hp 35s 関数電卓

日本語版ユーザーズガイド

Edition 1

HP part number　F2215AA-90001

Notice

REGISTER YOUR PRODUCT AT: www.register.hp.com

THIS MANUAL AND ANY EXAMPLES CONTAINED HEREIN ARE PROVIDED "AS IS" AND ARE SUBJECT TO CHANGE WITHOUT NOTICE. HEWLETT-PACKARD COMPANY MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THIS MANUAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT AND FITNESS FOR A PARTICULAR PURPOSE.

HEWLETT-PACKARD CO. SHALL NOT BE LIABLE FOR ANY ERRORS OR FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH THE FURNISHING, PERFORMANCE, OR USE OF THIS MANUAL OR THE EXAMPLES CONTAINED HEREIN.

Hewlett-Packard Company
16399 West Bernardo Drive
MS 8-600
San Diego, CA 92127-1899
USA

Printing History

Edition 1 February 2007

Japanese Edition 4 May 2008

もくじ

Part 1 基本操作

Part 2 プログラミング

Part 3 付録

Part 1

基本操作

1

はじめに（Getting Started）

このマークにはご注意下さい。このマークがあるときは、RPN モードでのキー入力例が記載されています。ALG モードでは異なるキー入力になります。

付録 C で ALG モードでの使用方法が記載されています。

重要な予備知識

電源のオン・オフ

電源を入れるには、 [C]キーを押します。 [C] キーの下に ON という字がプリントされているはずです。

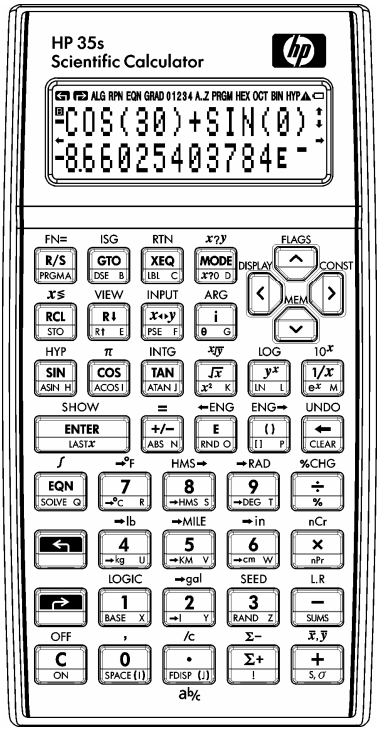
電源を切るには、[↰][C] と操作します。つまり、[↰](shift key, シフトキー)を押して離した後, [C]キーを押します ([C]キーの上には黄色で OFF という字がプリントされています)。この電卓にはコンティニアスメモリ(*Continuous Memory*)が搭載されており、電池が持続する限り、電源を切っても入力済みのデータに一切の影響はありません。

省エネのため 10 分間操作しないと自動的に電源がオフになります。電池がなくなりつつあることを示すマーク (▭) がディスプレイに出たら、すぐに電池を交換して下さい。電池の交換方法は付録 A をご参照下さい。

コントラスト調整

コントラスト(明るい部分と暗い部分との**輝度**の差)は、見る角度や光の具合、コントラスト設定により変化します。コントラストの調整は [C] キーを押しながら [+] と [−] でおこないます。

キーボードとディスプレイの主要部分

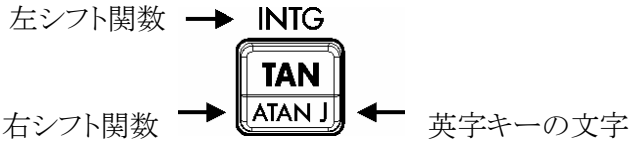


シフトキー

それぞれのキーには3つの機能が割り当てられています。キーの上に印刷されている機能、左シフト機能（黄色）、右シフト機能（青）の3つです。シフト機能は、それぞれのキーの上部に黄色で、下部に青で印刷されています。シフトキー（ か ）を押してから、目的のキーを押すとシフト機能が利用できます。たとえば、電源を切るには シフトキーを押して離した後、 キーを押します。

シフトキー や を押すと、それぞれの*表示記号* や がディスプレイ上部に表示されます。次のキーを押すまで有効です。シフトキーをキャンセルし、この表示を消すには、同じシフトキーをもう一度押して下さい。

英字キー（Alpha Keys）



上図のように、ほとんどのキーは右下に英字が表示されています。文字を入力するとき（たとえば、プログラムや変数にラベルをつけるとき）は、そのキーが「アクティブ」であることを示すための **A..Z** 表示記号が画面にあらわれます。

変数については第3章を、ラベルについては第13章をご参照下さい。

カーソルキー（Cursor Keys）

4つのカーソルキーには、矢印が表示されています。このマニュアルでは [>], [<], [^], [v] と表示します。

バックスペースとクリア

まず、入力のクリアと数値の訂正、ディスプレイに表示されたものを全てクリアする方法を解説します。

クリアするためのキー

キー	詳細
[←]	**バックスペース**。 式の入力途中であれば、[←] を押すと入力カーソル（_）位置の左側の文字を消去します。それ以外の場合、計算結果や入力済みの数値があれば、[←] を押すとその値がゼロに置き換わります。また、エラーメッセージやメニューが表示されていれば、それらがクリアされます。 プログラム入力モードや、式編集モードで [←] キーは下記のように機能します。 ■ 式編集モード（Equation–entry mode） 式の入力途中か編集中であれば、[←] キーは挿入カーソルの左側の文字を消去します。その他の場合、つまり、式の入力が完了して挿入カーソルが表示されていない場合は、[←] キーはその式の全てを消去します。 ■ プログラム入力モード（Program-entry mode） プログラム行が入力途中か、編集中であれば、[←] キーは挿入カーソルの左側の文字を消去します。その他の場合、つまり、プログラム行の入力が完了している場合は、[←] キーはその行の全てを消去します。
[C]	**クリア**または**キャンセル**。 表示中の数値をクリアし、ゼロに置き換えます。あるいは、現在の状況をキャンセルします（メニュー、メッセージ、プロンプト、カタログ、式入力モード、プログラム入力モードなど）

クリアするためのキー(続き)

キー	詳細
[↩] [CLEAR]	クリア(*CLEAR*)メニュー (x VARS ALL Σ STK CLVARx) には次のようなオプションがあります。すなわち、x(X-レジスタの数値)のクリア、全てのダイレクト変数をクリア、メモリをクリア、統計データのクリア、全てのスタックと間接変数のクリアです。 [3](3ALL)を選択すれば、メモリの全てをクリアする前に CLR ALL? Y N という確認メニューが出ます。 プログラム入力のときは、, 3ALL の代わりに 3PGM と表示されます。[3] (3PGM) を選択すると、全てのプログラムをクリアする前に CLR PGRMS? Y N という確認メニューが出ます。 式入力のときは、3ALL の代わりに 3EQN と表示されます。[3] (3EQN) を選択すると、全ての式をクリアする前に CLR EQN? Y N という確認メニューが出ます。 [6] (CLVARx) を選択すると、このコマンドが入力行にコピーされます。間接変数のアドレスを示す3個の数字を入力してください。このアドレスより大きな間接変数がクリアされます。 例: CLVAR056 はアドレスが 56 より大きい間接変数を全て消去します。

メニューの利用

キーボードに表示されたものより、もっと強力な機能が HP 35s にはたくさん用意されています。つまり、16 個のキーがメニューキーとして割り当てられており、16 個の追加メニューが利用できます。

HP 35s メニュー一覧

メニュー名	メニュー詳細	章
	数値関数	
L.R.	$\hat{x}$ $\hat{y}$ r m b 線形回帰(linear regression)： カーブフィッティング(curve fitting)と線形推定(linear estimation)	12
$\overline{x}$, $\overline{y}$	$\overline{x}$ $\overline{y}$ $\overline{x}_w$ 統計変数 x と y の相加平均(arithmetic mean)、統計変数 x の加重平均(weighted mean)	12
s,σ	s_x s_y σ_x σ_y 標本標準偏差(sample standard deviation)、母標準偏差(population standard deviation)	12
CONST	41 個の物理定数。 参考:「第4章 物理定数」	4
SUMS	n Σx Σy Σx^2 Σy^2 Σxy 統計データの和(**sum**mation**s**)	12
BASE	DEC HEX OCT BIN d h o b 基数変換(**base** conversions …10 進数:**dec**imal, 16 進数:**hex**adecimal, 8 進数:**oct**al, 2 進数:**bin**ary)	12
INTG	SGN INT÷ Rmdr INTG FP IP 符号(SGN, Sign)、整数除算の商(INT÷, Integer Division)、余り(Rmdr, Remainder from Division)、最大の整数(INTG, Greatest Ingeger)、小数部(FP, Fractional Part)、整数部(IP, Integer Part)	4,C
LOGIC	AND XOR OR NOT NAND NOR 論理演算子(Logic operators)	11

	プログラミング関数	
FLAGS	SF CF FS? セット、クリア、テストのフラグ	14
x?y	≠ ≤ < > ≥ = X と Y レジスタの比較	14
x?0	≠ ≤ < > ≥ = X レジスタとゼロとの比較	14
	その他の機能	
MEM	VAR PGM メモリの状態(有効なメモリのバイト数); 変数の目録、プログラム(ラベル)の目録	1, 3, 12
MODE	DEG RAD GRAD ALG RPN 角度モードと操作モード	4, 1
DISPLAY	FIX SCI ENG ALL . , 1,000 1000 x↔y x+yi r∠θ 固定小数点表記、科学表記、全浮動小数点表記; 位取り記号(. または ,); 複素数表記(RPN モードでは xiy と rθa のみ利用可能)	1
R↓ R↑	X Y Z T ALG モードでスタック(X, Y, Z, および T レジスタ)を表示させる	C
CLEAR	メモリの一部をクリアするための関数。 詳細は [→] [CLEAR] (P.1-5)の表を参照。	1, 3, 6, 12

メニュー関数の使いかた :

1. メニューキーを押し、メニューを表示させます
2. カーソルキー [▶] [◀] [▲] [▼] を使って目的のオプションに下線を移動させます
3. 目的のオプションに下線がある状態で [ENTER] キーを押します

メニューのオプションに数値が割り当てられている場合、[ENTER] キーの代わりにその数値を選択しても良いです。

CONST や SUMS のような、いくつかのメニューは複数ページにまたがっています。

この種のメニューでは⬆ や ⬇ という表示記号が出ます。カーソルキー [›] と [‹] を使って表示中のオプションを選択するか、[⌄] と [⌃] で次ページ・前ページに移動してください。

例:

この例では、DISPLAY メニューを使って固定小数点表記に変更し、6÷7 の計算をしてみます。最後に、全浮動小数点表記に変更します。

キー	表示	詳細
	0 0	初期表示
[↩] [DISPLAY]	1FIX 2SCI 3ENG 4ALL	DISPLAY メニューに入る
[1] or [ENTER]	FIX _	FIX コマンドが第２行にペーストされる
[4]	0.0000 0.0000	4 桁の固定小数点表示
[6] [ENTER] [7] [÷]	0.0000 0.8571	割り算を実行
[↩] [DISPLAY] [4]	0 8.57142857143E-	全浮動小数点表記

メニューを使うことで、機能や関数をインタラクティブに選択でき、関数の名前を覚えたり、キーボードから探す手間が省けます。

メニューの終了

メニューから選択が終了すると、上記の例のようにメニューは自動的に閉じます。選択前にキャンセルする場合は、次の操作を行ってください。

- ２レベル CLEAR や MEM メニューの場合、[←] キーを押すとひとつ前のメニューに戻ります。P. 1-5 の [↩] [CLEAR] を参照して下さい。
- その他のメニューは [←] または [C] でキャンセルします。

キー	表示
1 2 3 · 5 6 7 8	123.5678_
◀ DISPLAY	1FIX 2SCI ⬇ 3ENG 4ALL
← または C	123.5678_

■ 他のメニューキーを押すと古いメニューが新しいものに置き換わります。

キー	表示
1 2 3 · 5 6 7 8	123.5678_
◀ DISPLAY	1FIX 2SCI ⬇ 3ENG 4ALL
▶ CLEAR	1X 2VARS ⬇ 3ALL 4Σ
C	123.5678

RPN モードと ALG モード

計算式の入力方式は ALG (Algebraic、代数) モードと RPN (Reverse Polish Notation、逆ポーランド) モードから選択できます。

RPN モードでは、途中の計算結果が自動的にスタックに入るため入力式にカッコが不要となります。

ALG モードでは教科書通りの標準的な式が利用できます。

RPN モードを選択するには:

MODE 5 (5RPN) と操作します。RPN モードのときはディスプレイ上部に **RPN** と表示されます。

ALG モードを選択するには:

MODE 4 (4ALG) と操作します。ALG モードのときはディスプレイ上部に **ALG** と表示されます。

例:

1 + 2 = 3 を計算してみます。

RPN モードでは第1の数値を入力し、[ENTER]キーを押します。続いて第2の数値、最後に演算子[+]を入力して下さい。

ALG モードでは一般的な電卓と同じです。第1の数値、演算子[+]、第2の数値、最後に[ENTER]という順番になります。

RPN モード	ALG モード
1 [ENTER] 2 [+]	1 [+] 2 [ENTER]

ALG モードでは計算結果と入力式が表示されます。RPN モードでは結果のみが表示され、入力式は表示されません。

Note

ALG と RPN どちらも選択できますが、このマニュアルでは基本的にすべてのページで RPN を利用します。このマニュアルの欄外に " ✓ ✓ " が出ているところでは、RPN と ALG で操作が異なります。ご注意下さい。ALG モードの利用方法は付録 C で解説します。

アンドゥ(Undo)

Undo キー

Undo キーの機能は状況によって変化しますが、基本的には「前に戻る」というよりも、「削除したものを復活させる」ための機能です。数値関数が実行された後、ディスプレイ第2行のデータを復元する方法の詳細は、第2章「Last X レジスタ」をご参照下さい。[←] や [C] でデータが消去された後、[UNDO] と操作して次のデータを復元します。

- **削除したデータ**
- **式モードで削除した式**
- **プログラムモードで削除したプログラム行**

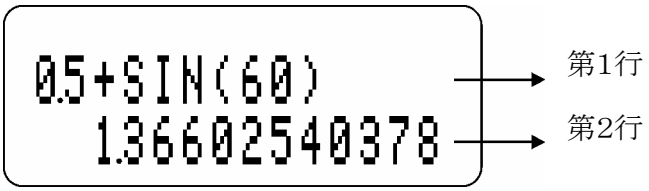
さらに、Undo は CLEAR メニューでクリアされたレジスタ値を復元させることもできます。Undo 操作は削除やクリア操作の直後でなければいけません。削除されたオブジェクトを復元させるとき、中間の演算も一緒に復元されます。また、Undo で復元したものをさ

らに編集することもできます。 ↶ UNDO と操作し、下記のものを再編集できます。

- ← で削除した、式の中の数値
- C でクリアした、編集途中の式
- 式モードやプログラムモードにおいて ← で削除した、式やプログラムの中の文字

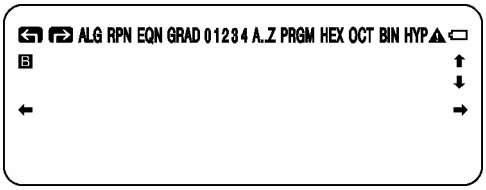
Undo 操作は利用できるメモリ量にも依存します。ご注意下さい。

ディスプレイと表示記号



ディスプレイは2行の表示行と表示記号から構成されています。

1行に 13 文字を超えて表示されるときは、左にスクロールします。入力データは ALG モードでは第1行、RPN モードでは第2行に表示されます。14 桁を超える場合は指数部が3桁までの E sign (指数表記)で表示されます。



上図に示すディスプレイの記号は、「表示記号」(*annunciators*)と呼びます。それぞれ、特定の意味を持ちます。

HP 35s 表示記号

表示記号	意味	章
B	"**B** (Busy)" は操作中、計算中、プログラム実行中にあらわれます。	
▲ ▼	分数表示モード(Fraction–display mode、[FDISP])で "▲" と "▼" の一方のみが表示されているときは、表示中の分数の厳密値が、内部で持っている数値よりも小さいか大きいかを表しています。"▲▼" のどちらも出ていなければ、分数の厳密値と内部の値が等しいことを表しています。	5
	左シフトがアクティブ	1
	右シフトがアクティブ	1
RPN	RPN モード	1, 2
ALG	ALG モード	1, C
PRGM	プログラム入力がアクティブ	13
EQN	式入力モードがアクティブ、または式を評価中、あるいは、計算を実行中	6
0 1 2 3 4	セットされたフラグ。(5 から 11 までのフラグの場合は表示されません)	14
RAD or **GRAD**	ラジアン(RAD)かグラジアン(GRAD)が角度モードに設定されています。デフォルト設定である度数法(DEG)のときは何も表示されません。	4
HEX OCT BIN	アクティブな数値の基数。10 進数(DEC, デフォルト値)のときは何も表示されません。	11
HYP	双曲線関数(hyperbolic)がアクティブ。	4, C

HP 35s 表示記号(続き)

表示記号	意味	章
⬅,➡	ディスプレイの第1行または第2行の左右に表示しきれない文字列があります。第1行で非表示になってしまった文字列があるときは、省略記号(...)が表示されます。RPN モードでは、[›] と [‹] キーを使ってスクロールさせて下さい。ALG モードでは [↩] [›] または [↩] [‹] と操作します。	1, 6
⬆,⬇	式リスト、変数一覧、プログラム行、メニューページ、並びにプログラム一覧を表示させるとき、[˅] および [˄] キーがアクティブであることを示します。	1, 6, 13
A..Z	英字キーがアクティブ	3
⚠	注意！ 特別な状態にあるか、エラーになっています。	1
▭	電池が少なくなっています	A

数値の入力

取扱可能な最大値は $\pm 9.999999999999^{499}$ です。計算結果がこの範囲を超えると "OVERFLOW" というエラーメッセージが表示記号⚠と一緒にしばらく表示されます。その後、このメッセージはオーバーフロー範囲に近い値で、かつ表示可能なものに置き換わります。

この最大値を超える値を入力すると、"INVALID DATA"と表示されます。このエラーメッセージをクリアしないと次の入力が行えません。

認識可能な最小値は $\pm 10^{-499}$ です。この範囲の値を入力するとディスプレイにゼロが表示されます。同様に、計算結果がこの範囲の値であるときもゼロと表示されます。

数値の符号を変更する

[+/−] キーで符号を変更できます。

- 負の値を入力するには、数値を入力した後で [+/−] キーを押します。
- ALG モードでは、数値の入力前に [+/−] キーを押しても良いです。

■ 数値の符号を再度変更するには、 [+/-] キーをもう一度押して下さい。(指数の場合は [+/-] キーは仮数、つまり指数ではない方の数の符号のみを変更します)

10 の指数

ディスプレイで表示される指数

10 のべき乗で表される数値(たとえば 4.2×10^{-5})は **E** を使って表示されます。4.2×10^{-5} の場合は 4.2**E**-5 となります。

表示可能な数値を超えると、指数表記に自動的に変更されます。

たとえば、表示形式が FIX 4 (小数点以下を4桁に固定)にしてこの振る舞いを確認してみましょう:

キー	表示	解説
[0] [.] [0] [0] [0] [0] [6] [2]	0.000062_	入力した数値
[ENTER]	0.0001	表示形式にあわせて丸めが発生
[0] [.] [0] [0] [0] [0] [4] [2] [ENTER]	4.2000E-5	丸めると数値がゼロになってしまうため、自動的に科学表記へ変更

10 の指数をキー入力

10 の指数を入力するには、 [E] キーを使います。たとえば、1000000(百万、ゼロが6個)を入力するときは [1] [E] [6] と操作します。下記の例もご参照下さい。

例:

プランク定数(Planck's constant、6.6261×10^{-34})を入力してみます

キー	表示	解説
[6] [.] [6] [2] [6]	0	仮数の入力
[1]	6.6261_	
[E]	0	$\times10^{x}$ と同意義
	6.621E_	

3 4 +/- ENTER	6.621E-34 6.621E-34	指数を入力

前述の「百万」のように、仮数が1のときは 1 E に続けて指数を入力します。

その他の指数関数

10 の指数や常用対数(つまり底が 10)を計算するには、 ↰ 10^x と操作します。その他の指数や対数は y^x を使います(詳細は第4章)。

入力カーソルの挙動について

数値を入力すると、カーソル (_) と空白がディスプレイにあらわれます。数値を入力すると、点滅したカーソル (_) がディスプレイにあらわれます。カーソルは次の数値がどこに入るか、また、その入力が完了していないことを示しています。

キー	表示	解説
1 2 3	123_	入力は完了しておらず、数値は未確定です。

計算するために**関数を実行する**と、数値が確定されてカーソルは消えます。

✓	$\sqrt{x}$	11.0905	入力が完了。

ENTER キーを押すと入力が完了します。2つの数値を入力するときは ENTER キーで分割します。最初の数値を入力後 ENTER キーを押し、次の数値を入力して下さい。

✓	1 2 3 ENTER	123.0000	入力完了した数値
✓	4 +	127.0000	第２の数値

入力が完了していない(カーソルが表示されている)ときは、バックスペースキー ← で最後の数字を削除できます。入力が完了したら(カーソルが非表示)、 ← は C と同等に作用し、数値全体をクリアします。お試し下さい。

数値の範囲とオーバーフロー

取扱可能な最小値は-9.99999999999 × 10^{499} で、最大値は 9.99999999999 × 10^{499} です。

■ 計算結果がこの範囲を超えると、値は -9.99999999999 × 10^{499} または 9.99999999999 × 10^{499} となり、OVERFLOW が表示されます。

四則演算と一般的な関数

HP 35s には RPN モードと ALG モードが用意されています。それぞれのモードにより入力方式が変化します。ここでは RPN と ALG モードの相違点を、単一引数の場合と2つの引数の場合で解説します。

単一の引数または単項演算子

いくつかの演算は単一の引数をとるものがあります。たとえば [1/x]、[x²]、[LN]、[SIN] です。これらの演算は RPN か ALG モードかによって入力方法が異なります。RPN モードでは、最初に引数となる数値を入力し、演算のキーを押します。数値を入力してから [ENTER] キーを押し、更に演算のキーを押すと、ディスプレイの第1行に入力した数値が、第2行に計算結果が表示されます。[ENTER] キーを押さずに演算キーを押すと、第1行には変化が無く、第2行目に計算結果が表示されます。ALG モードでは、最初に演算キーを押すと演算の関数とともにカッコが表示されます。引数となる数値をカッコの中に入力し、[ENTER] キーを押してください。第1行に式が、第2行に計算結果が表示されます。次の例で相違点を解説します。

例:

RPN モードと ALG モードでそれぞれ、3.4^2 を計算します。

キー	表示	解説
[MODE] [5] (5RPN)		RPN モードに変更（必要であれば）
[3] [.] [4]	0 3.4	数値を入力
[↶] [x²]	0 11.56	２乗を計算する演算キー

MODE 4 (4ALG)		ALG モードに変更
↷ x^2	SQ()	２乗を計算する演算キー
3 . 4	SQ(3.4)	カッコの間に数値を入力
ENTER	SQ(3.4)	計算するために ENTER キーを押す
	11.56	

この例では2乗の演算キーは x^2 ですが、SQ() と表示されます。ALG モードではこれと同じように、キーとディスプレイ表示が異なるものがあります。このような演算キーを下表にまとめます。

キー	RPN と RPN プログラム	ALG、式、ALG プログラム
x^2	X^2	SQ()
$\sqrt{x}$	$\sqrt{}x$	SQRT()
e^x	e^x	EXP()
10^x	10^x	ALOG()
$1/x$	1/x	INV()

２つの引数または二項演算子

2つの引数をとる演算には、+、÷、y^x、nCr などがあります。単項演算子と同様に、RPN と ALG モードで入力方法が異なります。RPN モードでは、最初の数値を入力してから第二の数値を x レジスタに入れ、二項演算子を入力します。ALG モードでは、伝統的な中置記法(infix notation)と、それぞれの演算に対応した方式の2種類があります。相違点を次の例で解説します。

例

RPN と ALG モードでそれぞれ、2+3 および ${}_6C_4$ を計算します。

キー	表示	解説
MODE 5 (5RPN)		RPN モードに変更（必要であれば）
2 ENTER 3	2 3_	2 を入力し、ENTER で確定。3 を x レジスタに入力。注意：カーソルが 3 の後ろに出たままにします。3 の後に ENTER は押さないで下さい！
+	0 5	加算キーを押す

6 ENTER 4	6 4_	6 を入力し、ENTER で確定。4 を x レジスタに入力。
◀ nCr	5 15	組合せ(combination)キーを押し、計算を実行。
MODE 4 (4ALG)		ALG モードに変更
2 + 3 ENTER	2+3 5	式と結果が表示されます。
◀ nCr	nCr(,)	組合せキーを押す
6 ▶ 4	nCr(6,4)	6 を入力し、カーソルキーでカンマの右に移動。続いて 4 を入力。
ENTER	nCr(6,4) 15	計算を実行

ALG モードで中置記法を用いる演算子は +, −, ×, ÷, y^x です。その他の演算で2つの引数をとるものは f(x,y) という関数になります。x は第1引数、y は第2引数です。RPN モードではスタックの X と Y が2つの引数として用いられます。つまり、関数 f(x,y) においてx レジスタの値が x、y レジスタの値が y となります。

y の x 乗根($\sqrt[x]{y}$) はこのルールの例外です。たとえば RPN モードで $\sqrt[3]{8}$を計算するときは 8 ENTER 3 ◀ $\sqrt[x]{y}$ と操作します。ALG モードでは ◀ $\sqrt[x]{y}$ 3 ▶ 8 ENTER となります。

単項演算子の場合と同様に、いくつかの演算は RPN モードと ALG モードで表示が異なります。これを下表にまとめます。

キー	RPN と RPN プログラム	ALG、式、ALG プログラム
y^x	y^x	^
$\sqrt[x]{y}$	$x\sqrt{\ }y$	XROOT(,)
INT÷	INT÷	IDIV(,)

結合法則をもつ + や × のような演算子は、引数の順番は計算結果に影響しません。その他の演算で引数の順番を間違えてしまったとき、RPN モードであれば $x \leftrightarrow y$ キーで x レジスタと y レジスタを入れ替えて修正できます。詳細は第2章の「スタックの X レジスタと Y レジスタを交換」をご参照下さい。

表示形式の制御

全ての数値は 12 桁の精度で保存されます。ただし、表示桁は Display メニュー [↩] [DISPLAY] で変更することができます。最初の4つのオプション (FIX, SCI, ENG, ALL) はディスプレイの表示桁を指定できます。このとき、数値は丸め処理されてから表示されます。内部で複雑な計算が行われるときは、中間結果が15桁の精度になることもあります。

固定小数点表記(Fixed–Decimal Format、FIX)

FIX 表記では小数部(小数点の右側の数値)の桁を11までに指定できます。プロンプト FIX_ に続けて桁を指定して下さい。10 桁や 11 桁の場合はそれぞれ、[.][0] や [.][1] と指定します。

たとえば、123.456.7089 では "7", "0", "8", "9" が小数です。FIX 4 かそれ以上の桁にすれば全て表示されます。

表示可能な範囲(最大 10^{11}、最小 10^{-11})を超えたときは、自動的に科学表記になります。

科学表記(Scientific Format、SCI)

科学表記では整数部(小数点の左側)の桁が 1 となり、小数部は 11 桁まで指定でき、指数部は 3 桁までとなります。プロンプト SCI_ に続けて小数部の桁を指定して下さい。10 桁や 11 桁の場合はそれぞれ、[.][0] や [.][1] と指定します。

たとえば、1.2346E5 であれば "2", "3", "4", "6" が小数部となり、SCI 4 にすれば全ての桁が表示されます。"E" の次の "5" は 10 の指数をあらわし、この表示は 1.2346×10^5 を意味します。

12 桁以上の計算のとき、12 桁を超える桁の精度は維持されません。

工学表記(Engineering Format、ENG)

工学表記は科学表記に近いですが、指数部が 3 の倍数になるという点が異なります。よって整数部(小数点の左側)は 3 桁までになります。メガやキロ、ミリなど、10^3 で単位が決まっているときは便利です。

プロンプト ENG_ に続けて、最初の有効数字に続く有効数字の数(つまり、表示される桁より 1 小さい桁)を指定して下さい。10 桁や 11 桁の場合はそれぞれ、[.][0] や [.][1] と指定します。

たとえば、123.46E3 であれば "2", "3", "4", "6" が「最初の有効数字に続く有効数字」となり、SCI 4 にすれば全ての桁が表示されます。"E" の次の "3" は 10 の指数をあらわし、この表示は 123.46 × 10^3 を意味します。

[↰] [←ENG] または [↰] [ENG→] と操作すると、指数が 3 の倍数で変化し、仮数(指数以外の部分)もそれにつれて調整されます。

例:

ここでは 12.346E4 という数値を使って工学表記と、[↰] [←ENG] ならびに [↰] [ENG→] 操作について解説します。入力形式は RPN モードとします。

キー	表示	解説
[↰] [DISPLAY] [3] (3E ENG)	ENG_	工学表記を選択
[4]	0.0000E0	4 を入力(第 1 桁に続く有効桁が 4)
	0.0000E0	
[1] [2] [.] [3] [4] [6]	123.46E3	数値 12.346E4 を入力
[E] [4] [ENTER]	123.46E3	
[↰] [←ENG] or	123.46E3	
[↰] [ENG→]	123.46E3	
[↰] [←ENG]	123.46E3	指数を 3 の倍数で増加
	0.12346E6	
[↰] [ENG→]	123.46E3	指数を 3 の倍数で減少
	123.46E3	

全表記 (ALL)

全表記はデフォルト設定です。12 桁までの精度で全ての桁を表示します。表示できないときは自動的に科学表記で表示されます。

位取り: ピリオドとコンマ (.) (,)

可読性向上のため、HP 35s では位取りの記号をコンマまたはピリオドに指定できます。また、位取り無しでも表示できます。次の例で解説します。

例

数値 12,345,678.90 を入力し、小数点をカンマに変えます。次に位取り記号を消してみます。最後に、デフォルト設定に戻します。ここでは RPN モードを用います。

キー	表示	解説
↶ DISPLAY 4 (4 ALL)		全表示を選択（ALL）
1 2 3 4 5	12,345,678.9	デフォルト設定では位取りにカンマ、小数点にピリオドとなっています。
6 7 8 . 9 ENTER	12,345,678.9	
↶ DISPLAY 6 (6 ,)	12,345,678.9	小数点をカンマに変更します。位取りは自動的にピリオドになります。
	12,345,678.9	
↶ DISPLAY 8 (8 1000)	12345678.9 12345678.9	位取り記号を消します。
↶ DISPLAY 5 (5 .)	12,345,678.9	デフォルト形式に戻します。
↶ DISPLAY 7 (7 1,000)	12,345,678.9	

複素数(Complex number)の表示形式 (x i y, x+y i, r θ a)

複素数は x i y, x+y i, r θ a 形式で表示できます。ただし、x+y i 形式は ALG モードでのみ利用できます。次の例では、複素数 3+4i をそれぞれの形式で表示させてみます。

例

複素数 3+4i をそれぞれの形式で表示

キー	表示	解説
MODE 4 (4 ALG)		ALG モード
3 i 4 ENTER	3 i 4	複素数を入力。デフォルト形式の 3i4 で表示されます。
	3 i 4	
↶ DISPLAY .	3 i 4	x+yi 形式に変更
1 (1 1 x+y i)	3+4 i	

[↩] [DISPLAY] [.]	3i4	rθa 形式に変更。半径は 5 で角度は
[0] (10r8a) or	5053.1301023542	約 53.13° です。
[↩] [DISPLAY] [^]		
[^] [>] [ENTER]		

12 桁の全表示

表示桁の変更により、ディスプレイに出ている桁が変化しますが、内部精度は変わりません。全ての数値は常に、12 桁の精度で保存されています。

たとえば数値 14.8745632019 を入力したとき、表示設定が FIX 4 であれば "14.8746" と表示されますが、最後の 6 桁 ("632019") は内部に存在しています。

一時的に数値を全精度で表示させるには、[↩] [SHOW] と操作します。この操作では仮数部(指数以外)のみが表示されます。[SHOW] ボタンを離すと消えます。

	キー	表示	解説
✓	[4] [5] [ENTER] [1] [.] [3] [×]	58.5000	計算を実行 (FIX 4)
	[↩] [DISPLAY] [2] (2SCI) [2]	5.85E1	科学表記： 小数部２桁と指数
	[↩] [DISPLAY] [3] (3ENG) [2]	58.5E0	工学表記
	[↩] [DISPLAY] [4] (4ALL)	58.5	全表記。最後のゼロは省かれます。
	[↩] [DISPLAY] [1] (1FIX) [4]	58.5000	固定小数点表記 4 桁。指数はありません。
✓	[1/x]	0.0171	58.5 の逆数
	[↩] [SHOW] (hold)	170940170940	[SHOW] ボタンを離すまで全精度で表示

分数(Fractions)

HP 35s では分数の入力と表示ができます。小数と分数を切り替えて表示することもできます。分数の表示は a b/c となります。a は整数、b と c は正の整数です。ただし、b は 0≤b<c という条件を、c は 1<c≤4095 を満たすものとします。

分数の入力

分数は次の方法でいつでもスタックに入力できます。

1. 整数部を入力し、[.]を押します。この[.] は分数と整数部を分割するためのものです。
2. 分子を入力し、再度[.]を押します。この[.] は分子と分母を分割します。
3. 分母を入力し、[ENTER] か演算キーを押します。入力した数値または計算結果がディスプレイに表示されます。

[.]キーの下に赤で印刷された *a b/c* は、このキーが分数の入力に使われることを忘れないための措置です。

次の例では分数の入力と表示を解説します。

例

帯分数 12 3/8 を入力し、分数形式と小数形式で表示させます。さらに 3/4 を加算します。ここでは RPN モードを用います。

キー	表示	解説
[1][2][.][3]	0 12.3	通常通り小数点が表示されます。
[.][8]	0.0000 12 3/8_	２度目の[.] を押すと、ディスプレイが分数モードに切り替わります。
[ENTER]	12.3750 12.3750	入力が完了すると現在の表示モードに戻ります。
[→][FDISP]	12 3/8 12 3/8	分数表示モードに変更
[.][3][.][4]	12 3/8 0 3/4_	3/4 を入力。整数部が無いため、最初の入力は [.] です。(0 ¾ と入力することもできます)
[+]	0 13 1/8	12 3/8 に 3/4 を加算
[→][FDISP]	0 13.1250	元の表示モードに戻します。

分数についての詳細は「第5章　分数」をご参照下さい。

メッセージ

エラーが発生すると表示記号 ⚠ が出てきます。通常は表示記号と一緒にエラーメッセージも表示されます。

- **メッセージをクリアするには [C] か [←] を押します。RPN モードではエラーの発生前のスタックに戻ります。ALG モードでは最後の式に戻り、エラーを引き起こした位置に編集カーソルが移動します。**
- **その他のキーでもメッセージはクリアされますが、クリアされるだけであり、キーの機能は動作しません。**

メッセージの表示なく表示記号 ⚠ のみが出る場合は、その状況にあっていないキーが押されています。たとえば、[·] [·] と操作すると ⚠ が表示されます。2回目の [·] はこの状況では意味をなさないからです。

全てのメッセージを「付録 F メッセージ」に掲載しました。

電卓のメモリ

HP 35s に搭載されたメモリは全てのデータ（変数、式、プログラム）をあわせて 30KB です。

メモリの状況を調べる

[↶] [MEM] と操作すると次のメニューが表示されます。

```
1VAR    2 PGM
nnn   mm,mmm
```

ただし:

nnn 間接変数の量

mm,mmm 利用できるメモリのバイト数

1 (1VAR)を押すとダイレクト変数が表示されます（参考：「第３章　VAR 一覧による変数のレビュー」）。2 (2PGM)を選択すればプログラム一覧が表示されます。

1. 変数一覧を表示させるには1 (1VAR)を押します。プログラム一覧は2 (2PGM)です。
2. それぞれ ⌄ や ⌃ でスクロールできます。
3. 変数やプログラムを削除するには、表示中に ↩ CLEAR と操作します。
4. 一覧の表示を終了するにはCを押します。

全てのメモリをクリア

全てのメモリを削除すると、数値、式、プログラムが全てクリアされますが、モードと表示形式の設定には影響はありません。（設定のクリアは「付録 B メモリのクリア」をご参照下さい。）

全てのメモリをクリアするには：

1. 4 (4ALL)を選択します。誤操作をさけるため、確認のプロンプト CLR ALL? Y N が出ます。
2. ＜ (Y) ENTER と操作します。

2

RPN: メモリスタックの自動処理

ここでは RPN モードでメモリスタックがどのように動作しているかを解説します。単に電卓を使用するだけであれば、この章の内容を理解する必要はありません。しかし、利用方法の幅が大きく広がりますし、特にプログラミングのときは役立ちます。

「Part 2 プログラミング」に、プログラム中のデータ処理とスタックとの関係が記載されています。

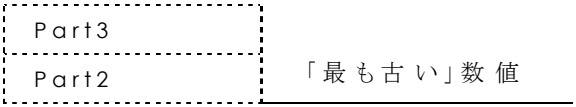
スタックとは

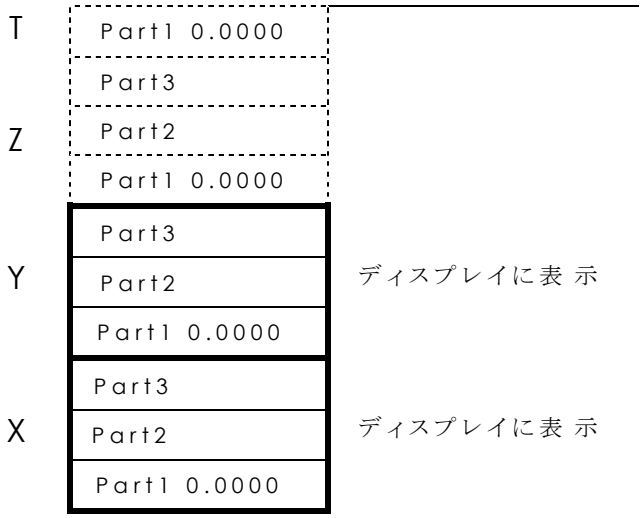
HP 35s では、込み入った計算における式中のカッコ無しに、簡単に計算できます。これは「計算結果の迅速かつ自動的な保存」という機能のおかげです。つまり、**RPN のメモリスタックを自動保存**という機能がキーポイントになります。

HP 電卓の操作ロジックはポーランドの論理学者ヤン・ウカシェヴィチ (Jan Łukasiewicz, 1878-1956) の「ポーランド方式」がベースになっています。

一般的な方式の場合、式中の変数や数値の**間に**演算子が適時追加されます。これに対し、ウカシェヴィチの方法は変数・数値の**前に**演算子を配置します。スタックを最大限効率的に利用するため、HP はこの方法を変更して演算子を数値の**後に**配置することにしました。よってこの方法を「逆ポーランド方式（*RPN, Reverse Polish Notation*）」と呼んでいます。

スタック（stack、積み重ねる）は 4 個の保存場所から構成されています。この保存場所をレジスタ（register、記録）と呼びます。レジスタはスタックに「積み重ね」て保存されます。4 個のレジスタは順番に X, Y, Z, T という名前がついています。「最も古い」数値は T (*top*) レジスタに保存されます。スタックは電卓の作業領域とも言えます。





「最も新しい」数値は X レジスタに存在し、ディスプレイの第2行に表示されます。

それぞれのレジスタは次の3部分に分かれています。

- **実数と 1 次元ベクトルの場合、**part 1 **のみが使われます。**part 2 **と** part 3 **は** null **値になります。**
- **複素数と 2 次元ベクトルの場合、**part 1 **と** part 2 **が使われます。**part 3 **は** null **値になります。**
- **3 次元ベクトルの場合、**part 1**、**part 2**、**part 3 **の全てが使われます。**

プログラムの計算実行時には、中間結果を一時保存したり、プログラムやサブルーチン間でのデータ(変数)のやりとりや、入力の受付、出力の受け口にするためにスタックが用いられます。

X と Y レジスタのディスプレイ表示

X と Y レジスタは基本的にディスプレイに表示されている数値ですが、メニュー、メッセージ、式、プログラム行が表示されている場合は除きます。いくつかの関数には x や y を含む名前のものがあります。

このような関数はそれぞれ、X および Y レジスタを利用します。たとえば、↶ 10^x は 10 のべき乗を X レジスタの数値で計算します。

X レジスタのクリア

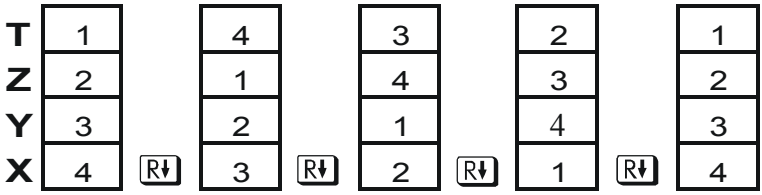
[f] [CLEAR] [1] (×) と操作すると**常に** X レジスタがクリアされ、ゼロになります。プログラム中でも、X レジスタをクリアするときはこの操作を行います。[C] キーはこれと対照的に、現在の表示をクリアしたり、キャンセルしたりと、状況に依存した動作となります。X レジスタが表示されているときであれば [f] [CLEAR] [1] (×) と同じ動作です。また、[←] も状況に依存した機能で、X レジスタが表示されており、**かつ**、入力が完了していれば（カーソルが出ていない状態であれば） [f] [CLEAR] [1] (×) のような動作をします。

スタックの閲覧

R↓ (Roll Down、ロールダウン)

[R↓] *(roll down)* **キー**はスタック中の全てのレジスタの中身を下方向に回転させます。1回押すとレジスタの中身が1個分回転します。これにより、X および Y レジスタを通してレジスタ中の数値を確認できます。

スタックに 1, 2, 3, 4 を入れておきます([1] [ENTER] [2] [ENTER] [3] [ENTER] [4])。[R↓] を4回押すと、図のようにすべての数値を確認できます。

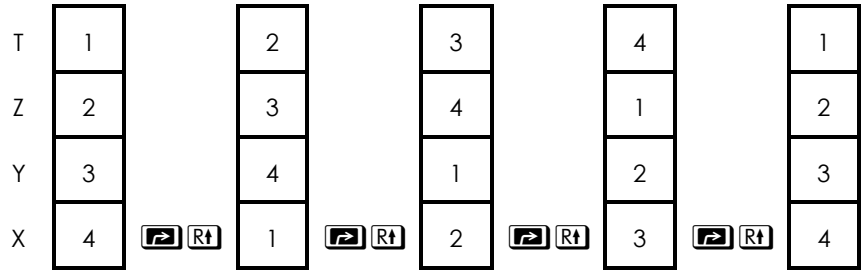


X レジスタの数値が T レジスタへ、T レジスタの数値が Z レジスタへというように回転します。レジスタの**中身**だけが回転し、レジスタ自身は移動しないことに注意して下さい。また、表示されているのは X と Y レジスタだけで、Z と T レジスタは表示されていません。

R↑ (Roll Up、ロールアップ)

[f] [R↑] *(roll up)* キーは [R↓] と同様の機能ですが、回転させる方向が「上」になります。これも、1操作につき1個のレジスタの中身が回転します。

X レジスタの数値が Y レジスタへ、T レジスタの数値が X レジスタへ、というように回転します。



X レジスタと Y レジスタを交換

スタックの位置を操作するもう一つの機能が [x⇄y] キーです。このキーは X と Y レジスタの中身を入れ替えますが、その他のレジスタには影響しません。[x⇄y] を2回押すと元のレジスタに戻ります。

[x⇄y] キーは、計算の過程で使われることが多いです。
たとえば、9 ÷ (13 × 8) を考えてみます。ひとつの方法は
[1] [3] [ENTER] [8] [×] [9] [x⇄y] [÷]
ですが、[1] [3] [ENTER] [8] [×] [9] [x⇄y] [÷] とすることも可能です。
なお、「左から右」に入力する方式であれば
[9] [ENTER] [1] [3] [ENTER] [8] [×] [÷] となります。

Note

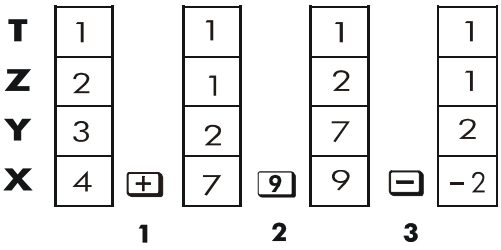
スタックに保存されるのは常に 4 個の数値のみであるという点にご注意下さい。T レジスタ（レジスタ最上部）の数値は 5 番目の数値が入力されると消えます。

計算時のスタック

スタックの中身は自動的に上下します。X レジスタに新しい数値が入力されたときは上方向に移動し、X レジスタと Y レジスタの数値を使って演算されたときは下方向へ移動し、X レジスタには計算結果が入ります（スタックが「落ち」ます）。

たとえば、スタックが 1, 2, 3, 4 であるとします。計算を実行するとスタックは図のように上下します。

3+4−9



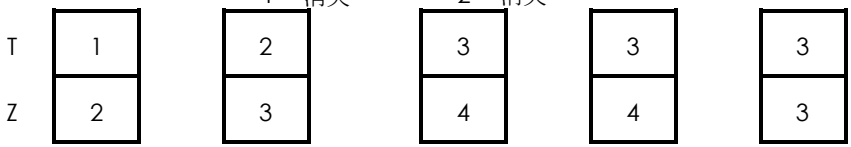
1. スタックの中身が「落ち」ます。それにつれて、T(top)レジスタが Z レジスタに**コピー**されます。
2. スタックが「上がり」ます。T レジスタの数値は消えます。
3. スタックが落ちます

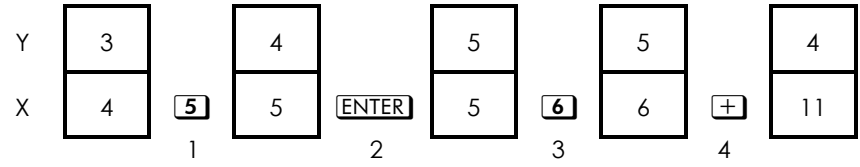
- スタックが上がるとき、T レジスタには Z レジスタの中身がコピーされ、それまでの T レジスタの中身は消えます。スタックは 4 個の数値のみが保存されるからです。
- スタックが自動的に移動するため、新しい計算を行うときでも、多くの場合、X レジスタを明示的にクリアする必要はありません。
- ほとんどの演算は、X レジスタに新たな数値が入力されるとスタックが上がるようになっています。スタックの上昇が発生しない演算については「付録 B」をご参照下さい。

