，関数適用，足し算，かけ算の順であった場合，$f(x) + 2y$ を括弧を省略せずに書くとどうなるか．

[*2] 昇順とは，小さい方から大きい方へ順に並べられていること．
[*3] 昇順に一列に並べるアルゴリズムは本授業でも別の日に扱う．

解答例. $(f(x)+2)y$ □

1.3 漸化式

例：階乗

高校数学では，非負整数 n の**階乗** $n!$ とは n 以下のすべての正整数の積である[*4]．ただし，$0!=1$ とする．例えば，

$$5! = 5\times(4\times(3\times(2\times 1))) \tag{9}$$
$$= 120 \tag{10}$$

である．「n 以下のすべての正整数の積」というのは，$n=0$ のときは 1 であり，$n\geq 1$ のときは，「n」と「$n-1$ 以下のすべての正整数の積」の積とも考えられる．この考えに基づくと階乗は

$$n! = \begin{cases} 1 & \textbf{if } n=0 \\ n\times(n-1)! & \textbf{if } n\geq 1 \end{cases} \tag{11}$$

と定義できる．ここで 2 行目では，n の階乗を定義するのに $n-1$ の階乗が用いられている．すなわち，ここではある問題の解を求めるのに，それよりも小さいサイズの同じ問題の解を用いている．このことを**再帰**という．この再帰的定義を用いると

$$\begin{aligned} 5! &= \{\ \text{階乗の定義 (11)}\ \} \\ &\quad 5\times(5-1)! \\ &= \{\ \text{減算の定義}\ \} \\ &\quad 5\times 4! \\ &= \{\ \text{階乗の定義 (11)}\ \} \\ &\quad 5\times(4\times(3-1)!) \\ &= \{\ \text{減算の定義と階乗の定義 (11)}\} \\ &\quad 5\times(4\times(3\times(2\times(1\times 0!)))) \\ &= \{\ \text{階乗の定義 (11)}\ \} \\ &\quad 5\times(4\times(3\times(2\times(1\times 1)))) \\ &= \{\ \text{乗算の定義}\ \} \\ &\quad 120 \end{aligned}$$

となる．すなわち，5! を等式変形して 120 が得られる．式 (11) は，場合分けが排反であるので，n を非負整数としたとき，

$$0! = 1 \tag{12}$$
$$n! = n\times(n-1)! \qquad \textbf{if } n\geq 1 \tag{13}$$

と書き直せる．

問題 10 (B 問題)**.** 等式 12 と等式 13 を用いて 3! の値を導き出しなさい．（途中の計算を省略しないこと）

[*4] ちなみに，大学の数学では，階乗の概念は複素数の引数（ひきすう）を取る Γ 関数に拡張される．

解答例.

$$
\begin{aligned}
3! &= \{ \text{ 階乗の定義 (13) } \} \\
&\quad 3 \times (3-1)! \\
&= \{ \text{ 減算の定義 } \} \\
&\quad 3 \times 2! \\
&= \{ \text{ 階乗の定義 (13) } \} \\
&\quad 3 \times (2 \times (2-1)!) \\
&= \{ \text{ 減算の定義 } \} \\
&\quad 3 \times (2 \times (1 \times 0!)) \\
&= \{ \text{ 階乗の定義 (12) } \} \\
&\quad 3 \times (2 \times (1 \times 1)) \\
&= \{ \text{ 乗算の定義 } \} \\
&\quad 3 \times (2 \times 1) \\
&= \{ \text{ 乗算の定義 } \} \\
&\quad 3 \times 2 \\
&= \{ \text{ 乗算の定義 } \} \\
&\quad 6
\end{aligned}
$$

□

問題 11 (B 問題)**.** $n!$ が n の関数であることを説明せよ．ヒント：n から $n!$ を確定する規則は何であるか.

解答例.「階乗せよ」という規則である. □

$n!$ は，演算子 $!$ が引数 n を取っている．このように，引数の後ろに書く関数（演算子）を**後置**という．一方，上に出てきた $f(x)$ のように引数の前に書く関数は**前置**という．なお，$x - y$ に出てくる演算子 $-$ は**中置**という．なお，中置演算子 $-$ は，次のように前置の関数 sub に書き換えることができる.

$$sub(x, y) = x - y$$

問題 12 (B 問題)**.** 等式 12 と等式 13 の後置演算子 $!$ を前置の関数 $fact$ で書き換えよ.

