第1章 集計

集計処理とは

データ処理で行われる繰返し処理の代表として集計処理があります。集計処理とは，複数のデータを合計して結果を求めることで，一般的には次のような式を繰り返し実行することで実現します。

集計変数　←　集計変数　＋　加算項目（変数，定数）

繰返し処理の中で，この演算式が実行されていくことで，集計変数の中に，次々と加算項目が足し込まれていくことになります。このとき，注意するべきことは，集計変数の初期値の設定です。変数は，宣言しただけでは値が未定義の状態です。この演算式では，右辺にある「集計変数」の値が未定義となり，演算ができなくなってしまいます。そのため一般的には，繰返し処理に入る前に，集計変数に０を初期値として設定します。

カウンタと同様に，変数を集計に利用するために特別な定義（宣言）は必要なく，通常の変数と同様の宣言を行います。そのため，アルゴリズム中では，集計用の変数とカウンタ用の変数の区別がつきにくくなることがあります。一般的には，集計用の変数には「sum」や「total」などの変数名をつけ，その役割を明示するよう工夫します。

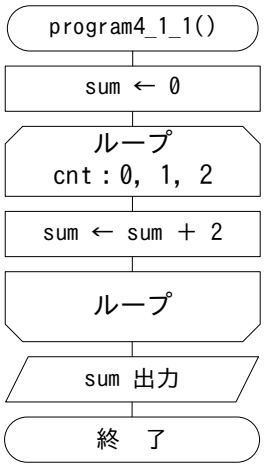
次からは，集計変数に足し込んでいく加算項目の変化に着目してどのようにアルゴリズムを考えていくべきなのかを説明します。

なお，以降の「擬似言語のトレース表」では，擬似言語の仕様にあわせて判定結果に「true」または「false」を使用します。

加算項目が定数の場合

次の図は，「２」という定数を３回加算する流れ図と擬似言語，そのトレース表です。集計変数 sum とカウンタ変数 cnt の動きの違いに注意して，結果を確認してみましょう。

【プログラム 4-1-1】

流れ図	擬似言語
	1 ○program4_1_1() 2 　整数型: sum 3 　整数型: cnt 4 　sum ← 0 5 　for (cnt を 0 から 2 まで 1 ずつ増やす) 6 　　sum ← sum ＋ 2 7 　endfor 8 　sum を出力

【実行結果】　6

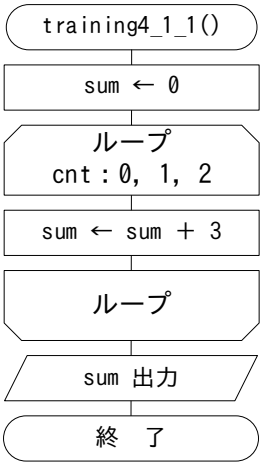
【擬似言語のトレース表】

行番号	判定結果	sum	cnt	出力結果
1	―	―	―	
2	―	領域確保	―	
3	―	未定義	領域確保	
4	―	0	未定義	
5	true	0	0	
6	―	2	0	
7	―	2	0	
5	true	2	1	
6	―	4	1	
7	―	4	1	
5	true	4	2	
6	―	6	2	
7	―	6	2	
5	false	6	3	
8	―	6	3	6

トレース表を確認すると，繰返し処理のたびに，カウンタ変数 cnt の値が１ずつ増えていき，集計変数 sum には２が足し込まれていくことがわかります。

練習問題 4-1-1

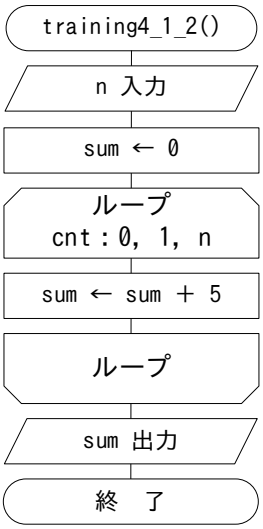
次の流れ図を，for～endfor 文を用いた擬似言語で記述してみましょう。また，トレース表を用いて，擬似言語をトレースしてみましょう。

流れ図	擬似言語
	1 ○training4_1_1() 2 　整数型：sum 3 　整数型：cnt 4 5 6 7 8

行番号	判定結果	sum	cnt	出力結果
1	―	―	―	

練習問題 4-1-2

次の流れ図を，for～endfor 文を用いた擬似言語で記述してみましょう。また，ｎに２が入力された場合の擬似言語の処理の流れを，トレース表を用いてトレースしてみましょう。

流れ図	擬似言語
	1 ○training4_1_2() 2　整数型：n 3　整数型：sum 4　整数型：cnt 5 6 7 8 9 10

行番号	判定結果	n	sum	cnt	出力結果
1	―	―	―	―	