ENTER の動作

数値を入力した後、ENTER キーを押すことで次の数値を入力できるようになります。このときスタックはどのように動作しているでしょうか？ 再度、スタックに 1, 2, 3, 4 が入っているものとして考えてみましょう。さらに、新しい2つの数値を足し算(5+6)してみます。

5+6





1. スタックの上昇
2. スタックの上昇と、X レジスタのコピー
3. ここではスタックの上昇は**発生しません**
4. スタックの下落と T レジスタのコピー

ENTER キーにより、X レジスタの数値が Y レジスタにコピーされます。その次に入力（あるいは再呼び込み…recall）する数値は、最初の数値が残っている X レジスタに上書きされます。これにより、2 個の数値を別々に入力することができます。

この ENTER によるコピー機能を使って、スタックを迅速にクリアできます。0 ENTER ENTER ENTER と操作して下さい。全てのレジスタがゼロになります。ただし、必ずしも計算の前にスタックをクリアしなければならない、というわけではありません。

同じ数値を再利用する

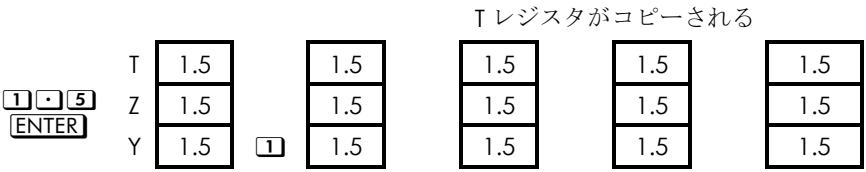
ENTER によるコピー機能を使えば、他にも便利な機能が利用できます。たとえば同じ数値を足し算するときは、その数値を入力したあと、ENTER + と操作します。

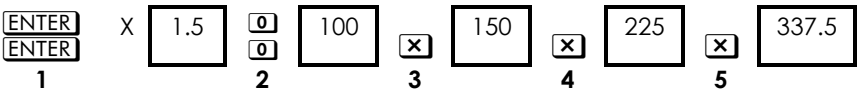
スタックを全て同じ数値で満たす

ENTER によるコピー機能とスタックの下降（T から Z へ）を一緒に使うと、スタックをある数値で満たすことができます。

例：

あるバクテリアは1日で 50%増殖します。最初 100 匹だったら、3 日目の終わりの時点で何匹になっていますか？





1. スタックを増殖率で満たします
2. 最初の個体数を入力
3. １日目の個体数を計算
4. ２日目の個体数を計算
5. ３日目の個体数を計算

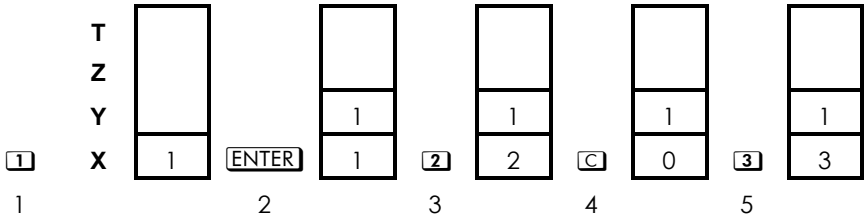
スタックのクリア

X レジスタをクリアすると、X レジスタにはゼロが入ります。次に数値キーを入力するか、数値を再呼び出し(recall)すると、このゼロに上書きされます。

X レジスタをクリアするには4つの方法があります。

1. C を押す
2. ← を押す
3. f CLEAR 1 (1×) と操作する(プログラムの時に使います)
4. f CLEAR 5 (SSTK) と操作すると、全てのレジスタがクリアされます

たとえば1と3を入力したかったのに1と2を入力してしまったとします。次の操作で訂正できます。



1. スタックの上昇
2. スタックの上昇と X レジスタのコピー
3. X レジスタに上書き

4. X レジスタをクリアしたことでゼロが入る
5. X レジスタに上書き入力

LAST X レジスタ

LAST X レジスタはスタック操作用の便利な機能で、直前の数値演算が実行されたときの X レジスタを保持しています。(数値演算とは[√x]のように、ある数値をもとに計算結果を生み出す操作です。) [↩] [LAST x] により、この値を X レジスタに戻します。

この機能は次のような場合によく使われます。

1. エラーの訂正
2. 数値の再利用

付録 B に LAST X レジスタに X レジスタの値を保存する演算の一覧があります。

LAST X を使ったエラーの訂正

単項演算子の計算を訂正

単項演算子の計算において間違って入力したら、[↩] [LAST x] で計算に使っていた数値を回復させることができます。(誤操作による計算結果を完全にスタックから除去したければ、その前に [C] でクリアして下さい)

[↩] [%] と [↰] [%CHG] は単項演算子ではありませんが、スタックの下降が発生しないため、単項演算子と同じように訂正できます。

例:

ある計算結果 $4.7839 \times (3.879 \times 10^5)$ の平方根を計算したかったのに間違えて[e^x]を押してしまったとします。再計算するには手間がかかります。このようなときは、[↩] [LAST x] [√x] と操作して下さい。

二項演算子の計算を訂正

二項演算子 ([+], [y^x], [nCr] など) での誤操作は、[↩] [LAST x] とその演算子の逆の演算で訂正できます。

1. [↩] [LAST x] で第 2 引数 (つまり直前の操作の x) を復活させます。

2. 逆の演算を行います。これにより元の第１引数が出てきます。元の第２引数は LAST X レジスタに入っています。よって：
   - **間違った演算**を実行したときは: [↩] [LAST x] で元のスタックの内容に戻したうえで正しい演算を実行します。
   - **第２引数**を間違ったときは、正しいものを入力して演算を再度実行します。

第１引数を間違ったときは、正しい第1引数を入力してから [↩] [LAST x] で第2引数を再度呼び込み、演算を再度実行します。(誤操作による計算結果はスタックに残ります。クリアするにはこの操作の前に [C] を押します。)

例:

次の計算を誤操作したと考えて下さい。

$$16 \times 19 = 304$$

誤操作には次の3種類があります。

間違った計算の例	ミスの種類	訂正方法
[1] [6] [ENTER] [1] [9] [−]	演算子の誤入力	[↩] [LAST x] [+] [↩] [LAST x] [×]
[1] [5] [ENTER] [1] [9] [×]	第１引数の誤入力	[1] [6] [↩] [LAST x] [×]
[1] [6] [ENTER] [1] [8] [×]	第２引数の誤入力	[↩] [LAST x] [÷] [1] [9] [×]

LAST X で数値を再利用

直前の数値を再利用するときも [↩] [LAST x] が便利です。直前の計算の第2引数、つまり X レジスタに入っていた数値のみがこのボタンで再利用できます。

例:

$$\frac{96.704 + 52.3947}{52.3947}$$ を計算して下さい

9 6	T	t	5 2	t		t
· 7	Z	z	· 3	z		t
0 4	Y	96.7040	9 4	96.7040		z
ENTER	X	96.7040	7	52.3947	+	149.0987
	LAST X	/		/	+	52.3947

	T	t	t
	Z	z	t
↻	Y	149.0987	z
	X	52.3947	÷	2.8457
LAST x				
	LAST X	52.3947	52.3947

キー	表示	説明
9 6 · 7 0 4 ENTER	96.7040	最初の数値を入力
5 2 · 3 9 4 7 +	149.0987	中間結果
↻ LAST x	52.3947	+演算のときの X レジスタを再呼び込み
÷	2.8457	最終結果

例:

太陽以外で、地球から近い距離にある恒星はリゲル・ケンタウルス(距離は 4.3 光年)とシリウス(距離は 8.7 光年)です。光の速度 c (9.5×10^{15} m/年) を使って、これらの距離をメートル表記に変換して下さい。

リゲル・ケンタウルス: 4.3 光年 × (9.5×10^{15} m/年)

シリウス: 8.7 光年 × (9.5×10^{15} m/年)

キー	表示	詳細
4 . 3 ENTER	4.3000	リゲル・ケンタウルスまでの距離
9 . 5 E 1 5	9.5E15_	光速 c
×	4.0850E16	リゲル・ケンタウルスの距離をメートルへ変換
8 . 7 ↻ LAST x	9.5000E15	光速 c を再度呼び出す
×	8.2650E16	シリウスの距離をメートルへ変換

RPN モードでの連鎖計算

RPN モードではスタックの自動的な上昇・下降により、中間結果の保持が簡単になります。カッコを使ったり、明示的に保存しておく必要も、再入力する必要もありません。

カッコ無しでの作業

(12 + 3) × 7 という計算を考えてみます。

紙に書いて計算するときは最初に(12 + 3)の中間結果を求めます。
（皆さんもそうですよね？）

$$(12 + 3) = 15$$

この中間結果に 7 をかけ算します。

$$(15) \times 7 = 105$$

HP 35s の RPN モードでカッコ無しに入力するにはどうしたらよいでしょうか？

キー	表示	解説
1 2 ENTER 3 +	15.0000	まず中間結果を求めます

中間結果が出た後で ENTER キーを押す必要はありません。**計算された**結果であるため、自動的にスタックへ保存されるからです。

キー	表示	解説
7 ×	105.0000	演算キーを押すと計算が実行されます。この結果も、のちの計算で利用できます。

次の例に進みます。ENTERキーは連続した数値を分割する場合に**のみ**必要になるということに注意して下さい。たとえば式を入力し始めるときなどです。演算子(+、 −など) も数値を分割します。演算キーを押すと計算結果がスタックに保存されます。この計算結果は次の計算で利用できます。

2 ÷ (3 + 10)を計算して下さい:

キー	表示	解説
3 ENTER 1 0 +	13.0000	(3 + 10) をまず計算
2 x⇔y ÷	0.1538	13 を入力して順番を変更： 2÷13 にするため。

4 ÷ [14 + (7 × 3) − 2] を計算して下さい:

キー	表示	解説
7 ENTER 3 ×	21.0000	(7 × 3)を計算
1 4 + 2 −	33.0000	分母を計算
4 x⇔y	33.0000	分子の 4 を中間結果 33 の前に配置
÷	0.1212	4÷33 を実行

複数のカッコがある式の入力でも、これまでと同じ方法で解決できます。たとえば紙で(3 + 4) × (5 + 6)を計算するときのことを考えてみます。まず(3 + 4) を計算します。次に(5 + 6)を計算します。最後に、それぞれの中間結果をかけ算します。

HP 35s でも同じです。ただし、中間結果を書き残しておく必要はありません。

キー	表示	解説
3 ENTER 4 +	7.0000	(3+4)を計算
5 ENTER 6 +	11.0000	(5+6)を計算
×	77.0000	中間結果同士をかけ算

練習問題 1

問題 1:

$$\frac{\sqrt{(16.3805 x 5)}}{0.05} = 181.0000$$

解答 1:

[1] [6] [·] [3] [8] [0] [5] [ENTER] [5] [×] [√x] [·] [0] [5] [÷]

問題 2:

$$\sqrt{[(2+3)\times(4+5)]} + \sqrt{[(6+7)\times(8+9)]} = 21.5743$$

解答 2:

[2] [ENTER] [3] [+] [4] [ENTER] [5] [+] [×] [√x] [6] [ENTER] [7] [+] [8] [ENTER] [9] [+] [×] [√x] [+]

問題 3:

(10 – 5) ÷ [(17 – 12) × 4] = 0.2500

解答 3:

[1] [7] [ENTER] [1] [2] [−] [4] [×] [1] [0] [ENTER] [5] [−] [x◂▸y] [÷]
または
[1] [0] [ENTER] [5] [−] [1] [7] [ENTER] [1] [2] [−] [4] [×] [÷]

計算の順番

紙で計算するのと同じように、カッコの内側から外側へという順番で計算するのもひとつの方法です。しかし「左から右へ」という順番で入力する方法もあります。

たとえば次の計算を考えます。

$$4 \div [14 + (7 \times 3) - 2]$$

カッコの内側から、という規則で入力すれば(7 × 3)が最初です。このときの入力キーは [7] [ENTER] [3] [×] [1] [4] [+] [2] [−] [4] [x◂▸y] [÷] となります。

「左から右へ」とすれば、次の入力キーになります。

[4] [ENTER] [1] [4] [ENTER] [7] [ENTER] [3] [×] [+] [2] [−] [÷].

入力するキーは増えます。最初の中間結果は (7 × 3) で、「カッコの内側から」のときと同じです。「左から右へ」の場合は非可換性の演算（[−] や [÷]）でも [x◂▸y] キーを使わなくて済みます。

ただし、最初の方法「カッコの内側から」のほうが次の点では便利です。

- キー入力が少ない
- 必要なレジスタが少ない

Note

「左から右へ」方式をとった場合、中間結果を必ず **4 個**までにしてください。そうしないとスタックからあふれてしまいます。（スタックは 4 個までしか記憶できません）

先ほどの例で「左から右へ」方式を用いると全てのレジスタを活用することになります。

キー	表示	解説
[4] [ENTER] [1] [4] [ENTER]	14.0000	4 と 14 をスタックに入力
[7] [ENTER] [3]	3_	7 と 3 をスタックに入力。この時点で全てのスタックが埋まります
[×]	21.0000	中間結果.
[+]	35.0000	中間結果
[2] [−]	33.0000	中間結果
[÷]	0.1212	最終結果

練習問題 2

次の問題で RPN を練習しましょう。

問題 1:

(14 + 12) × (18 – 12) ÷ (9 – 7) = 78.0000

解答例 1:

[1] [4] [ENTER] [1] [2] [+] [1] [8] [ENTER] [1] [2] [−] [×] [9] [ENTER] [7] [−] [÷]

問題 2:

$23^2 - (13 \times 9) + 1/7 = 412.1429$

解答例 2:

2 3 ↷ x^2 1 3 ENTER 9 × − 7 $1/x$ +

問題 3:

$$\sqrt{(5.4 \times 0.8) \div (12.5 - 0.7^3)} = 0.5961$$

解答例 3:

5 · 4 ENTER · 8 × · 7 ENTER 3 y^x 1 2 · 5 $x \leftrightarrow y$ −
÷ $\sqrt{x}$

あるいは

5 · 4 ENTER · 8 × 1 2 · 5 ENTER · 7 ENTER 3 y^x −
÷ $\sqrt{x}$

問題 4:

$$\sqrt{\frac{8.33 \times (4 - 5.2) \div [(8.33 - 7.46) \times 0.32]}{4.3 \times (3.15 - 2.75) - (1.71 \times 2.01)}} = 4.5728$$

解答例 4:

4 ENTER 5 · 2 − 8 · 3 3 × ↷ LASTx 7 · 4 6 −
0 · 3 2 × ÷ 3 · 1 5 ENTER 2 · 7 5 − 4 · 3 ×
1 · 7 1 ENTER 2 · 0 1 × − ÷ $\sqrt{x}$

3

変数にデータを保存

HP 35s には 30KB のメモリが搭載されており、数値、式、およびプログラムを保存できます。数値は**変数**という場所に保存します。変数には A から Z までの名前がついています。（覚えやすい名前の変数に保存すると良いでしょう。たとえば銀行残高に G や Z を、光の速度を C など）

例:

ここでは変数 A に 3 を入力してみます。最初に RPN で、次に ALG モードとします。

キー	表示	説明
[MODE] [5] (5 RPN)		RPN モードへ変更（必要であれば）
[3]	0.0000 3_	値 (3) を入力
[↗] [STO]	STO_	変数に保存(**Sto**re)コマンドを実行。プロンプトがあらわれるとともに表示記号 A...Z が出ます。
[A]	0.0000 3.0000	変数 A に数値 3 が保存され、スタックに戻ります。
[MODE] [4] (4 ALG)	3.0000	ALG モードに変更
[3] [↗] [STO] [A]	3▶A_	再度、STO コマンドを実行。プロンプトがあらわれるとともに表示記号 A...Z が出ます。
[ENTER]	3▶A 3.0000	変数 A に数値 3 が保存され、その結果が第 2 行に表示されます。

ALG モードでは計算式を指定して、その計算結果を変数に保存できます。計算式そのものではなく、計算結果の数値が変数に保存されます。

例:

キー	表示	説明
1 + 3 ÷ 4	1+3÷4▶G	式を入力し変数に保存
↗ STO G ENTER	□□□□□1.7500	

キーボードでピンクに表示された英字はそれぞれ、変数 A..Z に割り当てられています。表示記号 A..Z が出たときはこれらのキーを押してください。

変数 X, Y, Z, T とスタックの X レジスタ, Y レジスタ, Z レジスタ, T レジスタとは別のものです。

数値の保存と読込

数値とベクトルは保存(**Sto**re、↗ STO) や読込(**Rec**all、RCL) ボタンで変数へ保存、ならびに変数から読込できます。数値は実数、虚数、10 進数、分数などがサポートされています。

表示されている数値(X レジスタ)をダイレクト変数にコピーするには:

↗ STO **変数文字** ENTER.

ダイレクト変数に保存された数値を読み込むには:

RCL **変数文字** ENTER.

例: 数値の保存

アボガドロ定数 (Avogadro Constant、約 6.0221×10^{23}) を変数 A に保存

キー	表示	説明
6 . 0 2 2 1 E 2 3	6.0221E23_	アボガドロ定数
↗ STO A	6.0221E23▶A_	変数入力のための "▶" プロンプト
ENTER	6.0221E23▶A 6.0221E23	アボガドロ定数を変数 A に保存。 同時に数値入力も完了します。

C	_	ディスプレイの表示をクリア
RCL	A..Z	表示記号 **A..Z** が出てきます。
A ENTER	A= 6.0221E23	変数 A からアボガドロ定数を読込

保存された変数を読み込むには読込コマンド(RCL、Recall)を使います。次の例で示すように、このコマンドは RPN と ALG モードで少し異なります。

例:

以前の例で保存した変数 G から数値 1.75 を読み込みます。ALG モードでの作業です。

キー	表示	説明
RCL G ENTER	G 1.7500	RCL を押すと A...Z モードがアクティブになります。このとき、ディスプレイ第 1 行に変化はありません。

ALG モードでは変数を式の中に入れることもできます。たとえば 15-2×G を計算してみます。これまで通り G=1.75 とすると:

キー	表示	説明
1 5 − 2 ×	15-2×G	
RCL G ENTER	11.5000	

では RPN モードではどうなるでしょうか。

キー	表示	説明
MODE 5 (5RPN)		RPN モードへ変更
RCL	RCL _	RPN モードでは RCL ボタンは編集行に変数をコピーします
G	1.7500 1.7500	ENTER を押す必要はありません

変数の閲覧

VIEW コマンド([↰] [VIEW])は X レジスタに数値をコピーすることなく、変数を表示します。表示は「変数=数値」という形式です。

数値の桁が多すぎて表示しきれないときは[↱] [›] や [↱] [‹] を使ってスクロールさせて下さい。閲覧を終了するには[←] か [C] ボタンを押します。VIEW コマンドはプログラミングの時によく使われますが、スタックを変更せずに変数を確認したいときはいつでも使用できます。

メモリー覧

メモリー覧 ([↰] [MEM]) は利用可能なメモリ量などを表示します。次の形式になっています:

```
1.VAR  2. PGM

nnn    mm,mmm
```

nnn は利用されている間接変数の量、mm,mmm は利用可能なメモリ量のバイト数です。

間接変数については第14章を参照して下さい。

VAR(変数)一覧

デフォルト設定では全てのダイレクト変数 A から Z には数値ゼロが入っています。ゼロでない数値を保存すると、VAR 一覧 ([↰] [MEM] [1](1VAR)) で閲覧できます。

例:

この例では変数 C に数値 3 を、変数 D に数値 4 を、変数 E に数値 5 を入力します。VAR 一覧で確認したあと、クリアしてみます。RPN モードを使います。

キー	表示	説明
[↱] [CLEAR] [2](2VARS)		ダイレクト変数を全てクリア
[3] [↱] [STO] [C]	4	変数 C に 3 を、変数 D に 4 を、
[4] [↱] [STO] [D]	5	変数 E に 5 を保存
[5] [↱] [STO] [E]		
[↰] [MEM] [1](1VAR)	C=	VAR 一覧を表示

□□□□□□3

一覧で ▽ および △ キーが利用できることを示す表示記号⬆ と ⬇があらわれます。ただし、分数表示モードの場合、分数の精度を示す表示記号▲ と ▼ は表示されません。VAR 一覧の使い方を次の例で示します。

▽	D=	ゼロではない値を持つ次のダイレクト変数(D=4)にスクロールします。
	4	
▽	E=	もう一度スクロールして E=5 を表示
	5	

続いて、VAR 一覧で変数をクリアしてゼロに戻す方法を示します。
E をクリアしてみましょう:

↗ CLEAR	C=	これで E はクリアされ、その値はゼロになるため、VAR 一覧には出なくなります。操作後、E の次の変数である C が表示されます。
	3	

次に、変数 C をスタックにコピーしてみます。

ENTER	5	C=3 が X レジスタにコピーされます。数値 5（前の操作で削除した E=5）は Y レジスタに移動します。
	3	

VAR 一覧の表示を終了するには← または C を押します。

変数にゼロを代入すると、クリアするのと同じ効果となります。全ての変数を一括してクリアするには ↗ CLEAR 2 (2VARS) と操作します。全てのダイレクト変数がゼロだった場合は VAR 一覧にはエラーメッセージ ALL VARS = 0 が表示されます。

変数が表示可能な桁より多い数値を持つ場合、▷ や ◁ でスクロールさせて下さい。

保存した変数を使った計算

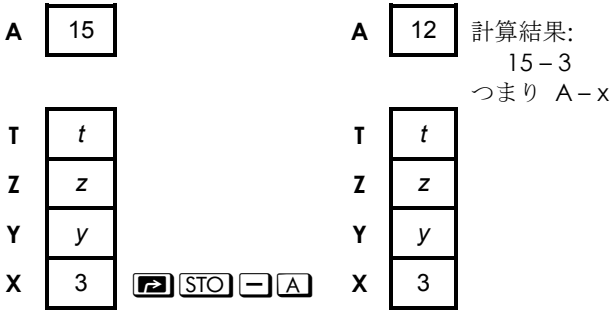
「保存した変数を使った計算」機能を使うと、変数に保存された値を**スタックに読み込まずに**計算で利用できます。計算では X レジスタの値と、指定された変数が持つ数値を使います。

ストレージ(STO)計算

[f] [STO] [+], [f] [STO] [−], [f] [STO] [×], [f] [STO] [÷] と操作するとストレージ(STO, 変数に保存した数値の読込)による計算ができます。変数に保存された数値は、スタックに読み込まれることなく、直接に計算に使われます。X レジスタの値とストレージの値を使って計算し、計算結果をその変数に戻します。スタックには影響はありません。

変数が持つ新しい数値 = その変数が持っていた以前の数値 {+, −, ×, または ÷} x

たとえば、A(15)を X レジスタ(3、ディスプレイに表示中)と引き算して更新したい場合、[f] [STO] [−] [A] と操作します。A=12 で、3 はディスプレイに表示したままとなります。



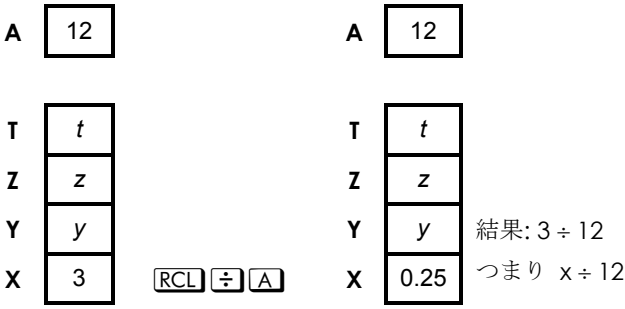
再読込 (RCL, Recall) による計算

再読込(RCL, Recall)による計算は [RCL] [+], [RCL] [−], [RCL] [×], あるいは [RCL] [÷] と操作します。X レジスタと再読込した変数が持つ数値が計算され、計算結果が X レジスタに保存されます。この操作では X レジスタのみが変更されます。変数に保存されていた数値はそのままです。

新しい x = 以前の x {+, −, ×, ÷} 変数

たとえば X レジスタの数値が 3(ディスプレイに表示中)で、これを変数 A(12 が保存されている)で割り算する場合を考えます。 RCL ÷ A と操作して下さい。x = 0.25 となり、A=12 はそのままです。

この操作はプログラムでも利用できます。1行で RCL + A と入力でき、2行で RCL A, + とするよりもメモリを半分に節約できます。



✔ **例:**

変数 D, E, F にそれぞれ数値 1, 2, 3 が保存されているとします。ストレージ計算を使ってそれぞれの変数に 1 を加算します。

キー	表示	説明
1 → STO D	1.0000	それぞれの数値を変数に保存
2 → STO E	2.0000	
3 → STO F	3.0000	
1 → STO + D → STO + E → STO + F	1.0000	*D*, *E*, *F* に 1 を加算
↺ VIEW D	D= 2.0000	変数 D の現在の値を表示
↺ VIEW E	E= 3.0000	
↺ VIEW F	F= 4.0000	
←	1.0000	ＶＩＥW表示を終了。X レジスタの

値が表示されます。

引き続き、変数 D, E, F に数値 2, 3, 4 が保存されているとします。数値 3 を変数 D で割り算したうえで E をかけ算、F を加算します。(3 ÷ D × E + F)

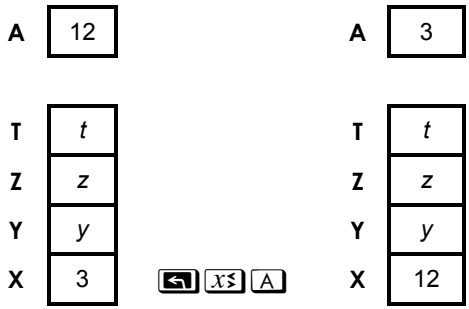
キー	表示	説明
3 RCL ÷ D	1.5000	3 ÷ *D* を計算
RCL × E	4.5000	3 ÷ *D* × *E*
RCL + F	8.5000	3 ÷ *D* × *E* + *F*

ある変数と x を交換

x の数値(X レジスタ、ディスプレイに表示中)をある変数が持つ数値と交換するには ↔ x⇄ と操作します。この操作は他のレジスタ、つまり Y、Z、T レジスタには影響しません。

例:

キー	表示	説明
1 2 ↔ STO A ENTER	12.0000	変数 A に 12 を保存
3	3_	X レジスタに 3 を入力
↔ x⇄ A	12.0000	X レジスタと変数 A の数値を交換
↔ x⇄ A	3.0000	X レジスタと変数 A の数値を再交換



変数 I と J

直接アクセスできる特別な変数がIとJです。他の変数と同じように数値を保持しますが、(I)ならびに(J)を使って他の変数（統計レジスタを含む）を参照できます。

(I)は 0 キーに、(J)は . キーに表示されています。これは間接呼び出しというプログラミング技法で、詳細は「第 14 章 変数とラベルの間接呼び出し」に掲載されています。

4

実数の関数

この章では実数を演算する関数のほとんどをカバーします。プログラムで使われる数値関数、たとえば絶対値を求める ABS なども含みます。実数関数には次のようなグループがあります。

- **指数**(Exponential)**と対数**(Logarithmic)
- **除算後の商**(Quotient)**と余り**(Division)
- **べき乗**(Power): ([y^x] と [$\sqrt[x]{y}$])
- **三角関数**(Trigonometric)
- **双曲線関数**(Hyperbolic)
- **百分率**(Percentage)
- **物理定数**(Physics constants)
- **座標、角度、単位の変換**(Conversion)
- **確率**(Probability)
- **数値の部分**(Parts of numbers)

四則演算は第1章と第2章でカバーされています。上記以外の関数(根の決定、積分、複素数、基数変換、統計)はこれ以降の章に記載します。この章での例は全て RPN モードを用いています。

✓ 指数関数と対数関数

ディスプレイに数値を表示させ、その関数のボタンを押します。[ENTER] を押す必要はありません。

目的	操作
自然対数 (底 *e*)	[→] [LN]
常用対数 (底 10)	[←] [LOG]
e を底とする指数	[→] [e^x]
10 を底とする指数	[←] [10^x]

✓ 除算後の商と余り

2つの実数をわり算したときの商(INT÷、Quotient)と余り(Rmdr、Remainder)をそれぞれ [◱] [INTG] [2](2INT÷) と [◱] [INTG] [3](3Rmdr) で求めることができます。

1. 第１の実数を入力
2. 第１の数値と第２の数値を分けて入力するため [ENTER] を押す
3. 第２の数値を入力 (ここでは [ENTER] は押さないで下さい)
4. 関数キーを押す

例:

除算 58 ÷ 9 の商と余りを求めます

キー	表示	説明
[5] [8] [ENTER] [9] [◱] [INTG] [2](2INT÷)	6.0000	商
[5] [8] [ENTER] [9] [◱] [INTG] [3](3Rmdr)	4.0000	余り

✓ べき乗

RPN モードで y の x 乗を入力するには、y [ENTER] x [y^x] と操作します。(ただし、y>0 のとき x は任意の数で、y<0 のとき x は正の数)

計算対象	操作	結果
15^2	[1] [5] [↷] [x^2]	225.0000
10^6	[6] [◱] [10^x]	1,000,000.0000
5^4	[5] [ENTER] [4] [y^x]	625.0000
$2^{-1.4}$	[2] [ENTER] [1] [.] [4] [+/−] [y^x]	0.3789
$(-1.4)^3$	[1] [.] [4] [+/−] [ENTER] [3] [y^x]	-2.7440

RPN モードで y の x 乗根を求めるには、y [ENTER] x [◱] [$\sqrt[x]{y}$] と操作します。ただし、$y < 0$ のとき x は整数とします。

計算対象	操作	結果
$\sqrt{196}$	1 9 6 $\sqrt{x}$	14.0000
$\sqrt[3]{-125}$	1 2 5 +/− ENTER 3 ↶ $\sqrt[x]{y}$	-5.0000
$\sqrt[4]{625}$	6 2 5 ENTER 4 $\sqrt[x]{y}$	5.0000
$\sqrt[.4]{.37893}^{-1}$	. 3 7 8 9 3 ENTER 1 . 4 +/− ↶ $\sqrt[x]{y}$	2.0000

三角関数

工学的なπ

X レジスタに 12 桁の精度で π を入力するには ↶ π と操作します。

（表示される桁は表示設定に依存します。） ↶ π は π の近似値をスタックに出力する関数ですから ENTER を押す必要はありません。

π は超越数ですから 35s が出力するのはあくまで π の近似値であり、厳密値ではありません。

角度モードの設定

角度モードは、三角関数の計算でどの単位を使うかの設定です。すでに入力された数値を変換するものでは**ありません**。（この章で後述する「変換関数」をご参照下さい）

360 [deg] = 2π [rad] = 400 [grad] です。

角度モードの設定には MODE を押します。オプションを選択するメニューが次のように表示されます。

オプション	詳細	表示記号
DEG	度(Degree)に設定します。度を 10 進数で入力します。60 進数(度、分、秒)ではありません。	none
RAD	ラジアンに設定	**RAD**
GRAD	グラジアンに設定	**GRAD**

✓ 三角関数

数値 x が X レジスタとしてディスプレイに表示されているとします。

計算対象	操作
sin(x)	SIN
cos(x)	COS
tan(x)	TAN
asin(x)	↰ ASIN
acos(x)	↰ ACOS
atan(x)	↰ ATAN

Note

無理数である π は、電卓の内部精度である 15 桁を超えて保持できないため、これを用いた計算は厳密解になりません。これは三角関数の演算で顕著にあらわれます。たとえば、sin(π) [rad] はゼロではなく -2.0676×10^{-13} となります。

例:

cos(5/7 π) [rad] と cos(128.57°) [deg] が有効桁4桁まで等しいことを確認します。

キー	表示	説明
MODE 2 (2RAD)		角度モードをラジアンにします。表示記号 **RAD** があらわれます。
. 5 . 7 ENTER	0.7143	小数点表記で 5/7 が表示されます
↰ π × COS	-0.6235	cos(5/7 π)

MODE 1 (1DEG)	-0.6235	角度モードを度に変更（表示記号はありません）
1 2 8 · 5 7 COS	-0.6235	cos(128.57°)を計算。cos(5/7 π)と有効桁４桁で同じになります。

プログラミングの注意:

関数 atan は角度 θ を求めるときに便利ですが、ゼロ割に注意して下さい。

$$\theta = \arctan(y/x).$$

x = 0 だと y/x が計算できないためエラーメッセージ **DIVIDE BY 0** が出ます。

双曲線関数

数値 x が X レジスタとしてディスプレイに表示されているものとします:

計算対象	操作
sinh(x)	↶ HYP SIN
cosh(x)	↶ HYP COS
tanh(x)	↶ HYP TAN
asinh(x)	↶ HYP ↷ ASIN
acosh(x)	↶ HYP ↷ ACOS
atanh(x)	↶ HYP ↷ ATAN

✓ 百分率関数

百分率関数は × や ÷ と違い、元となる数値(Y レジスタ) がそのまま保持されます。x の y に対するパーセンテージを計算し、結果は X レジスタに上書きされます。Y レジスタに入っていた数値、ならびに計算結果は、再入力することなく次の計算で再利用できます。

計算対象	操作
y の x%	y [ENTER] x [→] [%]
y から x への変化率 (y≠ 0)	y [ENTER] x [←] [%CHG]

例:

$15.76(消費税抜き)の商品を購入したとき、消費税が 6%であれば消費税額と総額はいくらになるでしょうか？

セント単位で丸めて表示させるため、表示設定を FIX 2 として下さい。

キー	表示	説明
[←] [DISPLAY] [1](FIX) [2]		表示を小数点以下２桁に指定
[1] [5] [.] [7] [6] [ENTER]	15.76	
[6] [→] [%]	0.95	6% の消費税額を計算
[+]	16.71	総額 (税抜き価格 + 6%)

昨年、価格が $16.12 だった商品が今年は $15.76 となりました。価格の変化率は？

キー	表示	説明
[1] [6] [.] [1] [2] [ENTER]	16.12	
[1] [5] [.] [7] [6] [←] [%CHG]	-2.23	今年は約 2.2%下落ました
[←] [DISPLAY] [1](FIX) [4]	-2.2333	表示形式を FIX 4 に変更

Note

%CHG 関数では２つの数値の順番が重要です。あくまで y から x への変化率であり、順番が異なると符号も大きさも違ってきます。

物理定数

41 の物理定数(Physics CONstants) を CONST メニューから呼び出すことができます。 [↑] [CONST] で次のメニューが出てきます。

CONST メニュー

記号	詳細	値
c	真空中の光の速度	299792458 $m\ s^{-1}$
g	重力加速度	9.80665 $m\ s^{-2}$
G	万有引力定数	$6.673\times10^{-11}\ m^3\ kg^{-1}s^{-2}$
Vm	理想気体のモル体積	0.022413996 $m^3\ mol^{-1}$
NA	アボガドロ定数	$6.02214199\times10^{23}\ mol^{-1}$
R∞	リュードベリ定数	10973731.5685 m^{-1}
eV	電気素量	$1.602176462\times10^{-19}$ C
me	電子の質量	9.10938188×10^{-31} kg
mP	陽子の質量	1.67262158×10^{-27} kg
mn	中性子の質量	1.67492716×10^{-27} kg
mμ	ミュー粒子の質量	1.88353109×10^{-28} kg
k	ボルツマン定数	$1.3806503\times10^{-23}\ J\ K^{-1}$
h	プランク定数	$6.62606876\times10^{-34}\ J\ s$
$\overline{h}$	プランク定数を 2 pi で割ったもの(ディラック定数)	$1.054571596\times10^{-34}\ J\ s$
Φo	磁束量子	$2.067833636\times10^{-15}$ Wb
ao	ボーア半径	$5.291772083\times10^{-11}$ m
εo	真空の誘電率	$8.854187817\times10^{-12}\ F\ m^{-1}$
R	気体定数	$8.314472\ J\ mol^{-1}\ k^{-1}$
F	ファラデー定数[	$96485.3415\ C\ mol^{-1}$
u	原子質量単位	1.66053873×10^{-27} kg
μo	真空中の透磁率	$1.2566370614\times10^{-6}\ NA^{-2}$
μB	ボーア磁子	$9.27400899\times10^{-24}\ J\ T^{-1}$
μN	核磁子	$5.05078317\times10^{-27}\ J\ T^{-1}$
μP	陽子の磁気モーメント	$1.410606633\times10^{-26}\ J\ T^{-1}$
μe	電子の磁気モーメント	$-9.28476362\times10^{-24}\ J\ T^{-1}$
μn	中性子の磁気モーメント	$-9.662364\times10^{-27}\ J\ T^{-1}$

記号	詳細	値
μμ	ミュー粒子の磁気モーメント	$-4.49044813\times10^{-26}$ J T^{-1}
re	電子の古典半径	$2.817940285\times10^{-15}$ m
Zo	真空のインピーダンス	376.730313461 Ω
λC	電子のコンプトン波長	$2.426310215\times10^{-12}$ m
λCn	中性子のコンプトン波長	$1.319590898\times10^{-15}$ m
λCp	陽子のコンプトン波長	$1.321409847\times10^{-15}$ m
α	微細構造定数	7.297352533×10^{-3}
σ	シュテファン・ボルツマン定数	5.6704×10^{-8} W m^{-2} K^{-4}
t	セルシウス度(摂氏)	273.15
atm	標準大気圧	101325 Pa
γP	陽子磁気回転比	267522212 $s^{-1}T^{-1}$
C1	第 1 放射定数	374177107×10^{-16} W m^2
C2	第 2 放射定数	0.014387752 m K
Go	コンダクタンス量子	7.748091696×10^{-5} S
e	自然対数の底*e*(ネイピア数)	2.71828182846

参照: Peter J.Mohr and Barry N.Taylor, CODATA Recommended Values of the Fundamental Physical Constants: 1998, Journal of Physical and Chemical Reference Data,Vol.28, No.6,1999 and Reviews of Modern Physics,Vol.72, No.2, 2000.

定数の入力方法:

1. 入力位置にカーソルをあわせます
2. [◀] [CONST] と操作して物理定数メニューを開きます
2. [▶] [◀] [▲] [▼] でスクロールし、目的の定数の下にアンダーラインを移動させ([◀] [CONST] で 1 ページずつ移動できます)、最後に [ENTER] を押します。

物理定数を式やプログラム中で使うときは、その値ではなく名前で参照したほうが良いです。

変換関数

HP 35s には4種類の変換があります。次の数値を相互に変換できます。

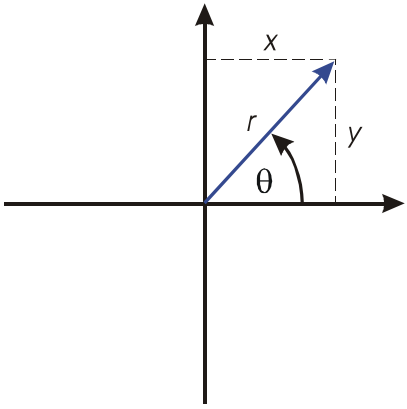
- **複素数の直交形式と極形式**
- **角度の単位：度、ラジアン、グラジアン**
- **時間と角度(度分法)の表記における 10 進数と 60 進数**
- **その他、様々な単位系** (cm/in, kg/lb など)

直交座標と極座標の変換にはそれぞれ、特定のキーが割り当てられています。左シフト(黄)に割り当てられた変換と、右シフト(青)に割り当てられた変換はそれぞれ逆方向のものになります。この種の変換では、変換元となる数値は、変換対象の単位系で測定したものと考えます。たとえば [→°F] を使ってある数値を華氏に変換するとき、その数値は摂氏で測定された温度であると見なします。

この章の例では RPN モードを使っています。ALG モードでは関数を最初に入力し、次に変換する数値を入力します。

直交形式と極形式の変換

極座標系 (*r*,*θ*) と直交座標系 (x,y) を次の図に示します。角度 θ の単位は角度モードに指定された単位系になります。 θ の範囲は±180°、± π [rad]、あるいは ±200[grad] となります。



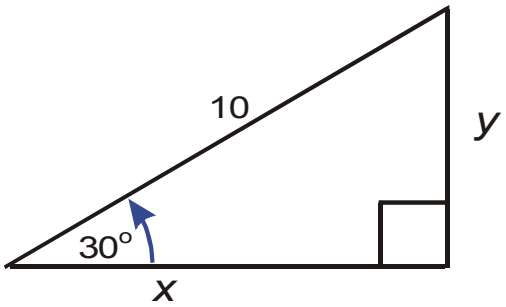
極座標系と直交座標系を変換するには:

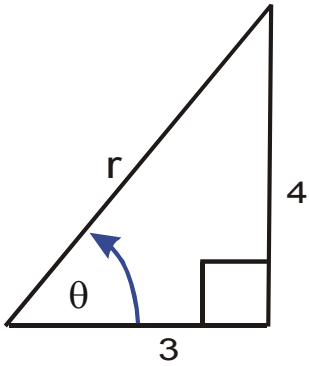
まず、表示モードを変更して複素数を表示できるようにします。複素数の表示であればどの表示モードでも結構ですが、入力する方法が異なってきます。複素数の表示モードに変更するには次のように操作します。

1. RPN モードでは [↶] [DISPLAY] と操作し、[9] (9×iy) あるいは [·] [0] (10rθa) を選択します。(ALG モードでは [·] [1] (11x+yi)を選択することもできます)
2. 座標値を入力します (x [i] y や、x [+] y [i]、あるいは r [↷] [θ] a)
3. [ENTER]

例: 極座標から直交座標への変換

次の図の直角三角形で、左の三角形の辺 x と y の長さ、および、右の三角形の斜辺の長さ r と角度 θ を求めます。





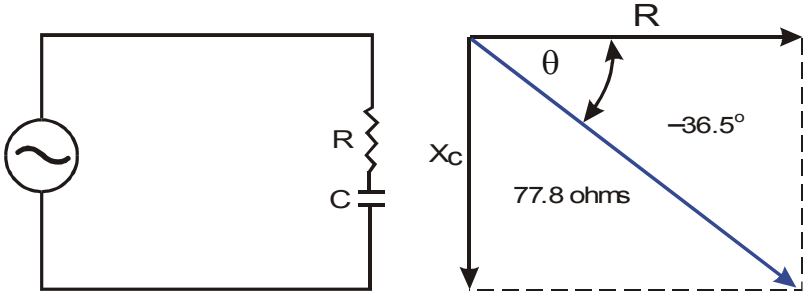
キー	表示	説明
[MODE] [1] (1DEG) [↶] [DISPLAY] [9] (9×iy) [1] [0] [↷] [θ] [3] [0] [ENTER]	8.6603i5.0000	角度を Degree に、複素数の表示モードにそれぞれ変更 rθa (極座標) を xiy (直交座標)に変更

[↺] [DISPLAY] [.] [0] (10rθa)	10.0000θ30.0000	複素数の表示モードを変更
[3] [i] [4] [ENTER]	5.0000θ53.1301	xiy (直交座標) を rθa (極座標)に変更

例: ベクトルの変換

次の図の RC 回路で総インピーダンスが 77.8 [Ω]、電圧の電流に対する遅れが 36.8° であるとします。この回路の抵抗 R と容量性リアクタンス (capacitive reactance) X_C を求めます。

右図のベクトル図を使います。インピーダンスは極座標の大きさ *r* で、電圧の遅れは角度 θ [deg] です。このベクトルを直交座標に変換すれば *x* の値が求める *R* [Ω] で、*y* の値が求める X_C [Ω] となります。



キー	表示	説明
[MODE] [1] (1DEG)		角度を Degree に、表示モードを複素数に指定
[↺] [DISPLAY] [9] (9×iy)		
[7] [7] [.] [8] [↷] [θ]	77.8θ-36.5	電圧の遅れを角度 θ として、総インピーダンス [Ω] を *r* として入力
[3] [6] [.] [5] [+/−]		
[ENTER]	62.5401i-46.2772	計算結果。*x* が求める *R* [Ω] で、*y* が求める X_C [Ω] です。

時間の変換

HP 35s は 10 進数と 60 進数を相互に変換できます。この機能は特に、時間や度分法の角度の取り扱いにおいて便利です。たとえば、度分法で計測された角度が 10 進数で D.ddd…だったとき、これを D.MMSSss という形式に変換できます。ここで、D は度の整数部で、ddd… は計測のときの端数(度の小数部)、MM は分をあらわす整数、SS は秒をあらわす整数、ss は秒の小数部分です。

✓ 10 進数と時間・分・秒の変換

1. 変換元の数値を入力
2. [↩] [→HMS] と操作し、時間/角度、分、秒に変換します。
   あるいは [↪] [HMS→] で 10 進数に変換します。

例: 時間の表示形式を変換

1時間の 1/7 は何分何秒でしょうか？ 表示形式は FIX 6 にしてください。

キー	表示	説明
[↩] [DISPLAY] [1](1FIX) [6]		表示形式を FIX 6 に
[·] [1] [·] [7]	0.000000 0 1/7	10 進数で 1/7 を入力
[↪] [→HMS]	0.000000 0.083429	答えは 8 分 34.29 秒
[↩] [DISPLAY] [1](1FIX) [4]	0.000000 0.0834	表示形式を FIX 4 にしてみます

✓ 角度の変換

ラジアンへの変更のとき、X レジスタに入力された数値は degree 単位であるものとして計算します。また、degree への変更のとき、X レジスタに入力された数値はラジアン単位であるものとして計算します。

degree とラジアンの変換

例

ここでは 30 [deg] から π/6 [rad] へ変換します。

キー	表示	説明
3 0	0.0000 30_	degree で角度を入力
↶ →RAD	0.0000	ラジアンへ変換。
	0.5236	計算結果 0.5236 は約 π/6 です。

単位変換

HP 35s のキーボードには単位を変換するキーが 10 個あります。

変換元	変換先	操作	結果
1 lb	kg	1 ↷ →kg	0.4536 (kg)
1 kg	lb	1 ↶ →lb	2.2046 (ポンド)
32 °F	°C	3 2 ↷ →°C	0.0000 (°C)
100 °C	°F	1 0 0 ↶ →°F	212.0000 (°F)
1 in	cm	1 ↷ →cm	2.5400 (cm)
100 cm	in	1 0 0 ↶ →in	39.3701 (インチ)
1 gal	l	1 ↷ →l	3.7854 (リットル)
1 l	gal	1 ↶ →gal	0.2642 (ガロン)
1 MILE	KM	1 ↷ →KM	1.6093 (km)
1 KM	MILE	1 ↶ →MILE	0.6214 (マイル)

確率関数

✓ 階乗(Factorial)

正の整数 x (0 ≤ x ≤ 253) の階乗を計算するには、[→] [!] (右シフト + [Σ+] キー)と操作します。

✓ ガンマ(Gamma)