解答.（前置にすることに注意...）非負整数 n を用いて

$$
\begin{aligned}
&fact(0) = 1 \\
&fact(n) = n \times fact(n-1) \qquad \textbf{if } n \geq 1
\end{aligned}
$$

と表せる. □

問題 13 (D 問題)**.** $fact(5)$ を $fact$ の定義を用いて整数値を導き出しなさい.

解答例.

$$
\begin{aligned}
fact(5) &= \{\ fact\text{の定義 } (14)\ \} \\
&\quad 5 \times fact(5-1) \\
&= \{\ \text{減算の定義}\ \} \\
&\quad 5 \times fact(4) \\
&= \{\ fact\text{の定義 } (14)\ \} \\
&\quad 5 \times (4 \times fact(3-1)) \\
&= \dots \\
&\quad 5 \times (4 \times (3 \times (2 \times (1 \times fact(0))))) \\
&= \{\ fact\text{の定義 } (14)\ \} \\
&\quad 5 \times (4 \times (3 \times (2 \times (1 \times 1)))) \\
&= \cdots \\
&= 120
\end{aligned}
$$

□

詳しい説明

前の項から，その次に続く項を定める規則を表した式を**漸化式**という．

初項 a, 公差 d の等差数列 $\{a_n\}_{n\geq 0}$ は

$$a_0 = a \tag{14}$$
$$a_n = a_{n-1} + d \qquad \textbf{if } n \geq 1 \tag{15}$$

というように，初項と漸化式で定められる．初項 a, 項比 r の等比数列 $\{a_n\}_{n\geq 0}$ は

$$a_0 = a \tag{16}$$
$$a_n = r \times a_{n-1} \qquad \textbf{if } n \geq 1 \tag{17}$$

という等式で定められる．

問題 14 (A 問題)**.** 等差数列 $4, 8, 12, \dots$ を定める式を求めなさい．この数列は第何項から 20 以上になるか．

解答例. この数列は

$$
\begin{aligned}
a_0 &= 4 \\
a_n &= a_{n-1} + 4 \qquad \textbf{if } n \geq 1
\end{aligned}
$$

と表せる．$\{a_n\}_{n\geq 0}$ を第 0 項から順に書き出すと $4, 8, 12, 16, 20, \dots$ であり，第 4 項から 20 以上になる． □

問題 15 (B 問題)**.** 等差数列 $2, 4, 6, \dots$ と等差数列 $3, 6, 9, \dots$ とに共通に含まれる数が作る数列の第 10 項は何か．ただし，数列の初項は第 0 項とする．（ヒント：順に書き出しても解ける）

解答例．この数列を第 0 項から順に書き出すと，

$$6, 12, 18, 24, 30, 36, 42, 48, 54, 60, 66 \ldots$$

となっており，第 10 項は 66 である． □

等差数列や等比数列以外は，一般項を求めることが難しい．しかし，次回以降で学ぶようにコンピュータの助けをかりると，人の手では煩雑すぎて計算が難しい項の値を求めたり，それらの値から特徴を探ったりすることができるようになる．

問題 16 (D 問題)**.** 次の式で定まる数列の第 4 項 a_4 を求めよ．

$$\begin{aligned} a_0 &= 1 \\ a_n &= 2 \times a_{n-1} + 1 \qquad \textbf{if } n \geq 1 \end{aligned}$$

解答例．数列は $1, 3, 7, 15, 31, \ldots$ となり，第 4 項は 31 である． □

問題 17 (D 問題)**.** 3 つの文字 a, b, c を繰り返しを許して，左から順に n 個並べる．ただし，a の次は必ず b であるとする．例えば，1 個並べられたときは，

$$b, c$$

の 2 つの場合がある．また，2 個並べられたときは，先頭文字が a のときは必ず次が b であるが，b および c は任意の文字がくるので

$$ab, bb, cb, bc, cc$$

の 5 つの場合がある．したがって $z_1 = 2$ および $z_2 = 5$ である．このような列の個数 z_n を定める式を求めよ．また，z_5 を求めよ．

解答例．数列 $\{z_n\}_{n\geq 1}$ は，等式

$$\begin{aligned} z_1 &= 2 \\ z_2 &= 5 \\ z_n &= 2z_{n-1} + z_{n-2} \qquad \textbf{if } n \geq 3 \end{aligned}$$

で表される．漸化式は，長さ n の列の場合の数は，その先頭が，a のときは次が必ず b でその後ろから始まる長さ $n-2$ の長さの列の場合の数になり，それ以外のときはその後ろから始まる長さ $n-1$ の長さの列の場合の数になることから求まる．z_5 は以下の通り求まる：

$$\begin{aligned} z_5 &= 2z_4 + z_3 \\ &= 2(2z_3 + z_2) + z_3 \\ &= 5z_3 + 2z_2 \\ &= 5(2z_2 + z_1) + 2z_2 \\ &= 12z_2 + 5z_1 \\ &= 12 \times 5 + 5 \times 2 \\ &= 70 \end{aligned}$$