整数ではない正の値 x のガンマ関数 Γ(x) を計算するには、(x − 1)を入力してから [→] [!] と操作します。x! 関数で Γ(x + 1) が計算されます。

✓ 確率

組合せ(Combinations)

異なる *n* 個のものから *r* 個のものを選ぶ組合せ(**C**ombination)を計算するには、最初に *n* を入力し、[←] [nCr] と操作し、最後に *r* を入力して下さい。*n* と *r* は正の整数です。*r* を選ぶとき、重複することはありません。また、選択の順番は無関係です。

✓ 順列(Permutations)

異なる *n* 個のものから *r* 個取り出して一列に並べたときの、並べ方の総数(順列、**P**ermutation)を計算するには、最初に *n* を入力し、[→] [nPr] と操作し、最後に *r* を入力して下さい。*n* と *r* は正の整数です。*r* を選ぶとき、重複することはありません。また、選択の順番が違うと別の並べ方としてカウントされます。

✓ 乱数シード値(Seed)

数値 x を乱数生成器のシード値に指定するには、[←] [SEED] と操作します。

✓ 乱数生成器

0 < x < 1 の範囲で乱数(**Rand**am number)を生成するには [→] [RAND] と操作します。(この乱数は疑似乱数であり、一様分布する数列から取り出したものです。次の文献で紹介されたスペクトルテストを通過しています: D. Knuth, *The Art of Computer Programming,* vol. 2, *Seminumerical Algorithms,* London: Addison Wesley, 1981)

RANDAM 関数は乱数の生成時にシード値を使っています。生成された乱数は、次に生成される乱数のシード値になります。よって、同じシード値から生成された一群の乱数では同じものが出てくる可能性があります。SEED 関数を使って新しいシード値を指定できます。メモリがクリアされたときシード値はゼロにリセットされます。シード値がゼロのときは、シード値は自動決定されます。

例: 従業員の組合せ

ある会社に14人の女性と10人の男性従業員がいるとします。この中から6人の委員を選出するとしたら、異なる組合せの総数はいくつでしょうか？

キー	表示	説明
2 4 ENTER 6	24 6_	合計 24 人から 6 人を取り出す
↶ nCr	134,596.0000	組合せの総数

ランダムに選択するとき、女性が6人になる確率はいくつでしょうか？ 確率を計算するにはその場合の数を、全ての組合せの数でわり算します。

キー	表示	説明
1 4 ENTER 6	14 6_	女性 14 人から 6 人を選ぶ
↶ nCr	3,003.0000	6 人の女性が選ばれる組合せ
x◂▸y	134,596.0000	組合せの総数をXレジスタに移動させる
÷	0.0223	女性のみの組合せ÷全ての組合せ。これが確率となります。

数値の部分を取り出す

これらの関数はプログラミングでよく使われます。

整数部(IP, Integer part)

✓ 小数部をゼロに置き換えて整数部のみにするには [↰] [INTG] [6] (6IP)と操作します。たとえば、14.2300 の整数部は 14.0000 です。

小数部(FP, Fractional part)

✓ 整数部をゼロに置き換えて小数部のみにするには [↰] [INTG] [5] (5FP)と操作します。たとえば、14.2300 の小数部は 0.2300 です。

絶対値(ABS, Absolute value)

X レジスタの数値をその絶対値で置き換えるには [↱] [ABS] と操作します。複素数やベクトルでは下記のようになります:

1. rθa 形式の複素数は r を出力
2. xiy 形式の複素数は $\sqrt{x^2+y^2}$ を出力
3. ベクトル [A1,A2,A3, ...An] は $|A|=\sqrt{A_1{}^2+A_2{}^2+\cdots+A_n{}^2}$ を出力

複素数の偏角(Argument value)

複素数の偏角は[↰] [ARG]で出力します。偏角は次のようになります:

1. rθa 形式の複素数では a
2. xiy 形式の複素数では atan(y/x)

符号(Sign)

✓ x の符号を出力するには [↰] [INTG] [1](1SGN) と操作します。x が負の場合は –1.0000 が、ゼロの場合は 0.0000 が、正の場合は 1.0000 が表示されます。

最大の整数(INTG, Greatest integer)

✓ [↰] [INTG] [4] (4INTG) と操作すると、与えられた数値以下で最も大きい整数を出力します。

例:

数値の部分を出力する関数の操作方法をまとめると下表の通りです。

目的	操作方法	表示
2.47 の整数部	2 · 4 7 ↶ INTG 6 (6IP)	2.0000
2.47 の小数部	2 · 4 7 ↶ INTG 5 (5FP)	0.4700
–7 の絶対値	7 +/– ↷ ABS	7.0000
9 の符号	9 ↶ INTG 1 (1SGN)	1.0000
-5.3 以下で最も大きな整数	5 · 3 +/– ↶ INTG 4 (4INTG)	-6.0000

RND 関数(↷ RND) はxを表示形式で指定された桁数で丸めます。(内部精度は12 桁のままです。) 分数表示モードにおける RND 関数のふるまいは第5章をご参照下さい。

5

分数 (Fraction)

第1章の「分数」節では分数の入力、表示、計算の基本を記載しました。この章ではより詳細な情報を解説します。分数の入力と表示をまとめると次のようになります。

- 分数を入力するには [·] キーを２回押します。
  整数部、１回目の [·]、分子、２回目の [·]、分母の順番で入力します。
  たとえば 2 3/8 の場合は、[2] [·] [3] [·] [8] となります。
  5/8 であれば、[·] [5] [·] [8] または [0] [·] [5] [·] [8] です。
- 分数表示モードのオンとオフは [↗] [FDISP] と操作します。分数表示モードがオフになると、分数表示モードがオンになる前の表示モードに戻ります。分数表示モードがオンのとき、他の表示モードにすると分数表示モードはオフになります。
- 分数は 10 進数とほとんど同じように取り扱われますが、RND 関数だけは別です。詳細はこの章で解説します。

この章の例では、特に明記しない限り RPN モードが使われています。

分数の入力

ほとんど全ての分数をキーボードから入力できます。仮分数、つまり分子が分母より大きい分数でも問題ありません。

例:

キー	表示	説明
[↗] [FDISP]		分数表示モードをオン
[1] [·] [5] [ENTER]	1 1/2	1.5 を入力。分数として表示されます。
[1] [·] [3] [·] [4] [ENTER]	1 3/4	1 3/4.を入力
[↗] [FDISP]	1.7500	x を小数で表示
[↗] [FDISP]	1 3/4	x を分数で表示

もし結果が例と同じにならない場合は、分数の表示設定が何かの拍子に変わっています。この章の「分数の表示設定」をご参照下さい。次節において、入力できる分数と、できない分数についての例を記載しました。

ディスプレイでの分数

分数表示モードでは、数値は内部で小数として評価されたあと、その小数に最も近い分数として表示されます。さらに、その 12 桁精度の小数と表示している分数の差があれば、精度の表示記号▲▼が表示されます。(統計レジスタは例外で、常に小数として表示されます。)

表示のルール

入力した分数は表示されるものと異なる場合があります。デフォルト設定の状態では次のルールで表示しています。(このルールを変更するには、この章の「分数の表示設定」をご参照下さい)

- **整数部と真分数（分子が分母より小さい分数）で表示**
- **分母は 4095 以下とする。**
- **分数は可能な限り約分される**

例:

入力された値と表示される分数の例です。比較のため内部の 12 桁の値もあわせて表記します。表示記号▲ ▼はそれぞれの行の最後に掲載しています。

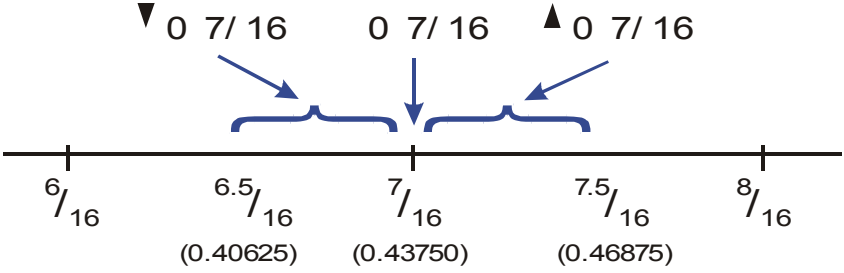
入力値	内部値	表示される分数
$2\ ^{3}/_{8}$	2.37500000000	2 3/8
$14\ ^{15}/_{32}$	14.4687500000	14 15/32
$^{54}/_{12}$	4.50000000000	4 1/2
$6\ ^{18}/_{5}$	9.60000000000	9 3/5
$^{34}/_{12}$	2.83333333333	2 5/6 ▼
$^{15}/_{8192}$	0.00183105469	0 7/3823 ▲
$12345678\ ^{12345}/_{3}$	12349793.0000	12349793
$16\ ^{3}/_{16384}$	16.0001831055	16 1/4095

精度の表示記号

表示中の分数の精度をディスプレイ右端に出る表示記号 ▲ ▼ で確認できます。この記号は、表示中の分数と内部の 12 桁精度の小数を比較して表示しています。

- **表示記号が出ていない場合は、内部の 12 桁精度の小数が表示中の分数と完全に一致していることを示しています。**
- **表示記号 ▼ が出ている場合、内部の 12 桁精度の小数は表示中の分数よりわずかに小さいことを示しています。このとき、正確な分子と表示中の分子との差は 0.5 より小さいものとなります。(正確な分子の方が小さい)**
- **表示記号 ▲ が出ている場合、内部の 12 桁精度の小数は表示中の分数よりわずかに大きいことを示しています。このとき、正確な分子と表示中の分子との差は 0.5 より小さいものとなります。(正確な分子の方が大きい)**

次の図は表示中の分数と近似値との関係をあらわしています。▲は正確な分子が表示中の分子よりも「少し大きい」ことを、 ▼ は「少し小さい」ことを示します。



分数の表示設定を変えたときはこの点が特に重要になります。(次の「分数の表示設定」を参照して下さい。) たとえば、設定で全ての分数の分母を 5 に設定すると、$^2/_3$ は 0 3/5▲ と表示されます。正確な分数は約 $^{3.3333}/_5$ であり、$^3/_5$ の分子より「少し大きい」からです。同じように、$-^2/_3$ は正確な分子である約 3.3333は 3より「少し大きい」ため −0 3/5▲ と表示されます。

まれに、期待しない表示記号が出ることがあります。たとえば 2 $^2/_3$ を入力すると、入力したのが正確な分数だとしても、2 2/3▲ と表示されます。これは常に内部の 12 桁精度の小数と表示中の分数とを比較しているからです。また、内部の数値が整数部を持つ場合は、その分数部分は 12 桁より少ない精度で保持されているため、分数との比較では 12 桁を全て使うことができなくなります。

分数の表示設定

デフォルト設定では前述のルールで分数を表示しますが、このルールは変更できます。

- **分母の最大値を指定できます。**
- **３つの表示形式から選択できます。**

変更方法は次の通りです。

分母の最大値

全ての分数で分母の最大値を指定できます。分数が *a b/c* のとき、/c が最大値を指定できる分母です。

/c は**最大値**のみが指定できます。表示のときに使われる分母は分数の表示設定に依存します。(次項で詳細を記載します)

- **分母の最大値を指定するには、最大値を入力したあと [↰] [/c] と操作します。自動的に分数表示モードがオンになります。4095 を超える値は指定できません。**
- **X レジスタにその最大値を呼び出すには [1] [↰] [/c] と操作します。**
- **デフォルト値である 4095 に戻すには、[0] [↰] [/c] と操作するか 4095 より大きな数値を/c に指定して下さい。この操作でも分数表示モードがオンになります。**

/c 機能は X レジスタの数値における整数部の絶対値を読み込みます。LAST X レジスタの数値は変更しません。

表示中の分数が長すぎてディスプレイに収まらないときは、表示記号➡ が出てきます。その場合は [→] [<] と [→] [>] を使ってスクロールさせて確認できます。分数表示から小数表示に一時的に戻すには [↩] を押してから [SHOW] を押し続けて下さい。

例:

分母の最大値を 3125 に指定し、ディスプレイに収まりきれない分数を表示してみます。

キー	表示	説明
[3] [1] [2] [5] [↩] [/c]		分母の最大値を 3125 に指定
[1] [4] [→] [e^x]	0 1202604□888/31	表示しきれない数値があることを示す表示記号が出ます
[>]	0 25	スクロールさせて分母の残りを表示させて下さい。

注意:

1. ALG モードではディスプレイ第 1 行に式を入力してから [↩] [/c] と操作することもできます。その場合は式を計算し、計算結果が分母の最大値として認識されます。
2. ALG モードでは、計算結果を /c 機能の引数として与えることができます。ディスプレイ第 2 行に計算結果を出した状態で [↩] [/c] と操作します。第 2 行の表示が分数形式だったときは、その整数部が分母の最大値となります。
3. 複素数およびベクトルは /c 機能の引数に指定できません。指定するとエラーメッセージ INVALID□DATA が表示されます。

分数の表示形式の選択

分数の表示には3つの形式があります。次のルールに沿って最も精度の高い状態で表示されます。

- **精度優先**　分母は /c で指定された値を超えない範囲で任意の数になり、可能な限り約分されます。たとえば、分母を可能な範囲で制限なく表示させたければ /c に 4095 を設定します。これがデフォルト設定です。
- **分母を最大値の約数に**　分母を /c で指定された値の約数とし、可能な限り約分されます。たとえば、米国の古い株価を計算していて 53 1/4 や 37 7/8 という形式にしたければ /c を 8 に設定します（訳注:米国の株価は 2001 年から小数点表示に移行しました)。 あるいは、/c が 12 だった場合は分母は 2, 3, 4, 6, 12 のいずれかになります。
- **分母を固定**　分母は常に /c で指定された値になります。約分は発生しません。たとえば、時間を見やすく 1 25/60 (/c を 60 にしておく) と表示できます。

分数の表示モードの設定には3つのフラグが用意されています。フラグにはそれぞれ 7、8、9 という番号が割り振られています。それぞれのフラグはクリアされた状態かセットされた状態かを保持しています。それぞれの目的は次の通りです。

- フラグ 7 は分数表示モードのオンとオフをあらわします。クリア=オフ、セット=オンです。
- フラグ 8 は /c の値の約数のみを使うか、/c の値を最大値とする任意の値を使うかの設定です。クリア=任意の値を使う、セット=約数のみを使う、というフラグです。
- フラグ 9 はフラグ 8 がセットされている状態でのみ有効で、分数を約分するかどうかの設定です。クリア=約分する、セット=約分しない、となります。

フラグ 8 と 9 がクリアかセットかにより、次の表のように3種類の分数表示形式があります。

分数の表示形式	フラグ	
	8	9
精度優先	クリア	—
分母を最大値の約数に	セット	クリア
分母を固定	セット	セット

フラグ 8 と 9 を設定するには次のように操作して下さい。(フラグはプログラムで特に便利ですので第14章で詳細を記載します。)

1. [↓] [FLAGS] と操作してフラグメニューを出します。

2. フラグのセットには 1 (1SF、Set Flag)を選択しフラグ番号（8 や 9）を入力します。フラグのクリアには 2 (2CF、Clear Flag)を選択し、フラグ番号（8 や 9）を入力します。フラグの確認には 3 (3FS?、Flag Set?) を選択しフラグ番号を入力します。クリアするには C か ← を押して確認ダイアログにYES か NO を選択して下さい。

例:

この例では π を使って3種類の分数表示形式を試してみます。分数表示モードがオンであり、フラグ 8 がデフォルト状態（クリア）であるものとします。

キー	表示	説明
4 0 9 5 ↶ /c		分母の最大値 /c をデフォルト設定の 4095 に戻します。
↶ π	0	精度優先。
	3□16/113	フラグ 8 = クリア
↶ FLAGS 1 (1SF)	0	フラグ 8 = セット
8	3□116/819	最大値の約数を分母とします。(819 * 5 = 4095)
↶ FLAGS 1 (1SF)	0□0/4095	フラグ 9 = セット
9	3□580/4095	分母を固定
↶ FLAGS 2 (2CF)	0	デフォルト形式に戻します。
8 ↶ FLAGS 2 (2CF) 9	3□16/113	(精度優先)

分数表示の例

次の表は小数 2.77 が3種の分数表示形式でどのように表示されるかをまとめています。/c は2種類用意しています。

分数の表示形式	2.77 の表示	
	/c = 4095	/c = 16
精度優先	2 77/100 (2.7700)	2 10/13▲ (2.7692)
分母を最大値の約数に	2 1051/1365▲ (2.7699)	2 3/4▲ (2.7500)
分母を固定	2 3153/4095▲ (2.7699)	2 12/16▲ (2.7500)

次の表は異なる数値が3種の分数表示形式でどのように表示されるかを示しています。/c は 16 としています。

分数の表示形式＊	入力された数値と表示される分数				
	2	2.5	$2\,^{2}/_{3}$	2.9999	$^{16}/_{25}$
精度優先	2	2 1/2	2 2/3▲	3▼	2 9/14▼
分母を最大値の約数に	2	2 1/2	2 11/16▼	3▼	2 5/8▲
分母を固定	2 0/16	2 8/16	2 11/16▼	3 0/16▼	2 10/16▲
＊ /c は 16 としています					

分数の丸め

分数表示モードがオンであれば、RND 関数は X レジスタの数値を表示中の分数に対して最も近い小数に置き換えます。この丸めはフラグ 8 と 9 の状態にも依存します。分数と変換後の小数が一致していれば、精度をあらわす表示記号は消えます。そうでなければ（誤差があれば）精度の表示記号は点灯します。（この章の前のほうにある「精度の表示記号」もご参照ください。）

式やプログラム中でも、RND 関数は分数表示モードがオンの場合は上記と同じく、分数を丸めます。

例:

56 $^{3}/_{4}$ インチの長さを6等分したいとします。巻き尺の最小単位が $^{1}/_{16}$ インチのとき、ひとつ何インチになりますか？ また、そのときの累積誤差は？

キー	表示	説明
↩ FLAGS ENTER 8		フラグ 8 をセット

1 6 ↶ /c		分母の最大値を $^{1}/_{16}$ に指定。（フラグ 8 と 9 は前の例と同じ状態とします）
5 6 · 3 · 4 ↷ STO D	56 3/4	距離を変数 D に保存
6 ÷	9 7/16▲	６分割すると $9\,^{7}/_{16}$ インチより少し長い
↷ RND	9 7/16	表示されている分数に丸めます
6 ×	56 5/8	６倍を計算
RCL D −	-0 1/8	累積誤差
↶ FLAGS 2 (2CF) 8	-0 1/8	フラグ 8 をクリア
↷ FDISP	-0.1250	分数表示モードをオフ

式の中での分数

式の中でも分数を使用できます。式が表示されるときは、式の中の全ての数値は入力時の形式で表示されます。また、分数表示モードは式の入力時でも有効です。

式を計算するときと、変数に数値を保存するときも分数を取り扱えます。ただし、表示される数値はそのときの表示形式に依存します。

式と分数の詳細は第6章をご参照下さい。

プログラム中での分数

式の場合と同様に、プログラム中でも分数を取り扱いできます。数値は入力時の形式で表示されます。

プログラムの実行時は、分数表示モードがオンであれば分数が表示されます。INPUT プロンプトに分数を入力することもできます。プログラムの計算結果はそのときの表示形式となります。

/c 関数とフラグ 7、8、9 のクリアとセットにより、分数表示形式をプログラムで制御できます。第 14 章「フラグ」もご参照下さい。

プログラムと分数については第 13 章、第 14 章も参考にして下さい。

6

式の入力と計算

式の使いかた

HP 35s で式(equation)を利用するにはいくつかの方法あります。

- 計算のための式（この章）
- 未知の変数を求める式（第７章）
- 積分に利用する式（第８章）

例: 式による計算

まっすぐな断面を持つ円筒の体積をいくつも計算する必要があるとします。

$$V = .25\ \pi d^2\ l$$

d は円筒の直径、l は円筒の長さです。

✓ 直径や長さが変わるたびに、何度も同じキーを押せば計算することは可能です。たとえば、[.][2][5][ENTER][↶][π][×][2][.][5][↷][x²][×][1][6][×] と押せば直径 2.5 cm で長さが 16 cm であれば、体積が 78.5398 [cm³] だとわかります。しかし、式を利用すれば直径、長さ、体積の関係を HP 35s に**記憶させる**ことができます。これで何度も同じキーを押す必要はありません。

式モード(**Eq**uatio**n** mode)に設定するには次のように操作します。

キー	表示	説明
[EQN]	EQN□LIST□TOP または入力済みの式が第２行に表示されます	式モードを選択します。 表示記号 **EQN** が出てきます。
[RCL]		新しい式を入力します。 [RCL]ボタンを押すと表示記号 **A..Z** が出て変数名を入力できるようになります。
[V][↶][=]	V=_	[RCL][V] で V を入力します

[.] [2] [5]	V= 0.25_	数値の入力のため "_" カーソルが出てきます
[×] [↶] [π] [×]	V=0.25×π×_	[×] を押すとその数値の入力が完了します
[RCL] [D] [y^x] [2]	V=0.25×π×D^ 2_	[y^x] で ^ が入力されます
[×] [RCL] [L]	V=0.25×π×D^2×L_	
[ENTER]	V=0.25×π×D^2×L	入力を完了し、式が表示されます。
[↶] [SHOW]	CK=49CA LN=14	チェックサムと式の長さが表示されます。入力ミスを確認できます。

入力した式のチェックサム(CK、**c**heck**k**sum)と長さ(LN、l**en**gth)を比較することで、入力に誤りが無いかどうかを確認できます。(参考: この章の最後の「式の確認」)

式の評価の手順(*V* の計算):

キー	表示	説明
[ENTER]	D? 値	式の右辺の変数を入力するプロンプトが出ます。プロンプトに従って、最初に *D* を入力します。ディスプレイには *D* の現在の値が表示されます。
[2] [.] [1] [.] [2]	D? 2 1/2_	長さ $2\,{}^{1}/_{2}$ を分数で入力します。
[R/S]	L? 値	*D* の入力が終わると *L* の入力が促されます。*L* の現在値も表示されます。
[1] [6] [R/S]	V= 78.5398	*L* を保存し、体積 *V* を計算します。

式の操作のまとめ

作成した全ての式は「式リスト」に保存されます。このリストは式モードをオンにすると表示されます。

式の入力には確実なキー入力が求められます。チェックの方法などの詳細はこの章で後述します。

式リストでは一度に２つの式が表示されます。現在アクティブな式はディスプレイ第２行のほうです。

キー	解説
EQN	式モードの開始と終了。
ENTER	表示中の式を評価します。式が**代入式**であれば右辺を評価して結果を左辺に代入します。式が**等価式**であれば XEQ キーを押したのと同様に動作します。（この章の「式の種類」を参照して下さい。）
XEQ	表示中の式を評価します。式中に “=” があれば “–” に変換して計算します。
SOLVE	指定した未知の変数を求めます。（詳細は第 7 章）
↶ /	表示中の式を、指定した変数で積分します。（詳細は第 8 章）
←	表示中の式を削除するか、カーソルの左側の要素を削除します。
< または >	表示中の式の編集に入ります。カーソルの移動のみであり、内容の削除は行われません。
↷ < または ↷ >	現在の式の表示をスクロールさせます。
^ または v	式リストを上下に移動させます。
↷ ^ または ↷ v	式リストの最上位か最下位に移動します。
↶ SHOW	式のチェックサム（確認のための値）と長さ（メモリのバイト数）を表示します。
↶ UNDO	最後に削除された要素や式を再現します。
C	式モードの終了

式はプログラムでも利用できます。詳細は第 13 章をご覧下さい。

式リストに式を入力

式リスト（equation list）は入力済みの式の集合です。リストはメモリに保存されています。式を入力すると、自動的に式リストに追加されていきます。

式の入力方法:

メモリ量の制限を除き、式リストには何個でも追加できます。

1. まず、通常の操作モードにあることを確認して下さい。通常の操作モードではたいてい、ディスプレイに数値が表示されています。たとえば変数のやプログラムの一覧を表示しているときは式を入力できません。
2. EQN を押します。式モードがアクティブであることを示す表示記号 **EQN** と式リストが表示されます。
3. 式の入力を開始します。その前まで表示されていた式が移動して、入力中のものに置き換わります。その前まで表示されていた式には影響はありません。入力ミスがあったら必要に応じて ← か ↶ UNDO キーを使います。
4. 入力を完了するため ENTER を押すと、式リストに保存されます。(代わりに C を押すと、式が保存されてから式モードが終了します。)

式には変数、数値、ベクトル、関数、カッコを含めることができます。詳細は次節の例で解説します。

式の中の変数

式の中では全ての変数、*A* から *Z*、および *(I)* と *(J)* が利用できます。それぞれの変数に使用回数の制限はありません。(変数 *(I)* と *(J)* の詳細は第 14 章「変数とラベルの間接呼び出し」をご参照下さい。)

式に変数を入力するには RCL ***変数*** と操作します。RCL ボタンを押したあと、表示記号 **A..Z** が出てきて変数キーを押すことができる状態になります。

式の中の数値

式の中には有効な数値(2 進数、8 進数、16 進数、実数、複素数、分数)であれば全て入力できます。数値は常に表示形式 ALL で 12 文字まで表示されます。

式で数値を入力するときも計算のときと同じキー(.、+/−、E など)を使います。ただし、引き算で +/− は使えません。

式の中の関数

HP 35s のたくさんの関数が式の中に入力できます。入力可能な関数の一覧は、この章の「式で利用できる関数」に登場します。また、「付録 B 操作もくじ」にも掲載されています。

式の中に関数を入力するときも、計算のときと同じキーを使います。

- 式では、"+" や "÷" のような**中置**演算子(infix operator) は引数の**間に**入れます。
- “cos” や “ln” のような**前置**関数(prefix function)は、関数名に続けて引数を入力します。これらの関数キーを押すと必要に応じて一組のカッコが入り、左カッコの直後に入力カーソルが移動します。これにより、関数キーに続けてそのまま引数を入力できます。
- 関数に複数の引数がある場合は [↶] [0] で分割してください。

式の中のカッコ

カッコを使えば、計算の順番を制御できます。カッコを挿入するには [( )] を押して下さい。(詳細はこの章の「演算の順序」をご覧下さい。)

例: 式の入力

式 $r = 2 \times c \times (t - a) + 25$ を入力してみます。

キー	表示	説明
[EQN]	V=0.25×π×D^2×L	式リストで最後に使った式が表示されます。
[RCL] [R] [↶] [=]	R=_	変数 *R* で新しい式の入力を開始
[2]	R= 2_	数値を入力
[×] [RCL] [C] [×]	R=2×C×_	中置演算子を入力
[( )]	R=2×C×(_)	カッコを一組入力
[RCL] [T] [−] [RCL] [A] [▷] [+] [2] [5]	=2×C×(T−A)+ 25 _	カッコの中身を入力し、カーソルを移動。さらに最後の数値を入力。
[ENTER]	R=2×C×(T−A)+25	式の入力を終了し、式全体を表示

キー	表示	説明
↩ SHOW	CK=9E5F LN=14	チェックサムと長さを表示
C		式モードの終了

式の表示と選択

式リストには 2*2 lin. solve と 3*3 lin. solve という、2つの組み込み済みの式があります。式リストの表示と選択は次のように操作します。

式の表示:

1. まず EQN を押します。式モードがオンになり、表示記号 **EQN** が表示されます。ディスプレイには式リストが表示されます。
   - EQN LIST TOP　式リストのポインタがトップのであることを示しています。
   - 現在の式（最後に表示していた式）
2. 式リストは ^ または v でスクロールできます。リストの最後にスクロールすると次はリストの最初にもどります。EQN LIST TOPはリストの最初を意味します。

長い式を表示させる

1. まず、前述のように式リストを表示させてください。式は 14 文字までしか一度に表示できないため、これを超えていたら表示記号 ➡ が出てきます。
2. 表示中の式を先頭から編集するには > キーを押します。式の末尾から編集するには < です。編集する位置まで < や > でカーソルを移動させて下さい。このときも、式が表示しきれなければ表示記号⬅ ➡が登場します。
3. ディスプレイの端から反対の端まで移動させるには ↩ < あるいは ↩ > を使います。

式の選択:

まず、前述のように式リストを表示させます。ディスプレイ第2行に表示されている式が、どんな場合も操作対象の式になります。

例: 式の閲覧

キー	表示	説明
EQN	R=2×C×(T-A)+25	式リストで編集対象の式を表示
>	R=2×C×(T-A)+25	式の左端にカーソルが出ます
ENTER <	=2×C×(T-A)+25_	カーソルの右端にカーソルを移動
C		式モードの終了

式の編集と削除

入力中の式は、そのまま編集あるいは削除(クリア)ができます。式リストでも編集または削除が実行できます。ただし、組み込み済みの式 2*2 lin. solve と 3*3 lin. solve は削除できません。これらの式の間に新たな式を挿入しようとすれば、3*3 lin. solve の後に挿入されます。

入力中の式を編集するには:

1. カーソルの移動には < または > キーを使います。カーソルの直前の位置に文字が入力できます。
2. カーソルの移動後、← を押すとカーソルの直前にある数値や関数を削除できます。式の編集行に何も入っていないときは← を押しても何も変化はありませんが、このときに ENTER キーを押すと式が削除され、ディスプレイにはその前の式（式リストの順番で）が表示されます。
3. 式リストへの保存は ENTER (または C) を押します。

入力済みの式を編集するには:

1. まず、目的の式を表示し、> を押して編集カーソルを式の先頭でアクティブにします。式の後端でカーソルをアクティブにするには < を押します。（前述の「式の表示と選択」の通りです。）
2. 編集カーソルがアクティブな状態で、新しい式の入力における操作と同じように編集できます。
3. 編集した式を式リストに保存するには ENTER (または C) を押します。以前のものに上書きされます。

式の編集でメニューを使う:

1. 式の編集時に設定メニュー（MODE, や →DISPLAY、 →CLEAR など）を呼び出すと編集状態が終了します。
2. 式の編集時に挿入メニューあるいは閲覧メニュー（L.R.、 →$\bar{x},\bar{y}$、 →S,σ、 →SUMS、 →BASE、 →LOGIC、 R↓、 →MEM、 →CONST など）を呼び出すと挿入のあとに編集モードに戻ります。
3. 式の編集モードでは $x?y$、FLAGS、 →$x?0$ は利用できません。

保存済みの式を削除:

式リストをスクロールさせて目的の式をディスプレイ第2行に表示させ、← を押します。

全ての保存済みの式を削除:

EQN モードで →CLEAR と操作し、3 (3EQN)を選択します。CLR EQN? Y N というプロンプトがでます。最後に ‹ (Y) ENTER と操作します。

例: 式の編集

前の例の式で、数値 25 を削除します。

キー	表示	説明
EQN	R=2×Cx(T-A)+25	目的の式を式リストに表示
‹	=2×Cx(T-A)+25_	式の後端でカーソルをアクティブにします
←←←	=2×CxCOS(T-A)_	+25 を削除
ENTER	R=2×Cx(T-A)	式リストに保存
C		式リストを終了

式の種類

HP 35s では3種類の式が利用できます。

- **等式(Equality equation)** ひとつの "=" を含む式で、左辺がひとつ以上の変数を含むものです。たとえば、$x^2 + y^2 = r^2$ は等式です。
- **代入式(Assignment equation)** ひとつの "=" を含む式で、左辺がひとつの変数のみの式です。たとえば $A = 0.5 \times b \times h$ は代入式です。
- **式(Expression equation)** "="を含まない式です。たとえば $x^3 + 1$ は式です。

式により計算するときは、全ての種類の式が使用できますが、式の種類によって評価方法が異なります。未知の変数を計算するときは等式か代入式を使うと良いでしょう。積分を利用するときは式(expression)を使います。

式の評価

式の機能で便利な機能が、式から数値を計算する「評価(evaluate)」です。（式を解いたり、積分する機能は第 7 章と 8 章をご参照下さい。）

等式（"=" をもつ式）で左辺と右辺に差がある場合、式の「基本値」は左辺と右辺の差になります。"=" は “−” として取り扱われます。この基本値で等式の左右のバランスを計測することができます。

HP 35s には評価のためのキーが2つ、[ENTER] と [XEQ] があります。等式の場合は、他の種類の式と結果が少し異なります。

- [XEQ] は式の種類に関わらずその式の値を出力します。
- [ENTER] は等式**以外**の場合に式の値を出力します。等式の場合は右辺の値のみを出力し、その値を左辺の変数に代入(enter)します。

次表では式の評価の方法を2種類、掲載します。

式の種類	ENTERの結果	XEQの結果
等式: $g(x) = f(x)$ 例: $x^2 + y^2 = r^2$	$g(x) - f(x)$ $x^2 + y^2 - r^2$	
代入式: $y = f(x)$ 例: $A = 0.5 \times b \times h$	$f(x)$ * $0.5 \times b \times h$ *	$y - f(x)$ $A - 0.5 \times b \times h$
式: $f(x)$ 例: $x^3 + 1$	$f(x)$ $x^3 + 1$	
* 計算結果が式の左辺の変数(この例では A)に保存されます。		

式の評価:

1. 目的の式を表示させます。(前述「式の選択と表示」)
2. ENTER または XEQ を押します。 すると、必要な変数のための数値入力を求めるプロンプトが出てきます。(式の基本値が変化するときはその結果を自動的に保存します。.)
3. それぞれのプロンプトで求められる数値を入力します。
   - 入力した数値が正しければそのままR/Sキーを押します。
   - 正しくないときは訂正してからR/Sキーを押します 。(この章で後述する「式のプロンプト」もご参照下さい。)

計算を中断するにはCまたはR/Sを押します。ディスプレイ第2行にINTERRUPTEDと表示されます。

式の評価では基本的に、スタックに数値を保存しません。式の中と変数のみが影響されます。ただし、式の基本値はXレジスタに保存されます。

式の評価にENTERキーを使う

式リストにある式が表示されている状態では、ENTERキーを押すことでその式を評価できます。(式の入力中にENTERを押しても、式が保存されるだけで評価はされません。)

- 式が代入式の場合は、右辺のみが評価され、その結果はXレジスタと左辺の変数に保存されます。 変数はディスプレイに保存されます。ENTER は基本的に、左辺の変数を計算するためのものです。

■ 等式や式(expression)の場合は[XEQ]キーと同じで、その式の全てが評価されます。結果は X レジスタに保存されます。

例: ENTER キーで式を評価

この章の最初の方にある、円筒の体積を求める式を使います。

キー	表示	説明
[EQN] (必要であれば [^])	V=0.25×π×D^2×L	目的の式を表示
[ENTER]	D? 2.5	代入式の評価を開始。結果は変数 *V* に保存されます。右辺の変数の入力を求めるプロンプトが出てきます。変数 D の現在値は 2.5 [mm]です。
[3][5][R/S]	L? 16	*D* を入力すると次は *L* です。*L* の現在値は 16 [mm]です。
[2][0][×][1][0][0] [0][ENTER]	V=	長さ *L* を入力すると体積 *V* を計算後、変数 *V* に保存し、表示します。
[R/S]	19,242,255.0033	
[÷][1][E][6] [ENTER]	19.2423	立方ミリメートルから立方リットルに変換します。ただし、変数 *V* の値は変化しません。

式の評価に XEQ キーを使う

式リストに表示されている式は[XEQ]キーでも評価できます。このキーでは式の種類によらず、その式の全てが評価され、結果が X レジスタに保存されます。

例: XEQ キーで式を評価

前回の例と同じ式を使います。ただし、今回は円筒の直径を 35.5 [mm] としたときとの差を求めます。

キー	表示	説明
[EQN]	V=0.25×π×D^2×L	目的の式を表示

XEQ	V? 19,242,255.0033	計算するため式の評価を開始。以降、全ての変数の入力を求めるプロンプトが登場します。
R/S	D? 35	*V* はそのままにします。*D* を入力。
3 5 · 5 R/S	L? 20,000	新しい *D* を保存。続けて *L* を入力。
R/S	-553,705.7051	*L* はそのままにして計算します。左辺と右辺の差が表示されます。
÷ 1 E 6 ENTER	-0.5537	立方ミリメートルをリットルに変換。

式の基本値は、古い体積(V)と新しい体積(新しい *D* を使って計算された値)との差になります。この例では古い体積の方が小さかったことを意味しています。

式のプロンプト

式を評価すると、必要な変数の入力プロンプトが登場します。プロンプトでは変数名とその現在値が `X?2.5000` のように表示されます。名前のない間接変数 (I) や (J) にはプロンプトは出ません。この場合は自動的に間接変数の現在値が使われます。（参照：第 14 章）

- **値の変更がないとき:** R/S を押します。
- **値を変更するとき:** 新しい値を入力し R/S を押します。新しい値は X レジスタの古い値に上書きされます。値の入力には関数も使えます。関数を使った場合は R/S ボタンで入力値を計算します。たとえば、RPN モードでは 2 ENTER 5 y^x R/S、ALG モードで 2 y^x 5 ENTER R/S と操作します。ENTER キーを押す前にはディスプレイ第２行にその計算式が表示されます。ENTER キーを押した後は計算結果になります。
- **プロンプトをキャンセルするとき:** C を押します。変数の現在値が X レジスタに残り、その X レジスタがディスプレイ第２行に表示されます。数値の入力中に C キーを押すと、入力値がクリアされゼロになります。プロンプトのキャンセルにはもう一度 C を押します。
- **プロンプトによって隠された数字を表示するとき:** ↶ SHOW と操作します。

RPN モードでは、プロンプトが出るたびに変数の現在値が X レジスタにコピーされますが、スタックの移動は発生しません。プロンプトに従って数値を入力すると、その値は X レジスタにも入ります。[R/S]を押すとスタックの移動が発生し、計算された数値はスタックに保存されます。

式の文法

ここでは、式が評価されるときのルールを解説します。

- 演算子の間の相互作用
- 式の中で利用できる関数
- 文法エラーのチェック

演算の順番

式の中の演算は次の順序になります。

順番	演算	例
1	カッコ	(X+1)
2	関数	SIN(X+1)
3	指数 (y^x)	X^3
4	単項マイナス ($+/-$)	-A
5	乗算と除算	X×Y, A÷B
6	加算と減算	P+Q, A−B
7	等号	B=C

たとえば、全ての演算子がカッコの中に入っていたら、カッコの中の演算が優先されます。

例:

式	意味
A×B^3=C	$a \times (b^3) = c$
(A×B)^3=C	$(a \times b)^3 = c$
A+B÷C=12	$a + (b/c) = 12$
(A+B)÷C=12	$(a + b) / c = 12$
%CHG(T+12,A-6)^2	[%CHG ($(t + 12)$, $(a - 6)$)]^2

式で使える関数

式で利用できる関数は下表の通りです。付録 G「操作もくじ」にも掲載してあります。

LN	LOG	EXP	ALOG	SQ	SQRT
INV	IP	FP	RND	ABS	!
SGN	INTG	IDIV	RMDR		
SIN	COS	TAN	ASIN	ACOS	ATAN
SINH	COSH	TANH	ASINH	ACOSH	ATANH
→DEG	→RAD	HMS→	→HMS	%CHG	XROOT
→L	→GAL	→MILE	→KM	nCr	nPr
→KG	→LB	→°C	→°F	→CM	→IN
SEED	ARG	RAND	π		
+	–	×	÷	^	
sx	sy	σx	σy	$\bar{x}$	$\bar{y}$
$\bar{x}$ w	$\hat{x}$	$\hat{y}$	*r*	*m*	*b*
n	Σx	Σy	Σx^2	Σy^2	Σxy

✔ 1つまたは2つの引数をとる前置関数を入力すると、自動的にカッコが一組追加されます。

✔ 2つの引数をとる前置関数は %CHG、XROOT、IDIV、RMDR、nCr、nPr です。引数の間はカンマで区切ります。

XROOT の引数は RPN モードと式では反対になります。たとえば –8[ENTER]3 [x√y] は XROOT(3,-8) です。

RPN モードでは、2つの引数をとる関数は全て、その順番が Y, X となっています。たとえば 28[ENTER] 4 [↩] [nCr] は式では nCr(28,4)です。

2つの引数をとる関数では、第2の引数が負の値の場合は注意して下さい。次の場合は負の引数を入力できます。

```
%CHG(-X,-2)
%CHG(X,(-Y))
```

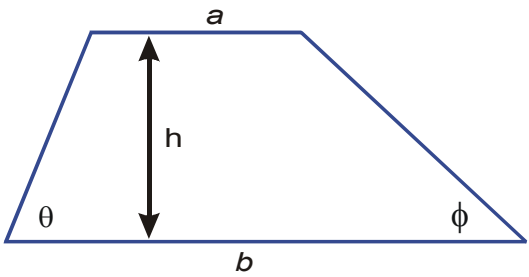
次の 8 個の関数が通常の操作と式に入力するときで異なる名前がついています。

RPN 操作	式での関数
x^2	SQ
$\sqrt{x}$	SQRT
e^x	EXP
10^x	ALOG
1/x	INV
$\sqrt[x]{y}$	XROOT
y^x	^
INT÷	IDIV

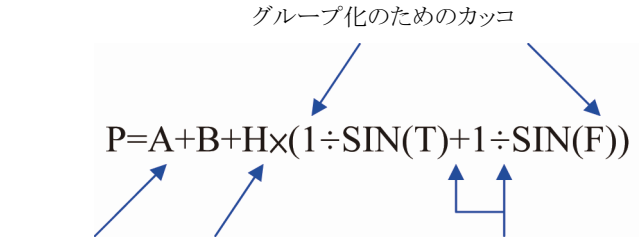
例: 台形の辺の長さ

次の式で台形の辺の長さが計算できます。教科書には次のように掲載されています。

$$\text{辺の長さ} = a + b + h\left(\frac{1}{\sin\theta} + \frac{1}{\sin\phi}\right)$$



HP 35s のルールに従って、次のように式を入力します。



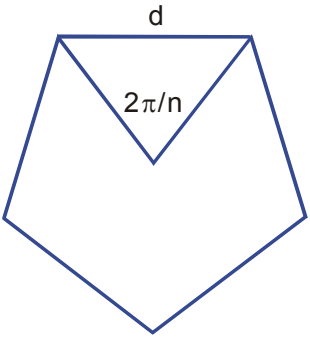
次のように入力することもできます。ここでは除算 `1÷SIN(T)` の代わりに逆関数 `INV(SIN(T))` を使っています。sin 関数は inv 関数の中に「入れ子（ネスト）」になっている点にご注意下さい。（inv は [1/x] で入力します。）

```
P=A+B+H×(INV(SIN(T))+INV(SIN(F)))
```

例: 正多角形の面積

一辺の長さが *d* である正 *n* 角形の面積は次のようになります。

$$\text{面積} = \frac{1}{4} n d^2 \frac{\cos(\pi / n)}{\sin(\pi/n)}$$



式はたとえば、次のようになります。

```
A=0.25×N×D^2×COS(π÷N)÷SIN(π÷N)
```

目的の式を入力するとき、演算子と関数の組合せをどう入力するか確認して下さい。

入力時のキーストロークは次のようになります。

EQN RCL A ↶ = · 2 5 × RCL N × RCL D y^x 2 ×
COS ↶ π ÷ RCL N › ÷ SIN ↶ π ÷ RCL N ENTER

文法エラー

式の文法は評価されるとき、はじめてチェックされます。もしエラーがあればエラーメッセージ SYNTAX ERROR が表示され、最初のエラーが発生した位置にカーソルが表示されます。式を編集するかエラーを補正して下さい。(この章で前述の「式の編集と削除」もご参照下さい。)

評価するまではエラーがあっても良いので、エラーになる可能性がある式でも作成できます。この点は第 13 章で解説するプログラムで特に便利です。

式の確認

式を閲覧しているとき(入力しているときではありません)、↶ SHOW でその式のチェックサム(CK, **c**heck**s**um)と長さ(LN, **le**n**g**th)を確認できます。SHOW キーは押し続けて下さい。

チェックサムは4桁の 16 進数で、その式が持つ独自の値です。式を間違って入力したときはチェックサムが異なってきます。長さはその式のメモリ使用量のバイト数です。

チェックサムと長さにより、入力が正しかったかどうかを確認できます。このマニュアルの例と、お手元の HP 35s の結果は一致するはずです。

例: チェックサムと長さを表示

この章の最初の方で出てきた、円筒の体積の例を使ってチェックサムと長さを確認します。

キー	表示	説明
EQN (必要であれば ^)	V=0.25×π×D^2×L	目的の式を表示させます。
↶ SHOW (押し続ける)	CK=49CA LN=14	式のチェックサムと長さを表示

(離す)	V=0.25×π×D^2×L	式を再表示
C		式モードの終了

7

式を解く (Solve)

第 6 章では、代入式 で ENTER キーを使って左辺の変数に計算結果を代入する方法を解説しました。SOLVE を使えば、全ての種類の式の、全ての変数を計算できます。

たとえば次の式を考えます。

$$x^2 - 3y = 10$$

y が既知であれば、SOLVE で未知数 x を求めることができます。x が既知の場合も同様に未知数 y を計算できます。次のような文章問題にも利用できます。

利掛け x 仕入原価 = 売価

この式の3変数のうち、任意の2変数がわかれば SOLVE で残りの変数を求めることができます。

式に変数がひとつしかない場合や、未知数がひとつのみの場合は、SOLVE はその式の根(root)を計算します。式の根を計算できるのは、等式 や代入式で左辺と右辺が完全に等しいときと、等号のない式(expression)でその式がゼロのときになります。

式を解く方法

変数を求めるために式を解くには、次のように操作します(組込み関数の式は別です)。

1. EQN を押して目的の式を表示します。必要であれば第 6 章「式リストに式を追加」の方法で式を追加します。
2. ↷ SOLVE と操作し未知数を入力します。たとえば ↷ SOLVE X だと x について解きます。その他の変数があれば、その値を求めるプロンプトが出てきます。
3. それぞれのプロンプトで必要な数値を入力します:
   - 表示された値が正しければそのまま R/S です。
   - 異なっていれば再入力するか、計算させるためにキー R/S を押します。(詳細: 第 6 章「式のプロンプト」)

計算中に C または R/S を押すと計算を中断できます。

根が見つかれば、指定された変数に保存され、ディスプレイにも表示されます。さらに X レジスタにはその根が、Y レジスタにはゼロまたは直前の推定値が、Z レジスタには D 値の平方根（これはゼロになるはず）が入ります。

数学的に込み入った状態の式では、根が見つからないことがあり、NO ROOT FOUND というメッセージがでます。詳細はこの章で後述の「結果の確認」と「付録 D 結果の調査」、「付録 D SOLVE が根を発見できなかったときは」をご参照下さい。

式を未知の変数について解くとき、適切な**初期推定値**があれば演算が容易になります。適切な初期推定値により計算速度が向上し、求める解に直接近づけることができ、さらに、状況によっては複数の解を発見できることもあります。詳細はこの章の「SOLVE の初期推定値を選択」をご参照下さい。

例: 線形運動方程式を解く

物体の自由落下の運動方程式は下記のようになります。

$$d = v_0 t + {}^1/_2 g t^2$$

ただし、d は距離、v_0 は初速度、t は時間、g は重力加速度です。

式の入力:

キー	表示	説明
f CLEAR 3 (3ALL) v (Y)		メモリをクリア
EQN	3*3□lin.□solve EQN LIST TOP	式モードに入ります。
RCL D g = RCL V × RCL T + . 5 × RCL G × RCL T y^x 2	D=V×T+_ ←=V×T+0.5×G×T^ 2_	式を入力
ENTER	D=V×T+0.5×G×T^2	入力を完了。式の左端から表示されます。
g SHOW	CK=FB3C LN=15	チェックサムと長さ

g (重力加速度) は、単位系が変化したときのために変数として入力します。
(9.8 m/s^2 や 32.2 ft/s^2 など。)

静止状態から 5 [秒] 落下したときの移動距離をメートル単位で求めます。すでに式モードに入っていて、目的の式が表示された状態ですので、そのまま SOLVE を実行して *D* を求めます。

キー	表示	説明
[↷] [SOLVE]	SOLVE_	未知数を指定します
[D]	V?	未知数として *D* を選択。
	value	続いて変数 *V* のプロンプト。
[0] [R/S]	T?	*V* に 0 を入力。
	value	続いて変数 *T* のプロンプト。
[5] [R/S]	G?	*T* に 5 を入力。
	value	続いて変数 *G* のプロンプト。
[9] [.] [8] [R/S]	SOLVING	*G* に 9.8 を入力。
	D=	未知数 *D* について解きます。
	122.5000	

同じ式で別の問題を解いてみます。ある物体が静止状態から 500 [m] 落下するまでにかかる時間を求めます。

キー	表示	説明
[EQN]	D=V×T+0.5×G×T^2	式を表示
[↷] [SOLVE] [T]	D?	未知数として *T* を選択。
	122.5	続いて変数 *D* のプロンプト。
[5] [0] [0] [R/S]	V?	*D* に 500 を入力。
	0	続いて変数 *V* のプロンプト。
[R/S]	G?	*V* は 0 のままにします。
	9.8	続いて変数 *G* のプロンプト。

R/S	SOLVING T= 10.1015	G は 9.8 のままにします。 未知数 *T* について解きます。

例: 理想気体の状態方程式を解く

理想気体では次の方程式が成り立ちます。

$$P \times V = N \times R \times T$$

ただし、*P* は圧力(Pa または N/m^2)、*V* は体積(リットル)、*R* は気体定数(0.0821 litter–atm/mole–K または 8.314 J/mole–K)、温度 *T*(ケルビン: K=°C + 273.1)です。

式を入力:

キー	表示	説明
EQN RCL P ×	P×_	式モードに入り、新しい式を作成
RCL V ◀ =		
RCL N ×		
RCL R × RCL T	P×V=N×R×T_	
ENTER	P×V=N×R×T	入力完了
◀ SHOW	CK=EDC8 LN=9	チェックサムと長さ

2リットルの容器に 0.005 [mol] の二酸化炭素が入っていて、その温度は 24°C です。この二酸化炭素を理想気体として圧力を計算して下さい。すでに式モードに入っていて目的の式がディスプレイに表示されているため、そのまま SOLVE できます。

キー	表示	説明
▶ SOLVE P	V? *value*	未知変数として *P* を指定。続いて変数 *V* のプロンプト。
2 R/S	N? *value*	*V* に 2 を入力。 続いて変数 *N* のプロンプト

[.] [0] [0] [5] [R/S]	R? *value*	*N* に .005 を入力。 続いて変数 *R* のプロンプト
[.] [0] [8] [2] [1] [R/S]	T? *value*	*R* に .0821 を入力。 続いて変数 *T* のプロンプト
[2] [4] [+] [2] [7] [3] [.] [1] [ENTER] [R/S]	T? 297.1000 SOLVING P= 0.0610	*T* をケルビンに変換 *T* に 297.1 を入力。 未知数 *P* について解きます。

5リットルのフラスコに窒素が入っています。気体の温度が 18°C のとき、圧力は 0.05 [atm] です。気体の密度を求めてください。(窒素の分子量 28 を使って、$N \times 28/V$ で密度を求めます)

キー	表示	説明
[EQN]	P×V=N×R×T	目的の式を表示
[↗] [SOLVE] [N]	P? 0.0610	未知変数として *N* を指定。続いて変数 *P* のプロンプト。
[.] [0] [5] [R/S]	V? 2.0000	*P* に .05 を入力。 続いて変数 *V* のプロンプト
[5] [R/S]	R? 0.0821	*V* に 5 を入力。 続いて変数 *R* のプロンプト
[R/S]	T? 297.1000	*R* はそのままにします。 続いて変数 *T* のプロンプト
✓ [1] [8] [ENTER] [2] [7] [3] [.] [1] [+]	T? 291.1000	*T* をケルビンに変換して入力
[R/S]	SOLVING N= 0.0105	*T* に 291.1 を入力。 未知数 *N* について解きます。

✓	2 8 ×	0.2929	グラム単位で質量を計算。 *N* × 28 です。
✓	RCL V ÷	0.0586	密度。[gram / liter]

組み込みの式を解く

組み込みの式は2つあります:

"2*2 lin. solve" (Ax+By=C,　Dx+Ey=F)

"3*3 lin. solve" (Ax+By+Cz=D,　Ex+Fy+Gz=H,　Ix+Jy+Kz=L)

どちらかを選んで ↗ SOLVE を押します。他の式と異なり XEQ、ENTER、/ キーは受け付けられません。SOLVE を実行すると、2*2 の場合は 6 変数(A から F)を入力して x と y を求めます。3*3 の場合は 12 変数(A から L)を入力して、x, y, z を求めます。結果は変数 x, y, z に保存されます。解の数が無限になる場合や、解のない場合でも取り扱うことができます。

例: 次の連立方程式を解きましょう

$$\begin{cases} x + 2y = 5 \\ 3x + 4y = 11 \end{cases}$$

キー	表示	説明
EQN	3*3 lin. solve EQN LIST TOP	式モードに入ります
⌄	EQN LIST TOP 2*2 lin. solve	組み込みの関数を表示
↗ SOLVE	A? value	変数 *A*
1 R/S	B? value	変数 *A* に 1 を入力。 続いて変数 *B*
2 R/S	C? value	変数 *B* に 2 を入力。 続いて変数 *C*
5 R/S	D? value	変数 *C* に 5 を入力。 続いて変数 *D*
3 R/S	E? value	変数 *D* に 3 を入力。 続いて変数 *E*

4 R/S	F?		変数 *F* に 4 を入力。
	value		続いて変数 *F*
1 1 R/S	X=	⬆	変数 *F* に 11 を入力。
	1.0000	⬇	*x* と *y* を求めます。
⌄	y=	⬆	*y* の値
	2.0000	⬇	

SOLVE の理解と制御

SOLVE は最初に、未知の変数を直接解くことを試みます。それができなかった場合は、反復計算 (iterative procedure) に切り替えます。この計算では2つの初期推定値を使って未知変数を探り出します。これらの推定値を使って、更により良い推定値を求めます。その繰り返しによって式がゼロになるような未知数を求めていきます。

,SOLVE が式を評価するときは、XEQ のときと同じ方法、つまり、全ての等号 " = " をマイナス記号 " – " に置き換える方法を使います。たとえば理想気体の例であれば、$P \times V - (N \times R \times T)$ となります。これにより、等式や代入式でも左辺と右辺をバランスさせることができ、等号無しの式は全体がゼロに等しいものとすることで、どの種類の式でも同様に根を求めることができます。

解くことが困難な式でも、ユーザが初期推定値をうまく指定すれば根を発見できることもあります。(後述の「SOLVE の初期推定値を選択」を参考にして下さい。) SOLVE が根を発見できなかったときは NO ROOT FND と表示されます。

SOLVE の動作の詳細は「付録 D」をご参照下さい。

結果の確認

SOLVE の計算が終わると、その結果はスタックに保存されます。

- X レジスタには計算した未知数の根が入ります。クリアするには C を押します。この数値を未知数に代入すると式はゼロになります。
- ✓ Y レジスタには根を求める直前の推定値が入ります。Y レジスタは R↓ で確認できます。この数値は X レジスタと同じになるはずですが、もし同じでなければ、X レジスタに入った根は**近似値** (*approximation*) になっています。このとき、X と Y レジスタは近い値になります。

- Z レジスタには根における D 値になります。(Z レジスタをディスプレイ第２行に表示させるにはもう一度 R↓ を押します。) 根が完全に求められたときはゼロになります。ゼロでない場合は根は近似値ですが、Z レジスタの値はゼロに近い値になっているはずです。

計算がNO ROOT FNDで終わったときは、根が見つからなかったことを示しています。(このメッセージをクリアするには C か ← を押します。) X レジスタと Y レジスタには、根の計算に使われた、最後の推定値が保存されます。Z レジスタには最後の推定値に置ける式の値が入ります。

- X レジスタと Y レジスタが近い値ではない場合や、Z レジスタがゼロに近い値ではないときは、X レジスタに入っている値も根では無い可能性が高いです。
- X レジスタと Y レジスタが近い値であり、かつ Z レジスタがゼロに近ければ、X レジスタの値は根の近似値である可能性が高いです。

SOLVE 計算の中断(Interrupt)

計算を中断するには C または R/S を押します。すると、INTERRUPTED というメッセージが出て、その時点で最も良い根の推定値が未知変数に保存されます。スタックに保存せずにこの推定値を確認するには ↶ VIEW と操作します。ただし、いったん閲覧すると SOLVE の再開はできません。

SOLVE の初期推定値を選択

2つの初期推定値は次のように決定されます。

- 現在、未知変数に保存されている数値
- X レジスタの数値

これらの数値は、推定値を入力したかどうかに関わらず計算で使われます。ひとつの推定値のみを変数に入力した場合は、第2の推定値は第1のものと同じ値になります。というのは、変数に入力した直後は X レジスタ(ディスプレイに表示されている数値)にも同じ値が入っているからです。(このような場合は2つの異なる推定値にするため、片方の推定値は自動的に少し変更されます。)

推定値を指定すると、次の利点があります。

- 解を発見するための検索範囲を狭めることで計算時間が減ります。

- 根が複数あるとき、推定値によって根の範囲を指定できます。たとえば線形な運動の式を考えます。

$$d = v_0 t + {}^1/_2 g t^2$$

t は２つの解があります。推定値を入力することで、求めたい根の範囲があらかじめ指定できます。

この章の最初の方の例では、変数 T を解くのに推定値を入力する必要はありませんでした。というのは、この例の最初で変数 T に数値を入力して変数 D について解いた後だったからです。T に残った値は条件変更の後についても良い(現実的な)値だったため、そのまま推定値として使われました。

- 式の性質上、未知変数について確定的な根がない場合、推定値によって根の発見が妨げられることがあります。たとえば：

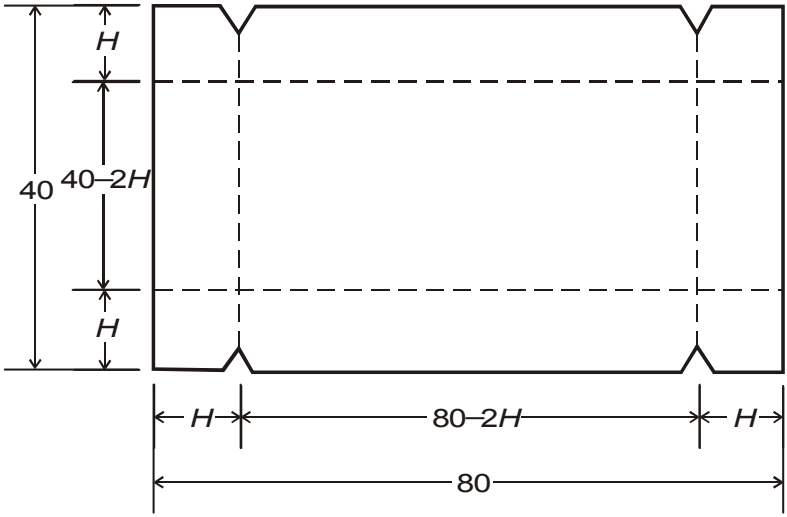
$$y = t + \log x$$

$x \leq 0$ のときはエラーになります。（メッセージ NO ROOT FND が出ます）

次の例では式の解が複数ありますが、推定値によって求める根を発見しています。

例: 根の発見のために推定値を使う

幅 40 [cm] 長さ 80 [cm] の板状の金属を使って、容積が 7500 [cm^3] のフタのない直方体の箱を作成します。箱の高さ H を求めて下さい。直方体ですから 4 辺は同じ長さになります。H は可能な限り大きくしてください。



高さを *H* としましたので、直方体の長さは (80 – 2*H*) で幅は (40 – 2*H*) です。容積 V は:

$$V = (80 - 2H) \times (40 - 2H) \times H$$

で、式を簡素化すると次の通りです。

$$V= (40 - H) \times (20 - H) \times 4 \times H$$

式を入力します。

キー	表示	説明
EQN RCL V ↶ =	V=_	式モードを開始し、式を入力。
( ) 4 0 − RCL H › × ( ) 2 0 − RCL H › × 4 × RCL H ENTER ↶ SHOW	V=(40-H)_ (40-H)×(20-H)_ H)×(20-H)×4×H_ V=(40-H)×(20-H CK=49A4 LN=19	式の入力を完了し、表示。 チェックサムと長さ

求める解としては、*H* が長くて背が高いものと、*H* が短くて背が低いものが考えられます。*H* が長い方を求めたいので、初期推定値に大きな値を指定すると良いでしょう。しかし板の幅は 40 [cm]であり、*H* は 20 [cm] より長くすることはできないため、初期推定値として 10 [cm] と 20 [cm] を指定します。

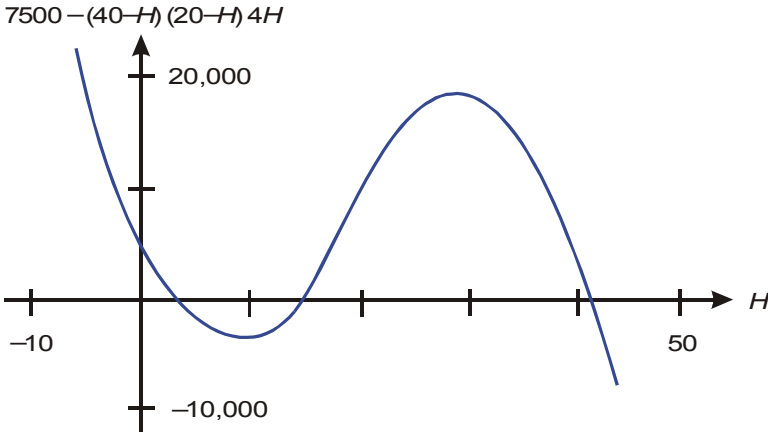
キー	表示	説明
C		式モードを終了
1 0 ↗ STO H		初期推定値を２つ入力
ENTER 2 0	20_	
EQN	V=(40-H)×(20-H	現在の式を表示
↗ SOLVE H	V?	*H* について解きます。
	value	続けて変数 *V* のプロンプト。
7 5 0 0 R/S	H=	変数 *V* に 7500 を入力。
	15.0000	*H* の根が出てきます。

次に、根の精度をチェックしましょう。つまり、正確な根であったかどうかを、Y レジスタに保存された直前の推定値と Z レジスタの式の値を確認するということです。

	キー	表示	説明
✓	R↓	15.0000	Y レジスタの値は最終結果を出す直前の推定値です。最終結果と Y レジスタの数値が一致しているので、完全な根と判断できます。
✓	R↓	0.0000	Z レジスタの値は、根を式に代入すると式がゼロになる、ということを示しています。

よって、求める直方体は 50 × 10 × 15 [cm] となります。もし、高さの上限(20 [cm])が無く、初期推定値に 30 [cm] と 40 [cm] を使えば *H* は 42.0256 [cm] になりますが、これは条件に合致しない根です。また、初期推定値 に 0 [cm] と 10 [cm] を使った場合は *H* は 2.9774 [cm] になりますが、これも出題の条件に合いません。

どのような推定値が良いか判断しにくいときは、グラフが手助けとなることも多いです。未知変数に数値を当てはめてみて XEQ ボタンを押すと、そのときの式の値が得られます。先ほどの例では容積 *V* を 7500 で固定にして、未知変数 *H* の値を x 軸に、式の値を y 軸にプロットすると次の図のようになります。式の値は右辺と左辺の差になるという点にご注意下さい。



もっと詳しく

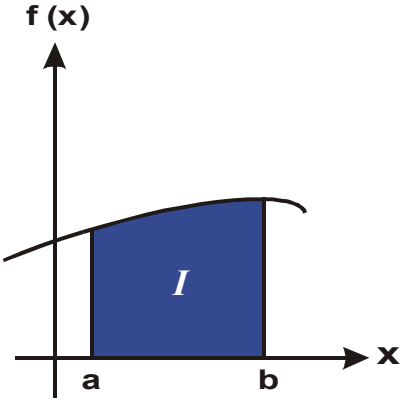
この章では、幅広い例をもとに未知変数や根の SOLVE についての手順を解説しました。付録 D では SOLVE のアルゴリズム、結果の解釈、根が見つからないときの原因、ならびに正しくない結果になってしまう場合の条件について解説します。

8

積分

数学、科学、工学などにおける多くの問題には、関数の積分の計算が要求されます。関数が *f(x)* で与えられたとき、区間 [a, b] における積分は次のように表現されます。

$$I = \int_a^b f(x)\,dx$$



関数 *f(x)* をプロットしたとき、*f(x)* とグラフの x 軸、区間をあらわす x = *a* および x = *b* で囲われた領域の面積が *I* になります。(ただし、*f(x)* は積分区間において負にならないものとします)

∫ ボタン(∫FN) は表示中の式を指定された変数 (∫FN d_) で積分します。式には複数の変数があっても問題ありません。

式を積分する (∫FN)

式を積分するには:

1. 被積分関数が式リストに保存されていなければ、入力し（参考: 第６章「式リストに式を入力」)、式モードを終了します。式の種類は等式のない式(expression)とします。

2. 次の操作で積分区間を入力します： 最初に**下限**を入力して [ENTER]、続けて上限を入力します。
3. 次のように式を表示させます： [EQN]を押し、必要に応じてスクロールさせ（[^] や [v]）、目的の式を表示させます。
4. 次のように積分する変数を選択します:: [↵] [/] 「変数名」と押します。すると計算が始まります。

[/] は他の計算よりも多くのメモリを消費します。もし [/] によってメッセージ MEMORY FULL が出たら、付録 B を参考にして下さい。

[C] か [R/S] を押すと積分の計算を中止できます。メッセージINTERRUPTEDがディスプレイ第2行に出ますが、積分は再開することができません。また、中断された場合は積分についての情報は一切残りません。

表示形式の設定は計算結果の表示だけではなく、精度にも影響します。表示形式が ALL の場合や、FIX、SCI、ENG で大きな桁が指定されていた場合は、精度は向上しますが、遙かに多くの計算時間がかかります。結果の不確定値はYレジスタに保存され、積分区間はTレジスタとZレジスタに押し出されます。詳細はこの章で後述する「積分の正確度」をご参照下さい。

同じ式で条件を変えて積分するには:

✓ 積分区間が同じであれば [R↓] [R↓] と操作して X レジスタと Y レジスタに上限値と下限値を移動させます。前述のステップ 3 から再開して下さい。積分区間を変えるにはステップ 2 から開始します。

異なる式を積分するにはステップ 1 からやり直して下さい。

例: ベッセル関数（Bessel Function）

次式はゼロ次の第1種ベッセル関数です。

$$J_0(x) = \frac{1}{\pi}\int_0^{\pi} \cos(x \sin t)dt$$

x が 2 および 3 のときのベッセル関数を計算して下さい。

まず、被積分関数を入力します。

cos (*x* sin *t*)

キー	表示	説明
[↗] [CLEAR] [3] (3ALL) [<] (Y)		メモリをクリア
[EQN]	3*3 lin, solve EQN LIST TOP	式モードに入ります。
[COS] [RCL] [X]	COS(X)	被積分関数を入力
[×] [SIN]	COS(X×SIN( )	
[RCL] [T]	COS(X×SIN(T))	
[>] [>]	COS(X×SIN(T))_	
[ENTER]	COS(X×SIN(T))	式の入力を完了すると、式の左端から表示されます。
[↖] [SHOW]	CK=E1EC LN=13	チェックサムと長さ
[C]		式モードを終了

入力した式を x = 2、区間 [0, π] で積分します。

キー	表示	説明
[MODE] [2] (2RAD)		角度をラジアンに設定
✓ [0] [ENTER] [↖] [π]	3.1416	積分区間を下限値、上限値の順で入力。
[EQN]	COS(X×SIN(T))	被積分関数を表示
[↖] [∫]	∫FN d_	積分の変数を指定するプロンプト
[T]	X? *value*	*X* の値を指定するプロンプト
[2] [R/S]	INTEGRATING ∫ = 0.7034	x = 2. 次の積分を開始。 $\int_0^{\pi} f(t)$
✓ [↖] [π] [÷]	0.2239	J_0 (2) の計算結果

続けて、同じ積分区間で $J_0(3)$ を計算します。積分の計算で得た結果を π で割る操作をしたので、スタックから積分区間 [0, π] が消えてしまいました。再入力して下さい。

	キー	表示	説明
✓	0 ENTER ◀ π	3.1416	積分区間を下限値、上限値の順で入力。
	EQN	COS(X×SIN(T))	被積分関数を表示。
	◀ /	∫FN d_	積分の変数を指定するプロンプト
	T	X? 2.0000	*X* の値を指定するプロンプト
	3 R/S	INTEGRATING ∫= -0.8170	x = 3. 次の積分を開始。 $\int_0^{\pi} f(t)$
✓	◀ π ÷	-0.2601	$J_0(3)$ の計算結果

✓ **例: 正弦積分**

通信理論における問題(たとえば理想的な経路におけるパルス伝達)では、正弦積分と呼ばれる、次のような積分の計算が求められます。

$$Si(t) = \int_0^t \left(\frac{\sin x}{x}\right) dx$$

Si (2) を計算しましょう。

被積分関数を入力します。

$$\frac{\sin x}{x}$$

積分の下限値である x = 0 で式を評価すると、エラー(DIVIDE BY 0) になります。しかし、積分区間が極端に小さかったり、サンプル点の数が極端に多かったりしない限り、積分のアルゴリズムでは通常、下限値ならびに上限値で式を評価することはありませんので問題ありません。

キー	表示	説明
EQN	3*3 lin. solve EQN LIST TOP	式モードに入る
SIN RCL X	SIN(X)	式を入力
▶	SIN(X)_	カッコの外に移動

÷ RCL X	SIN(X)÷X_	
ENTER	SIN(X)÷X	式の入力を終了
↶ SHOW	CK=0EE0 LN=8	チェックサムと長さ
C		式モードの終了

次に x について区間 [0, 2] で積分します。

キー	表示	説明
MODE 2 (2RAD)		角度をラジアンにします
0 STO X ENTER 2	2_	積分区間を下限値、上限値の順で入力。
EQN	SIN(X)÷X	対象の式を表示
↶ ∫ X	INTEGRATING ∫ = 1.6054	*Si*(2) の計算結果

積分の精度

HP 35s では、積分の計算で厳密解を求めることができないため、結果は近似値になります。近似値の正確度(accuracy)は被積分関数の計算に依存します。被積分関数の計算は、丸めと定数の精度に影響されます。

被積分関数がある一部で急激に変化する場合は、積分の計算が不正確になることがありますが、その可能性はとても低いです。むしろ、被積分関数の全体的な特性と、その取り扱いに関するテクニックが問題になるケースが多いです。この点は付録 E で検証します。

正確度の特定

ディスプレイの表示設定 (FIX、SCI、ENG、ALL) により、積分計算の正確度が決定されます。より多い桁の表示設定であれば、より精度は高くなります(それに伴って計算時間がかかります)。より少ない桁の表示設定では、計算時間は少なくなりますが、被積分関数の正確度は指定された桁までとなりますので、その分、誤差は大きくなります。

積分の正確度を特定するには、表示形式を計算結果に必要な桁を超えない桁に指定しておきます。この桁で積分計算の正確度・精度が決定されます。

分数モードがオンのとき(フラグ 7 がセットされているとき)は、その直前に使われていた表示設定が用いられます。

正確度の確認

積分の計算が終わったとき、推定した不確定値 (*uncertainty*) が Y レジスタに保存されます。Y レジスタの確認には [x⇄y] を押して下さい。

たとえば *Si(2)* の積分が 1.6054 ± 0.0002 だとしたら、0.0002 が不確定値です。

例: 正確度を指定

表示形式を SCI 2 とし、前回の例の *Si(2)* を積分します。

キー	表示	説明
[↶] [DISPLAY] [2] (2SCI) [2]	1.61E0	科学表記の２桁に設定し、被積分関数を小数点以下２桁の正確度とします。
✓ [R↓] [R↓]	0.00E0 2.00E0	区間として使った値を Z および T レジスタから、X および Y レジスタに移動して再利用します。
[EQN]	SIN(X)÷X	目的の式を表示
[↶] [∫] X	INTEGRATING ∫ = 1.61E0	積分値は小数点以下２桁で近似されます。
[x⇄y]	1.61E-2	近似の不確定値

積分値は 1.61±0.0161 です。この計算で得られた積分の近似値の小数点以下2桁まで(表示中の数値)には、不確定値の影響がないので、この近似値は正確であると言えます。

もし近似値の不確定値が許容される範囲を超えていたら、表示形式の桁を増やして再計算して下さい(被積分関数 *f(x)* をより高い正確度で再計算します)。一般的に、表示桁をひとつ増やすと、積分の不確定値もひと桁小さくなります。

例: 正確度を変更

直前の例の *Si(2)* で表示桁の設定を SCI 4 に変更してみます。

キー	表示	説明
↶ DISPLAY 2 (2SCI) 4	1.6079E-2	表示桁を SCI 4 に変更。最後に表示していた不確定値がディスプレイに残っています。
R↓ R↓	0.0000E0 2.0000E0	区間として使った値を Z および T レジスタから、X および Y レジスタに移動して再利用します。
EQN	SIN(X)÷X	目的の式を表示
↶ ∫ X	INTEGRATING ∫ = 1.6054E0	積分を実行
x⇄y	1.6056E-4	SCI 2 のときと比べて、不確定値が約 1/100 になりました。
DISPLAY 1 (2SCI) 4	0.0002	表示形式を FIX 4 に戻します。
MODE 1 (1DEG)	0.0002	角度モードを Deg に戻します。

この例では不確定値を参考にすれば、積分値が小数点以下 3 桁までしか正確とは言えないことになります。しかし実は、厳密解に比べても、今回の結果は **7 桁**の正確度になっています。不確定値は控えめに計算されているため、ほとんどの場合、近似値は不確定値が示すものより**やや正確度が高い**です。

もっと詳しく

この章では、幅広い例をもとに積分についての手順を解説しました。付録 E では積分のアルゴリズム、正しくない結果が出てしまうときの条件、計算時間がかかってしまうときの条件、ならびに計算の近似について解説します。

9

複素数

HP 35s では次の形式の複素数を扱えます。

x+iy x+yi r∠θ

複素数の四則演算(+, –, ×, ÷)、三角関数(sin, cos, tan)、数学関数(–z, 1/z, $z_1^{z_2}$, ln z, e^z) が利用できます。(z_1 と z_2 は複素数とします。)

ALG モードでは x+yi 形式のみが利用できます。

複素数を入力するには:

形式: x+iy

1. 実数部を入力
2. [i] を押す
3. 虚数部を入力

形式: x+yi

1. 実数部を入力
2. [+]を押す
3. 虚数部
4. [i] を押す

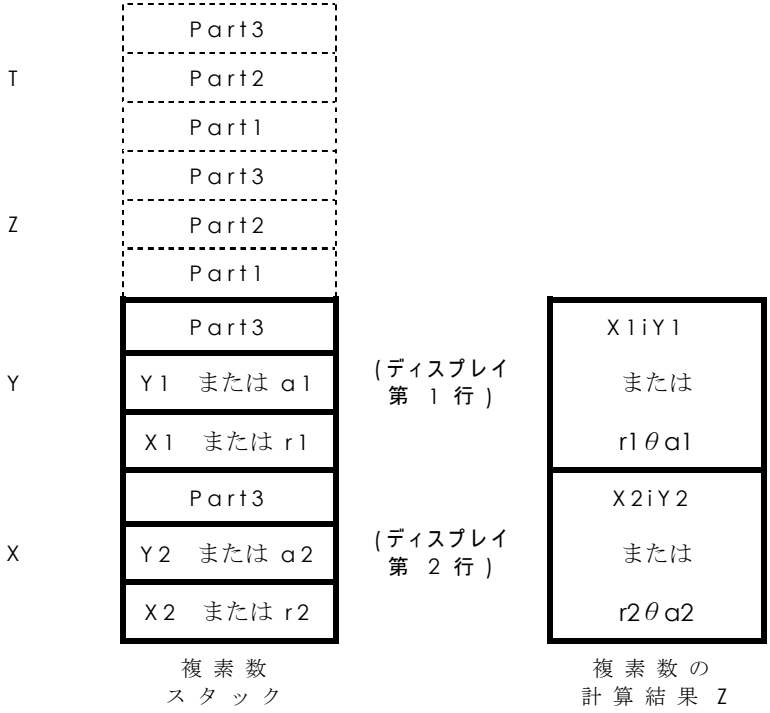
形式: r∠θ

1. r の値を入力
2. [↵] [θ] を押す
3. θ の値を入力

この章の例ではすべて、特に注意のない限り RPN モードを用いるものとします。

複素数スタック

複素数はスタックレベルの Part 1 と Part 2 を使います。RPN モードでは、X レジスタの Part 1 と Part 2 に保存された複素数がディスプレイ第2行に、Y レジスタの Part 1 と Part 2 に保存された複素数がディスプレイ第1行に表示されます。



複素数の計算

ALG モードならびに RPN モードで、複素数も実数と同じように計算できます。

複素数の計算

1. 前項の方法で複素数 z を入力
2. 関数を選択

複素数 z に利用できる単項演算子

目的	操作
符号の変更 –z	[+/−]
逆数 1/z	[1/x]
自然対数 ln z	[→] [LN]
指数関数 e^z	[→] [e^x]
sin(z)	[SIN]
cos(z)	[COS]
tan(z)	[TAN]
絶対値 ABS(z)	[→] [ABS]
偏角 ARG(z)	[←] [ARG]

✓ 2つの複素数を二項演算子で計算:

1. 最初の複素数 z_1 を前述の方法で入力
2. 第二の複素数 z_2 を前述の方法で入力
3. 二項演算子を選択

2つの複素数 z_1 と z_2 の計算

目的	操作
加算 $z_1 + z_2$	[+]
減算 $z_1 - z_2$	[−]
乗算 $z_1 \times z_2$	[×]
除算 $z_1 \div z_2$	[÷]
べき乗 $z_1^{z_2}$	[y^x]

例:

複素数の四則演算と三角関数の例です。

sin (2i3) の計算

キー	表示	説明
↺ DISPLAY 9 (9×iy)		表示形式の設定
2 i 3 SIN	9.1545i-4.1689	結果は 9.1545 *i* –4.1689

次の式を計算してみます。

$$z_1 \div (z_2 + z_3),$$

ただし、z_1 = 23 *i* 13、z_2 = –2i1、z_3 = 4 *i*– 3 とします。

次のような操作になります。

キー	表示	説明
↺ DISPLAY 9 (9×iy)		表示形式の設定
2 3 i 1 3 ENTER	23.0000i13.0000 23.0000i13.0000	z1 を入力
2 +/- i 1 ENTER	-2.0000i1.0000 -2.0000i1.0000	z2 を入力
4 i 3 +/- +	23.0000i13.0000 2.0000i-2.0000	$(z_2 + z_3)$ を計算。 結果は 2 *i* –2
÷	2.5000i9000	$z_1 \div (z_2 + z_3)$ を計算。 結果は 2.5 *i* 9

次に、(4 *i* –2/5)×(3 *i* –2/3) を計算します。

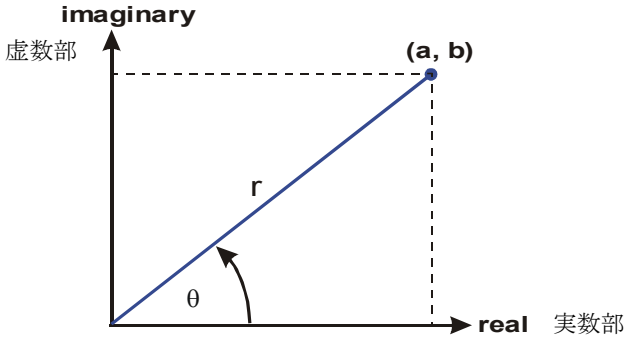
キー	表示	説明
↺ DISPLAY 9 (9×iy)		表示形式の設定
4 i · 2 · 5 +/- ENTER	4.0000i-0.4000 4.0000i-0.4000	4i-2/5 を入力
3 i · 2 · 3 +/-	4.0000i-0.4000 3i-0 2/3	3i-2/3 を入力
×	11.7333i-3.8667	結果は 11.7333i-3.8667

続けて、$e^{z^{-2}}$ を計算します。ただし、z = (1*i* 1) とします。

キー	表示	説明
1 i 1 ENTER	1.0000i1.0000 1.0000i1.0000	1i1 を入力
2 +/- y^x	0.0000i-5.0000	z^{-2} を計算。 結果は 0i-5
↻ e^x	0.8776i-0.4794	最終結果は 0.8776 *i*– 0.4794

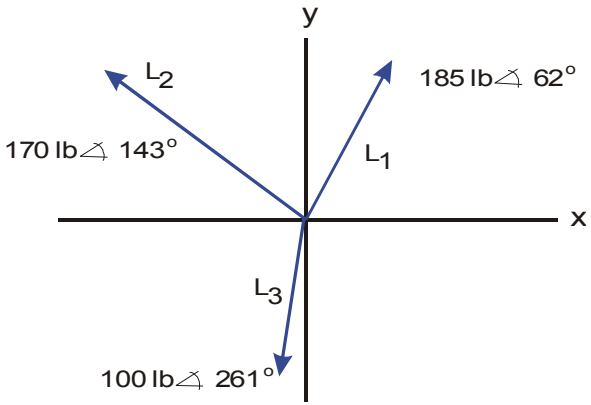
複素数と極座標

現実における多くの問題は、実数を極座標で取り扱います。この方式ではひと組の数値を複素数と同じように取り扱いますから、複素数の操作によって計算ができます。



例: ベクトル加算

次の3つの荷重の合計を求めます。
(訳注: 図の lb は力の単位「重量ポンド」で、1 [lbf] = 4.4482216152605 [N] です。)



キー	表示	説明
MODE 1 (1DEG)		角度を Deg に設定
◀ DISPLAY . 0 (10r8a)		複素数モードに設定
1 8 5 ▶ θ	185.0000θ62.0000	L_1 を入力
6 2 ENTER	185.0000θ62.0000	
1 7 0 ▶ θ	170.0000θ143.00...	L_2 を入力
1 4 3 ENTER	170.0000θ143.000➡	
1 0 0 ▶ θ	185.0000θ62.0000	L_3 を入力し、$L_2 + L_3$ を
2 6 1 +	151.4529θ178.660➡	計算
+	178.9372θ111.148➡	$L_1 + L_2 + L_3$ を計算
▶ >	⬅9	結果を表示させるため スクロール

複素数の表示形式を、この例と違うものを用いて計算することもできますが、結果の表示は DISPLAY メニューでの設定に依存します。

次に、1i1+3θ10+5θ30 を計算します。

キー	表示	説明
MODE 1 (1DEG)		角度を Deg にします
◀ DISPLAY . 0 (10r8a)		複素数モードをセット
1 i 1 ENTER	1.4142θ45.0000	1i1 を入力
	1.4142θ45.0000	
3 ▶ θ 1 0	3.0000θ10.0000	3θ10 を入力
ENTER	3.0000θ10.0000	
5 ▶ θ 3 0	1.4142θ45.0000	5θ30 を入力し、3θ10
+	7.8861θ22.5241	を加算
+	9.2088θ25.8898	1i1 を加算。計算結果は 9.2088θ25.8898

式における複素数

式には複素数を入力できます。式を表示するときは、全ての数値形式、xiy や rθa などが表示されます。

式を評価するときの変数にも複素数を入力できます。計算結果の表示は表示形式の設定に依存します。これは ALG モードでも同じです。

複素数を含む式でも、SOLVE と積分が実行できます。

プログラムにおける複素数

プログラムでも複素数を入力できます。たとえばプログラムで 1i2+3 θ 10+5 θ 30 は次のようになります。

プログラム行: (ALG モード)	詳細
F001 LBL F	プログラムを開始
F002 1↵2+3θ10+5θ30	
F003 RTN	

プログラムを実行して変数の入力プロンプト INPUT が出たときにも複素数を入力できます。結果の表示は表示形式の設定に依存します。

複素数を含むプログラムについても、SOLVE と積分が実行できます。

10

ベクトルの演算

数学的な観点からは、ベクトルは複数の要素を持つ、1行あるいは1列の行列です。

物理でのベクトルは、2つか3つの成分から構成され、位置、速度、加速度、力、モーメント、線運動量、角運動量、角速度、角加速度などの物理量をあらわします。

ベクトルを入力するには:

1. [→] [ ] を押します。
2. ベクトルの最初の数値を入力します。
3. [←] [,] を押し、２次元ベクトルや３次元ベクトルの第２の数値を入力します。
4. [←] [,] を押し、３次元ベクトルの第３の数値を入力します。

HP 35s では３次元を超えるベクトルは取り扱いできません。

ベクトルの演算

✓ **加算と減算:**

ベクトルの加算と減算には、2つのベクトルが同じ次元である必要があります。異なる次元のベクトルを加算・減算しようとすると、エラーメッセージ INVALID DATA が出ます。

1. 第１のベクトルを入力
2. 第２のベクトルを入力
3. [+] または [−] を入力

✔ [1.5,-2.2]+[-1.5,2.2] を計算します。

キー	表示	説明
MODE 5 (5RPN)		RPN モードに変更(必要であれば)
↷ [] 1 . 5 ↶	[1.5000,-2.2000]	Enters [1.5,-2.2]
, +/- 2 . 2	[1.5000,-2.2000]	
ENTER		
↷ [] +/- 1 . 5	[1.5000,-2.2000]	Enters [-1.5,2.2]
↶ , 2 . 2	[-1.5,2.2]	
+	0.0000	ベクトルの加算
	[0.0000,0.0000]	

[-3.4,4.5]-[2.3,1.4] を ALG モードで計算します。

キー	表示	説明
MODE 4 (4ALG)		ALG モードに切り替え
↷ [] +/- 3 . 4 ↶ , 4 . 5 ›	[-3.4,4.5]_	[-3.4,4.5] を入力
− ↷ [] 2 . 3 ↶ , 1 . 4	◀3.4,4.5]-[2.3,1.4]	[2.3,1.4] を入力
ENTER	[-3.4,4.5]-[2.3,..	計算を実行
	[-5.7000,3.1000]	

✔ スカラーとベクトルの乗算と除算:

1. ベクトルを入力
2. スカラーを入力
3. × か ÷ を押す

✔ [3,4]x5 を計算します。

キー	表示	説明
MODE 5 (5RPN)		RPN モードへ変更
↷ [] 3 ↶ , 4	[3.0000,4.0000]	[3,4] を入力
ENTER	[3.0000,4.0000]	
5	[3.0000,4.0000] 5_	5 をスカラーとして入力
×	0.0000 [15.0000,20.0000]	乗算を実行

[-2,4]÷2 を ALG モードで計算します。

キー	表示	説明
MODE 4 (4ALG)		ALG モードへ変更
↷ [] +/- 2 ↶ , 4 ›	[-2,4]_	[-2,4] を入力
÷ 2	[-2,4]÷2	5 をスカラーとして入力
ENTER	[-2,4]÷2 [-1.0000,2.0000]	除算を実行

ベクトルの大きさ

絶対値の関数 “ABS” がベクトルに使われると、そのベクトルの大きさを計算します。ベクトル A=(A1, A2, …An) の大きさは次のように定義されています。

$$|A| = \sqrt{A_1^{\,2} + A_2^{\,2} + \cdots + A_n^{\,2}}$$

1. ↷ ABS を押す
2. ベクトルを入力
3. ENTER を押す

たとえば [5,12] の大きさは ：

→ ABS → [] 5 ← , 1 2 ENTER 結果は 13 です。
RPN モードでは MODE 5 (5RPN) → [] 5 ← , 1 2 → ABS となります。

内積 (Dot product)

関数 DOT は同じ次元の2つのベクトルの内積を求めるときに使います。異なる次元のベクトルを与えるとエラーメッセージ INVALID DATA が出ます。

2次元ベクトル [A, B] と [C, D] の内積は [A, B]•[C, D]= A x C +B x D です。

3次元ベクトル [A, B, X] と [C, D,Y] の内積は [A, B, X]•[C, D, Y] =
A x C +B x D+X x Y です。

1. 第１ベクトルを入力
2. × を押す
3. 第２ベクトルを入力
4. ENTER を押す

注意： 記号 × は「ドット積」を意味しており、「クロス積」ではありません。クロス積については第 17 章で解説します。

ベクトル [1, 2] と [3, 4] の内積を計算します。

キー	表示	説明
MODE 4 (4ALG)		ALG モードにします
→ [] 1 ← , 2 >	[1,2]_	最初のベクトル [1,2] を入力
× → [] 3 ← , 4	[1,2]x[3,4]	内積の × と第２のベクトルを入力
ENTER	11.0000	内積は 11

✓ ベクトル[9, 5] と [2, 2] の内積を計算します。

キー	表示	説明
MODE 5 (5RPN)		RPN モードにします
→ [] 9 ← , 5	[9.0000,5.0000]	最初のベクトル [9,5] を入力
ENTER	[9.0000,5.0000]	

↷ [] 2 ↶ , 2	[9.0000,5.0000] [2,2]	第２のベクトル [2,2] を入力
×	28.0000	× で内積を計算。 結果は 28

ベクトル間の角度

2つのベクトル A と B の角度は $\theta = \mathrm{ACOS}(\mathrm{A}\cdot\mathrm{B}/|A||B|)$ です。

ベクトル A=[1, 0] と B=[0,1] の角度を計算します。

キー	表示	説明
MODE 4 (4ALG)		ALG モードに変更
MODE 1 (1DEG)		角度を Deg に変更
↷ ACOS	ACOS()	acos 関数
↷ [] 1 ↶ , 0 >	ACOS([1,0])	ベクトル A [1,0] を入力
× ↷ [] 0 ↶ , 1 >	ACOS([1,0]×[0,1])	ベクトル B [0,1] を入力し、内積を求める
÷ ↷ ABS ↷ [] 1 ↶ , 0 >	←,1]÷ABS([1,0]→	ベクトル A [1,0] の大きさ
÷ ↷ ABS ↷ [] 0 ↶ , 1	←1,0]÷ABS([0,1]→	ベクトル B [0,1] の大きさ
ENTER	ACOS([1,0]×[0,... □□□□□□□90.0000	ベクトルの角度は 90 度

✓ ベクトル A=[3,4] と B=[0,5] の角度を計算します。

キー	表示	説明
MODE 5 (5RPN)		RPN モードに変更
MODE 1 (1DEG)		角度を Deg に
↷ [] 3 ↶ , 4	90	２つのベクトルの内積
ENTER ↷ [] 0 ↶ , 5 ×	20.0000	

[→] [[]] 3 [←] [,] 4	20.0000	ベクトル [3,4] の大きさ
[→] ABS	5.0000	
[→] [[]] 0 [←] [,] 5	5.0000	ベクトル [0,5] の大きさ
[→] ABS	5.0000	
×	20.0000	乗算
	25.0000	
÷	90	除算
	0.8000	
[→] ACOS	90	角度は 36.8699 度
	36.8699	

式におけるベクトル

ベクトルは実数と同様に、式でも、式の中の変数でも利用できます。変数にはそのプロンプトに対してベクトルを指定できます。

ベクトルを含む式は SOLVE を実行できますが、未知変数がベクトルの場合は機能が制限されます。

ベクトルを含む式は積分もできますが、計算結果が実数か1次元ベクトル、第2と第3要素がゼロであるベクトルになるものに限定されます。

プログラムでのベクトル

実数や複素数と同様に、ベクトルもプログラムで利用できます。

たとえば [5, 6] +2 x [7, 8] x [9, 10] をプログラムで入力してみます。

プログラム行	説明
00001 LBL□G	プログラムを開始
00002 [5,6] + 2 × [7,8] ×[9,10]	[5,6]+2 x [7, 8] x [9, 10]
00003 RTN	

変数のプロンプトでもベクトルを入力できます。ベクトルを含むプログラムも SOLVE と積分で利用できます。

変数やレジスタからベクトルを作成

プログラムモードやその実行時には、メモリに保存された変数、スタックレジスタ、間接レジスタからの数値をベクトルに含めることができます。

ALG モードではベクトルの入力を [→] [ ] で開始します。RPN でも同じキーを使いますが、[→] [ ] の前に [EQN] を押す必要があります。

数値が保存された文字変数をベクトルに与えるには [RCL] を押してから変数の文字を指定します。

スタックレジスタをベクトルの要素として入力するには [R↓] の後に [▶] か [◀] キーで下線を移動させ、目的のスタックレジスタで [ENTER] キーを押します。

間接変数 (I) や (J) をベクトルの要素として入力するには [RCL] の後に (I) または (J) を押します。

たとえば、ベクトル [C, REGZ, (J)] を RPN モードで作成するには
[EQN] [→] [ ] に続けて、
[RCL] [C] [←] [,] [R↓] [▶] [ENTER] [←] [,] [RCL] [(J)] [ENTER]
と操作します。