□

例：フィボナッチ数列，様々な数列

1 歩で 1 段か 2 段で登る登り方で，n 段の階段を上りきる登り方は，**フィボナッチ**（Fibonacci）数列 $\{fib_n\}_{n\geq 0}$ によって表される．また，以下を仮定すると，n ヶ月目のウサギの数は，フィボナッチ数列によって表される．

- 子ウサギは生後 1 ヶ月経つと親ウサギになる．
- 親ウサギは毎月 1 羽の子ウサギを産む．
- 0 ヶ月目にはウサギは全くおらず，1 ヶ月目には親ウサギが 0 匹，子ウサギが 1 匹いたとする．

フィボナッチ数列 $\{fib_n\}_{n\geq 0}$ は

$$fib_0 = 0 \tag{18}$$
$$fib_1 = 1 \tag{19}$$
$$fib_n = fib_{n-1} + fib_{n-2} \qquad \textbf{if } n \geq 2 \tag{20}$$

という等式で定められる．

問題 18 (B 問題)．フィボナッチ数列 $\{fib_n\}_{n\geq 0}$ の第 0 項から第 10 項まで求めよ．

解答例．

$$0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55$$

□

問題 19 (B 問題)．式 20 は，なぜ漸化式であるか．（ヒント：漸化式の定義は何であったか）

解答例．前の項から，その次に続く項を定める規則を表した式を漸化式といった．式 20 は，fib_{n-1} と fib_{n-2} からその次の項 fib_n を表した式であるため漸化式である．同様に，式 13 は，$fact(n-1)$ からその次の項 $fact(n)$ を表した式であるため漸化式である． □

問題 20 (D 問題)．0 から n までの数字の和を表す式 $0+1+2+\cdots+n$ の計算順序を明確にするために括弧

をつけるとする．何通りの括弧の付け方があるかを考える．例えば，$n = 1, 2, 3$ ではそれぞれ，

$$\begin{aligned} 0+1 &= (0+1) \\ &\quad 1 \text{ なら } 1 \text{ 通り } (A_1 = 1). \end{aligned}$$

$$\begin{aligned} 0+1+2 &= (0+(1+2)) \\ &= ((0+1)+2) \\ &\quad 2 \text{ なら } 2 \text{ 通り } (A_2 = 2). \end{aligned}$$

$$\begin{aligned} 0+1+2+3 &= (0+(1+(2+3))) \\ &= (0+((1+2)+3)) \\ &= ((0+1)+(2+3)) \\ &= ((0+(1+2))+3) \\ &= (((0+1)+2)+3) \\ &\quad 3 \text{ なら } 5 \text{ 通り } (A_3 = 5). \end{aligned}$$

となる．0 から n までの数字の和を表す式の括弧の付け方が A_n 通りであるとして，数列 $\{A_n\}_{n\geq 0}$ を定める式を求めなさい．

解答．A_n は漸化式を用いると

$$\begin{aligned} A_0 &= 1 \\ A_n &= \sum_{k=0}^{n-1} A_k A_{n-k-1} \qquad \textbf{if } n \geq 1 \end{aligned}$$

と表せる．なお，一般式は

$$A_n = \frac{1}{n+1}\left(\begin{array}{c} 2n \\ n \end{array} \right)$$

である． □

問題 21 (D 問題)**.** n 人に送る招待状と封筒をそれぞれ n 枚用意した．それぞれの封筒に招待状を 1 枚ずつ入れるとすると，招待状がすべて誤った宛名に入れられる場合の数を a_n とする．数列 $\{a_n\}$ を定める式を求めよ．また，a_4 を計算せよ．a_{10} は計算できるだろうか．

解答例．

$$\begin{aligned} a_1 &= 0 \\ a_2 &= 1 \\ a_n &= (n-1)(a_{n-1} + a_{n-2}) \qquad \textbf{if } n \geq 3 \end{aligned}$$

a_4 は 9，a_{10} は 1334961 である． □

1.4 順列・組合せ

例えば，A, B, C と書かれたボールから 2 個取り出して並べると，

AB　AC　BA　BC　CA　CB

の 6 通りの場合がある．また，同じ条件で取り出す組合せのみを考えると

AB　AC　BC

の 3 通りの場合がある．

一般の場合を考える．与えられたいくつかのものの中から，一定個数のものを取り出し，一列に並べる並べ方の 1 つ 1 つを**順列**という．n 個のものから k 個取った順列の総数は ${}_nP_k$ と表され，