11

基数(Base)と論理(Logic)演算

BASE メニュー ([→] [BASE]) では、10 進数、2 進数、8 進数、16 進数の入力と変換ができます。

LOGIC メニュー ([→] [LOGIC]) では論理関数が利用できます。

BASE メニュー

メニュー	説明
DEC	10 進数(**Dec**imal)モード。これがデフォルト設定です。
HEX	16 進数(**Hex**adecimal)モード。このモードでは表示記号 **HEX** がつきます。数値は 16 進数で表示されます。 RPN モードでは [SIN]、[COS]、[TAN]、[√x]、[y^x]、[1/x] が A から F のショートカットに割り当てられています。 ALG モードで A から F を入力するには [RCL] の後に A、B、C、D、E、F と操作します。
OCT	8 進数(**Oct**al)モード。このモードでは表示記号 **OCT** がつきます。数値は 8 進数で表示されます。
BIN	2 進数(**Bin**ary)モード。このモードでは表示記号 **BIN** がつきます。数値は 2 進数で表示されます。数値が 12 桁を超える場合は [→] [>] ならびに [→] [<] キーによりスクロールさせて閲覧して下さい。(この章で後述の「長い 2 進数を表示させる」を参考にして下さい)
d	数値の最後に出ていれば、それが 10 進数であることを示しています。
h	数値の最後に出ていれば、それが 16 進数であることを示しています。16 進数の入力には、数値の後に "h" を追加します。

o	数値の最後に出ていれば、それが 8 進数であることを示しています。8 進数の入力には、数値の後に “o” を追加します。
b	数値の最後に出ていれば、それが 2 進数であることを示しています。2 進数の入力には、数値の後に “b” を追加します。

例: 基数変換

基数の変換は次の操作でおこないます。

125.99_{10} を 16 進数、8 進数、2 進数に変換しましょう。

キー	表示	説明
1 2 5 ↻ BASE	7Dh	10 進数を 16 進数に変換
2 (2HEX)		
↻ BASE 3 (3OCT)	175o	8 進数
↻ BASE 4 (4BIN)	1111101b	2 進数
↻ BASE 1 (1DEC)	125.0000	

注意: 10 進数以外のときは、数値の整数部のみが表示されます。分数はそのまま（分数を削除する操作をするまで）保持され、再び 10 進数が選択されたときに表示されます。

$24FF_{16}$ を 2 進数に変換してみましょう。14 桁（表示可能な最大桁）以上になります。

キー	表示	説明
↻ BASE 2 (2HEX)	24FFh	"F" を入力するため 1/x キーを使います。
2 4 1/x 1/x ↻ BASE 6 (6h)		
↻ BASE 4 (4BIN)	10010011111111➡	桁が多いため表示しきれません。表示記号➡により右側に数値が続いていることがわかります。
↻ >	⬅b	残りの数値を表示させます。全桁は 10010011111111_b です。
↻ <	10010011111111➡	最初の 14 桁を再表示させま

す。

[→] [BASE] [1] (1DEC)	9,471.0000	10 進数に戻します。

2/8/10/16 進数をあらわす基数記号 n（b/o/d/h）を入力するときは[BASE]メニューを利用します。基数記号が無いときは 10 進数と見なします。

注意:

ALG モードでは:

1. 計算結果の基数は、現在の基数モード設定に依存します。
2. アクティブなコマンド行がない場合（点滅するカーソルがディスプレイ1行目に表示されていない場合）、基数の変換操作の結果は第2行に表示されます。
3. [ENTER]キーを押した後や基数モードの変更後は、ディスプレイ第２行に表示中の計算結果の末尾に、基数 2/8/16 をあらわす記号 b/o/h を自動的に追加します。
4. 式を再編集するには [←] や [→] を押します。

RPN モードでは：

ディスプレイ第2行に数値を入力し[ENTER] キーを押してから、基数を変換します。ディスプレイ第1行と第2行に表示中の数値の基数が変換され、基数 2/8/16 をあらわす記号 b/o/h が追加されます。

第2行の数値の桁が多いときは[→] [←] や [→] [→]で移動させます。

論理(LOGIC) メニュー

メニューラベル	説明
AND	**2つの引数のビット単位の論理積** "AND" 例: AND(1100b,1010b)=1000b
XOR	**2つの引数のビット単位の排他的論理和** "XOR" 例: XOR(1101b,1011b)=110b
OR	**2つの引数のビット単位の論理和** "OR" 例: OR(1100b,1010b)=1110b
NOT	**引数の成分のビット単位の論理否定** "NOT"。 対応する成分が反対になります。 例: NOT(1011b)= 11111111111111111111111111111110100b
NAND	**2つの引数のビット単位の否定論理積** "NAND" 例: NAND(1100b,1010b)=11111111111111111111111111111111111110111b
NOR	**2つの引数のビット単位の否定論理和** "NOR" 例: NOR(1100b,1010b)= 11111111111111111111111111111110001b

"AND", "OR", "XOR", "NOT", "NAND", "NOR" は論理関数としても利用できます。引数に分数、複素数、ベクトルが与えられると "INVALID□DATA" というエラーが出ます。

基数 2, 8, 16 の演算

どんな基数でも [+], [−], [×], [÷] で計算できます。

16進数(HEX)モードで利用できないのは [√x], [e^x], [LN], [y^x], [1/x], [Σ+] です。ただし、その他のほとんどの演算でも、下記のように小数部が切り捨てられるため、意味ある結果になることは少ないです。

2進数、8進数、16進数での演算では、2の補数 (2's compornent)で行われ、整数のみが使われます。

- **数値に小数部が含まれるときは、整数部のみが演算に使われます。**
- **演算の結果は常に整数になります。(全ての小数部は切り捨てられます)**

基数変換では数値の表示のみを変更するのに対し、X レジスタの演算では X レジスタの数値そのものが入れ替わります。

演算により有効なビットを表現できないときは、ディスプレイに**OVERFLOW**と表示され、正または負の表示可能な最大値が表示されます。

例:

16 進数、8 進数、2 進数の演算の例です。

$12F_{16} + E9A_{16} = ?$

	キー	表示	説明
	[↗] [BASE] [2] (2HEX)		基数を 16 とします。 表示記号 **HEX** が出ます。
✓	[1] [2] [1/x] [↗] [BASE] [6] (6h) [ENTER] [y^x] [9] [SIN] [↗] [BASE] [6] (6h) [+]	FC9h	結果

$7760_8 - 4326_8 = ?$

	[↗] [BASE] [3] (3OCT)	7711o	基数を 8 とします。 表示記号 **OCT** が出ます。 表示中の数値が 8 進数に変換されます。
✓	[7] [7] [6] [0] [↗] [BASE] [7] (7o) [ENTER] [4] [3] [2] [6] [↗] [BASE] [7] (7o) [−]	3432o	結果

$100_8 \div 5_8 = ?$

✓	[1] [0] [0] [↗] [BASE] [7] (7o) [ENTER] [5] [↗] [BASE] [7] (7o) [÷]	14o	結果の整数部

$5A0_{16} + 1001100_2 = ?$

✓	[↗] [BASE] [2] (2HEX) [5] [SIN] [0] [↗] [BASE] [6] (6h) [ENTER]	5A0h	基数を 16 に設定。 表示記号 **HEX** が出てきます
	[↗] [BASE] [4] (4BIN) [1] [0] [0] [1] [1] [0] [0]	1001100b	基数を 2 に設定。 表示記号 **BIN** が出ます。

	[→] [8] (8b)		この操作で数値の入力が完了しますから、数値の間に [ENTER] を押す必要はありません。
✓	[+]	10111101100b	基数 2 での結果
	[→] [BASE] [2] (2HEX)	5ECh	基数 16 での結果
	[→] [BASE] [1] (1DEC)	1,516.0000	10 進数に戻します。

数値の内部表現

基数変換のときに表示される数値が変化しますが、保存された数値そのものが入れ替わるわけではありません。よって、10 進数の場合は演算で利用されるまでは小数部の切り捨てが行われません。

数値が 16 進数、8 進数、2 進数で表示されているときは、36 ビット（16 進数で 9 桁、8 進数で 12 桁）で表示されます。先頭のゼロは省略されますが、符号に関わる部分ですから重要です。たとえば、125_{10} を 2 進数に変換すると次のように表示されます。

1111101b

36 桁表示では下記の通りです。

000000000000000000000000000001111101b

負の数値

2 進数のいちばん左（最初の桁）のビットは符号をあらわします。(1) になっているときは負です。先頭のゼロが省略されて表示されているときはこのビットはゼロ（つまりその数値が正ということ）をあらわします。負の数値は、対応する正の 2 進数の 2 の補数になります。

キー	表示	説明
[5] [4] [6] [→] [BASE] [2] (2HEX)	222h	正の 10 進数を入力し、16 進数に変換
[+/−] [5] [4] [6] [ENTER]	FFFFFFDDEh	2 の補数（符号の変更）

[↗] [BASE] [4] (4BIN)	1111111111111➡	2 進数に変換。 表示記号➡ が登場します。先頭ビットが 1 なので負の数値だとわかります。
[↗] [›]	⬅1111111111101➡	スクロールさせて結果を確認します。
[↗] [›]	⬅11011110b	いちばん右のビットを表示させます。
[↗] [BASE] [1] (1DEC)	-546.0000	負の 10 進数

数値の範囲

2 進数の内部表現のサイズである 36 ビットにより、取り扱い可能な数値の範囲が制限されています。16 進数では 9 桁、8 進数では 12 桁、2 進数では 36 桁となります。10 進数にも 11 桁の制限がありますが、他の基数の範囲はこれでカバーされます。

基数変換における数値の範囲

基数	正の最大値	負の最大値
16	7FFFFFFFh	800000000h
8	377777777777o	400000000000o
2	011111111111111111111111111111111111b	100000000000000000000000000000000000b
10	34,359,738,367	–34,359,738,368

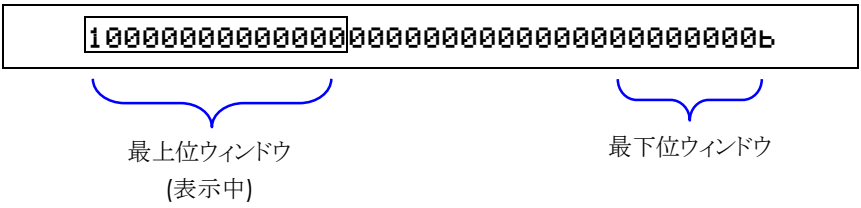
この範囲を超える数値は 10 進数以外を選択していると入力できません。

2 進数、8 進数、16 進数において、10 進数で入力した数値がこの範囲を超えていた場合、メッセージ TOO BIG が表示されます。これが表示されるとオーバーフローとなり、演算のときは、その数値の代わりにそれに近い正か負の最大値が用いられます。

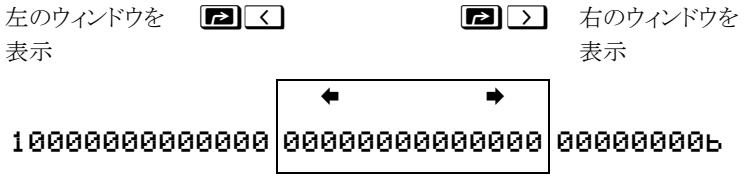
長い 2 進数でのウィンドウ

2 進数は最大で 36 桁になります。長くなると 14 桁ごとに表示されます。表示中の 14 桁をウィンドウと呼びます。

36 ビットの数値



2 進数が 14 桁を超える場合は、表示記号◀ か ▶ (あるいはその両方)が出てきて、どちらかの方向に表示できない桁があることを示します。[ ][ ] か [ ][ ] と操作してウィンドウを移動させることができます。



プログラムと式で基数を利用する

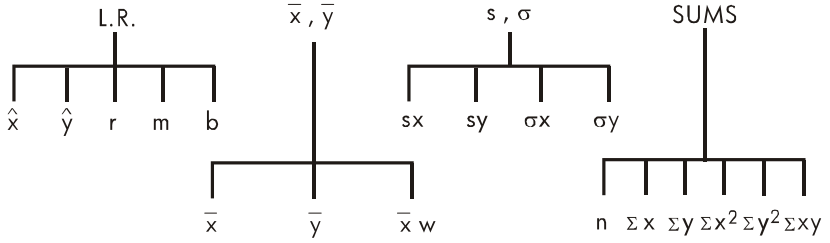
プログラムと式では、基数の設定が影響します。2 進数、8 進数、16 進数をプログラムや式に入力することができますし、変数に数値を入力するプロンプトでも同じです。
結果は現在の基数で表示されます。

12

統計(Statistics)

HP 35s の統計(Statistics)メニューで、次に示す 1 変数または 2 変数データ(実数)の統計処理ができます。

- **平均** (mean)、**標本** (sample)、**母標準偏差** (population standard deviation)
- **線形回帰** (linear regression)、**線形推定** (linear estimation、$\hat{x}$ と $\hat{y}$)
- **加重平均** (weighted mean、x に対する y の加重)
- **統計データの総和** (summation): n, Σx, Σy, Σx^2, Σy^2, Σxy



統計データの入力

1 変数および 2 変数の統計データが Σ+ (または ↩ Σ-)キーを使って、それぞれ同様の方法で入力 (または削除) できます。入力したデータは 6 個の**統計レジスタ**(−27 から −32) に統計データの総和として蓄積されます。これは SUMS メニューで確認できます。(↗ SUMS と操作して n Σx Σy Σx² Σy² Σxy を表示させます)

注意

統計データを新たに入力するときは、統計レジスタに保存された前回のデータをクリアして下さい。
(↗ CLEAR 4 (4Σ) と操作します)

1 変数データの入力

1. まず、既存のデータをクリアするため [→] [CLEAR] [4] (4Σ) と操作します。
2. それぞれの x 変数を入力し、[Σ+] キーを押します。
3. ディスプレイには蓄積された統計データの数（標本数）n が表示されます。

[Σ+] キーを押すと、Y レジスタの値も y として統計レジスタに保存されるため、x とあわせて 2 つの変数が保存されます。これにより、たとえ x の数値しか入力していなくても、あるいは、x と y のデータ数が異なっていても線形回帰の計算が可能になり、y に基づく数値を表示できるようになります。ただし、このような場合でもエラーは発生しませんが、結果は意味のないものになるでしょう。

データの入力が完了した直後に、その数値を再度呼び出すには [→] [LAST x] と操作します。

✓ 2 変数データの入力

統計データが 2 変数のときは、最初に従属変数（データセットの第 2 変数）を入力し [ENTER] キーを押します。続けて、独立変数（データセットの第 1 変数）を入力し [Σ+] キーを押します。

1. 既存の統計データをクリアするには [→] [CLEAR] [4] (4Σ) と操作します。
2. 最初に y の数値を入力し [ENTER] キーを押します。
3. 次に、対応する x の数値を入力し [Σ+] キーを押します。
4. ディスプレイには入力済みの統計データのセット数をあらわす n が表示されます。
5. 次の x, y のセットを入力していくと、それに応じて n が更新されていきます。

直前に入力した x の値を再度読み込むには [→] [LAST x] と操作します。

入力済みのデータを修正

統計データの入力で誤りがあったときは、間違ったデータを削除して新しいデータを追加して下さい。x と y のペアの、どちらか一方のみ修正したいときでも、両方一緒に削除して再入力する必要があります。

統計データの修正方法:

1. 間違っているデータを再入力し、Σ+キーの代わりに [↩] Σ−と操作します。この操作によりデータの削除とともに *n* も減ります。
2. 正しいデータを入力し、Σ+キーを押します。

✓ 入力直後のデータが間違っていたときは、[↩] LAST *x* でそのデータを呼び出し、[↩] Σ− で削除できます(誤入力した *y* のデータはまだ Y レジスタに残っており、誤入力した x のデータは LAST X レジスタにあります)。統計データを削除したときは、Y レジスタの数値がディスプレイ第1行に、*n* がディスプレイ第2行に表示されます。

例: 次の表の左側の x と *y* の数値を入力し、右側のように訂正します。

x, y の初期値	x, y の訂正値
20, 4	20, 5
400, 6	40, 6

	キー	表示	説明
	[↪] CLEAR [4] (4Σ)		既存の統計データを全てクリア
✓	[4] ENTER [2] [0] Σ+	4.0000 1.0000	最初のデータを入力
✓	[6] ENTER [4] [0] [0] Σ+	6.0000 2.0000	入力済みの標本数をあらわす *n* を表示。
	[↪] LAST *x*	6.0000 400.0000	LAST X レジスタから直前の x 呼び出し。直前の *y* は Y レジスタに残っています。
	[↩] Σ−	6.0000 1.0000	直前のデータセットを削除
✓	[6] ENTER [4] [0] Σ+	6.0000 2.0000	最後のデータを再入力
✓	[4] ENTER [2] [0] [↩] Σ−	4.0000 1.0000	最初のデータを削除
✓	[5] ENTER [2] [0] Σ+	5.0000 2.0000	最初のデータを再入力。統計レジスタには2組のデータが入っています。

統計の計算

データの入力が終わったら、統計メニューが利用できます。

統計メニュー

メニュー	キー	説明
L.R.	↶ L.R.	線形回帰: 線形推定 $\hat{x}$ $\hat{y}$ とカーブフィッティング r m b。 詳細はこの章の「線形回帰」をご参照下さい。
$\overline{x}$, $\overline{y}$	↶ x,y	平均メニュー: $\overline{x}$ $\overline{y}$ $\overline{x}_w$ この章で後述の「平均」をご参照下さい。
S,σ	↷ S,σ	標準偏差メニュー: s_x s_y σ_x σ_y この章で後述の「標本標準偏差」と「母標準偏差」をご参照下さい。
SUMS	↷ SUMS	総和メニュー: n Σx Σy Σx^2 Σy^2 Σxy この章で後述の「総和の統計」をご参照下さい。

平均 (mean)

平均とは、あるグループの数値の総計をデータの個数で除算したものです。

- x の平均を求めるには ↶ x,y ($\overline{x}$) と操作します。
- y の平均を求めるには ↶ x,y › ($\overline{y}$) と操作します。
- y を加重あるいは頻度として使った x の**加重平均**を求めるには ↶ x,y › › ($\overline{x}_w$) と操作します。加重には整数も小数も利用できます。

例: 平均(1変数)

ある製品の製造にかかる時間の平均を求めてみます。製造者のなかの6人をピックアップして、各人が費やした時間を計測したところ、次のようになりました。(単位: 分)

15.5	9.25	10.0
12.5	12.0	8.5

平均は下記のように求めます。(全てのデータを x として取り扱います)

キー	表示	説明
[↻] [CLEAR] [4] (4Σ)		統計レジスタのクリア
[1] [5] [.] [5] [Σ+]	1.0000	最初のデータを入力
[9] [.] [2] [5] [Σ+] [1] [0]		残りのデータを入力。
[Σ+] [1] [2] [.] [5] [Σ+] [1]	6.0000	標本数は 6 になります。
[2] [Σ+] [8] [.] [5] [Σ+]		
[↺] [x,y] ($\bar{x}$)	$\bar{x}$ $\bar{y}$ $\bar{x}_W$ 11.2917	製造にかかる時間の平均

例: 加重平均(2 変数)

ある製造業の会社は、ある部品を 1 年で 4 回購入します。昨年の購入は下表の通りでした。

部品単価 (x)	$4.25	$4.60	$4.70	$4.10
部品の数 (y)	250	800	900	1000

この部品の購入量による加重平均を求めましょう。y が加重(頻度)になることと、y は価格 x の前に入力するという点にご注意下さい。

	キー	表示	説明
✓	[↻] [CLEAR] [4] (4Σ)		統計レジスタのクリア
	[2] [5] [0] [ENTER] [4] [.]		データを入力。
✓	[2] [5] [Σ+]		n が表示されます。
	[8] [0] [0] [ENTER] [4] [.]		
	[6] [Σ+]	900.0000	
✓	[9] [0] [0] [ENTER] [4] [.]	3.0000	
	[7] [Σ+]		
✓	[1] [0] [0] [0] [ENTER] [4]	1,000.0000	合計 4 個の標本が入力さ
	[1] [Σ+]	4.0000	れました。
	[↺] [x,y] [›] [›] ($\bar{x}_W$)	$\bar{x}$ $\bar{y}$ $\bar{x}_W$ 4.4314	購入量による価格の加重平均

標本標準偏差 (Sample Standard Deviation)

標本標準偏差は、平均値からのデータの散らばり具合を計測するものです。標本標準偏差では**母集団**が十分に大きく、標本は有限であると仮定して、分母を $n-1$ として計算します。

- x の標準偏差を求めるには [↗] [S.σ] (sx) と操作します。
- y の標準偏差を求めるには [↗] [S.σ] [›] (sy) と操作します。

このメニューの (σx) と (σy) は次節「母標準偏差」で解説します。

例: 標本標準偏差

前述した製造時間の**平均**の例において、製造時間の標準偏差 (s_x) を求めてみましょう。

15.5	9.25	10.0
12.5	12.0	8.5

製造時間の標準偏差を求めます(全てのデータを x として取り扱います)。

キー	表示	説明
[↗] [CLEAR] [4] (4Σ)		統計レジスタのクリア
[1] [5] [.] [5] [Σ+]	1.0000	最初の時間を入力
[9] [.] [2] [5] [Σ+] [1] [0]		残りのデータを入力。
[Σ+] [1] [2] [.] [5] [Σ+] [1]		標本数は 6 になります。
[2] [Σ+] [8] [.] [5] [Σ+]	6.0000	
[↗] [S.σ] (sx)	sx sy σx σy 2.5808	時間の標準偏差を計算

母標準偏差 (Population Standard Deviation)

母標準偏差は、平均値からの散らばり具合を計測するものです。母標準偏差では標本が母集団であると仮定し、分母を n として計算します。

- x の母標準偏差を求めるには [↗] [S.σ] [›] [›] (σx) と操作します。
- y の母標準偏差を求めるには [↗] [S.σ] [›] [›] [›] (σy) と操作します。

例: 母標準偏差

ある老人の 4 人の孫の身長がそれぞれ 170、173、174、180 cm だとします。彼らの身長の母標準偏差を計算しましょう。

キー	表示	説明
[↗] [CLEAR] [4] (4Σ)		統計レジスタのクリア
[1] [7] [0] [Σ+] [1] [7] [3] [Σ+] [1] [7] [4] [Σ+] [1] [8]		データ入力。標本数は 4 になります。
[0] [Σ+]	4.0000	
[↗] [S,σ] [›] [›] (σx)	sx sy σx σy 3.6315	母標準偏差の計算

線形回帰 (Linear Regression, L.R.)

線形回帰(線形推定、Linear Estimation とも呼ばれます)は、x、 y のデータセットの相関関係をあらわす直線を求めるための統計手法です。

Note

エラーメッセージ STAT ERROR を回避するため、L.R. メニューの機能を使う**前に**データ入力を完了させて下さい。

L.R. (線形回帰) メニュー

メニューキー	説明
ŷ	データにフィットするように計算された直線に基づき、与えられた y に対する x の推定(予想)値。
x̂	データにフィットするように計算された直線に基づき、与えられた x に対する y の推定(予想)値。
r	(x, y) データの相関係数。この係数の範囲は –1 から +1 で、求めた直線がどれだけ標本に近いかをあらわします。
m	直線の傾き
b	直線の y 切片

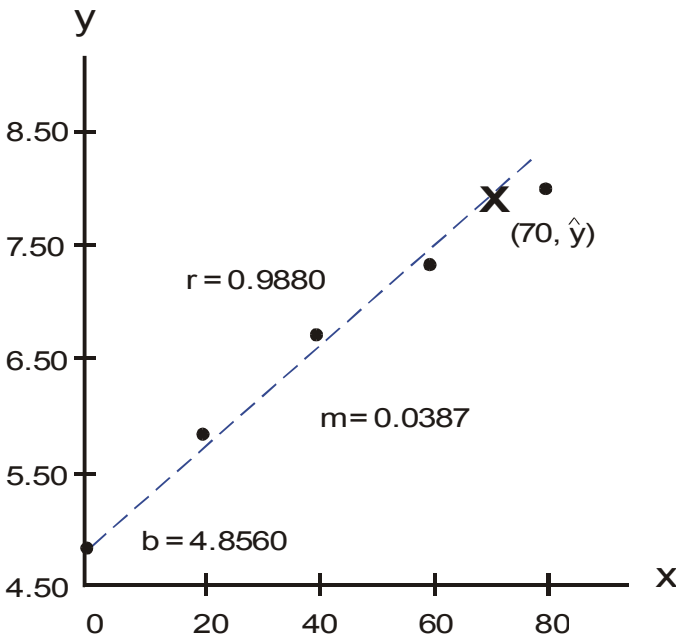
- ■ x（または y）の推定値を求めるには、対応する y（または x）の値を入力し、[↩] [L.R.] ($\hat{x}$)（または [↩] [L.R.] [›] ($\hat{y}$)）と操作します。
- ■ 線形回帰の直線の特性をあらわす数値を表示させるときは[↩] [L.R.] と操作し、r、m、b を選択します。

例: カーブフィッティング (Curve Fitting)

ある穀物の収穫量が、窒素肥料の量に依存しているとします。次の標本を使って収穫量と肥料の量における相関係数、傾き、y 切片を計算してみましょう。

X，肥料の量 (kg／ヘクタール)	0.00	20.00	40.00	60.00	80.00
Y，生産量 (トン／ヘクタール)	4.63	5.78	6.61	7.21	7.78

キー	表示	説明
[↩] CLEAR [4] (4Σ)		以前の統計データをすべてクリア
[4] [·] [6] [3] ENTER [0] [Σ+]		データを入力すると *n* が表示されます
[5] [·] [7] [8] ENTER [2] [0] [Σ+]	7.2100	
[6] [·] [6] [1] ENTER [4] [0] [Σ+]	4.0000	
[7] [·] [2] [1] ENTER [6] [0] [Σ+]		
[7] [·] [7] [8] ENTER [8] [0] [Σ+]	7.7800 5.0000	5 個の標本が入力されました
[↩] [L.R.] [›] [›] (r)	$\hat{x}$ $\hat{y}$ r m b 0.9880	線形回帰メニュー。 相関係数：求めた直線に対してデータがどれくらい近いかを示す係数。
[›]	$\hat{x}$ $\hat{y}$ r m b 0.0387	直線の傾き
[›]	$\hat{x}$ $\hat{y}$ r m b 4.8560	直線の y 切片



では、70 kg の肥料を与えたらどうなるでしょうか？ 推定値の機能を使って予想してみます。

	キー	表示	説明
✓	C 7 0	7.7800 70_	想定される x の値
	↩ L.R. > (ŷ)	x̂ ŷ r m b 7.5615	収穫量の推定値 [トン／ヘクタール]

データの精度に関する制限

HP 35s では有限の精度を使っているため、計算は丸めによる制限があります。次の2つの例をご参照下さい。

巨大な数値と微小な数値の標準化 (normalizing)

標本データ同士の数値の大きさの差がごく微小であるときは、標準偏差と線形回帰を正しく計算できない可能性があります。これを避けるには、データそのものを入力するのではなく、ある中間値(たとえば平均値でも良いです)からの差を求め、データを標準化する方法があります。x を標準化して入力した場合は、$\overline{x}$ と $\hat{x}$ の計算後に調整する必要があります。また、$\hat{y}$ と b も調整が必要です。

たとえば、x が 7776999、7777000、および 7777001 だったとき、まず標準化データとして −1, 0, および 1 を入力します。$\overline{x}$ と $\hat{x}$ を求めてからそれぞれに 777000 を加算して下さい。b には 7777000 × m を加算して下さい。$\hat{y}$ の計算では、x の値に対して元の数値より 7777000 少ないものを入力するという点にご注意下さい。

データ間の差が微小なときと同様に、差が巨大な場合も精度が悪くなります。この場合も標準化して入力して下さい。

削除したデータの影響

標本データを入力後に [↩] [Σ−] で削除しても、統計レジスタに保存されてしまった丸めの分は元に戻りません。ほとんどの場合、この丸め分は深刻な問題にはなりませんが、もし削除前のデータが修正後のものに比べて遙かに大きい値だったときは、丸めの影響は大きくなります。このような際はいったん全てのデータをクリアしてから再入力して下さい。

総和と統計レジスタ

統計レジスタには 6 個の独立なメモリ領域が割り当てられていて、それぞれに 6 種の総和が蓄積されていきます。

統計データの合計

[↪] [SUMS] キーで統計レジスタを確認できます。

- (n) は標本データ数を呼び出します。
- [›] (Σx) は x の合計を表示します。
- [›] [›] (Σy) は y の合計を表示します。

■ [›][›][›] (Σx²), [›][›][›][›] (Σy²), および [›][›][›][›][›] (Σxy) はそれぞれ、x の 2 乗の合計、y の 2 乗の合計、x と y の積の合計を表示します。これらの数値は、この電卓で用意された関数では求めることができない統計値を計算するような場合に利用します。

統計データの入力後、統計レジスタを確認するには [↶] [MEM] [1] (1VAR) [ENTER] と操作し、[^] と [v] でスクロールさせて下さい。

例: 統計レジスタの確認

まず、標本データ (1,2) および (3,4) を [Σ+] キーで統計レジスタに保存します。それから、保存された統計値を確認します。

	キー	表示	説明
	[↷] [CLEAR] [4] (4Σ)		統計レジスタのクリア
✓	[2] [ENTER] [1] [Σ+]	2.0000 1.0000	最初のデータセット (1,2) を入力
✓	[4] [ENTER] [3] [Σ+]	4.0000 2.0000	次のデータセット (3,4) を入力
	[↶]	n=	⬆ VAR 一覧で n レジスタを表示。
	[MEM] [1] (1VAR)	2.0000	⬇
	[^]	Σxy= 14.0000	⬆ Σxy レジスタを表示 ⬇
	[^]	Σy²= 20.0000	⬆ Σy² レジスタを表示 ⬇
	[^]	Σx²= 10.0000	⬆ Σx² レジスタを表示 ⬇
	[^]	Σy= 6.0000	⬆ Σy レジスタを表示 ⬇
	[^]	Σx= 4.0000	⬆ Σx レジスタを表示 ⬇
	[^]	n= 2.0000	⬆ *n* レジスタを表示 ⬇
	[C]	4.0000 2.0000	VAR 一覧の表示を終了

統計レジスタへのアクセス

HP 35s の統計レジスタは次表のように割り当てられています。合計に関するレジスタを式やプログラムで呼び出すときは番号ではなく名前で参照して下さい。

統計レジスタ

レジスタ	番号	解説
n	-27	標本データの数
Σx	-28	標本の x の合計
Σy	-29	標本の y の合計
Σx^2	-30	標本の x の 2 乗の合計
Σy^2	-31	標本の y の 2 乗の合計
Σxy	-32	標本の x と y の積の合計

統計レジスタを呼び出したいときは、レジスタ番号 (-27 から -32) を *I* または *J* に保存して下さい。レジスタ番号を入力し、STO (I) または (J) と操作します。呼び出した統計レジスタの値を確認するには、↶ VIEW (I) または (J) (あるいは RCL (I) または (J)) と操作します。SUMS メニューで統計レジスタの数値を呼び出すこともできます。詳細は第 14 章の「変数とラベルの間接呼び出し」をご参照下さい。

Part 2

プログラミング

13

プログラムの基礎

このマニュアルの Part 1 では、関数や操作について「手動」で行う際の解説をしました。つまり、それぞれの操作をひとつひとつキーを押した場合についてです。また、式を登録することで同様の計算をするときの手間を省くことについても述べました。

Part 2 ではプログラムについて解説します。プログラムにより反復計算や複雑な計算、希望する方法による計算が可能になります。

この章ではプログラムについての一連の操作を解説します。次章「プログラミングテクニック」では、サブルーチンと条件分岐を解説します。

例: 簡単なプログラム

半径 r が 5 の円の面積 A を計算するときは、$A = \pi r^2$ ですから次のように計算します。

RPN モード: 5 x^2 ↶ π ×

ALG モード: 5 y^x 2 × ↶ π ENTER

結果は 78.5398 になります。

もし半径が異なる他の円の面積も求めたいとしたらどうでしょうか？

上記のキーストロークで半径だけ変更してもう一度入力することもできますが、あらかじめキーストロークをプログラムとして登録しておくことができます。

RPN モード	ALG モード
0001 x^2	0001 SQ(x)×π
0002 π	
0003 ×	

このプログラムでは、プログラム実行時の X レジスタ（表示中の数値）を半径として計算しています。実行すると面積を計算し、結果を X レジスタに保存します。

RPN モードでプログラムをメモリに保存するには次の操作を行います。

キー (RPN モード)	表示	説明
[↗] [CLEAR] [3]		メモリのクリア
(3ALL) [<] (Y) [ENTER]		
[↗] [PRGM]		プログラム入力モード (表示記号 **PRGM** が出ます)
[GTO] [·] [·]	PRGM TOP	プログラムポインタをリセットして PRGM TOP に移動させます。
[↗] [x^2]	0001 $\times^2$	(半径)^2
[↖] [π]	0002 π	
[×]	0003 ×	面積 = πx^2
[↗] [PRGM]		プログラム入力モードの終了

プログラムを実行して半径 5 の円の面積を求めるには次の操作を行います。

キー (RPN モード)	表示	説明
[GTO] [·] [·]		プログラムの先頭に移動します。
5 [R/S]	78.5398	計算結果

ALG モードでプログラムをメモリに保存するには次の操作を行います。

キー (ALG モード)	表示	説明
[↗] [CLEAR] [3]		メモリのクリア
(3ALL) [<] (Y) [ENTER]		
[↗] [PRGM]		プログラム入力モード (表示記号 **PRGM** が出ます)
[GTO] [·] [·]	PRGM TOP	プログラムポインタをリセットして PRGM TOP に移動させます。
[↗] [x^2] [RCL] [X] [>] [×] [↖] [π]	0001 SQ(X)×π	面積 = πx^2
[↗] [PRGM]		プログラム入力モードの終了

プログラムを実行して半径 5 の円の面積を求めるには次の操作を行います。

キー (ALG モード)	表示	説明
GTO ▫ ▫		プログラムの先頭に移動します。
5 STO X ENTER	5▶X □□□5.0000	5 を変数 X に保存
R/S	78.9358	計算結果

引き続き、次節でもこの例を使ってプログラムの概念と方法を解説します。

プログラムの設計

次のトピックではプログラム行に入力できるコマンドを示しています。プログラムに入力したコマンドは、(当然ですが)プログラムの閲覧時に表示され、実行時の挙動に影響します。

モードの選択

RPN モードで作成され、保存されたプログラムは RPN モードで編集ならびに実行して下さい。同様に ALG モードで作成され、保存されたプログラムは ALG モードで編集ならびに実行して下さい。間違っていると正しい結果は得られません。

プログラムの境界 (LBL と RTN)

複数のプログラムをメモリに保存するときは、プログラムの先頭に A001 LBL A のようなラベルが必要になります。また、それぞれのプログラムの最終行に A005 RTN のようにリターンを定義して下さい。

行番号には対応するラベル(たとえばA)が付与されます。

プログラム ラベル

プログラムとその一部(サブルーチン)にはラベルから開始しなければなりません。ラベルを記録するには次の操作を行います。

[↩] [LBL] 文字キー

ラベルには A から Z の文字をひとつ使います。この文字は第 3 章で紹介した変数にも使われていますが、変数のものとは無関係です。同じラベルは重複してプログラムで利用できません。もし、重複して使ったときはエラーメッセージ DUPLICAT,LBL が出ます。

メモリにプログラムがひとつしかないときはラベル無しで登録することもできます。しかし、次のプログラムを入力するときは隣り合うプログラムの間にラベルが必要になります。

プログラムには 999 行までの上限があります。

プログラムの return

プログラムとサブルーチンは return で終了しなければなりません。return の入力は次のように行います:

[↩] [RTN]

プログラムの実行が終わったとき、RTN 行があるとポインタが PRGM TOP (プログラムのトップ) に移動します。

RPN、ALG および式をプログラムで利用する

キーボードで計算するときと同じ方法がプログラムでも利用できます。

- RPN の利用 (第 2 章で解説したように、スタックを使用します)
- ALG の利用 (付録 C で解説)
- 式の利用 (第 6 章で解説)

この章の円の面積を求める例では RPN を使っていました。プログラムでは RPN 以外にも**式** (equation) を利用することができます (例題をこの章で後述します)。多くのプログラムでは RPN と式の、それぞれの良いところを組み合わせたものになるでしょう。

RPN の利点	式と ALG の利点
メモリ使用量が少ない	書きやすさと読みやすさ
実行速度が速い	プロンプトを自動で表示できる

式を含むプログラムが実行されると、その式は式リストで [XEQ] キーを押したときと同様に評価されます。プログラムの評価では等式に定義された "=" は "–" に置き換えられます。等式に割り当てられた [ENTER] と同等の機能はプログラムには存在しません。式で計算した数値を変数に保存するには、まず、等号のない式 (expression) を定義し、STO で変数に保存して下さい。

どちらのタイプでも、入力と出力、およびプログラムの流れの指定は RPN にすることができます。

データの入出力

ひとつ以上の入力または出力があるプログラムでは、いかに入出力するかを指定できます。

入力では INPUT 関数により、変数の入力を促すプロンプトが定義できます。あるいは、式に定義された変数のプロンプトの定義や、あらかじめスタックに保存された数値の獲得ができます。

出力では VIEW 関数により変数を表示したり、式から出てきたメッセージを表示したり、ディスプレイ第 1 行にプロセスを表示、プログラムの結果をディスプレイ第 2 行に表示、あるいはスタックに数値を残すことができます。

入出力の詳細はこの章で後述する「データの入力と表示」で解説します。

プログラムの入力

[→] [PRGM] と操作するとプログラム入力モードの切り替えが行われます。プログラム入力モードでのキー入力は、メモリにプログラム行として保存されます。コマンドや式はそれぞれの、ひとつのプログラム行を占有します。ALG モードでは式を直接プログラムの中に記述できます。

プログラムをメモリに保存するには:

1. [→] [PRGM] と操作し、プログラム入力モードに入ります。
2. [GTO] [.] [.] と操作し、PRGM TOP を表示させます。これにより**プログラムポインタ**がその他のプログラムの前に移動します。プログラム行を入力すると、その他の行の前に挿入されます。

   プログラムが不要になったときは [→] [CLEAR] [3] (3PGM) と操作してメモ

リから削除します。すると、本当に削除するかどうかの確認のため、メッセージ CLR PGMS? Y_Nが表示されます。削除を実行するには ◀ (Y) ENTER と操作します。

3. ▶ LBL *英字* と操作してプログラムに A から Z の１文字で**ラベル**を定義します。面積 (Area) のプログラムに対して "A" にするなど、プログラムの内容をあらわす文字が良いでしょう。

   もしメッセージ DUPLICAT.LBLが表示されたら、違う文字を使って下さい。あるいは、既存のプログラムをクリアしても良いです。◀ MEM 2(2PGM) と操作しのうえ ▲ や ▼でラベルを探し、▶ CLEAR C でクリアして下さい。
4. 電卓での操作をプログラムとして保存するには、手動の操作と同じキーを押して下さい。キーボードに表示されておらず、メニューから呼び出すタイプの関数も多いという点にご注意下さい。プログラム行に式を入力する方法は後述します。
5. ◀ RTN と操作してリターン行でプログラムを終了させます。これにより、実行後にプログラムポインタがPRGM TOPに移動します。
6. C (または ▶ PRGM) でプログラム入力モードを終了させます。

プログラム行における数値は入力した通りの精度で保存され、表示形式は ALL か SCI になります。(数値が短縮されて表示されるときは ◀ SHOW で全桁を表示させて下さい。)

プログラム行に式を入力するには:

1. EQN キーでしき入力モードをアクティブにします。表示記号 **EQN** が表示されます。
2. 式リストのときと同様に式を入力します。詳細は第 6 章をご参照下さい。入力ミスは ← で削除のうえ訂正します。
3. ENTER で式の入力を終了します。すると、表示は式の左端に移動します。（ただし、式リストには保存されません）

式の入力が終わったら、◀ SHOW で式のチェックサムと長さを確認できます。表示し続けるには SHOW キーを押し続けて下さい。

長い式ではプログラム行に表示記号➡ や ⬅ が出ます。▶ ◀ と ▶ ▶ でスクロールできます。

関数のクリアとバックスペース

プログラム入力モードでは次の点が通常と異なります。

- [C]キーによりプログラム入力が完了します。通常のように数値をクリアしてゼロにすることはありません。
- プログラム行の閲覧時に[←]キーを押すと表示中のプログラム行が削除されます。[<] / [>]キーで編集を開始します。プログラム行の編集状態では[←]キーはカーソル直前の文字を削除します。
- X レジスタをクリアするときは[↗] [CLEAR] [1] (1×)を使います。

プログラムに行を追加したり削除するときは、必要に応じて GTO と XEQ の引数が調整されます。

たとえば次のプログラムで:

```
A001 LBL A
A002 2+3
A003 1+2
A004 GTO A003
```

行 A002 を削除すると、行 A004 は「A003 GTO A002」に自動で更新されます。

プログラムでの関数名

プログラム行で使われる関数(サブルーチン)の名前はキーやメニュー、式と同じにする必要はありません。キーやメニューにフィットする短縮された名前にすることが多いです。

例: ラベル付きのプログラムを入力

次に示す操作により、円の面積を計算するプログラムが削除され、ラベルとリターンを含む新しいプログラムが保存されます。入力ミスがあれば [←] キーで入力行の全体を削除のうえ、再入力して下さい。

キー (RPN モード)	ディスプレイ	説明
[↗] [PRGM]		プログラム入力モード (表示記号 **PRGM** が点灯)

[↗] [CLEAR] [3]		プログラムメモリのクリア
(3PGM) [<] (Y) [ENTER]	PRGM TOP	
[↗] [LBL] [A]	A001 LBL A	ラベルを A とします。(area の A)
[↗] [x^2]	A002 $\times^2$	プログラム行を３つ入力
[↖] [π]	A003 π	
[×]	A004 ×	
[↖] [RTN]	A005 RTN	プログラム終了
[MEM] [2] (2PGM)	LBL A LN=15	ラベル A の長さ（バイト数）を表示
[↖] [SHOW]	CK=DAF1 LN=15	チェックサムと長さを表示
[C] [C]		プログラムの入力完了 (表示記号 **PRGM** がオフ)

チェックサムが異なっているときは、プログラムに入力ミスがあることを意味しています。

例: 式を含むプログラムの入力

次のプログラムでは前の例のような RPN ではなく、式を使って円の面積を求めています。

キー (RPN モード)	表示	説明
[↗] [PRGM] [GTO] [.] [.]	PRGM TOP	プログラム入力モードをアクティブにします。ポインタはメモリのトップに移動します。
[↗] [LBL] [E]	E001 LBL E	プログラムのラベルを E とします。(equation の E)
[↗] [STO] [R]	E002 STO R	半径を変数 R に保存
[EQN] [↖] [π] [×] [RCL] [R] [y^x] [2] [ENTER]	E003 π×R^2	式入力モードを選択し、式を入力。さらにプログラム入力モードに戻りま

		す。
[↩] [SHOW]	CK=7E5B LN=5	
[↩] [RTN]	E004 RTN	プログラムの終了
[↩] [MEM] [2] (2PGM)	LBL E LN=17	ラベル名とその長さ(バイト数)を表示。
[↩] [SHOW]	CK=2073 LN=17	チェックサムと長さを表示。
[C] [C]		プログラム入力モードの終了

プログラムの実行

プログラム入力モードがアクティブな状態(プログラム行が表示されていて表示記号 **PRGM** が点灯している状態)ではプログラムを実行できません。プログラム入力モードの終了には [C] ボタンを押して下さい。

プログラムの実行 (XEQ)

ラベル付きのプログラムを実行するには「[XEQ]**ラベル名**」と操作します。

プログラムをその最初の行から実行するには「[XEQ] ラベル名 [ENTER]」と操作します。たとえば[XEQ] [A] [ENTER] で、ディスプレイには "XEQ A001" と表示され、ラベル A の最初の行からスタートします。

その他の位置からスタートさせることもできます。「[XEQ] ラベル番号」と操作します。たとえば [XEQ] A005 のようになります。

プログラムがメモリにひとつだけしかない場合は、プログラムポインタを最初の行に移動させてから[R/S] (run / stop) キーで実行することもできます。表示記号 **PRGM** が出てきて、プログラムの実行中は表示記号 [B] が点灯します。

必要に応じてプログラム実行の前にデータを入力しておいて下さい。

例:

ラベル A と E のプログラムを実行し、3 個の異なる円の面積を求めます。円の半径をそれぞれ 5、2.5、2π とします。A と E の実行前に半径を入力するのを忘れないで下さい。

キー (RPN モード)	表示	説明
5 XEQ A ENTER	RUNNING 78.5398	半径を入力し、プログラム A を開始。計算結果の面積が表示されます。
2 . 5 XEQ E ENTER	19.6350	次の円の面積をプログラム E を使って求めます。
2 ↩ π ×		最後の円の面積をプログラム A で求めます。
XEQ A ENTER	124.0251	

プログラムのテスト

プログラムにエラーがあるのにその場所を特定できないときは、1行ずつ実行すると良いです。また、プログラムが長いものだったり複雑なときも、あらかじめテストする方が良いでしょう。1行ずつ実行すると、それぞれの行が実行されたときの結果がその都度わかり、既知のデータで正しい計算が行われることを確認できます。

1. 通常の実行時と同じように、プログラム入力モードがアクティブでないことを確認して下さい。(表示記号 **PRGM** がオフになっている状態)
2. LBL キーでプログラムポインタを最初の行にセットしてください。この操作ではプログラムポインタは移動しますが実行はされません。
3. ▽キーを押し続けてプログラム行を表示させます。▽キーを離すとその行が実行され、結果が表示され、X レジスタに保存されます。
   △キーで**前の**行に移動させることもできます。この操作ではプログラム行は実行されません。
4. プログラムポインタが次の行に移動します。エラーや結果の間違いを発見できるまで、あるいはプログラムの終了までステップ 3 を繰り返します。

プログラム入力モードがアクティブであれば、▽ と △ キーは単にプログラムポインタの移動のみでプログラム行の実行はされません。プログラム入力モードでカーソルキーを押し続けると行が自動スクロールします。

例: プログラムのテスト

プログラム A を1行ずつ実行してみます。テストのデータとして半径 5 を使います。プログラムの実行前に、プログラム入力モードが**オフ**であることを確認して下さい。

キー (RPN モード)	表示	説明
5 GTO A ENTER	5.0000	半径を入力し、プログラムカウンタをラベル A に移動させます。
⌄ (hold)	A001 LBL A	
(release)	5.0000	
⌄ (hold)	A002 x^2	半径の平方
(release)	25.0000	
⌄ (hold)	A003 π	π の値
(release)	3.1416	
⌄ (hold)	A004 ×	25π
(release)	78.5398	
⌄ (hold)	A005 RTN	プログラムの終了。
(release)	78.5398	結果が正しいことが確認できました。

データの入力と表示

HP 35s の**変数**は入力データや中間結果、最終結果の保存に利用できます。(変数についての詳細は第 3 章をご参照下さい。変数には A から Z の 1 文字が利用できますが、プログラムラベルとは無関係です。)

プログラムでは次の 3 種類の方法でデータを獲得できます。

- INPUT による獲得では、プロンプトを表示して入力を促し、変数へ数値を保存します。(最も使われる方法です)
- スタックからの獲得。(STO を使って変数に数値を保存することもできます)
- すでに変数に保存された数値を獲得

- **式の自動プロンプトからの獲得（フラグ 11 がセットされている場合に有効）。**
  式を利用するときは便利です。

プログラムの情報を表示させるには次の 3 種の方法があります。

- VIEW による方法では、変数の名前とその数値が表示されます。
  （最も手軽な方法です）
- スタックに出力する方法では X と Y レジスタの数値のみが表示できます。
  （PSE で X と Y レジスタの表示を 1 秒間ロックできます。）
- フラグ 10 がセットされていて式を表示させる方法。
  （式として評価されず、くメッセージとして表示されます）

次節でこれらの入出力方法の一部を解説します。

データの入力に INPUT を使う

INPUT(↩ INPUT **変数**) があるとそこでプログラムの実行が一時停止し、変数名とプロンプト、および変数に保存された既存の数値が次のように表示されます。

```
R?
0.0000
```

ここで、

"R" は変数名、
"?" はプロンプトであること、
0.0000 はその変数に保存されている数値を示しています。

プログラムを再開するには R/S *(run/stop)* ボタンを押します。数値を入力すると X レジスタにその数値が上書きされ、**さらに**その変数に保存されます。数値を入力しなかった場合は、変数に保存されている既存の数値が X レジスタに残ります。

円の面積のプログラムで INPUT を使うと次のようになります。

RPN モード	ALG モード
A001 LBL A A002 INPUT R A003 x^2 A004 π A005 × A006 RTN	A001 LBL A A002 INPUT R A003 SQ(R)×π A004 RTN

プログラムで **INPUT** 関数を使うには

1. どういう入力データを使うかを決定し、その変数の名前を割り当てます。（円の面積の例だと必要な入力データは半径のみで、その名前は R としました）
2. プログラムの最初に必要な変数の分の INPUT を定義しておきます。プログラムの後のほうでその変数を利用するときは「RCL**変数**」と操作することによりスタックに数値を呼び出します。

   INPUT は変数に保存する数値を X レジスタにも残すので、INPUT の直後であれば、その変数を再呼び出しする必要はありません。再呼び出ししなければその分メモリの使用量を削減できます。ただし、プログラムが長くなるとスタックの取り扱いが煩雑になりますから、（可能であれば）メモリを気にせずにプログラムの最初の方でデータを入力してしまって、必要なときに RCL で呼び出すと良いでしょう。

   プロンプトが出て、変数へのデータ入力待ちの状態においても計算ができます。計算を行うとスタックの状態が変更され、その後のプログラムによる計算に影響が出る可能性があります。つまり、INPUT の前後で X、Y、および Z レジスタの内容は変化するため、レジスタの状態に依存したプログラムは好ましくありません。プログラムの最初の方で全てのデータを収集し、あとで呼び出す方法では、スタックの内容変更による影響を防ぐことができます。

プロンプトへの応答:

プログラム実行のとき INPUT 行があると、そこで **R?0.0000** のように表示され、一時停止 します。表示される数値（X レジスタの数値）は変数に保存されている現在の値です。

- **数値を変更しないとき**: R/S キーを押します。
- **数値を変更するとき**: 新しい数値を入力し R/S キーを押します。入力された数値は X レジスタに上書きされます。必要であれば分数を入力することもできます。数値を計算してから入力したいときは、通常通りにキー入力して計算し、結果が出たら R/S キーで確定させます。たとえば RPN モードで 2 ENTER 5 y^x R/S 、ALG モードで 2 y^x 5 ENTER R/S と入力することができます。(ENTER キーを押す前にディスプレイ第 2 行に式が表示されます。ENTER キーを押した後は式の結果がディスプレイ第 2 行に入れ替わり、X レジスタに保存されます。)
- **INPUT プロンプトをキャンセルする**には、C キーを押します。変数に保存されていた値は X レジスタに残ります。R/S キーでプログラムを再開できます。するとキャンセルした INPUT プロンプトが再度あらわれます。数値の入力後に C キーを押すと入力した数値がクリアされゼロになります。再度 C キーを押すと INPUT プロンプトをキャンセルします。

VIEW を使ったデータ表示

VIEW をプログラムで定義しておく (↓ VIEW **変数名**) とプログラムが停止してその変数の内容を次のように表示します:

```
R=
78.5398
```

これは**表示のみ**であり、X レジスタへのコピーは行われません。分数表示モードがアクティブだった場合は分数として表示されます。

- ENTER キーを押すと表示中の数値は X レジスタにコピーされます。
- 数値が 2 進数や複素数、ベクトルなどで 14 桁より大きい場合、残りを表示させるには → ← ならびに → → と操作します。
- C (または ←) キーで VIEW の表示が消え、X レジスタが表示されます。
- → CLEAR で表示中の変数の内容をクリアします。

プログラムの再開には R/S キーを押します。

プログラムを停止させたくないときは後述の「停止させずに情報を表示する」をご参照下さい。

たとえば第 16 章の「正規分布」では T015 と T016 でルーチン T が X の結果を表示しています。ここの VIEW は RCL の直後に出ているという点に注目して下さい。RCL は必須ではありませんが、これにより VIEW で表示した変数を X レジスタにコピーすることで、手動での計算に利用できるので便利です。(VIEW の表示がされているときに ENTER キーを押しても同じ効果があります。) 第 16 章と第 17 章のその他の例でも、VIEW された変数が X レジスタにあるように設計されています。

メッセージの表示のために式を使う

式は評価されるまで文法チェックされません。よって、ほとんど全ての文字でも式としてプログラムに入力できます。入力するには任意のプログラム行で EQN を押すと開始します。数値、関数、シンボルキーも利用できます。英字の入力は RCL の後に一文字ずつおこないます。ENTER で式の入力を完了します。

フラグ 10 がセットされているときは**評価**される代わりに**表示**されます。これにより式をメッセージとして表示させることができます。(フラグについての詳細は第 14 章です)

メッセージが表示されるとプログラムは中断します。再開するには R/S を押します。表示されたメッセージが 14 文字より多い場合は表示記号➡が点灯します。 ▶ > ならびに ▶ < でスクロールさせて下さい。

プログラムを中断させたくないときは後述の「停止させずに情報を表示する」をご参照下さい。

例: プログラムにおける INPUT、VIEW、ならびにメッセージ.