$$ {}_n\mathrm{P}_k = n(n-1)(n-2)\cdots(n-k+1) $$

である．また，n 個のものから k 個を選ぶときの選び方の総数は**組合せ**と呼ばれ，${}_nC_k$ と表す．

順列の数は階乗を用いて

$$ {}_n\mathrm{P}_k = \frac{n!}{(n-k)!} \tag{21} $$

と表せる．組合せの数は順列の数と階乗を用いて

$$ {}_n\mathrm{C}_k = \frac{{}_n\mathrm{P}_k}{k!} \tag{22} $$

と表せる．

問題 22 (D 問題)**.** 5 人から 3 人を選び出して並べる方法は何通りであるか．また，10 人の中から 5 人を選び出して並べる方法は何通りであるか．

解答例．5 人から 3 人を選び出して並べる方法は，$60(= {}_5\mathrm{P}_3)$ 通りである．10 人の中から 5 人を選び出して並べる方法は，$30240(= {}_{10}\mathrm{P}_5)$ 通りである． □

問題 23 (D 問題)**.** 5 人から 3 人を選び出す方法は何通りであるか．また，10 人の中から 5 人を選び出す方法は何通りであるか．

解答例．5 人から 3 人を選び出す方法は，$10(= {}_5\mathrm{C}_3)$ 通りである．10 人の中から 5 人を選び出す方法は，$252(= {}_{10}\mathrm{C}_5)$ 通りである． □

組合せは，次のように再帰的定義で表すこともできる．

$$ \begin{aligned} {}_n\mathrm{C}_0 &= 1 \\ {}_n\mathrm{C}_n &= 1 \\ {}_n\mathrm{C}_k &= {}_{n-1}\mathrm{C}_k + {}_{n-1}\mathrm{C}_{k-1} \qquad \textbf{if } 1 \le k \le n-1 \end{aligned} $$

1.5 章末問題

問題 24 (B 問題)．入力された数の 2 乗を返す関数 *square* を定義せよ．

解答例．

$$square(x) = x^2$$

□

問題 25 (B 問題)．入力された数に 1 を加えたものを計算する関数 *plus1* を定義せよ．

解答例．

$$plus1(x) = x + 1$$

□

問題 26 (B 問題)．入力された数の 2 乗に 1 を加えたものを計算する関数 *sqplus1* を合成関数として定義せよ．

解答例．

$$sqplus1(x) = plus1(square(x))$$

□

問題 27 (B 問題)．リュカ数列 $\{L_n\}_{n\geq 0}$ は，第 0 項を 2，第 1 項を 1，それ以降の項をその前の 2 項の和として定められる．リュカ数列 $\{L_n\}_{n\geq 0}$ を定める式を求めなさい．（ヒント：フィボナッチ数列を定める式 18–20 において，第 0 項目の値を 2 に置き換えるとリュカ数列を定める式になる．）

解答例．

$$\begin{aligned} L_0 &= 2 \\ L_1 &= 1 \\ L_n &= L_{n-1} + L_{n-2} \qquad \textbf{if } n \geq 2 \end{aligned}$$

□

問題 28 (B 問題)．トリボナッチ数列 $\{T_n\}_{n\geq 0}$ は，第 0 項と第 1 項を 0，第 2 項を 1 それ以降の項をその前の 3 項の和として定められる．トリボナッチ数列 $\{T_n\}_{n\geq 0}$ を定める式を求めなさい．

解答例.

$$
\begin{aligned}
&T_0 = 0 \\
&T_1 = 0 \\
&T_2 = 1 \\
&T_n = T_{n-1} + T_{n-2} + T_{n-3} \qquad \textbf{if } n \geq 3
\end{aligned}
$$

□

問題 29 (B 問題). 数列 $\{a_n\}_{n\geq 1}$ の n 番目の項は，1 から n までの整数の 2 乗の和である．つまり，

$$
\begin{aligned}
a_1 &= 1^2 = 1 \\
a_2 &= 1^2 + 2^2 = 5 \\
a_3 &= 1^2 + 2^2 + 3^2 = 14 \\
a_4 &= 1^2 + 2^2 + 3^2 + 4^2 = 30
\end{aligned}
$$

である．数列 $\{a_n\}_{n\geq 1}$ を定める式を示しなさい．

解答例.

$$
\begin{aligned}
&a_1 = 1 \\
&a_n = n^2 + a_{n-1} \qquad \textbf{if } n \geq 2
\end{aligned}
$$