円筒の表面積と体積を求めるプログラムを作りましょう。プログラム名 C (cylinder)、表面積 *S* (surface area)、体積 *V* (volume)、半径 *R* (radius)、高さ H (height) とし、次の公式を使います。

$$V = \pi R^2 H$$

$$S = 2\pi R^2 + 2\pi RH = 2\pi R\,(R + H)$$

キー (RPN モード)	表示	説明
▶ PRGM ▶ CLEAR 3 (3PGM) < (Y) ENTER	PRGM TOP	プログラム入力モードに入り、メモリをクリアします。
▶ LBL C	C001 LBL C	プログラムにラベルを付け

キー (RPN モード)	表示	説明
		ます。
◀ INPUT R	C002 INPUT R	
◀ INPUT H	C003 INPUT H	半径と高さの入力プロンプト
EQN ◀ π ×		体積の計算
RCL R y^x 2 ×		
RCL H ENTER		
	C004 π×R^2×H	
◀ SHOW	CK=74FE	式のチェックサムと長さ
	LN=7	
STO V	C005 STO V	計算した体積を *V* に保存
EQN 2 × ◀		表面積を計算
π × RCL R ×		
( ) RCL R +		
RCL H ENTER	C006 2×π×R×(R+➡	
◀ SHOW	CK=19B3	式のチェックサムと長さ
	LN=11	
STO S	C007 STO S	表面積を *S* に保存
◀ FLAGS 1		式を表示するためフラグ
(1SF) • 0	C008 SF 10	10 をオン
EQN RCL V		表示されるメッセージを入力
RCL O RCL L		
▶ SPACE + ▶		
SPACE RCL A		
RCL R RCL E		
RCL A ENTER	C009 VOL + ARE➡	
◀ FLAGS 1		フラグ 10 をクリア
(2CF) • 0	C010 CF 10	
◀ VIEW V	C011 VIEW V	体積 *V* を表示
◀ VIEW S	C012 VIEW S	表面積 *S* を表示
◀ RTN	C013 RTN	プログラムの終了
◀ MEM 2	LBL C	ラベル C の長さ
(2PGM) ENTER	LN=67	

キー (RPN モード)	表示	説明
↶ SHOW	CK=97C3 LN=67	プログラムのチェックサムと長さ
C C		プログラムの入力をキャンセル

では、半径 $2\,1/2$ cm、高さ 8 cm の円筒の表面積と体積を求めてみましょう。

キー (RPN モード)	表示	説明
XEQ C ENTER	R? *value*	*C* を実行。*R* のプロンプトが登場します。（*value* のところには既存の数値が表示されます）
2 · 1 · 2	H?	分数として $2\,1/2$ を入力。
R/S	*value*	続いて *H* のプロンプト。
8 R/S	VOL + AREA	メッセージが表示されます。
R/S	V= 157.0796	体積 [cm^3]
R/S	S= 164.9336	表面積 [cm^2]

停止させずに情報を表示

通常、VIEW や式メッセージの表示によりプログラムが停止します。再開するには R/S を押します。

必要であれば情報の表示中でもプログラムの実行を続けることができます。VIEW や式の表示の**直後**に PSE (pause) を入れて下さい。すると情報が表示された後、1 秒間の停止後に再開します。この場合はスクロールやキーボード入力はできなくなります。

他の情報の表示があれば、ディスプレイはクリアされます。また、フラグ 7 がセットされていて RND (分数への丸め) が実行されたときもディスプレイがクリアされます。

PSE をプログラムに入力するには ↷ PSE と操作します。

プログラムを一行ずつ実行するとき、VIEW と PSE 行、ならびに式と PSE 行はひとまとめの操作として取り扱われます。

プログラムの中断と一時停止

中断(STOP)と一時停止(PSE)を指定する

- プログラム入力モードで [R/S] (*run/stop*) を押すと STOP が挿入されます。STOP は X レジスタの内容を表示し、プログラムの実行を中断します。キーボードで [R/S] を押すと再開できます。プログラムポインタをメモリのトップに移動させたくないときは RTN ではなく STOP を使って下さい。
- プログラム入力モードで [↗] [PSE] と操作すると PSE (*pause*) が挿入されます。PSE ではプログラムが約 1 秒間停止し、X レジスタの内容を表示します。ただし、PSE が VIEW や表示式 (フラグ 10 がセットされている状態での式) の直後にあるときは、その変数や式を X レジスタの代わりに表示し、1 秒間の停止後にもその表示が残ります。

実行中のプログラムを停止する

実行中のプログラムは [C] か [R/S] で停止できます。停止の前に現在行が実行されます。[R/S] (*run/stop*) で再開できます。

プログラムの停止後に [XEQ]、[GTO]、[↖] [RTN] を押すと、[R/S] による再開ができなくなります。その場合はプログラムを再実行して下さい ([XEQ] ラベル 行番号)。

エラーによる停止

プログラム実行中にエラーが発生したら、プログラム実行が停止してエラーメッセージが表示されます。(付録 F にメッセージとその状態の一覧が出ています)

エラーが発生したプログラム行を確認するには [↗] [PRGM] と操作します。すると、プログラムが停止した行が表示されます。(たとえばゼロ割が発生した行など)

プログラムの編集

メモリに保存されたプログラムに対して行の挿入、削除、編集ができます。式を含むプログラム行であればその式も編集できます。

プログラム行を削除するには:

1. 目的のプログラムやルーチンを選択し、[v] や [^] で削除する行を選択します。スクロールし続けるにはカーソルキーを長押しして下さい。
2. [←]で選択中の行が削除されます（削除後に Undo することもできます）。削除後はポインタがその前の行に移動します。（複数のプログラム行を連続して削除するときは、そのグループの最後の行から削除すると簡単です。）
3. 必要であれば続けて新しい行を入力すると、削除された行に置き換わります。
4. プログラム入力を終了します。([C] または [↵] [PRGM])

プログラム行を挿入するには:

1. 挿入する位置の**前の行**を選択します。
2. 新しい行を入力します。1 で選択した位置の**直後**に挿入されます。

たとえば A004 と A005 の間に新しい行を挿入したいときは、A004 を選択して新しい行を入力します。A005 に続く行は後方に移動し、それぞれ行番号がひとつ増えます。

プログラム行の演算子、式、等号式を編集するには:

1. 編集対象の行を表示させるか位置を指定します。
2. [>] か [<] でプログラム行の編集を開始します。編集カーソル "_" が点滅しますが、プログラム行の削除は行われません。
   [>] はプログラム行の先頭で編集カーソルがアクティブになります。
   [<] はプログラム行の末尾で編集カーソルがアクティブになります。
3. 編集カーソルを移動させ、[←] で数値や関数を削除したうえで、必要な分を再入力してください。(間違って[←] を押したときアンドゥ機能が使えます)

注意:

1. プログラム行の編集カーソルがアクティブのとき、[v] と [^]キーは無効になります。

2. プログラム行を編集しているとき(編集カーソルがアクティブのとき)は、プログラム行に何も無い状態だと ← は無効になります。そのプログラム行を削除したければ、そのまま ENTER を押して下さい。
3. 長いプログラム行を編集せずに閲覧したいときは → と ← が使えます。
4. ALG モードでは ENTER は関数として利用できません。プログラム行の有効化で使います。
5. 式は入力されたモードと現在のモードが異なっていても編集できます。

プログラムメモリ

プログラムメモリの閲覧

→ PRGM と操作するとプログラム入力モードへの切り替えが行われます(表示記号 **PRGM** が点灯し、プログラム行が表示されます)。プログラム入力モードがアクティブのとき、プログラムメモリの内容が表示されます。

プログラムメモリは PRGM TOP から開始します。プログラムのリストは循環していますから、プログラムポインタは末尾から先頭に(その逆も)ジャンプします。

プログラム入力モードがアクティブのときにプログラムポインタ(表示中の行)を変更するには、次の 4 種類の方法があります。

- → ↑ と → ↓ はラベルからラベルへと移動します。ラベルが定義されていなければ、表示中のプログラムの先頭か末尾に移動します。
- プログラム行をスクロールさせるには ↓ か ↑ を長押しします。
- ポインタを PRGM TOP に移動させるには GTO . . と操作します。
- ラベルと行番号を指定して移動するには「GTO . ラベル名 行番号」と操作します。

プログラム入力モードがアクティブでないとき(つまりプログラム行が表示されていないとき)でも、プログラムポインタを「GTO . ラベル名 行番号」で移動させることができます。

プログラム入力モードをキャンセルしてもプログラムポインタの位置は変更されません。

メモリの使用

プログラム入力中に MEMORY FULL というメッセージが出たら、そのときに入力しようとしたプログラム行を保存するプログラムメモリの領域が無いことを示しています。他のプログラムやデータをクリアすることで領域を確保できます。メモリのクリアについての詳細はこの章で後述の「プログラムのクリア」、または「付録 B 電卓のメモリ管理」をご参照下さい。

プログラム一覧 (MEM)

プログラム一覧では、全てのプログラムラベルと、それぞれのラベルに関連づけられたプログラム行が消費しているメモリのバイト数が確認できます。[◀] [MEM] [2] (2PGM) と操作して表示できます。一覧をスクロールさせるには [▼] や [▲] キーを使います。

プログラム一覧は次のように利用できます:

- プログラムメモリにおけるラベルと、それぞれのプログラムのメモリ消費量の確認。
- ラベル付きプログラムを実行。(実行するにはラベルの表示中に [XEQ] か [R/S] キーを押して下さい)
- ラベル付きプログラムの表示。(ラベルの表示中に [▶] [PRGM])
- 特定のプログラムを削除。(ラベルの表示中に [▶] [CLEAR])
- チェックサムの確認。([◀] [SHOW])

プログラム一覧では、それぞれのラベル付きプログラムが消費しているバイト数が次のように確認できます。

```
LBL C
LN=67
```

67 はそのプログラムが消費しているバイト数です。

プログラムの削除

特定のプログラムをメモリから削除(クリア)するには:

1. [◀] [MEM] [2] (2PGM) [ENTER] と [▼] および [▲] で目的のプログラムのラベルを表示させます。

2. [↗] [CLEAR] と操作
3. プログラム一覧から抜けるには [C] を、メモリ操作に戻るには [←] を押します。

全てのプログラムをクリアするには:

1. [↗] [PRGM] でプログラム行を表示させます。(表示記号 **PRGM** が出ます)
2. [↗] [CLEAR] [3] (3PGM) でプログラムメモリをクリアします。
3. 確認のためプロンプト CLR PGMS? Y N が出ます。[<] (Y) [ENTER] と操作します。
4. [↗] [PRGM] でプログラム入力モードを終了させます。

全てのメモリをクリア ([↗] [CLEAR] [3] (3ALL)) する操作でも、全てのプログラムをクリアできます。

チェックサム

チェックサム (*checksum*) は、あるラベル付きプログラムとそれに関連づけられたプログラム行に対する、特有の 16 進数の数値です。既存のプログラムでチェックサムが既知であれば、それを再入力するときの比較に利用できます。既知のチェックサムと同じものが再入力後にも表示されれば、全てのプログラム行が正しく入力されているとわかり確認できます。

チェックサムを表示させるには:

1. [↖] [MEM] [2] (2PGM) [ENTER] でプログラム一覧を表示させます。
2. 必要であれば、カーソルキーで目的のラベルを選択します。
3. [↖] [SHOW] を長押しすれば「CK=チェックサム」 と 「LN=長さ」が表示されます。[SHOW] を離すと表示が消えます。

たとえば、先ほどの「円筒」の例のプログラムに対するチェックサムを表示させるには:

キー (RPN モード)	表示	説明
[↖] [MEM] [2] (2PGM) [ENTER]	LBL C LN=67	ラベル C を表示。 67 バイトを消費しています。
[↖] [SHOW] (hold)	CK=97C3 LN=67	チェックサムと長さ

チェックサムがこれと異なるようでしたらプログラムが正しく入力できていないということです。

第16章と第17章の全ての例題プログラムには、確認のためのチェックサムがラベル付きのルーチンごとに掲載されています。

プログラム中におけるそれぞれの式にもチェックサムがあります。詳細はこの章で前述の「プログラム行に式を入力するには」をご参照下さい。

プログラムできない機能

次の機能は HP 35s のプログラムで利用できません。

- [↩] [CLEAR] [3] (3PGM)
- [↩] [CLEAR] [3] (3ALL)
- [←]
- [↓], [↑], [←], [→]
- [↩] [PRGM]
- [↩] [↓], [↩] [↑]
- [↪] [UNDO]
- [GTO] [.] [.]
- [GTO] [.] ラベル名 行番号
- [↪] [MEM]
- [↪] [SHOW]
- [EQN]
- [↩] [FDISP]
- [↩] [CLEAR] [6] (6CLVARx)

基数変換 (BASE) を使ったプログラム

[↩] [BASE] を使って基数モード(BASE mode)をプログラムでも利用できます。

キーボードで基数変換するときと同じように動作します。これにより、4 種類の基数を相互に変換し、演算し、さらに任意の基数で結果を表示するようなプログラムを作成することもできます。

10 進数以外の基数でプログラムを作成するときは、そのプログラムの中で基数モードを目的のものに設定して下さい。方法は通常の使用時と同じです。

プログラムにおける基数モードの選択

プログラムの先頭行で BIN, OCT, HEX(2 進数、8 進数、16 進数)を入力します。プログラムの最終行で DEC(10 進数)にしておけば、プログラムの実行後に 10 進数の設定に戻ります。

プログラムにおける基数モードの変更により、入力される数値の基数、ならびに出力する数値の基数が決定されますが、入力するプログラム行には影響しません。

プログラム行に入力された数値

プログラムの入力を開始する前に、基数モードを設定しておいて下さい。現在の基数モードの設定がプログラム結果に影響します。

表示記号で現在の基数設定が確認できます。10 進数設定のときとそれ以外のときの相違点を次の例で確認しましょう。数値はディスプレイの左側に沿って表示されます。

10 進数（DEC）設定のとき:

```
:
:
PRGM
A009  BIN
A010  10
:
:
```

10 進数は基数の記号"ь"が省略されます。

2 進数（BIN）設定のとき:

```
:
:
 PRGM  BIN
 A009  BIN
A010  10ь
:
:
```

2 進数は基数の記号"ь"が表示されます。

多項式とホーナーの方法（Horner's Method)

多項式のような式では、同じ変数が式の中で何度も登場します。たとえば次のような式です。

$$Ax^4 + Bx^3 + Cx^2 + Dx + E$$

ここでは変数 x が 4 回登場します。このような式をプログラムで計算するとき、RPN では RCL 機能を使うことで、保存済みの変数 x から繰り返し獲得することができます。

例:

RPN で $5x^4 + 2x^3$ を解くプログラムを作成し、x = 7 として実行してみましょう。

キー (RPN モード)	表示	説明
[↗] [PRGM] [GTO]		
[·] [·]	PRGM TOP	
[↗] [LBL] [A]	A001 LBL A	
[↖] [INPUT] [X]	A002 INPUT X	
[5]	A003 5	5
[RCL] [X]	A004 RCL X	
[4]	A005 4	
[y^x]	A006 y^x	x^4
[×]	A007 ×	$5x^4$
[RCL] [X]	A008 RCL X	
[3]	A009 3	
[y^x]	A010 y^x	x^3
[2]	A011 2	
[×]	A012 ×	$2x^3$
[+]	A013 +	$5x^4 + 2x^3$
[↖] [RTN]	A014 RTN	
[↖] [MEM] [2]	LBL A	ラベル A を表示。
(2PGM)	LN=46	消費バイト数は 46。
[↖] [SHOW]	CK=EA18	チェックサムと長さ
	LN=46	
[C] [C]		プログラム入力の終了。

入力した多項式に x = 7 を与えてみます。

キー (RPN モード)	表示	説明
[XEQ] [A] [ENTER]	X?	x の入力プロンプト
	value	
[7] [R/S]	12,691.0000	結果

より汎用性の高い式でも試してみましょう。
$Ax^4 + Bx^3 + Cx^2 + Dx + E$

```
A001 LBL A
A002 INPUT A
A003 INPUT B
A004 INPUT C
A005 INPUT D
A006 INPUT E
A007 INPUT X
A008 RCL X
A009 RCL× A
A010 RCL+ B
A011 RCL× X
A012 RCL+ C
A013 RCL× X
A014 RCL+ D
A015 RCL× X
A016 RCL+ E
A017 RTN
```

チェックサムと長さ: **9E5E 51**

14

プログラムのテクニック

第 13 章ではプログラミングの基礎について解説しました。この章では次に示すような、より複雑で有用なテクニックについて記載します。

- プログラムを簡潔にするため、ある特定の役割を担う部分にラベルをつけることでプログラムを分割する方法（つまり、サブルーチンです）。サブルーチンを使えば、ひと続きの処理を何度も利用するときにも便利です。
- 比較とフラグを利用して、サブルーチンや処理する行をある条件で分岐させる方法。
- カウンタを使ったループを用いて、特定の部分を指定された回数だけ実行する方法。
- 間接変数を使い、同じプログラム行から条件に応じて異なる変数にアクセスする方法。

プログラムのルーチン

プログラムはひとつ以上の**ルーチン**(routine)で構成されています。ルーチンはある特定の動作をする機能的な単位です。複雑なプログラムでは、タスクの分割とグループ化のために複数のルーチンが必要になります。複数のルーチンがあれば、プログラムの作成が楽になるだけではなく、可読性が増し、作成後の理解や変更が容易になります。

通常、ルーチンはラベルで開始し、プログラムの終了かルーチンの実行 で終了します (STOP や RTN) 。

サブルーチンのコール (XEQ, RTN)

サブルーチンは他のルーチンからコール (call) され、そのサブルーチンが完了したとき、元のルーチンに戻る (return) という性質のルーチンです。

- メモリにひとつのプログラムだけを保存するということであれば、ルーチンを多数のラベルで区切ることができます。しかし、複数のプログラムを保存したいときはメインプログラムのラベルの一部分として記載し、行番号で区別した方が良いです。
- あるサブルーチンから他のサブルーチンをコールすることもできます。

この章のフロー図 (Flow Diagram) では次の表記方法を用いています。

A005 GTO B001 →①	プログラムの実行が、この行から ← ① がついた行 (説明文では "1 から" と記載) に分岐します 。
B001 LBL B ←①	プログラムの実行が、→ ① がついた行 (説明文では "1 へ" と記載) からこの行に分岐しています。

次の例では、与えられた数値の符号を変換するサブルーチンをコールしています。サブルーチン E はルーチン D の D003 XEQ E001 という行からコールされ、その数値の符号を変換します。サブルーチン E は RTN で終了し、ルーチン D の D004 行に処理を戻します（結果を保存し、表示するため）。フロー図も参考にして下さい。

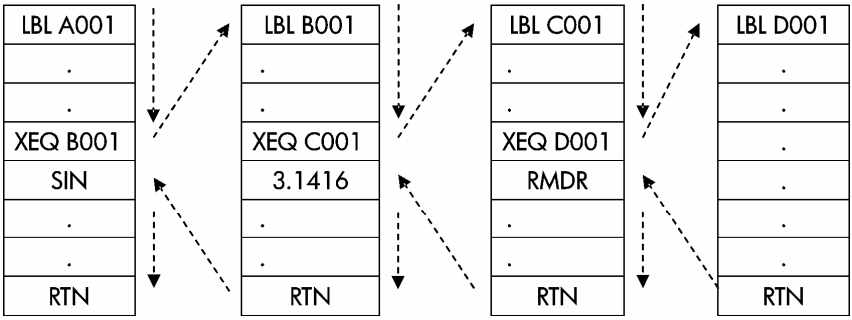
```
D001 LBL D                    開始行
D002 INPUT X
D003 XEQ E001    → ①          サブルーチン E をコール
D004 STO X       ← ②          ここに戻る
D005 VIEW X
D006 RTN
E001 LBL E       ← ①          サブルーチンの開始
E002 +/-                      その数値の符号を変更

E003 RTN         → ②          ルーチン D に戻る
```

サブルーチンの入れ子 (nesting)

サブルーチンから他のサブルーチンをコールすることができ、さらにそれを繰り返すこともできます。入れ子（nesting）の状態、つまり、サブルーチンから他のサブルーチンをコールする状態は、最上位のレベルを除いて 20 レベルの深さまで許容されています。

入れ子になったサブルーチンを図示すると次のようになります:

メインプログラム
(最上位)



プログラムの終了

入れ子のレベルが 20 を超えるとエラーとなり、メッセージ `XEQ OVERFLOW` が表示されます。

例: サブルーチンの入れ子

この例のサブルーチン S では次の式の値を計算します。

$$\sqrt{a^2 + b^2 + c^2 + d^2}$$

サブルーチン S は、(ここでは省略しますが)より長いプログラムの中の一部となっています。サブルーチン S から別のサブルーチン Q(サブルーチンの入れ子)をコールし、平方と加算を反復して計算させます。サブルーチンを入れ子にすることにより、メモリの消費を抑え、プログラムを簡潔にします。

RPN モードで記述します:

	S001 LBL S		サブルーチン S の開始
	S002 INPUT A		A を入力
	S003 INPUT B		B を入力
	S004 INPUT C		C を入力
	S005 INPUT D		D を入力
	S006 RCL D		データの呼び出し
	S007 RCL C		
	S008 RCL B		
	S009 RCL A		
	S010 x^2		A^2
	S011 XEQ Q001	→ ①	$A^2 + B^2$
② →	S012 XEQ Q001	→ ③	$A^2 + B^2 + C^2$
④ →	S013 XEQ Q001	→ ⑤	$A^2 + B^2 + C^2 + D^2$
⑥ →	S014 $\sqrt{x}$		$\sqrt{A^2 + B^2 + C^2 + D^2}$
	S015 RTN		メインルーチンに戻る
	Q001 LBL Q	← ①③⑤	入れ子になったサブルーチン
	Q002 x<>y		
	Q003 x^2		
	Q004 +		x^2 の加算
②④⑥ ←	Q005 RTN		サブルーチン S に戻る

分岐 (GTO)

サブルーチンをコールするときと同様に、プログラム処理を次の行ではなく、任意の行に転送することができます。これを分岐 (**branch**) と呼びます。

状況に依存しない分岐には GTO (*go to*) を使います。これはラベルと行番号でプログラム行を指定し、そこに分岐します。

GTO をプログラムに入力

「GTO ラベル名　行番号」で、指定されたプログラム行に処理が移動します。プログラムは引き続き、移動した行から実行されます。もとの行に処理が自動的に戻ってくることはありませんから、GTO はサブルーチンには利用できません。

たとえば第 16 章の「カーブフィッティング」では、**GTO Z 001** で最初の 3 つの独立な初期化ルーチンからラベル Z に移動します。ラベル Z は、それぞれの条件で共通の、プログラムの心臓部への入力点となっています。

```
S001 LBL S                ここから開始することもできます。
  .
  .
  .
S004 GTO Z001  →①         Z001 に移動


L001 LBL L                ここから開始することもできます。
  .
  .
  .
L004 GTO Z001  →①         Z001 に移動


E001 LBL E                ここから開始することもできます。
  .
  .
  .
E004 GTO Z001  →①         Z001 に移動


Z001 LBL Z     ←①         ここに移動
  .
  .
  .
```

キーボードで GTO を入力

プログラムポインタを特定の行に移動させるため、プログラムの実行を**開始しなくても** [GTO] が利用できます。

- **プログラムの先頭 PRGM TOPへの移動:** GTO ・ ・
- **特定の行への移動:** 「GTO **ラベル名 行番号」**(行番号< 1000)
  たとえば GTO A **0** **0** **5** で A005 へ移動します。ディスプレイには "GTO A005" と表示されます。
- **ラベルの先頭へ移動:** 「GTO **ラベル名** ENTER」
  たとえばGTO A ENTER **(長押し)**で A001 へ移動します。ディスプレイには "GTO A001" と表示されます。

条件分岐

プログラムの流れを変える方法には、GTO 以外にも、条件判定(*conditional test*)があります。条件判定では 2 つの数値を比較し、一致していなければ次の行をスキップします。

たとえば、A005 行で条件分岐 x=0? (x がゼロに等しいかどうか？)があったとき、X レジスタがゼロかどうか判定されます。X レジスタがゼロだった場合は、そのまま次行に進みます。X レジスタがゼロではないときは、次行を**スキップ**します。この例では A006 をスキップして A007 に進みます。このルールを「Do if true(真なら実行)」と呼びます。

```
                    A001 LBL A
                    .
                    .
                    .
真のとき次行へ      A005 x=0?        → ②   偽のとき次行をスキップ
          ① ←       A006 GTO B001
                    A007 LN          ← ②
                    A008 STO A
                    .
                    .
                    .
          ① →       B001 LBL B
                    .
                    .
                    .
```

上記の例は条件分岐を使った便利なテクニックです。判定の次行(真のときにのみ実行されます)に他のラベルへの分岐を設定しています。よって実質的には、条件判定により他のルーチンに分岐することになります。

条件分岐には次に示す 3 種類のものがあります。

- 比較判定。X レジスタと Y レジスタや、X レジスタとゼロを比較。
- フラグ判定。フラグの状態をチェックし、セットかクリアを判定します。
- ループのカウンタ。特定回数のループに利用します。

比較判定 (x?y, x?0)

比較演算子は全部で 12 個用意されています。[←] [x?y] か [→] [x?0] でそれぞれ、比較演算子メニューを表示します。

- x?y は x と y の比較
- x?0 は x とゼロの比較

x は X レジスタに保存された数値を参照し、y は Y レジスタに保存された数値を参照します。変数 X と Y ではないという点にご注意下さい。x?y と x?0 は数値を比較しますが、もしその数値が実数ではないときはエラーになり、メッセージ INVALID DATA が表示されます。

x?y と x?0 のメニューからどちらかを選び、メニューキーで必要な比較演算子を選択します。

比較演算子メニュー

x?y	x?0
≠ : x ≠ y?	≠ : x≠0?
≤ : x≤y?	≤ : x≤0?
< : x<y?	< : x<0?
> : x>y?	> : x>0?
≥ : x ≥y?	≥ : x≥0?
= : x=y?	= : x=0?

キーボードで条件分岐を実行すると、判定結果が YES や NO のようにディスプレイに表示されます。
たとえば、x =2 で y =7 のとき、 x<y を判定させると次のようになります。

	キー	表示
RPN モード	7 ENTER 2 [←] [x?y] [›] [›] (<) ENTER	YES
ALG モード	7 [x◂▸y] 2 [←] [x?y] [›] [›] (<) ENTER	YES

例:

第16章のプログラム「正規分布と逆正規分布」ではルーチンTで比較演算子 x<y? を利用しています。

プログラム行: (RPN モード)	解説
⋮	
T009 ÷	X *guess* の補正値を計算。
T010 STO+ X	補正値を新しい X *guess* として加算。
T011 ABS	
T012 0.0001	
T013 x<y?	補正値が適切かどうか判定。
T014 GTO T001	補正値が適切であれば最初のループに戻ります。適切でなければ処理を続けます。
T015 RCL X	
T016 VIEW X	計算した X の数値を表示。
⋮	

T009 行は X_{guess} の補正値を計算します。T013 行では、その補正値の絶対値と 0.0001 を比較しています。絶対値が 0.0001 より大きいときは T014 行を実行します ("Do If True") 。絶対値が 0.0001 以下のときは T015 行にスキップします。

フラグ判定

フラグにより、電卓の設定状態がわかります。それぞれのフラグにセット(真)かクリア(偽)の状態があります。**フラグ判定**は比較判定の一種であり、"Do If True" のルールが適用されます。フラグがセットされていれば次行が実行され、クリアのときは次行をスキップします。

フラグの意味

HP 35s には 12 個のフラグが用意されており、それぞれ 0 から 11 の番号が付けられています。全てのフラグはセットかクリアの状態を持ち、キーボード操作あるいはプログラムで判定できます。デフォルト設定では 12 のフラグが全てクリアです。全てのフラグをクリアするには「付録 B メモリのクリア」に掲載されている 3 つのキーを同時に押す方法を使います。 [↗] [CLEAR] [3] (3RLL) [<] (Y) [ENTER] はフラグには影響しません。

- **フラグ 0, 1, 2, 3, 4** には既定の意味は割り当てられていません。プログラム中で任意のフラグとして利用できます。（次の例をご参照下さい。）
- **フラグ 5** がセットされていると、プログラムの実行中にオーバーフローが発生したらメッセージ OVERFLOW と ⚠ を表示し、その実行が中断されます。オーバーフローが発生するのは、この電卓で取り扱い可能な数値の最大値を、プログラムの計算結果が超えたときです。オーバーフローとなったときの結果は、最大の正の値に置き換わります。フラグ 5 がクリアのときは、オーバーフローが発生しても実行が中断されませんが、プログラムの終了時にメッセージ OVERFLOW が少しの間、表示されます。
- **フラグ 6** はオーバーフロー TOO BIG が発生すると**自動的に**セットされます（ユーザが手動でセットすることもできます）。計算に影響があるわけではありませんが、オーバーフローになったかどうかの判定に利用できます。プログラムで 10 進数以外の基数を使っているときでも同様に、プログラム実行中に TOO BIG が発生したかどうかの状態をフラグ 6 は保持しています。

  フラグ 5 とフラグ 6 により、オーバーフロー状態を使ってプログラムを制御できます。フラグ 5 をセットしておけば、オーバーフローの発生した行の直後の位置でプログラムが停止します。また、プログラムの中でフラグ 6 を判定することで、オーバーフローの発生状況に応じてプログラムの流れを変更できます。
- **フラグ 7, 8, 9** は分数の表示方式を制御します。フラグ 7 はキーボードからでも変更できます。 [↗] [FDISP] で分数表示モードを切り替えると、フラグ 7 も切り替わります。

フラグ状態	分数のフラグ		
	7	8	9
クリア (デフォルト値)	分数表示モードがオフ。現在の表示形式で実数を表示。	分母に対し、/c を最大値とする任意の値を使う。	分数を約分する。
セット	分数表示モードがオン。実数を分数として表示。	分母を /c の約数とする。	分数を約分しない (フラグ 8 がセットされているときのみ有効)。

■ **フラグ 10** はプログラムにおける式の評価を制御します。

フラグ 10 がクリア(デフォルト設定)のときは、プログラム実行中に式が評価され、計算結果がスタックに残ります。

フラグ 10 がセットされていると、プログラム実行中に式がメッセージとして表示され、式を閲覧するときの状態と同じ挙動になります。

1. プログラムの実行は停止します。
2. プログラムポインタは次行に移動します。
3. ディスプレイに式が表示されますが、スタックには影響しません。この表示は [←] か [C] でクリアできます。その他のキーを押すと、そのキーの機能が実行されます。
4. 式の次行が PSE だったときは、1 秒間の停止後にプログラム実行が再開します。

フラグ 10 の状態を変更するには、キーボードで SF(Set Flag)または CF(Clear Flag)を指定するか、プログラムで SF や CF を使います。

■ **フラグ 11** はプログラムの中の式を評価するときのプロンプトの設定です。このフラグは、キーボードから実行したときに自動で表示されるプロンプトには**影響しません**。

フラグ 11 がクリア(デフォルト設定)のときは、式の評価、SOLVE、ならびに式の積分(∫FN)でもプログラムの処理が中断しません。式の中で変数が使われていることが判明した時点で、その変数の現在値が自動的に呼び出されます。INPUT プロンプトはこのフラグの影響を受けません。

フラグ 11 がセットであれば、式で使われている変数がその式で最初に登場するときにプロンプトが表示されます。プロンプトはそれぞれの変数について 1 回のみ表示されます。その変数が式の中で複数回登場しても、プロンプトは 1 回のみです。SOLVE のときは未知変数についてのプロンプトは表示されません。また、積分では被積分変数についてのプロンプトは表示されません。
プロンプトが表示されるとプログラム実行が停止します。[R/S] を押すと、変数に対して入力した値、または表示された値（その変数の現在の値）を使ってプログラム実行が再開します。プロンプトが出た後に数値を入力せずに[R/S]キーを押すと、その変数の現在の値が使われます。

フラグ 11 はプログラムで式の評価や SOLVE、積分が終わると自動的にクリアになります。フラグ 11 の状態を変更するには、キーボードで SF または CF を指定するか、プログラムで SF や CF を使います。

フラグの表示記号

フラグ 0、1、2、3、4 には表示記号が用意されており、セットになっているかどうかをディスプレイで確認できます。セットになっていると、それぞれの表示記号 **0**、**1**、**2**、**3**、**4** が点灯します。ただし、フラグ 5 から 11 には表示記号は存在しません。これらのフラグはキーボードで FS? を実行することで確認できます。詳細は後述の「フラグの利用」をご参照下さい。

フラグの利用

[↶] [FLAGS] でフラグの設定メニュー SF CF FS? が表示されます。

必要な機能を選択すると、フラグ番号 (0 から 11) の入力を促されます。
たとえばフラグ 0 をセットするには [↶] [FLAGS] [1](1SF) [0] と操作します。
フラグ 10 をセットするには [↶] [FLAGS] [1](1SF)[.][0] です。
フラグ 11 をセットするには [↶] [FLAGS] [1](1SF) [.][1] です。

フラグメニュー

メニューキー	説明
SF *n*	フラグ *n* をセット (Set Flag)。
CF *n*	フラグ *n* をクリア (Clear Flag)
FS? *n*	フラグ *n* の判定 (Flag Set?)

プログラムでのフラグ判定は条件判定の一種で、プログラムの実行に影響します。FS? *n* で、フラグ *n* がセットされているかどうかを判定します。セットされていればプログラムの次行が実行されます。クリアのときは、次行はスキップされます。この章で前述した、比較判定の "Do if True" ルールと同じですね。

キーボードでフラグ判定するときは、ディスプレイに "YES" または "NO" が表示されます。

フラグの状態に依存したプログラムを、状態に依存しないものに変更していく作業は、プログラミングの良い練習になるでしょう。現在のフラグ設定は、その前に実行したプログラムによって変更されているかもしれません。どのフラグでも、それがクリアかセットかを決めつけるのは危険です。状況によって自動的にセットになるフラグもあるので、必要であれば明示的に指定しておくべきです。次の例も参考にして下さい。

例: フラグの利用

プログラム行 (RPN モード)	説明
S001 LBL S	
S002 CF 0	*X* の自然対数に利用するフラグ 0 をクリア
S003 CF 1	*Y* の自然対数に利用するフラグ 1 をクリア
S004 INPUT X	*X* のプロンプト
S005 FS? 0	フラグ 0 がセットされていれば…
S006 LN	… *X* の自然対数を出力
S007 STO X	フラグの判定後、*X* に数値を保存
S008 INPUT Y	*Y* のプロンプト
S009 FS?1	フラグ 1 がセットされていれば…
S010 LN	… *Y* の自然対数を出力
S011 STO Y	フラグの判定後、*Y* に数値を保存
S012 VIEW X	*X* の値を表示
S013 VIEW Y	*Y* の値を表示
S014 RTN	

チェックサムと長さ: 16B3 42

この例では S002 CF0 と S003 CF1 により、フラグ 0 とフラグ 1 がクリアされています。よって S006 と S010 で X と Y に対する自然対数が計算されることはありません。

S002 を SF 0 に変更するとフラグ 0 がセットされ、S006 で X の自然対数が計算されます。

S002 を CF 0 に、S003 を SF 1 にすればフラグ 1 がセットされ、S010 で Y の自然対数が計算されます。

S001 が SF 0 で、S003 が SF 1 のときはフラグ 0 とフラグ 1 がセットされますから、S006 と S010 でそれぞれ、X と Y の自然対数が計算されます。

このプログラムを実行したときの挙動を確認しましょう。

キー (RPN モード)	表示	説明
XEQ S ENTER	X?	ラベル *S* を実行。
	value	*X* のプロンプトが表示されます。
1 R/S	Y?	*X* に 1 を保存。
	value	続けて *Y* のプロンプト。
1 R/S	X=	*Y* に 1 を保存。
	1.0000	フラグ判定の後に *X* の値を表示。
R/S	Y=	フラグ判定の後に *Y* の値を表示。
	1.0000	

フラグの設定を変更すれば、異なる結果になります。この例の確認が終わったら、 ↩ FLAGS 2 (2CF) 0 と ↩ FLAGS 2 (2CF) 1 でフラグ 0 とフラグ 1 をクリアしておきましょう。

例: 分数表示の制御

この例では分数表示の機能を利用しています。プロンプトで得られた数値と分母の最大値（/c）を使って分数を表示します。このプログラムでは分数の 3 個のフラグ（7、8、9）およびメッセージ表示のフラグ（10）も利用されています。

表示されるメッセージは MESSAGE のように記載しています。これは式として入力します。

1. EQN キーを押し、式入力モードをセットします。（表示記号 **EQN** がオン）
2. それぞれの英字について「RCL 文字キー」でメッセージを入力します。スペースは ↪ SPACE です。
3. メッセージの入力が終わったら ENTER キーで式入力モードを終了し、プログラム行として登録します。

	プログラム行 (RPN モード)	説明
F001	LBL F	分数プログラム F の開始
F002	CF 7	分数のフラグを全てクリア
F003	CF 8	
F004	CF 9	
F005	SF 10	メッセージを表示させるフラグをセット
F006	DEC	基数は 10 進数に
F007	INPUT V	変数 V のプロンプト
F008	INPUT D	変数 D のプロンプト (分母として利用、範囲は 2 から 4095)
F009	RCL V	
F010	DECIMAL	メッセージ DECIMAL を表示し、変数 V を 10 進数で表示
F011	PSE	
F012	STOP	
F013	RCL D	
F014	/C	/C を設定し、フラグ 7 をセット
F015	RCL V	
F016	MOST PRECISE	メッセージMOST PRECISEと分数を表示
F017	PSE	
F018	STOP	
F019	SF 8	フラグ 8 をセット
F020	FACTOR DENOM	メッセージFACTOR DENOMと分数を表示
F021	PSE	
F022	STOP	
F023	SF 9	フラグ 9 をセット
F024	FIXED DENOM	メッセージFIXED DENOMと分数を表示
F025	PSE	
F026	STOP	
F027	GTO F001	プログラムの先頭に戻る

チェックサムと長さ: BE54　123

このプログラムは分数の様々な表示形式を確認できます。

キー (RPN モード)	表示	詳細
[XEQ] [F] [ENTER]	V? *value*	ラベル *F* を実行。分数として表示する数値 *V* のプロンプトが表示されます。
[2] [.] [5] [3] [R/S]	D? *value*	変数 *V* に 2.53 を入力。 分母の最大値 *D* のプロンプト。
[1] [6] [R/S]	DECIMAL 16.0000 2.5300	*D* (/c) に 16 を入力。メッセージの後に入力した 10 進数を表示。
[R/S]	MOST PRECISE 2 8/15 ▼ 2 8/15	分数表示形式（分母は 16 を最大値とし、精度優先）を示すメッセージが 1 秒間と、続けて分数が表示されます。▼は「分子が 8 より少し少ない」ことを示しています。
[R/S]	FACTOR DENOM 2 1/2 ▲ 2 1/2	分数表示形式（分母は 16 の約数）を示すメッセージが 1 秒間と、続けて分数が表示されます。
[R/S]	FIXED DENOM 2 8/16 ▲	分数表示形式（分母は 16 で固定）を示すメッセージが 1 秒間と、続けて分数が表示されます。
[R/S] [C] [←] [FLAGS] [2] (2CF) [.] [0]	2.5300 2.5300	プログラムを停止し、フラグ 10 をクリア

ループ

後方、つまり前の行への分岐により、プログラムのある部分を複数回実行できます。これは「ループ (loop)」と呼ばれます。

```
D001 LBL D
D002 INPUT M
D003 INPUT N
D004 INPUT T
D005 GTO D001
```

このルーチンは**無限ループ**(infinite loop)の例で、初期データの収集に利用できます。3 個のデータを入力後、「XEQ **ラベル名**」と操作して手動でループを終了し、他のルーチンを実行します。

条件に応じたループ (GTO)

ある条件になったときに何かの操作をする場合で、ループの実行回数が想定できないときは、GTO を使って、条件に応じたループを作成すると良いです。

たとえば、次のルーチンでは値 A が定数 B 以下になるまで A–B を実行するループを使っています。

プログラム行 (RPN モード)	説明
S001 LBL S	
S002 INPUT A	
S003 INPUT B	
S004 RCL A	*A* を呼び出し (RCL) する方が、スタックの位置を覚えるより簡単です。
S005 RCL– B	*A* – *B* を計算
S006 STO A	古い *A* を新しい結果で上書き
S007 RCL B	比較のために *B* を呼び出し
S008 x<y?	B < 新しい *A* ?
S009 GTO S004	真: ループを繰り返す
S010 VIEW A	偽: 新しい *A* を表示
S011 RTN	

チェックサムと長さ: 2737 33

カウンタによるループ (DSE, ISG)

ループを回数を指定して実行したいときは、条件関数

[↩] [ISG] (**増加; より大きければスキップ** … *Increment; Skip if Greater than*)

または

[↪] [DSE] (**減少; 以下であればスキップ** … *Decrement; Skip if less than or Equal to*) を使います。

プログラムでそれぞれのループが実行されたとき、変数に保存されたカウンタ値が自動的に増加または減少します。次に、現在のカウンタ値と最終のカウンタ値を比較し、ループを続けるか脱出するかが決定されます。

カウント値を減少させるループには「[↪] [DSE] **変数名**」を使います。

カウント値を増加させるループには「[↩] [ISG] **変数名**」を使います。

この機能は BASIC 言語の FOR–NEXT ループに相当します:

```
FOR 変数名 = 初期値 TO 最終値 STEP 増減値
.
.
.
NEXT 変数名
```

DSE は FOR–NEXT ループで増減値が負の値の場合に相当します。

ISG ([↩] [ISG])や DSE([↪] [DSE])を入力すると、カウンタ値を格納するための変数名を求めるプロンプトが出てきます。(詳細は下記).

ループの制御値

ループ制御に指定された変数に保存される数値の形式は ±ccccccc.*fffii* です。

- ±ccccccc : **現在のカウンタ値** (1 から 12 桁)。**ループの実行に伴って変化します。**
- *fff* : **カウンタの最終値** (3 桁)。**ループの実行中に変化しません。指定されていなければ** 000 **が使われます。**
- *ii* : **増減値** (2 桁または指定無し)。**ループの実行中に変化しません。指定無しのときは** 01 **が使われ、増減値は** 1 **か** −1 **になります。**

ループの制御値が *ccccccc.fffii* のとき、DSE では *ccccccc* を *ccccccc — ii* として減少させ、新しい *ccccccc* と *fff* を比較します。もし *ccccccc* ≤ *fff* だったら次行をスキップします。

ループの制御値が *ccccccc.fffii* のとき、ISG では *ccccccc* を *ccccccc + ii* として増加させ、新しい *ccccccc* と *fff* を比較します。もし *ccccccc* > *fff* だったら次行をスキップします。

現在値 > 最終値であればループを継続	①→ ①←	W001 LBL W . . . W009 DSE A W010 GTO W001 W011 XEQ X001 . . .	→② ←②	現在値 ≤ 最終値であればループを終了
現在値 ≤ 最終値であればループを継続.	①→ ①←	W001 LBL W . . . W009 ISG A W010 GTO W001 W011 XEQ X001 . . .	→② ←②	現在値 > 最終値であればループを終了

たとえば、ループの制御値が 0.050 で条件関数が ISG のとき、カウンタはゼロから開始し、50 まで増加します。ループが実行されたときの増加値は 1 です。

ループの制御値が複素数のときは実数部が、ベクトルのときは最初の部分の数値が用いられます。

次の RPN プログラムでは ISG を用いて 10 回のループを使っています。ループカウンタ 1.010 は変数 Z に保存します。ループカウンタの先頭と末尾のゼロは省略できます。

```
L001 LBL L
L002 1.01
L003 STO Z
L004 ISG Z
L005 GTO L004
L006 RTN
```

[XEQ] [L] [ENTER]で実行した後、変数 Z を[↶] [VIEW] [Z] で確認できます。ループの制御値が 11.0100 になっているはずです。

変数とラベルの間接処理

間接処理 (indirect addressing) は高度なプログラムテクニックで、変数やラベルを**あらかじめ明示的に指定せずに**呼び出す方法です。プログラム実行中に、中間の計算結果(または入力)に依存して呼び出す変数やラベルが決定されます。

間接アドレスでは 4 個のキー [I], [(i)], [J] , [(j)] を使います。

これらのキーと、変数やラベルの A から Z を利用すると多くの機能が利用できます。

- **I と J は、他の変数を参照できる特別な変数です。他の変数 (A から Z) と同様に数値を保持します。**
- **(I) と (J) はプログラム用の機能で、**
  「変数 I と J に保存された数値で、呼び出す変数またはラベルを決定する」ことができます。
  この 2 つは間接アドレス (*indirect address*) **で登録されています。**
  これに対し、A から Z はダイレクトアドレス (*direct address*) **になっています。**

間接処理を使うときは、[I] と [(i)]、ならびに [J] と [(j)] をそれぞれ一緒に使います。

(I) や (J) は単体では、未定義（数値が保存されていない）状態か、または、制御できない（I や J にどんな数値が保存されるか制御できない）状態のどちらかです。

変数 "I" と "J"

他の変数と同様に、変数 I と J にも数値を保存したり、呼び出したり、操作することができます。SOLVE の未知変数や、積分の被積分変数として I や J を指定することもできます。次の表に、変数 I を利用できる関数をまとめています。(J も同じです)

STO I	INPUT I	DSE I
RCL I	VIEW I	ISG I
STO +,–, × ,÷ I	∫FN d I	x < > I
RCL +,–, × ,÷ I	SOLVE I	

(I) と (J) による間接処理

変数(またはラベル)A から Z を使う多くの関数で、変数(またはラベル)A から Z ならびに統計レジスタを間接に参照させるために (I) と (J) が利用できます。(I) と (J) は変数 *I* と *J* を使って、どの変数、ラベル、レジスタを参照するのかを決定します。次表でどの情報が参照されるかを示します。

I / J の内容	(I) / (J) が示すもの
-1	変数 A またはラベル A
. . .	. . .
-26	変数 Z またはラベル Z
-27	レジスタ *n*
-28	レジスタ Σx
-29	レジスタ Σy
-30	レジスタ Σx^2
-31	レジスタ Σy^2
-32	レジスタ Σxy
0	ここから名前のない 間接変数
. . .	. . .
800	最大のアドレスは 800
I <- 32 または I>800、 あるいは I が未定義のとき	エラー: INVALID (I)
J<-32 または I>800、 あるいは J が未定義のとき	エラー: INVALID (J)

INPUT**(I)** と INPUT**(J)** 、ならびに VIEW**(I)** と VIEW**(J)** は間接変数またはレジスタの名前と保存されたデータを表示します。

SUMS メニュー (統計データの和) では、統計レジスタの数値を再度呼び出すことができます。ただし他の操作、つまり STO、VIEW、INPUT では間接アドレスを使う必要があります。

次表の関数では **(I)** と **(J)** を利用できます。FN = の場合のみ、**(I)** と **(J)** はラベルを参照します。その他の関数は **(I)** と **(J)** は変数かレジスタを参照します。

STO**(I)/(J)**	INPUT**(I)/(J)**
RCL**(I)/(J)**	VIEW**(I)/(J)**
STO +, −,× ,÷, **(I)/(J)**	DSE**(I)/(J)**
RCL +, −,× ,÷, **(I)/(J)**	ISG**(I)/(J)**
X<>**(I)/(J)**	SOLVE**(I)/(J)**
FN=**(I)/(J)**	∫FN d**(I)/(J)**

名前のない変数とレジスタは SOLVE と積分ができません。

(I) / (J) によるプログラムの制御

STO (I) や STO (J) により、プログラムまたはその一部が実行されるたびに、変数 *I* や変数 *J* を他の変数の中身に変更できます。たとえば、STO(−1) であれば変数 A に保存された数値を呼び出します。この機能により、必要な変数またはプログラムラベルをプログラムの実行時まで未定義の状態に保つことができ、プログラムの柔軟性を確保できます。

間接処理はループのカウントや制御にたいへん便利な機能です。関数 DSE や関数 ISG のループの制御値を保存した変数の、メモリ上のアドレスを変数 *I* や変数 *J* に記録し、これを目次（*index*）として利用します。

(I) / (J) を使った式

式で変数を間接指定するときも、**(I)** や **(J)** が利用できます。 **(I)** と **(J)** は変数 *I* と *J* に保存された数値によって特定される変数（間接参照）を意味しますが、*I* または *J* はそれぞれ変数 *I* と *J* を意味します。(I) や (J) キーの代わりにカッコキーを使って入力した(I)と(J)も変数 I と J と見なされます。

名前のない間接変数

変数 *I* や変数 *J* に正の数値を保存すると、801 個までの間接変数にアクセスできます。次の例では間接変数を使っています。

プログラム行 (RPN モード)	説明
A001 LBL A	
A002 100	
A003 STO I	
A004 12345	
A005 STO (I)	間接レジスタであるアドレス 100 に “12345” を保存。利用するアドレスの範囲は “0 から 100”。
A006 150	
A007 STO I	
A008 67890	
A009 STO (I)	間接レジスタであるアドレス 150 に “67890” を保存。利用するアドレスの範囲は “0 から 150”。
A010 100	
A011 STO I	
A012 0	
A013 STO (I)	間接レジスタであるアドレス 100 に “0” を保存。利用するアドレスの範囲は “0 から 150” のまま。
A014 170	
A015 STO I	
A016 RCL(I)	アドレス 170 は未定義なので “INVALID (I)” と表示されます。
A017 RTN	

注意:

1. 未定義のストレージアドレスをコールすると、エラーメッセージ INVALID (I) が表示されます。(上記の A014 行)
2. 間接レジスタを使うと、アドレス 0 から最後のゼロでない数値が保存されたアドレスまでのメモリ領域が確保されます(この例だと当初 0～100 で、A009 行からは 0～150)。直接レジスタの使用後は、メモリを解放するために A013 行のようにゼロを上

書きしてください[†]。これは重要な作業です。それぞれの間接レジスタはプログラムメモリの 37 バイトを消費します。

3. 変数の数は最大で 800 です。

[†] 訳注：この例だと、アドレス 150 にゼロを保存すると間接レジスタのメモリ領域が全て解放されます。

15

プログラムの SOLVE と積分

プログラムの SOLVE

SOLVE を使って、式リストに入力された式をある変数について解く方法について、第 7 章で解説しました。ある関数を計算するプログラムも式と同じように、全ての変数について SOLVE できます。特定の条件で変化する式や、繰り返し計算が必要な問題を解くときは、特に便利な機能です。

プログラムを SOLVE するには:

1. **関数を定義したプログラムを入力（詳細はこの章で後述の「SOLVE のためのプログラムを書くには」をご参照下さい）。**
2. **「[↻] [FN=] ラベル名」で、SOLVE の対象のプログラムを選択。（このステップは同じプログラムを再度 SOLVE するときは不要です）**
3. **「[↻] [SOLVE] 変数名」で、未知変数について SOLVE を実行します。**

関数を定義したプログラムを SOLVE するには [FN=] を指定します。これは、式リストの式を SOLVE するときは不要な操作です。

計算を中断するには [C] か [R/S] を押します。するとメッセージ INTERRUPTED がディスプレイ第2行に表示されます。それまでの計算における最良の根が未知変数に保存されます。これをスタックに呼び出さずに確認するには [↻] [VIEW] と操作します。中断した計算を再開するには [R/S] キーを押します。

SOLVE のためのプログラムを書くには:

プログラムでは式を利用できます。入力モードは ALG も RPN でも、両者が混在していても問題ありません。

1. **プログラムはラベルから開始します。このラベル名は SOLVE が評価する関数になります。（(FN= ラベル名)**

2. 未知変数も含め、それぞれの変数に対して INPUT 行を定義します。多変数の関数においても、INPUT を定義しておけばどんな変数についてでも SOLVE を実行できます。未知変数に INPUT が定義されていても、SOLVE のときは無視されます。よって、未知変数も含めて全ての変数について、あらかじめ INPUT を定義しておくと便利です。

   INPUT が定義されていない変数があれば、その変数に保存された数値か、式プロンプトで入力された数値が使われます。
3. 関数を評価するための指示をプログラムに定義しておきます。
   - RPN や ALG の複数行で構成されるプログラム関数は、等号のない式（Expression）の形式で作成し、その式の値がゼロになるようにして下さい。たとえば式が *f(x) = g(x)* のとき、プログラムでは *f(x) – g(x)* と定義して、"=0" を省略して下さい。
   - 単一の式のみを含むプログラム関数では、式の種類は制限されません。つまり、等号式、代入式、等号を含まない式の全てが利用できます。定義した式はプログラムで評価され、右辺がゼロになるように調整されます。INPUT を定義せずに式の変数についてのプロンプトを表示させたいときは、フラグ 11 をセットしておいて下さい。
4. プログラムの最終行は RTN として下さい。プログラムの実行は X レジスタに関数の計算結果を出力して終了する必要があるからです。

例: ALG を使ったプログラム

「理想気体の法則」の未知変数を SOLVE で求める ALG プログラムを作成しましょう。この方程式は次の通りです。

$$P \times V = N \times R \times T$$

ただし変数は次の通りです。
P = 圧力 (気圧または N/m^2)
V = 容積 (リットル).
N = 気体のモル数
R = 気体定数 (0.0821 liter–atm/mole–K または 8.314 J/mole–K).
T = 温度 (ケルビン; K = °C + 273.1).

まず、プログラムモードに設定し、必要であればプログラムポインタをメモリの先頭に移動させます。

キー (ALG モード)	表示	説明
[→] [PRGM]		プログラムモードへの移行とポインタの移動
[GTO] [·] [·]	PRGM TOP	

プログラムを入力します:

プログラム行 (ALG モード)	説明
G001 LBL G	プログラム関数のラベル
G002 INPUT P	P: 圧力
G003 INPUT V	V: 体積
G004 INPUT N	N: 気体のモル数
G005 INPUT R	R: 気体定数
G006 INPUT T	T: 温度
G007 P×V=N×R×T	[EQN]を押して式を入力。 圧力 × 体積= モル数 × 気体定数 × 温度
G008 RTN	プログラムの終了

チェックサムと長さ: F425 33

プログラム入力モードをキャンセルするには [C] を押します。

このプログラム "G" を使って、2リットルの容器に 24°C で保存されている 0.005 モルの二酸化炭素の圧力を求めてみましょう。

キー (ALG モード)	表示	説明
[←] [FN=] [G]		プログラム "G" を選択。未知変数について SOLVE を実行します。
[→] [SOLVE] [P]	V?	未知変数は *P* にします。
	value	*V* のプロンプトが登場します。
[2] [R/S]	N?	*V* に 2 を保存。
	value	引き続き *N* のプロンプト。
[·] [0] [0] [5] [R/S]	R?	*N* に .005 を保存。
	value	続いて *R* のプロンプト

[.] [0] [8] [2] [1]	T?	*R* に .0821 を保存。
[R/S]	*value*	続いて *T* のプロンプト
[2] [4] [+] [2] [7]	T?	*T* を計算して保存。
[3] [.] [1] [ENTER]	297.1000	
[R/S]	SOLVING	*P* について SOLVE を実行。
	P=	圧力は 0.0610 気圧です。
	0.0610	

例: 式を使ったプログラム

「理想気体の法則」を式を使ったプログラムで解いてみましょう。

キー (RPN モード)	表示	説明
[↗] [PRGM]		プログラム入力モード。
[GTO] [.] [.]	PRGM TOP	プログラムポインタを先頭へ移動。
[↗] [LBL] H	H001 LBL H	ラベルを H に設定
[↖] [FLAGS] [1]		式のプロンプトを設定
(1SF) [.] [1]	H002 SF 11	
[EQN]		式を評価し、フラグ 11 をクリア。
[RCL] [P] [×]		
[RCL] [V] [↖] [=]		チェックサムと長さ： EDC8 9
[RCL] [N] [×]		
[RCL] [R] [×]		
[RCL] [T] [ENTER]	H003 P×V=N×R×T	
[↖] [RTN]	H004 RTN	プログラムの終了
[C]	0.0610	プログラム入力モードの終了

プログラムのチェックサムと長さ: DF52 21

では、前回の例から二酸化炭素の温度が 10 °C 下がったときの圧力の変化を計算しましょう。

キー (RPN モード)	表示	説明
[→] [STO] [L]	0.0610	前回計算した圧力を保存
[←] [FN=] [H]	0.0610	プログラム H を選択
[→] [SOLVE] [P]	V?	未知変数に *P* を選択。
	2.0000	*V* のプロンプト。
[R/S]	N?	*V* に 2 を入力。
	0.0050	続いて *N* のプロンプト。
[R/S]	R?	*N* に .005 を入力。
	0.0821	続いて *R* のプロンプト
[R/S]	T?	*R* に .0821 を入力。
	297.1000	続いて *T* のプロンプト。
[ENTER] [1] [0] [−]	T?	新しい *T* を計算。
	287.1000	
[R/S]	SOLVING	*T* に 287.1 を保存。
	P=	新しい条件で *P* を求めます。
	0.0589	
[RCL] [L] [−]	-0.0021	温度が 297.1 K から 287.1 K に変化するときの、圧力の変化を計算します。（負の場合は圧力の低下を意味します）

プログラムで SOLVE を利用する

プログラムの中に SOLVE を挿入しておくことも可能です。

可能であれば、「SOLVE 変数名」の行の前に初期推定値 (Initial Guess) のプロンプトを未知変数と X レジスタに定義しておくと良いです。プログラムの中で未知の変数について式を SOLVE させるには、次の 2 行を定義して下さい。

```
FN= ラベル名
SOLVE 変数名
```

プログラム中の SOLVE では、計算で得られた結果（変数=値）がディスプレイに表示されません。たとえば、SOLVE の処理後に更に計算を行う場合など、SOLVE の結果を表示するのがふさわしくないことがあるからです。結果を表示したければ「VIEW 変数名」という行を追加してください。

未知変数について解が見つからないときは、次のプログラム行がスキップされます（第 14 章でも登場した "Do If True" ルールです）。このような場合はプログラムのほうで、初期推定値や初期値を変更するなどの処理が必要になるでしょう。

例: プログラムの中の SOLVE

次に示すのは、XEQ X または Y と操作したとき、x または y を未知変数として SOLVE を実行するプログラムの抜粋です。

プログラム行 (RPN モード)	説明
X001 LBL X	*X* の条件設定
X002 24	*X* の目印
X003 GTO L001	メインルーチンへの分岐
チェックサムと長さ: 62A0 11	
Y001 LBL Y	*Y* の条件設定
Y002 25	*Y* の目印
Y003 GTO L001	メインルーチンへの分岐
チェックサムと長さ: 221E 11	
L001 LBL L	メインルーチン
L002 STO I	変数 *I* に目印を保存
L003 FN= F	SOLVE するプログラムを指定
L004 SOLVE(I)	目的の変数を SOLVE
L005 VIEW(I)	解を表示
L006 RTN	プログラムの終了
チェックサムと長さ: D45B 18	
F001 LBL F : : F010 RTN	*f* (x,*y*)の計算。 必要に応じて INPUT や式を定義します。

プログラムの積分

第 8 章では、式リストに追加された式を、ある変数について積分する方法について解説しました。ある関数を計算するプログラムであれば、そこで定義されたどの変数についても積分することができます。ある条件で被積分関数が変化したり、繰り返し計算が必要なときは特に便利です。

関数のプログラムを積分するには:

1. 被積分関数を定義するプログラムを作成。
   （後述の「∫FN のためのプログラムを書くには」を参照して下さい）
2. 「[↩] [FN=] ラベル名」で、被積分関数を定義したプログラムを指定。
   （前回と同じプログラムを再度積分するときは、このステップは不要）
3. 「上限値 [ENTER] 下限値」と積分区間を入力。
4. 「[↩] [∫] 変数名」で積分変数を指定。

関数プログラムを積分するには FN=が必要になります。(FN= は式リストの式を積分するときは不要でした)

[C] か [R/S] キーで積分の計算を中止できます。中止するとメッセージ INTERRUPTED がディスプレイ第2行に表示されますが、再開することはできません。計算が正常に完了するまでは、積分に関するいかなる情報も獲得できません。

積分計算の実行中に [XEQ] キーを押すと [∫] [FN=] の指定もキャンセルされます。この場合は [∫] [FN=] の指定からやり直して下さい。

∫FN のためのプログラムを書くには:

プログラムでは式を利用できます。入力モードは ALG も RPN でも、両者が混在していても問題ありません。

1. プログラムはラベルから開始します。このラベル名は被積分関数の名前になります。(FN= ラベル名)

2. 積分変数も含め、それぞれの変数に対して INPUT 行を定義します。多変数の関数においても、全ての変数に INPUT を定義しておけば、どの変数でも積分変数に指定できます。積分変数に INPUT が定義されていても自動的に無視されますから、積分変数も含めて全ての変数について、あらかじめ INPUT を定義しておくと便利です。

   INPUT が定義されていない変数があれば、その変数に保存された数値か、式プロンプトで入力された数値が使われます。
3. 関数を評価するための指示をプログラムに定義しておきます。
   - RPN や ALG の複数行で構成されるプログラム関数では、積分変数を計算する必要があります。
   - 単一の式のみを含むプログラム関数では、式の種類は制限されません。つまり、等号式、代入式、等号を含まない式の全てが利用できます。ただし、ほとんどの場合は等号式になるでしょう。INPUT を定義せずに式の変数についてのプロンプトを表示させたいときは、フラグ 11 をセットしておいて下さい。
4. プログラムの最終行は RTN として下さい。プログラムの実行は X レジスタに関数の計算結果を出力して終了する必要があるからです。

例: 式を使ったプログラムの積分

第 8 章の例は、次のような三角関数の積分でした。

$$Si(t) = \int_0^t \left(\frac{\sin x}{x}\right)dx$$

この被積分関数をプログラムで定義すると、次のようになります。

```
S001 LBL S          ラベル定義
S002 SIN(X)÷X       関数を等号のない式で定義
                    （式のチェックサムと長さ: 0EE0  8）
S003 RTN            サブルーチン終了
プログラムのチェックサムと長さ: D57E  17
```

このプログラムを、x について 0 から 2 の区間で積分します。

キー (RPN モード)	表示	説明
MODE 2 (2RAD)		角度をラジアンに設定。
↶ FN= S		ラベル S を被積分関数として選択。
0 ENTER 2	2_	下限・上限の順に区間を入力。
↶ / X	INTEGRATING ∫= 1.6054	ラベル S を区間 [0, 2] で積分し、結果を表示。
MODE 1 (1DEG)	1.6054	角度を Degree に戻す。

プログラムの中での積分

プログラムの中で積分を実行することも可能です。積分の前に必ず、区間を指定するか、そのプロンプトを定義しておいて下さい。また、精度と実行速度が、プログラムの実行時におけるディスプレイの表示形式に依存するという点にもご注意下さい。プログラムで積分を定義するには次の 2 つの方法があります。

FN= ラベル名

∫FN d 変数名

プログラムにおける ∫FN は、その計算結果 (∫= 数値) をディスプレイに表示しません。たとえば積分の後、更に計算を続ける場合などは結果の出力は適さないからです。結果を出力するときは PSE (↷ PSE) や STOP (R/S) 行を ∫FN の後に追加しておいて下さい。X レジスタに書き出された計算結果が表示されます。

フラグ 10(積分や SOLVE のときに式を表示させるフラグ)がセットされていて、PSE が式の直後にあると、プログラム行の実行時にその式が 1 秒間表示されます。その間は計算の実行が続けられますが、スクロールやキーボード入力はできません。

例: プログラムでの ∫FN

第 16 章「正規分布」では、正規分布の密度関数(normal density function)を積分しています。

$$\frac{1}{S\sqrt{2\pi}}\int_{M}^{D} e^{-\left(\frac{D-M}{S}\right)^2/2}\,dD.$$

関数 $e^{((D-M)\div S)^2\div 2}$ はラベル F のルーチンで計算しています。その他のルーチンでは既知の値に対するプロンプト、*Q(D)* の計算、正規分布曲線の上側確率の計算が行われます。積分はラベル Q のルーチンで設定され、実行されます。

```
Q001 LBL Q
Q002 RCL M      積分の下限値を呼び出す
Q003 RCL X      積分の上限値を呼び出す (X = D)
Q004 FN= F      関数を指定
Q005 ∫FN d D    ダミー変数 D を使って密度関数 F を積分
```

SOLVE と積分の制約

「SOLVE 変数名」と「∫FN d 変数名」では、SOLVE や ∫FN を含む他のルーチンをコールできません。つまり、繰り返し計算はできません。たとえば、多重積分を実行しようとすれば ∫(∫FN) というエラーになります。

また、SOLVE と ∫FN では、「FN=ラベル名」を含むルーチンをコールできません。実行すると SOLVE ACTIVE または ∫FN ACTIVE エラーになります。

SOLVE は ∫FN を含むルーチンをコールできません。SOLVE(∫FN) エラーになります。同様に ∫FN も SOLVE を含むルーチンをコールできません。∫(SOLVE) エラーになります。

プログラムで「SOLVE 変数名」や「∫FN d 変数名」を定義すると、サブルーチンの入れ子の深さの制限値 (20) がひとつ消費されます。(第 14 章の「サブルーチンの入れ子」をご参照下さい)

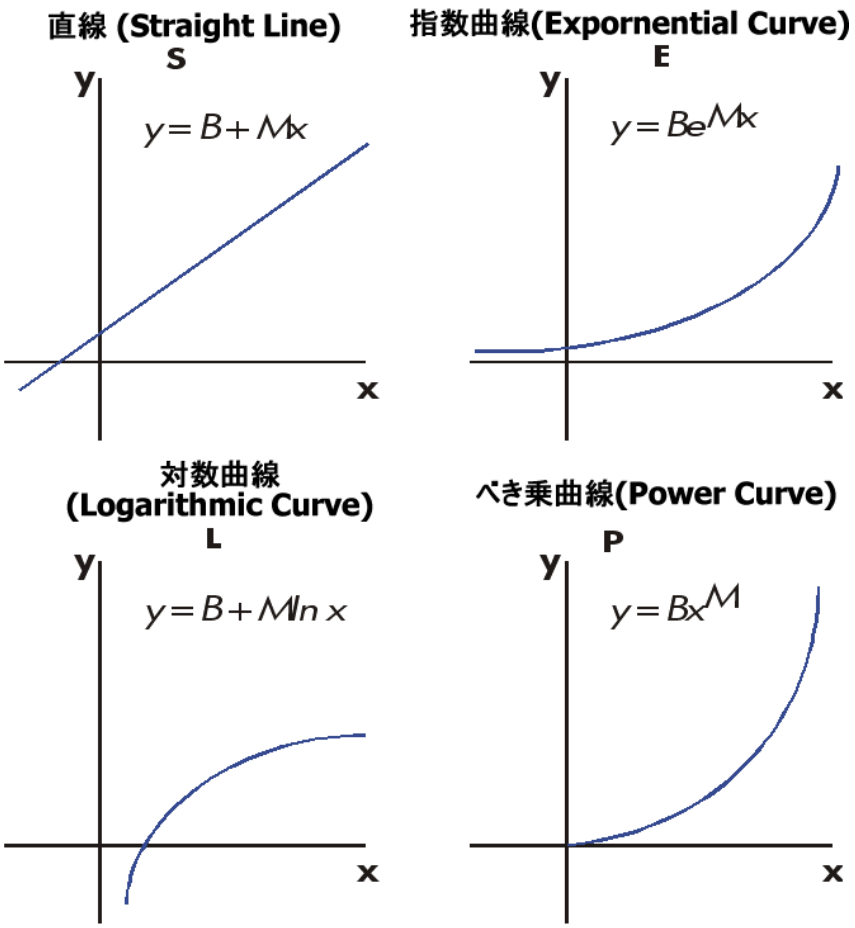
16

統計のプログラム

カーブフィッティング

この章のプログラムでは、与えられたデータを使って 4 種のモデル式にカーブフィッティングができます。4 種のモデル式として直線 (straight line) 、対数曲線 (logarithmic curve)、指数曲線 (exponential curve)、べき乗曲線 (power curve) を用意します。2 組以上の (x, y) を与えると相関係数 (correlation coefficient)*r* 、2 つの回帰係数 (regression coefficients) *m* と *b* が計算されます。予測値 $\hat{x}$ と $\hat{y}$ を計算するルーチンも作成します。(予測値の定義は第 12 章「線形回帰」をご参照下さい。)

カーブフィッティングで得られる曲線の例を次の図に示します。HP 35s に搭載された回帰関数を使って回帰係数が計算されます。



対数曲線にフィットさせるときは x が正とします。指数曲線のときは y が正とします。べき乗曲線のときは x と y の両方が正とします。これらの条件を満たさないときはエラー LOG(NEG)となります。

データの数値が巨大であるのに、データ間の差が小さいときなどは、計算精度が悪くなります。第 12 章「データの精度に関する制限」もご参照下さい。

プログラム:

プログラム行 (RPN モード)	説明
S001 LBL S	このルーチンでは直線モデルの設定を使います
S002 CF 0	In *X* のためのフラグ 0 をクリア
S003 CF 1	In *Y* のためのフラグ 1 をクリア
S004 GTO Z001	共通の入力ポイントである Z に移動

チェックサムと長さ: 8E85 12

L001 LBL L	このルーチンでは対数曲線モデルの設定を使います
L002 SF 0	In *X* のためのフラグ 0 をセット
L003 CF 1	In *Y* のためのフラグ 1 をクリア
L004 GTO Z001	共通の入力ポイントである Z に移動

チェックサムと長さ: AD1B 12

E001 LBL E	このルーチンでは指数曲線モデルの設定を使います
E002 CF 0	In *X* のためのフラグ 0 をクリア
E003 SF 1	In *Y* のためのフラグ 1 をセット
E004 GTO Z001	共通の入力ポイントである Z に移動

チェックサムと長さ: D6F1 12

P001 LBL P	このルーチンではべき乗曲線モデルの設定を使います
P002 SF 0	In *X* のためのフラグ 0 をセット
P003 SF 1	In *Y* のためのフラグ 1 をセット

チェックサムと長さ: 3800 9

Z001 LBL Z	全てのモデルに共通の入力ポイント
Z002 CLΣ	統計レジスタをクリア。[↗] [CLEAR] [4](4Σ) と入力します。
Z003 0	最初の入力のため、ループカウンタをゼロに設定。

チェックサムと長さ: 8611 10

W001 LBL W	入力ループの開始を定義
W002 1	入力プロンプトのため、ループカウンタを 1 に。
W003 +	
W004 STO X	*X* のプロンプトに表示させるためループカウンタを *X* に保存。

プログラム行 (RPN モード)	説明
W005 INPUT X	*X* に保存されたカウンタと、入力プロンプトを表示。
W006 FS? 0	もしフラグ 0 がセットされていたら . . .
W007 LN	. . . *X* の入力値の自然対数を計算
W008 STO B	収集ルーチンで利用するため、入力されたデータを保存
W009 INPUT Y	*Y* のプロンプト
W010 FS? 1	もしフラグ 0 がセットされていたら . . .
W011 LN	. . . *Y* の入力値の自然対数を計算
W012 STO R	
W013 RCL B	
W014 Σ+	*B* と *R* を統計レジスタの x, y のデータセットとして保存
W015 GTO W001	次の *X, Y* データセットの入力へ

チェックサムと長さ: 9560 46

U001 LBL U	やり直し (Undo) のためのルーチンを定義
U002 RCL R	最新のデータセットを呼び出し
U003 RCL B	
U004 Σ−	このセットを統計レジスタから削除
U005 GTO W001	次の *X, Y* データセットの入力へ

チェックサムと長さ: A79F 15

R001 LBL R	データ出力ルーチンの開始
R002 r	相関係数 *r* を計算
R003 STO R	*r* を変数 *R* に保存
R004 VIEW R	R を表示
R005 b	回帰係数 *b* を計算
R006 FS? 1	フラグ 1 がセットのときは *b* の逆対数 (底は e) を計算
R007 e^x	
R008 STO B	*b* を変数 *B* に保存
R009 VIEW B	B を表示
R010 m	回帰係数 *m* を計算
R011 STO M	*m* を変数 *M* に保存
R012 VIEW M	*M* を表示

プログラム行 (RPN モード)	説明

チェックサムと長さ: 850C 36

Y001 LBL Y	予測値 (estimation) の計算ループの開始
Y002 INPUT X	*X* の入力プロンプト。変更されれば保存。
Y003 FS?0	フラグ 0 がセットされていれば ...
Y004 GTO K001	K001 に移動
Y005 GTO M001	M001 に移動
Y006 STO Y	$\hat{y}$ を *Y* に保存
Y007 INPUT Y	*Y* の入力プロンプト。変更されれば保存。
Y008 FS?0	フラグ 0 がセットされていれば ...
Y009 GTO O001	O001 に移動
Y010 GTO N001	N001 に移動
Y011 STO X	次のループのため $\hat{x}$ を *X* に保存
Y012 GTO Y001	次の予測値の計算へ

チェックサムと長さ: C3B7 36

A001 LBL A	直線モデルのときの $\hat{y}$ を計算するサブルーチン
A002 RCL M	
A003 RCL× X	
A004 RCL+ B	$\hat{y} = MX + B$ を計算
A005 RTN	コールしたルーチンに戻る

チェックサムと長さ: 9688 15

G001 LBL G	直線モデルのときの $\hat{x}$ を計算するサブルーチン
G002 RCL Y	
G003 RCL− B	
G004 RCL÷ M	$\hat{x} = (Y - B) \div M$ を計算
G005 RTN	コールしたルーチンに戻る

チェックサムと長さ: 9C0F 15

B001 LBL B	対数曲線モデルのときの $\hat{y}$ を計算するサブルーチン
B002 RCL X	
B003 LN	

プログラム行 (RPN モード)	説明
B004 RCL× M	
B005 RCL+ B	$\hat{y} = M \ln X + B$ を計算
B006 RTN	コールしたルーチンに戻る

チェックサムと長さ: 889C 18

H001 LBL H	対数曲線モデルのときの $\hat{x}$ を計算するサブルーチン
H002 RCL Y	
H003 RCL− B	
H004 RCL÷ M	
H005 e^X	$\hat{x} = e^{(Y-B) \div M}$ を計算
H006 RTN	コールしたルーチンに戻る

チェックサムと長さ: 0DBE 18

C001 LBL C	指数曲線モデルのときの $\hat{y}$ を計算するサブルーチン
C002 RCL M	
C003 RCL× X	
C004 e^X	
C005 RCL× B	$\hat{y} = Be^{MX}$ を計算
C006 GTO M005	M005 に移動

チェックサムと長さ: 9327 18

I001 LBL I	対数曲線モデルのときの $\hat{x}$ を計算するサブルーチン
I002 RCL Y	
I003 RCL÷ B	
I004 LN	
I005 RCL÷ M	$\hat{x} = (\ln (Y \div B)) \div M$ を計算
I006 GTO N005	N005 に移動

チェックサムと長さ: 7219 18

D001 LBL D	べき乗曲線モデルのときの $\hat{y}$ を計算するサブルーチン
D002 RCL X	
D003 RCL M	
D004 y^X	

プログラム行 (RPN モード)	説明
D005 RCL× B	$Y = B(X^M)$ を計算
D006 GTO K005	K005 に移動

チェックサムと長さ: 11B3 18

J001 LBL J	べき乗曲線モデルのときの $\hat{x}$ を計算するサブルーチン
J002 RCL Y	
J003 RCL÷ B	
J004 RCL M	
J005 1/x	
J006 y^x	$\hat{x} = (Y/B)^{1/M}$ を計算
J007GTO 0005	O005 に移動

チェックサムと長さ: 8524 21

K001 LBL K	D001 と B001 のどちらを実行するかどうかを決定
K002 FS?1	フラグ 1 がセットされていれば ...
K003 XEQ D001	D001 を実行
K004 XEQ B001	B001 を実行
K005 GTO Y006	Y006 に移動

チェックサムと長さ: 4BFA 15

M001 LBL M	C001 と A001 のどちらを実行するかどうかを決定
M002 FS?1	フラグ 1 がセットされていれば ...
M003 XEQ C001	C001 を実行
M004 XEQ A001	A001 を実行
M005 GTO Y006	Y006 に移動

チェックサムと長さ: 1C4D 15

0001 LBL 0	J001 と H001 のどちらを実行するかどうかを決定
0002 FS?1	フラグ 1 がセットされていれば ...
0003 XEQ J001	J001 を実行
0004 XEQ H001	H001 を実行
0005 GTO Y011	Y011 に移動

チェックサムと長さ: 0AA5 15

プログラム行 (RPN モード)	説明
N001 LBL N	I001 と G001 のどちらを実行するかどうかを決定
N002 FS?1	フラグ 1 がセットされていれば ...
N003 XEQ I001	I001 を実行
N004 XEQ G001	G001 を実行
N005 GTO Y011	Y011 に移動

チェックサムと長さ: 666D 15

このプログラムのフラグについて:

フラグ 0 がセットされていれば、*X* の入力を自然対数と見なします。フラグ 1 は *Y* の入力を自然対数と見なします。

フラグ 1 がルーチン N でセットされているとき、I001 が実行されます。フラグ 1 がクリアのときは G001 が実行されます。

プログラムの使い方:

1. プログラムルーチンを入力したら、[C] キーで完了
2. [XEQ] を押してから次の操作でフィッティングさせる曲線の種類を選択
   - [S] [ENTER] : 直線 (Straight line)
   - [L] [ENTER] : 対数曲線 (Logarithmic curve)
   - [E] [ENTER] : 指数曲線 (Exponential curve)
   - [P] [ENTER] : べき乗曲線 (Power curve)
3. x の値を入力し [R/S]
4. y の値を入力し [R/S]
5. それぞれのデータについてステップ 3 と 4 を繰り返します。
   ステップ 3 の [R/S] 入力後にミスを発見したときは、プロンプト「Y? 数値」の表示中に再度 [R/S] キーを押し(「X? 数値」プロンプトが表示されます)、やり直し(Undo)のルーチンを実行するため [XEQ] [U] [ENTER] と操作します。すると、最後の入力データが削除されます。
   ステップ 4 の後にミスを発見したら、[XEQ] [U] [ENTER] と操作します。
   どちらの場合も、データの削除後はステップ 3 に戻ります。
6. 全てのデータを入力したら、[XEQ] [R] [ENTER] で相関係数 *R* を確認します。

7. 回帰係数 *B* を確認するため R/S を押します。
8. 回帰係数 *M* を確認するため R/S を押します。
9. 予測値 $\hat{x}$ と $\hat{y}$ を計算するルーチンを R/S で実行します。
10. ある *x* に対する予測値 $\hat{y}$ を計算させるには、プロンプト「X? 数値」で *x* を入力し R/S を押します。プロンプト Y? に続けて $\hat{y}$ が表示されます。
11. ある *y* に対する予測値 $\hat{x}$ を計算させるには、プロンプト「Y? 数値」が出るまで R/S を押してから、そのプロンプトに従って *y* を入力し R/S を押します。プロンプト X? に続けて $\hat{x}$ が表示されます。
12. 予測値を更に計算するにはステップ 11 を繰り返します。
13. 新しいデータセットを入力するときはステップ 2 に戻ります。

変数一覧

B	回帰係数（直線の *y* 切片）。一時保存用の変数としても使われています。
M	回帰係数（直線の傾き）
R	相関係数。一時保存用の変数としても使われています。
X	データ入力のときは、データセットのうち *x* の値。 予測値 $\hat{y}$ を計算するときは、計算の基になる *x* 。 計算の基になる *y* が与えられたときは、予測値 $\hat{x}$ 。
Y	データ入力のときは、データセットのうち *y* の値。 予測値 $\hat{x}$ を計算するときは、計算の基になる *y* 。 計算の基になる x が与えられたときは、予測値 $\hat{y}$ 。
統計レジスタ	統計データの収集と計算のため

例 1

次のデータを直線にフィッティングしてみます。3 個目のデータで入力ミスがあったとして、これを Undo ルーチンで修正してみましょう。データの入力が終わったら x = 37 に対する y の予測値と、y = 101 に対する x の予測値を計算します。

X	40.5	38.6	37.9	36.2	35.1	34.6
Y	104.5	102	100	97.5	95.5	94

キー (ALG モード)	表示	説明
XEQ S ENTER	X? 1.0000	直線モデルのルーチンを選択します
4 0 . 5 R/S	Y? *value*	最初のデータセットの x を入力
1 0 4 . 5 R/S	X? 2.0000	最初のデータセットの y を入力
3 8 . 6 R/S	Y? 104.5000	2 番目のデータセットの x を入力
1 0 2 R/S	X? 3.0000	2 番目のデータセットの y を入力

3 番目のデータセットの x に対し、37.9 ではなく 379 を入力してしまったら、次の操作で修正します。

キー (ALG モード)	表示	説明
3 7 9 R/S	Y? 102.0000	x に対して間違った数値を入力.
R/S	X? 4.0000	プロンプト X? を再表示させます
XEQ U ENTER	X? 3.0000	最後のデータセットを削除。 次のデータセットの入力に移ります。
3 7 . 9 R/S	Y? 102.0000	正しい 3 番目の x を入力
1 0 0 R/S	X? 4.0000	3 番目の y を入力
3 6 . 2 R/S	Y? 100.0000	4 番目の x を入力
9 7 . 5 R/S	X?	4 番目の y を入力

	5.0000	
3 5 · 1 R/S	Y?	5 番目の x を入力
	97.5000	
9 5 · 5 R/S	X?	5 番目の y を入力
	6.0000	
3 4 · 6 R/S	Y?	6 番目の x を入力
	95.5000	
9 4 R/S	X?	6 番目の y を入力
	7.0000	
XEQ R ENTER	R=	相関係数を計算
	0.9955	
R/S	B=	回帰係数 B を計算
	33.5271	
R/S	M=	回帰係数 M を計算
	1.7601	
R/S	X?	仮想値 x に対するプロンプト
	7.0000	
3 7 R/S	Y?	X に 37 を入力し $\hat{y}$ を計算
	98.6526	
1 0 1 R/S	X?	Y に 101 を保存し $\hat{x}$ を計算
	38.3336	

例 2

例 1 と同じデータを使って、対数曲線、指数曲線、べき乗曲線にカーブフィットを実行してみましょう。次の表では、それぞれの曲線タイプを実行するときのラベルと、計算結果(相関係数、回帰係数、x と y の予測値)を示しています。

なお、曲線タイプを変更したらデータを再入力しなければなりません。

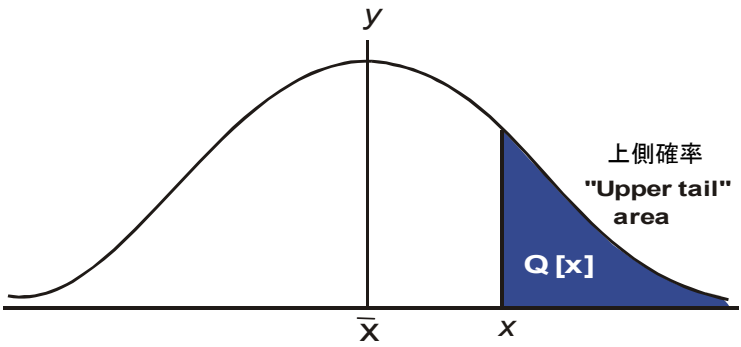
	対数	指数	べき乗
To start:	XEQ L ENTER	XEQ E ENTER	XEQ P ENTER
R	0.9965	0.9945	0.9959
B	–139.0088	51.1312	8.9730
M	65.8446	0.0177	0.6640

Y ($\hat{y}$ when X=37)	98.7508	98.5870	98.6845
X ($\hat{x}$ when Y=101)	38.2857	38.3628	38.3151

正規分布

正規分布(Normal distribution)は、データの平均についてのランダムな振る舞いをモデル化するときによく利用されます。正規分布は、釣り鐘状の次の図のように標本が平均 M(mean)に対して均等に分布していると見なし、その散らばり具合を標準偏差 S(standard deviation)とします。

このプログラムでは、標本データからランダムに選択した数値が、与えられた x よりも大きいものになる確率を求めます。これは上側確率(upper tail area) $Q(x)$ と呼ばれています。このプログラムではその逆、つまり、与えられた確率 $Q(x)$ に対する x も計算できます。



$$Q(x) = 0.5 - \frac{1}{\sigma\sqrt{2\pi}} \int_{\bar{x}}^{x} e^{-((x-\bar{x})\div\sigma)^2 \div 2} dx$$

正規分布曲線の式を積分するため、HP 35s に内蔵の積分機能を使っています。逆計算では、与えられた $Q(x)$ から反復計算で x を求めるため、ニュートン法を使います。

プログラム:

プログラム行 (RPN モード)	説明
S001 LBL S	このルーチンでは正規分布プログラムの初期化を行います
S002 0	平均値のデフォルト値を保存します
S003 STO M	
S004 INPUT M	平均値 *M* の入力プロンプト
S005 1	標準偏差のデフォルト値を保存
S006 STO S	
S007 INPUT S	標準偏差 *S* の入力プロンプト
S008 RTN	数値の表示を終了

チェックサムと長さ: 70BF 26

D001 LBL D	このルーチンでは与えられた*X*に対する*Q(X)* を計算します
D002 INPUT X	*X* のプロンプト
D003 XEQ Q001	上側確率の計算
D004 STO Q	計算結果を変数 Q に保存。次行の VIEW で表示します。
D005 VIEW Q	*Q(X)* を表示
D006 GTO D001	次の *Q(X)* を計算するため最初に戻る

チェックサムと長さ: 042A 18

I001 LBL I	このルーチンでは与えられた*Q(X)* に対する*X*を計算します
I002 INPUT Q	*Q(X)* のプロンプト
I003 RCL M	平均値を呼び出す
I004 STO X	平均値を X の推定値 X_{guess} として保存

チェックサムと長さ: A970 12

T001 LBL T	このルーチンは繰り返し計算のループです
T002 XEQ Q001	$Q(X_{guess}) - Q(X)$ の計算
T003 RCL- Q	
T004 RCL X	
T005 STO D	
T006 R↓	
T007 XEQ F001	X_{guess} の導関数を求めます

プログラム行 (RPN モード)	説明
T008 RCL÷ T	
T009 ÷	X_{guess} の補正値を求めます
T010 STO+ X	X_{guess} の補正値を加算し、新しい X_{guess} とします
T011 ABS	
T012 0.0001	
T013 x<y?	前回の計算結果との差を確認
T014 GTO T001	差が 0.0001 より大きければループの最初に戻ります。それ以外のときは処理を続けます。
T015 RCL X	
T016 VIEW X	計算で得られた X を表示
T017 GTO I001	次の X を求める処理に移動