□

問題 30 (B 問題). 数列 $\{a_n\}_{n\geq 1}$ の n 番目の項は，1 から n までの整数の 3 乗の和である．つまり，

$$
\begin{aligned}
a_1 &= 1^3 = 1 \\
a_2 &= 1^3 + 2^3 = 9 \\
a_3 &= 1^3 + 2^3 + 3^3 = 36 \\
a_4 &= 1^3 + 2^3 + 3^3 + 4^3 = 100
\end{aligned}
$$

である．数列 $\{a_n\}_{n\geq 1}$ を定める式を示しなさい．

解答例.

$$
\begin{aligned}
&a_1 = 1 \\
&a_n = n^3 + a_{n-1} \qquad \textbf{if } n \geq 2
\end{aligned}
$$

□

問題 31 (B 問題)．数列 $\{a_n\}_{n\geq 1}$ は 2 から n 番目までの偶数の 2 乗の和である．つまり，

$$\begin{aligned} a_1 &= 2^2 = 4 \\ a_2 &= 2^2 + 4^2 = 20 \\ a_3 &= 2^2 + 4^2 + 6^2 = 56 \\ a_4 &= 2^2 + 4^2 + 6^2 + 8^2 = 120 \end{aligned}$$

である．数列 $\{a_n\}_{n\geq 1}$ を定める式を示しなさい．

解答例.

$$\begin{aligned} a_1 &= 4 \\ a_n &= (2n)^2 + a_{n-1} \qquad \textbf{if } n \geq 2 \end{aligned}$$

□

問題 32 (B 問題)．数列 $\{a_n\}_{n\geq 1}$ の n 番目の項は，1 から n 番目までの奇数の 2 乗の和である．つまり，

$$\begin{aligned} a_1 &= 1^2 = 1 \\ a_2 &= 1^2 + 3^2 = 10 \\ a_3 &= 1^2 + 3^2 + 5^2 = 35 \\ a_4 &= 1^2 + 3^2 + 5^2 + 7^2 = 84 \end{aligned}$$

である．数列 $\{a_n\}_{n\geq 1}$ を定める式を示しなさい．

解答例.

$$\begin{aligned} a_1 &= 1 \\ a_n &= (2n-1)^2 + a_{n-1} \qquad \textbf{if } n \geq 2 \end{aligned}$$

□

問題 33 (B 問題)．$a_n = x^n$ とする (n は 0 以上の整数とする)．数列 $\{a_n\}_{n\geq 0}$ を定める式を示しなさい．

解答例.

$$\begin{aligned} a_0 &= 1 \\ a_n &= x \times a_{n-1} \qquad \textbf{if } n \geq 1 \end{aligned}$$

□

問題 34 (D 問題)．n 勝先取の勝負 (先に n 勝した方が勝つ勝負) を A と B が戦うとする．どの 1 試合でも A と B のそれぞれが勝つ確率は p と $(1-p)$ で，引き分けはないとする．A が i 勝 j 敗の時の優勝確率 $A_{i,j}$ を再帰的定義を用いて表しなさい．

解答例.

$$
\begin{aligned}
A_{n,j} &= 1 \\
A_{i,n} &= 0 \\
A_{i,j} &= p \times A_{i+1,j} + (1-p) \times A_{i,j+1} \qquad \textbf{if } i < n \wedge j < n
\end{aligned}
$$

□

1.6 本章のまとめ

本章では，プログラミングにおいて必要とされる数学を復習した．ここでみた多くの初歩的な数学概念が，次章以降のプログラミングにおいても役に立つことであろう．関数，項と式，漸化式などは今後も頻出する．

2 Haskell 環境の整備

2013 年度生以降の貸与ノート PC には Haskell Platform がインストールされているが，他のマシンで実習を行う環境を整備するには以下を参考にしてほしい．

2012 年度生の貸与ノート PC には Haskell Platform を簡易にインストールする方法がないので，代わりに GHC をインストールしてほしい．インストールにはパッケージ管理システム APT を利用する．まず，リポジトリの更新を行う．GNOME 端末などから

```
$ sudo apt-get update
```

と打鍵する．パスワードが聞かれたら正確に打鍵する．次に，GHC のインストールを行う．

```
$ sudo apt-get install ghc6
```

と打鍵する．確認のメッセージが聞かれた場合は `Y` (Yes) と打鍵する．

最近のオペレーティングシステムに Haskell Platform をインストールすることは簡単である．Windows や Mac には `http://www.haskell.org/platform/` においてインストーラが用意されている．最新の Ubuntu には haskell-platform というパッケージが用意されている．