チェックサムと長さ: EDF4 57

Q001 LBL Q	このサブルーチンでは上側確率 $Q(x)$ を求めます
Q002 RCL M	積分の下限値を呼び出す
Q003 RCL X	積分の上限値を呼び出す
Q004 FN= F	被積分関数にラベル F を指定
Q005 ∫FN d D	ダミー変数 D を使ってラベル F の関数を積分
Q006 2	
Q007 π	
Q008 ×	
Q009 √ ×	
Q010 RCL× S	$S \times \sqrt{2\pi}$ を計算
Q011 STO T	逆計算のルーチンのため、ここまでの結果を一時保存
Q012 ÷	
Q013 +/-	
Q014 0.5	
Q015 +	積分の下限に平均値を用いたので、0.5 を加算（前述の式の通り）
Q0016 RTN	コールしたルーチンに戻る

チェックサムと長さ: 8387 52

F001 LBL F	このサブルーチンでは正規分布関数 $e^{-((X-M)\div S)^2 \div 2}$ を定義

プログラム行 (RPN モード)	説明
	します。
F002 RCL D	
F003 RCL- M	
F004 RCL÷ S	
F005 x^2	
F006 2	
F007 ÷	
F008 +/−	
F009 e^x	
F010 RTN	コールしたルーチンに戻る

チェックサムと長さ: B3EB 31

このプログラムのフラグについて:

フラグは使われていません。

注意点:

このプログラムにおける計算精度は、数値の表示設定に依存します。平均値 ±3 標準偏差の範囲のデータであれば、ほとんどの事例では 4 桁の有効桁で十分でしょう。(この範囲には全データの 99.73%が含まれています)

平均値 ±5 標準偏差の範囲が最大の精度になります。有効桁が大きくなると計算時間が大幅に増大します。

ルーチン Q では 0.5 の代わりに「2 [1/x]」でも入力できます。

逆計算が不要であればそのルーチン(I と T)の入力も不要です。

プログラムの使い方:

1. プログラムの入力が終わったら [C] で完了します。
2. [XEQ] [S] [ENTER] で実行します。

3. *M* のプロンプトが出たら、母平均を入力して R/S。(母平均がゼロであればそのまま R/S)
4. *S* のプロンプトが出たら母標準偏差を入力して R/S。(母標準偏差が 1 であればそのまま R/S)
5. *Q(X)* を与えて *X* を計算するときはステップ 9 に移動します。
6. *X* を与えて *Q(X)* を計算するときは XEQ D ENTER と操作します。
7. プロンプトが出たら *X* の数値を入力して R/S を押します。結果の *Q(X)* が表示されます。
8. 同じ母平均と母標準偏差を使って、別の *X* について再計算するときは R/S を押してステップ 7 に戻ります。
9. *Q(X)* を与えて *X* を計算するときは XEQ I ENTER と操作します。
10. プロンプトに *Q(X)* を入力し、R/S を押します。計算結果 *X* が表示されます。
11. 同じ母平均と母標準偏差を使って、別の *Q(X)* について再計算するときは R/S を押してステップ 10 に戻ります。

変数一覧:

D	積分のためのダミー変数
M	母平均。デフォルト値は 0。
Q	上側確率。
S	母標準偏差。デフォルト値は 1。
T	逆計算を行うときのルーチンに $S \times \sqrt{2\pi}$ の値を受け渡すための一時的な変数。
X	上側確率の基準(左端の位置)を示す値。

例 1:

あなたはある友人に、今度の blind date*のお相手の「知性」が "3σ" だと言われたとします。トップから数えて何番目かを計算するには、平均値+3 標準偏差を超える知性を持っている人数を求めれば良いわけです。

あなたが住む都市で blind date に参加する可能性のある異性が 10,000 人いるとして、どれだけの人数が "3σ" の範囲外になるでしょうか？ 平均値と標準偏差はそれぞれ、デフォルト値を用いてください。

---

* 日本でいう「お見合い」ですね。:-)

キー (RPN モード)	表示	説明
XEQ S ENTER	M? 0.0000	ルーチンの初期化
R/S	S? 1.0000	平均値 M はデフォルトの 0 とします。
R/S	1.0000	標準偏差 *S* はデフォルトの 1 とします。
XEQ D ENTER	X? *value*	ラベル *D* を実行。 X のプロンプトが出ます。
3 R/S	Q= 0.0013	*X* に 3 を入力し、*Q(X)* を求めます。 平均値+3 標準偏差を超える人数の割合を求めます。
1 0 0 0 0 ×	13.4984	想定した人口を乗算し、人数を求めます。10,000 人の中で 13 から 14 位だとわかります。

この友人はときに物事を大げさに伝える傾向があるので、それを加味して "2σ" だった場合の計算もしてみましょう。プログラムの再実行には再度 R/S を押します。

キー (RPN モード)	表示	説明
R/S	X? 3.0000	プログラムの再開
2 R/S	Q= 0.0228	X の値として 2 を入力し、Q(X)を計算。
1 0 0 0 0 ×	227.5012	人口で乗算。 227 から 228 位です。

例 2:

ある試験の平均得点が 55 点だとします。標準偏差は 15.3 でした。正規分布曲線によるモデル化が適用できるとして、ランダムに選択した受験者が 90 点以上である確率はどうなるでしょうか？

受験者の中でトップ 10% に入るとすれば何点とれば良いでしょうか？

20%の受験者を落第にするときは、及第点は何点になるでしょうか？

キー (RPN モード)	表示	説明
XEQ S ENTER	M? 0.0000	初期化ルーチンを実行
5 5 R/S	S? 1.0000	平均に 55 を入力
1 5 · 3 R/S	15.3000	標準偏差に 15.3 を入力
XEQ D ENTER	X? *value*	ルーチン D を実行
9 0 R/S	Q= 0.0111	*X* に 90 を入力し、*Q(X)* を求めます

この計算から、約 1% の受験者のみが 90 点以上であると推察できます。

キー (RPN モード)	表示	説明
XEQ I ENTER	Q? 0.0111	逆計算のルーチンを実行
0 · 1 R/S	X= 74.6077	*Q(X)* に 0.1 (10 %) を保存し、*X* を求めます
R/S	Q? 0.1000	逆計算を再実行
0 · 8 R/S	X= 42.1232	*Q(X)* に 0.8 (100 % − 20%) を保存し、*X* を求めます

頻度を考慮した標準偏差

頻度を考慮した標準偏差 S_{xy} は、データ x_1, x_2, ... , x_n のそれぞれの頻度 (frequency) が f_1, f_2, ... , f_n であるときの標準偏差です。

$$S_{xg} = \sqrt{\frac{\sum x_i^2 f_i - \frac{(\sum x_i f_i)^2}{\sum f_i}}{(\sum f_i) - 1}}$$

このプログラムでは、データとそれぞれの頻度の入力、データ修正、標準偏差と加重平均の計算を行います。

プログラム:

プログラム行 (ALG モード)	説明
S001 LBL S	グループ化標準偏差の開始です
S002 CLΣ	統計レジスタをクリア(−27 から −32)
S003 0	
S004 STO N	カウンタ用の変数 *N* をクリア
チェックサムと長さ: E5BC 13	
I001 LBL I	統計データの入力ルーチン
I002 INPUT X	データを *X* に保存
I003 INPUT F	データの頻度を *F* に保存
I004 1	データ数 *N* の増量分
I005 STO B	
I006 RCL F	頻度 f_i を呼び出し
チェックサムと長さ: 3387 19	
F001 LBL F	総計の計算
F002 −27	
F003 STO I	レジスタのインデックス −27 を保存
F004 RCL F	
F005 STO+(I)	レジスタ −27 の $\sum f_i$ を更新
F006 RCL× X	$x_i f_i$
F007 STO Z	
F008 −28	
F009 STO I	レジスタのインデックス −28 を保存
F010 RCL Z	

プログラム行 (ALG モード)	説明
F011 STO+(I)	レジスタ −28 の $\sum x_i f_i$ を更新
F012 RCL× X	$x_i^2 f_i$
F013 STO Z	レジスタのインデックス −30 を保存
F014 −30	
F015 STO I	
F016 RCL Z	
F017 STO+(I)	レジスタ −30 の $\sum x_i^2 f_i$ を更新
F018 RCL B	
F019 STO+ N	*N* の増減
F020 RCL N	
F021 RCL F	
F022 ABS	
F023 STO F	
F024 VIEW N	現在のデータ数を表示
F025 GTO I001	次のデータ入力のため、ラベル *I* に移動

チェックサムと長さ: F6CB 84

G001 LBL G	統計特性を計算
G002 s×	標準偏差を計算
G003 STO S	
G004 VIEW S	標準偏差を表示
G005 $\bar{x}$	加重平均を計算
G006 STO M	
G007 VIEW M	加重平均を表示
G008 GTO I001	データの入力ルーチンへ移動

チェックサムと長さ: DAF2 24

U001 LBL U	Undo ルーチン
U002 −1	*N* の減少値
U003 STO B	
U004 RCL F	最後のデータの頻度を呼び出す
U005 +/−	f_i の符号変換
U006 STO F	

プログラム行 (ALG モード)	説明
U007 GTO F001	カウンタと総和を調整

チェックサムと長さ: 03F4 23

このプログラムのフラグについて:

フラグは使われていません。

プログラムの使い方:

1. プログラムの入力が終わったら [C] で完了します。
2. [XEQ] [S] [ENTER] で新しいデータを入力します。
3. データポイント x_j を入力し、[R/S]
4. 頻度 f_j を入力し、[R/S]
5. データ数の表示を確認して[R/S]
6. データの入力が完了するまでステップ 3 から 5 を繰り返す。
   ステップ 4 の[R/S]を入力後に x_j や f_j に入力ミスを発見したら、[XEQ] [U] [ENTER] で Undo ルーチンを実行し、[R/S]を押します。次にステップ 3 に戻って正しいデータを入力します。
7. データ入力が終わったら、[XEQ] [G] [ENTER] で標準偏差の計算と表示を行います。
8. 加重平均の表示のため[R/S]を押します。
9. データの追加には[R/S]を押してステップ 3 に戻ります。
   データ入力を最初からやり直すにはステップ 2 に戻ります。

X	データ
F	データの頻度
N	データ組の数
S	頻度を考慮した標準偏差
M	加重平均
i	間接アドレスを使った統計レジスタの補正に利用されるインデックス値。 Index variable used to indirectly address the correct statistics register.
レジスタ -27	総計 Σf_i
レジスタ -28	総計 $\Sigma x_i f_i$.
レジスタ -30	総計 $\Sigma x_i^2 f_i$.

例:

次のデータを使って頻度を考慮した標準偏差を求めましょう。

Group	**1**	**2**	**3**	**4**	**5**	**6**
x_i	5	8	13	15	22	37
f_i	17	26	37	43	73	115

キー (ALG モード)	表示	説明
XEQ S ENTER	X? *value*	最初の x_i のプロンプト
5 R/S	F? *value*	*X* に 5 を保存。 続いて最初の f_i のプロンプト
1 7 R/S	N= 1.0000	*F* に 17 を保存。 カウンタを表示。
R/S	X? 5.0000	第 2 の x_i のプロンプト
8 R/S	F? 17.0000	第 2 の f_i のプロンプト
2 6 R/S	N= 2.0000	カウンタを表示

R/S	X? 8.0000	第 3 の x_i のプロンプト
1 4 R/S	F? 26.0000	ここで入力ミスがあったとします。 続いて第 3 の f_i のプロンプト
3 7 R/S	N= 3.0000	カウンタを表示

上記の操作だと x_3 に 13 と入力すべきところを 14 としてしまいました。Undo ルーチンを実行してデータを取り消します。

変数一覧:

XEQ U ENTER	N= 2.0000	間違ったデータを削除。 カウンタが戻ります。
R/S	X? 14.0000	第 3 の x_i のプロンプト
1 3 R/S	F? 37.0000	第 3 の f_i のプロンプト
R/S	N= 3.0000	カウンタを表示
R/S	X? 13.0000	第 4 の x_i のプロンプト
1 5 R/S	F? 37.0000	第 4 の f_i のプロンプト
4 3 R/S	N= 4.0000	カウンタを表示
R/S	X? 15.0000	第 5 の x_i のプロンプト
2 2 R/S	F? 43.0000	第 5 の f_i のプロンプト
7 3 R/S	N= 5.0000	カウンタを表示
R/S	X? 22.0000	第 6 の x_i のプロンプト
3 7 R/S	F? 73.0000	第 6 の f_i のプロンプト

1 1 5 R/S	N= 6.0000	カウンタを表示
XEQ G ENTER	S= 11.4118	標準偏差 *sx* を計算し表示
R/S	M= 23.4084	加重平均 $\bar{x}$ の計算と表示
C	23.4084	VIEW をクリア

17

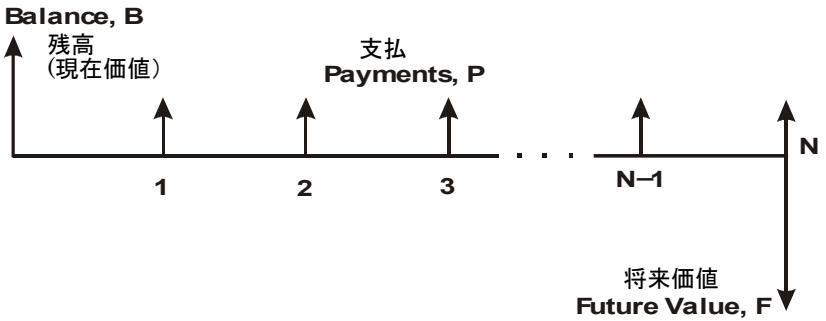
その他のプログラムと式

お金の時間的価値 (TVM, Time Value of Money)

貨幣の時間的価値 (TVM、Time Value of Money) の式には 5 個の変数が登場します。そのうち 4 個がわかれば、残りのひとつを求めることができます。消費ローンや住宅ローン、積立預金など、金融の多様な問題でこの式が幅広く利用されています。

TVM 式は次の通りです。

$$P\left[\frac{1-(1+I/100)^{-N}}{I/100}\right]+F(1+(I/100))^{-N}+B=0$$



貨幣価値の符号(残高 B、支払 P、将来価値 F)は、キャッシュフロー図の向きに一致します。受け取る(増える)ときは正の符号を、支払う(減る)ときは負の符号を用います。全ての金融問題には、貸す側と借りる側という、2 種類の観点があり、それぞれで符号が異なります。

式の入力:

次のように入力します。

```
P×100×(1-(1+I÷100)^-N)÷I+F×(1+I÷100)^-N+B
```

キー (RPN モード)	表示	説明
EQN	EQN LIST TOP または現在選択中の式	式モードの選択
RCL P × 1 0 0	P× 100_	式の入力を開始
× ( ) 1 −	P×100×(1-_)	
( ) 1 +	P×100×(1-(1+_))	
RCL I ÷ 1 0 0	⬅0×(1-(1+I÷ 100)_)➡	
> y^x	⬅(1-(1+I÷100)^_) ➡	
+/- RCL N >	⬅(1+I÷100)^-N)_	
÷ RCL I + RCL F ×	⬅100)^-N)÷I+F×_	
( ) 1 + RCL I	⬅^-N)÷I+F×(1+I_)	
÷ 1 0 0 >	⬅I+F×(1+I÷100)_	
y^x +/- RCL N	⬅×(1+I÷100)^-N_	
+ RCL B	⬅1+I÷100)^-N+B_	
ENTER	P×100×(1-(1+I÷ ➡	式の入力を完了
↩ SHOW (hold)	CK=CEFA LN=41	チェックサムと長さ.

注意点:

TVM 式では *I* がゼロのときゼロ割となり、エラー DIVIDE BY 0 が発生します。*I* が未知数のとき、SOLVE (↩ SOLVE I) の実行前の段階で *I* に現在保存されている値に確信が持てなければ 1 ↩ STO I でゼロ割を避けて下さい。

どれを未知変数にするかによって、プロンプトが出る順番が変化します。

SOLVE の使い方:

1. TVM 式で、利率 *I* について**最初に** SOLVE するときは 1 ↩ STO I と操作します。
2. EQN を押します。必要であれば ^ と v で式リストをスクロールさせ、TVM 式を選択します。

3. 次のうちどれかの操作を行います。
   a. 期間 N の計算は [→] [SOLVE] [N] です。
   b. 利率 I の計算は [→] [SOLVE] [I] です。
      月ごとの支払額が入力されていれば、計算結果の利率も月利 *i* になります。年利 *I* に直すには 12 [×] と操作して下さい。
   c. ローンや預金の現在の残高（現在価値）B の計算は [→] [SOLVE] [B] です。
   d. 定期支払額 P の計算は [→] [SOLVE] [P] です。
   e. 将来の残高（将来価値）F の計算は [→] [SOLVE] [F] です。
4. プロンプトに従って、4 つの既知の変数を入力します。数値の入力後は、それぞれ [R/S] キーで確定します。
5. 最後の [R/S] を押したときに未知変数が計算され、表示されます。
6. 未知変数を変更したり、既知の変数を変更して再計算するにはステップ 2 に戻ります。

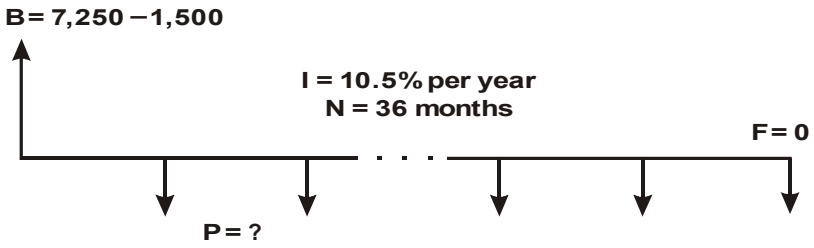
TVM 式では、初期推定値を入力しなくても SOLVE が十分に実行できます。

変数リスト:

N	期間
I	期間における利率(%)。 たとえば、年利が 15%のときは、1 年で 12 回の支払がありますから月利は 15÷12 = 1.25% になります。
B	預金やローンの現在の残高(現在価値)
P	支払額
F	預金やローンの将来の残高(将来価値)

例:

その 1. 自動車をローンで購入するとします。期間は 3 年(36 ヶ月)で、月毎の複利で年利 10.5%です。自動車の価格は$7,250 ドルで、頭金を$1,500 ドルとします。



キー (RPN モード)	表示	説明
DISPLAY 1 (1FIX) 2		米ドルの表示のため表示形式を FIX 2 とします。
EQN (必要であれば ⌄)	P×100×(1−(1+I÷➡	TVM 式の左端が表示されます。
↗ SOLVE P	I?	未知変数を P とします。
	value	*I* のプロンプトが出ます。
1 0 · 5 ENTER	I?	年利を月利に換算
1 2 ÷	0.88	
R/S	N?	*I* に 0.88 を保存。
	value	続いて *N* のプロンプト
3 6 R/S	F?	*N* に 36 を保存。
	value	続いて *F* のプロンプト
0 R/S	B?	*F* に 0 を保存。
	value	続いて *B* のプロンプト
7 2 5 0 ENTER	B?	現在価値 *B* を計算。
1 5 0 0 −	5,750.00	
R/S	SOLVING	*B* に 5750 を保存すると、月々
	P=	の支払額 *P* が計算されます。
	−186.89	

借りる側の観点で計算したので、計算結果は負になります。借りたときにお金(現在価値 *B*)を受け取るので、*B* は正です。

その 2. では、利率が何%のときに月々の支払額が$10ドル減るでしょうか？

キー (RPN モード)	表示	説明
EQN	P×100×(1-(1+I÷➡	TVM 式を表示
↗ SOLVE I	P?	未知変数に *I* を選択。
	-186.89	*P* のプロンプトが表示されます。
↗ RND	P?	セント単位で計算するため、小数
	-186.89	点以下 2 位で丸めます。
1 0 +	P?	新しい支払額を計算
	-176.89	
R/S	N?	*P* に–176.89 を保存。
	36.00	続いて *N* のプロンプト
R/S	F?	*N* に 36 を保存。
	0.00	続いて *F* のプロンプト
R/S	B?	*F* に 0 を保存。
	5,750.00	続いて *B* のプロンプト
R/S	SOLVING	*B* に 5750 を保存すると、月利が
	I=	計算されます。.
	0.56	
1 2 ×	6.75	月利を年利に換算

その 3. この年利(6.75%)のとき、2 年経過後のローン残高はいくらになるでしょうか？つまり、2 年後の将来価値 *F* はいくらでしょうか？

「その 2」の計算で、利率 *I* はゼロではなくなっています。この状態では、ゼロ割のエラー DIVIDE BY 0 にはなりません。

キー (RPN モード)	表示	説明
EQN	P×100×(1-(1+I÷➡	TVM 式を表示
↗ SOLVE F	P?	未知変数に *F* を選択。
	-176.89	*P* のプロンプトが表示されます。
R/S	I?	*P* は変更しません。
	0.56	続いて *I* のプロンプト
R/S	N?	*I* は変更しません。
	36.00	続いて *N* のプロンプト

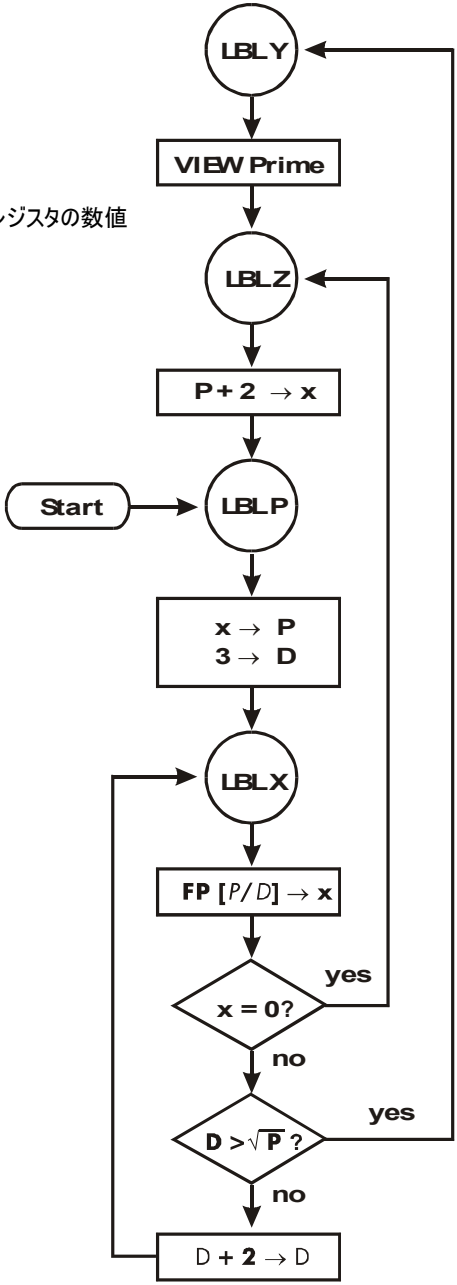
2 4 R/S	B? 5,750.00	*N* に 24 を保存。 続いて *B* のプロンプト
R/S	SOLVING F= -2,047.05	*B* は変更しません。 将来価値 *F* の計算が実行されます。結果は負になります。これは支払残高が残っていることを示しています。
↶ DISPLAY 1 (1FIX) 4		表示形式を FIX 4 に変更

素数ジェネレータ

このプログラムは、4 以上の正の整数の入力を受け付け、それが素数（Prime Number、その数自身と 1 でのみ割り切れる数値）のときは入力された数値を表示します。素数ではないとき、入力された数値に近い素数のうち、大きい方を出力します。

入力値が素数かどうかを見極めるため、全ての可能性のある数について試行します。素数でない場合は 2 を加算して更に試行します（1 を追加しても偶数になってしまうため）。素数が見つかるまで、この工程が繰り返されます。

注: x は X レジスタの数値



プログラム:

プログラム行 (ALG モード)	説明
Y001 LBL Y	このルーチンでは素数 *P* を表示します。
Y002 VIEW P	
チェックサムと長さ: 2CC5 6	
Z001 LBL Z	このルーチンは素数 *P* に 2 を加算します。
Z002 2+ P	
チェックサムと長さ: EFB2 9	
P001 LBL P	このルーチンでは入力値を変数 *P* に保存します。
P002 LASTx▶ P	
P003 FP(P÷2)	
P004 x<>y	
P005 0	
P006 x=y?	偶数かどうかの判定。
P007 1+P▶P	入力値が偶数であれば *P* に 1 を加算。
P008 3▶D	除算の試行のため変数 *D* に 3 を保存。
チェックサムと長さ: EA89 47	
X001 LBL X	このルーチンでは *P* が素数かどうかを判定します。
X002 FP(P÷D)	*P* ÷ *D* の小数部を求めます。
X003 x=0?	*P* ÷ *D* の小数部がゼロかどうかを判定。 (ゼロであれば素数ではありません)
X004 GTO Z001	素数ではないとき、次の可能性を試行。
X005 SQRT(P)	
X006 x<>y	
X007 D	
X008 x>y?	全ての可能性を試行したかどうかを判定。
X009 GTO Y001	全ての可能性が施行されていれば、素数の表示ルーチンに移動。
X010 2+D▶D	素数ではなかった数値に 2 を加算
X011 GTO X001	素数の判定に移動

プログラム行 (ALG モード)	説明

チェックサムと長さ: C6B5 53

このプログラムのフラグについて:

フラグは使われていません。

プログラムの使い方:

1. プログラムの入力が終わったら [C] で完了します。
2. 3 以上の正の整数を入力します。
3. [XEQ] [P] [ENTER] でプログラムを実行すると素数 *P* が表示されます。
4. 次の素数を表示するには [R/S] を押します。

変数一覧:

P	素数、または素数の可能性のある数値。
D	現在の *P* に対する除数。

注意:

入力が 3 以上であるかどうかのチェックは行われていません。

例:

789 の次の素数は何でしょうか？ さらにその次の素数は？

キー (ALG モード)	表示	説明
[7] [8] [9] [XEQ]	P= 797.0000	789 の次の素数を計算
[P] [ENTER]		
[R/S]	P= 809.0000	797 の次の素数を計算

ベクトルの外積 (Cross Product)

ベクトルの外積を求めるプログラムです。外積は次の通りです。

$$\mathbf{v}_1 \times \mathbf{v}_2 = (YW - ZV)\mathbf{i} + (ZU - XW)\mathbf{j} + (XV - YU)\mathbf{k}$$

ただし、

$\mathbf{v}_1 = X\,\mathbf{i} + Y\,\mathbf{j} + Z\,\mathbf{k}$

および

$\mathbf{v}_2 = U\,\mathbf{i} + V\,\mathbf{j} + W\,\mathbf{k}$

とします。

プログラム行 (RPN モード)	説明
R001 LBL R	ベクトルの入力・表示のためのルーチン *R* を開始
R002 INPUT X	*X* の入力の受け付け、または表示
R003 INPUT Y	*Y* の入力の受け付け、または表示
R004 INPUT Z	*Z* の入力の受け付け、または表示
R005 GTO R001	ベクトル入力のため R001 に移動
チェックサムと長さ: D82E 15	
E001 LBL E	ベクトル入力ルーチンの開始
E002 RCL X	*X*、*Y*、*Z* の数値を *U*、*V*、*W* にそれぞれ保存。
E003 STO U	
E004 RCL Y	
E005 STO V	
E006 RCL Z	
E007 STO W	
E008 GTO R001	ベクトル入力のため R001 に移動
チェックサムと長さ: B6AF 24	
C001 LBL C	外積の計算ルーチン *C* の開始。
C002 RCL Y	
C003 RCL× W	

プログラム行 (RPN モード)	説明
C004 RCL Z	
C005 RCL× V	
C006 −	
C007 STO A	*X* 成分である *YW – ZV* を保存
C008 RCL Z	
C009 RCL× U	
C010 RCL X	
C011 RCL× W	
C012 −	
C013 STO B	*Y* 成分である *ZU – WX* を保存
C014 RCL X	
C015 RCL× V	
C016 RCL Y	
C017 RCL× U	
C018 −	
C019 STO Z	*Z* 成分である *XV – YU* を保存
C020 RCL A	
C021 STO X	*X* 成分を保存
C022 RCL B	
C023 STO Y	*Y* 成分を保存
C024 GTO R001	ベクトル入力のため R001 に移動

チェックサムと長さ: 838D 72

例:

次の 2 つのベクトルの外積を求めましょう。

v1=2i+5j+4k

v2=i-2j+3k

キー	表示		説明
XEQ R ENTER	X?	value	ルーチン E を実行してベクトルの入力を開始。
1 R/S	y?	value	v2 の x 成分を入力
2 +/- R/S	z?	value	v2 の y 成分を入力
3 R/S	X?	1	v2 の z 成分を入力
XEQ E ENTER	X?	1	ルーチン E を実行し、v2 を U、V、W に変換。
2 R/S	y?	-2	v1 の x 成分を入力
5 R/S	z?	3	v1 の y 成分を入力
4 R/S	X?	2	v1 の z 成分を入力
XEQ C ENTER	X?	23	C ルーチンを実行します。外積の x 成分が表示されます。
R/S	y?	-2	外積の y 成分
R/S	z?	-9	外積の z 成分

Part 3

付録

A

サポートと電池について

サポート

電卓の使用方法などについてご不明な点があれば A-7 ページに掲載されている技術サポートにお問い合せ下さい。

想定されるご質問を次節に掲載しましたので、カスタマーサポートにお問い合せする前にご参照下さい。

よくあるご質問

Q: 電卓の動作が正しいかどうか確認するには？

A: A-5 ページを参考に、セルフテストを実行してみて下さい。

Q: 小数点がピリオド(.)ではなくコンマ(,)になっているが、ピリオドに直すには？

A: ↰ DISPLAY 5 (5 ⋅) と操作して下さい。(1-20 ページ)

Q: 小数点以下の桁を調整するには？

A: ↰ DISPLAY メニューを使います。 (1-19 ページ).

Q: メモリの一部をクリアするには？

A: ↱ CLEAR でクリア用のメニューを開きます。X レジスタ(X)、ダイレクト変数の全て(VARS)、メモリの全て(ALL)、統計データの全て(Σ)、スタックの全て(STK)、全てのスタックレベルおよび全ての間接変数(CLVAR%)から選択して下さい。

Q: 数値に表示された "E" とは何ですか？ (たとえば 2.51E-13)

A: 10 の指数です。この例だと 2.51×10^{-13} をあらわします。

Q: メッセージ MEMORY FULL が表示されている。何をすれば良いのか？

A: 次の操作の前にメモリの一部をクリアする必要があります。 (付録 B)

Q: sin(π) や tan(π) がゼロにならないのはなぜ？

A: πは無理数なので、この電卓の 12 桁の精度では正確に表現できないからです。

Q: 三角関数で計算結果が間違っている。

A: 角度の設定を確認して下さい。(MODE 1DEG, 2RAD, 3GRD)

Q: ディスプレイに出ている表示記号は何を意味するのか？

A: 電卓の状態をあらわしています。詳細は第 1 章「ディスプレイと表示記号」をご参照下さい。

Q: 表示が分数になっている。小数にするには？

A: ↗ FDISP と操作して下さい。

動作温度と動作湿度

次の範囲を超えた状態では正しく動作しない可能性があります。

- **動作温度**: 0 ～ 45 °C (32 ～113 °F).
- **保存温度**: –20 ～65 °C (–4 ～149 °F).
- **動作湿度・保存湿度**: **最大** 90%（**結露なきこと**）

電池の交換

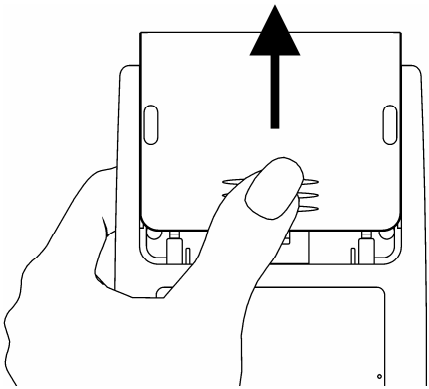
電池は 3V のリチウムコイン電池 CR2032 が 2 個使われています。

電池の表示記号(▭)が出たらすぐに電池を交換して下さい。電池の表示記号が出てディスプレイが暗くなってきたら、データが消失する可能性があります。消失した場合はメッセージ MEMORY CLEAR が表示されます。

電池を外したら**2 分以内**に新しい電池を入れて下さい。さもないと保存された情報が消えます。古い電池を外す前に新しい電池を手元に用意しておくと良いでしょう。

電池を入れ替えるには:

1. CR2032 ボタン電池を２個用意して下さい。電池の電極は指で触らないようにして、電池の縁を持って下さい。
2. 電卓の電源がオフになっていることを確認して下さい。**電池の入れ替えが完了するまで ON (C)を押さないで下さい。電池を外した状態で ON が押されると、連続メモリに保存された内容が消去されます**。
3. 電卓を裏返して、電池カバーをスライドさせて開きます。

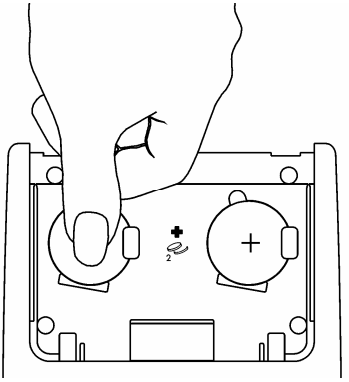


4. メモリの消失を防ぐため、２個の電池を同時に取り外さないで下さい。古い電池と新しい電池をひとつずつ交換して下さい。

警告

電池を分解したり、穴を空けたり、可燃物として廃棄しないで下さい。電池が破裂または爆発する可能性があります。
使用済み電池は、お住まいの自治体の指示に従って処分して下さい。

5. 新しい CR2032 リチウム電池を入れます。電池の「＋」が外側を向くように確認してから入れて下さい。



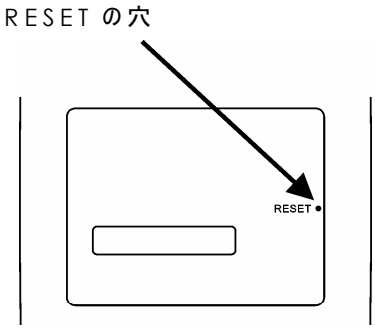
6. ステップ 4 から 5 を繰り返して残りの電池を入れ替えます。再度、それぞれの電池の「＋」が外側を向いていることを確認して下さい。
7. 電池カバーを装着します。
8. [C] で電源を入れて動作を確認して下さい。

動作テスト

電卓が正しく動作するかを確認するには、次のガイドラインをご参照下さい。正しく動作しない場合は A-7 ページに記載のサポートをご利用下さい。

■ **電源が入らないときは下記の 1 から 4 を、キーを押しても反応がない場合は 1 から 3 をお試し下さい。**

1. 電卓をリセットします。[C]を押しながら[GTO]を押して下さい。この操作を何回か繰り返す必要がある場合もあります。
2. メモリを消去します。[C]を押しながら[R/S]と [i] を同時に押して下さい。全てのボタンを離すと、メッセージ MEMORY CLEAR が出てメモリがクリアされます。
3. この章で前述の「電池の交換」を参考にして電池を外します。さらに、電卓側の電池の接点（＋と－の両方）に金属を同時に軽く当てて、ショートさせます。次に新しい電池を入れて下さい。電源を入れると MEMORY CLEAR と表示されます。
4. これまでの操作でまだキーに反応しないときは、電卓の裏側にある RESET と書かれた穴を、細くて尖ったもので押して下さい。通常は、この操作で保存データが消去されることはありません。



上記の操作でも正常に動作しないときは、技術サポートにお問い合せ下さい。

■ **電卓がキーに反応しているにもかかわらず、動作が不良と思えるときは:**

1. 次節を参考にセルフテストを実行して下さい。セルフテストに失敗するようであれば技術サポートにお問い合せ下さい。
2. セルフテストをパスしている場合は、操作ミスの可能性が残されています。このマニュアルを再読のうえ、「よくあるご質問」（A-1 ページ）をご確認ください。
3. それでも解決しなければ A-7 ページの技術サポートにご連絡ください。

セルフテスト

ディスプレイの表示が正常であるのに動作が正常でないと思えるときは、次の操作でセルフテストを実行して下さい。

1. [C]を押しながら[XEQ]を押して下さい。
2. 任意のキーを続けて８回押します。キーを押すごとに異なるパターンがディスプレイに表示されます。押し終わると © 2007 HP DEV CO，L，P，と表示され、続けてメッセージ KBD 01 が出ます。

3. 次の順にキーを押して下さい。

[R/S]→[GTO]→[XEQ]→[MODE]→[∧]→[<]→[>]→[RCL]→[R↓]→[x◂▸y]→[i]→[∨]→[SIN]→[COS]→[TAN]→[√x]→[y^x]→[1/x]→[ENTER]→[+/−]→[E]→[( )]→[←]→[EQN]→[7]→[8]→[9]→[÷]→[↶]→[4]→[5]→[6]→[×]→[↷]→[1]→[2]→[3]→[−]→[C]→[0]→[.]→[Σ+]→[+]

■ 順番通りにキーを押し、それが正しく認識されれば KBD に続けて 2 桁の 16 進数が表示されます。

- キー入力が順番通りではなかったときや、キー入力が認識できなかった場合は、その次のキー入力のときに失敗のメッセージが出ます。（ステップ 4 をご参照下さい）

4. セルフテストの結果は次のように出力されます:

- 35S-OK の場合、セルフテストに合格しています。ステップ 5 に進んで下さい。
- 35S-FAIL に続けて数値が出力された場合は、何らかの問題があります。順番通りにキーを入力しなかったことが原因であれば、電卓をリセット（C を押しながら GTO）してセルフテストを再実行して下さい。順番通りに入力したのにこのメッセージが出たときも、念のため再度セルフテストを実行して下さい。それでも再度同じメッセージが出るようでしたら電卓の不良が考えられます。技術サポートに出力されたメッセージをご連絡下さい。

5. セルフテストを終了するには電卓をリセット（C を押しながら GTO）して下さい。

C を押しながら MODE を押すと、セルフテストを自動実行します。終了するには任意のキーを押して下さい。

日本における保証

HP 35s Scientific Calculator にはご購入後1年間の保証がつきます。

HP 電卓の輸入代理店である株式会社ジュライは商品本体および付属品に対し、ご購入から上記の保証期間内における正常な動作を保証します。この保証期間において製品本体および付属品に不良が認められる場合、新品または新品同等品に交換いたします。

製品に不良が認められる場合、下記にご連絡ください。製品のご購入先は問いませんが、ご購入後 1 年以内であるという証明となるレシートや納品書などが必要になります。

ご連絡先:

株式会社ジュライ　青森支店

〒030-0802　青森県青森市本町 5-10-1

　電話　017-721-3733　　FAX　017-721-3734

技術サポート

アジア太平洋

国	電話番号
日本	+81 3 6666 9925 （国内からは 03-6666-9925）
Australia	1300-551-664 or 03-9841-5211
China	010-68002397
Hong Kong	2805-2563
Indonesia	+65 6100 6682
Malaysia	+65 6100 6682
New Zealand	09-574-2700
Philippines	+65 6100 6682
Singapore	6100 6682
South Korea	2-561-2700
Taiwan	+852 2805-2563
Thailand	+65 6100 6682
Vietnam	+65 6100 6682

ヨーロッパ・中東・アフリカ

国	電話番号
Austria	01 360 277 1203
Belgium	02 620 00 86
Belgium	02 620 00 85
Czech Republic	296 335 612
Denmark	82 33 28 44
Finland	09 8171 0281
France	01 4993 9006
Germany	069 9530 7103
Greece	210 969 6421
Netherlands	020 654 5301
Ireland	01 605 0356
Italy	02 754 19 782
Luxembourg	2730 2146
Norway	23500027
Portugal	021 318 0093
Russia	495 228 3050
South Africa	0800980410
Spain	913753382
Sweden	08 5199 2065
Switzerland	022 827 8780
Switzerland	01 439 5358
Switzerland	022 567 5308
United Kingdom	0207 458 0161

南アメリカ

国	電話番号
Anguila	1-800-711-2884
Antigua	1-800-711-2884
Argentina	0-800- 555-5000
Aruba	800-8000 ♦ 800-711-2884
Bahamas	1-800-711-2884
Barbados	1-800-711-2884

Bermuda	1-800-711-2884
Bolivia	800-100-193
Brazil	0-800-709-7751
British Virgin Islands	1-800-711-2884
Cayman Island	1-800-711-2884
Curacao	001-800-872-2881 + 800-711-2884
Chile	800-360-999
Colombia	01-8000-51-4746-8368 (01-8000-51- HP INVENT)
Costa Rica	0-800-011-0524
Dominica	1-800-711-2884
Dominican Republic	1-800-711-2884
Ecuador	1-999-119 ♦ 800-711-2884 (Andinatel) 1-800-225-528 ♦ 800-711-2884 (Pacifitel)
El Salvador	800-6160
French Antilles	0-800-990-011♦ 800-711-2884
French Guiana	0-800-990-011♦ 800-711-2884
Grenada	1-800-711-2884
Guadelupe	0-800-990-011♦ 800-711-2884
Guatemala	1-800-999-5105
Guyana	159 ♦ 800-711-2884
Haiti	183 ♦ 800-711-2884
Honduras	800-0-123 ♦ 800-711-2884
Jamaica	1-800-711-2884
Martinica	0-800-990-011 ♦ 877-219-8671
Mexico	01-800-474-68368 (800 HP INVENT)
Montserrat	1-800-711-2884
Netherland Antilles	001-800-872-2881 ♦ 800-711-2884
Nicaragua	1-800-0164 ♦ 800-711-2884

Panama	001-800-711-2884
Paraguay	(009) 800-541-0006
Peru	0-800-10111
Puerto Rico	1-877 232 0589
St. Lucia	1-800-478-4602
St Vincent	01-800-711-2884
St. Kitts & Nevis	1-800-711-2884
St. Marteen	1-800-711-2884
Suriname	156 ♦ 800-711-2884
Trinidad & Tobago	1-800-711-2884
Turks & Caicos	01-800-711-2884
US Virgin Islands	1-800-711-2884
Uruguay	0004-054-177
Venezuela	0-800-474-68368 (0-800 HP INVENT)

北アメリカ

国	電話番号
Canada	800-HP-INVENT
USA	800-HP INVENT

最新の情報は http://www.hp.com にログインしてご確認下さい。

ご利用上の注意

この装置は情報処理装置等電波障害自主規制協議会(VCCI)の基準に基づく第二情報技術装置です。この装置は家庭環境で使用することを目的としていますが、この装置がラジオやテレビに近接して使用されると、受信障害を引き起こすことがあります。

取扱説明書に従って正しい取り扱いをしてください。

B

メモリとスタック

この章では次の項目を解説します。

- メモリの割当と要求
- メモリへの影響なくリセットする方法
- メモリをクリアし、全ての設定をリセットする方法
- スタックの上昇に関連する操作

メモリの管理

HP 35s には、データ(変数、式、プログラム)の保存用メモリとして 30KB が用意されています。SOLVE、∫FN、および統計計算でもこのメモリを利用します。(∫FN はとりわけ多くのメモリを消費します。)

保存したデータは、ユーザが明示的にクリアするまでメモリに残ります。メッセージ **MEMORY FULL**が出たときは、その直前の操作のためのメモリが不足していることを示しています。次の方法でメモリをクリアして下さい。

- 数式を削除(第 6 章「式の編集と削除」をご参照下さい)
- プログラムを削除(第 13 章「プログラムの削除」をご参照下さい)
- ユーザメモリを削除([↗] [CLEAR] [3] (3ALL) と操作)

利用可能なメモリ量を調べるには、[↖] [MEM] と操作します。残りのバイト数が表示されます。

式リストにおいて、ある式が消費しているメモリを調べるには次の操作を行います。

1. [EQN] を押して式モードをアクティブにします。(EQN LIST TOP または選択済みの式の左端が表示されます)

2. 必要に応じて式リストを [^] や [v] でスクロールさせ、希望の式を選択します。
3. [↰] [SHOW] で、その式のチェックサム（16 進数）と長さ（バイト数）が CK=382E LN=41 のように表示されます。

あるプログラムの消費メモリを確認するには:

1. [↰] [MEM] [2] (2PGM) で、プログラムリストのラベルを表示します。
2. プログラムリストを [^] や [v] でスクロールさせ、希望のプログラムラベルと長さ（バイト数）を選択します。ディスプレイ表示は、たとえば LBL F LN=57 のようになります。
3. （必要に応じて）[↰] [SHOW] で 16 進数のチェックサムとプログラムの長さ（バイト数）を表示します。たとえば CK=9CC9 LN=57 のように表示されます。

あるプログラムに定義された式の消費メモリを確認するには:

1. 目的の式を含むプログラム行を表示します。
2. [↰] [SHOW] で、チェックサムと長さが CK=AB71 LN=15 のように表示されます。

リセット

電卓がキーに反応しないときや、挙動が通常と異なるときはリセットして下さい。リセットにより実行中の計算、プログラム入力、数値入力、プログラム実行、SOLVE の計算、∫FN の計算、VIEW の表示、INPUT 表示が中止されます。通常、保存されたデータはそのまま保存されます。

リセットするには [C] キーを押しながら [GTO] を押します。リセットできないときは電池を交換して下さい。それでもリセットできないときや、リセットしても動作が改善しないときは、次節の方法でメモリをクリアをお試し下さい。

キーに反応しないときは、先の細長いもので電卓の裏側の RESET と書かれた穴を押して下さい。

電卓を落下させたときと、電源が完全になくなったときは自動的にリセットされます。

メモリのクリア

メモリのクリアは通常、[↗] [CLEAR] [3] (3ALL) と操作します。この操作を CLEAR ALL と呼びます。

CLEAR ALL とは別に、より強力なクリア方法もあります。キーに反応しないときや、前述のリセットや電池交換を行っても動作が不良の場合は、次に示す MEMORY CLEAR 操作を行って下さい。これにより全てのメモリのクリアされ、電卓がリセットされ、さらに全ての表示形式とモード設定がデフォルト設定に戻ります。

1. [C] を押し続けます。
2. [R/S] を押し続けます。
3. [i] を押します。これで 3 個のキーを同時に押すことになります。次に、これらのキーを全て同時に離します。MEMORY CLEAR に成功すると MEMORY CLEAR と表示されます。

カテゴリ	CLEAR ALL	MEMORY CLEAR (デフォルト)
角度モード(Angles)	変更なし	Degrees（度）
基数(Base)モード	変更なし	Decimal（10 進数）
コントラスト	変更なし	Medium
小数点表記	変更なし	"."
桁区切り	変更なし	"1,000"
分母(/c の値)	変更なし	4095
表示形式	変更なし	FIX 4
フラグ	変更なし	クリア
複素数モード	変更なし	xiy
分数表示モード	変更なし	Off
乱数シード値	変更なし	ゼロ
式ポインタ	EQN LIST TOP	EQN LIST TOP
式リスト	クリア	クリア
FN = ラベル	クリア	クリア
プログラムポインタ	PRGM TOP	PRGM TOP
プログラムメモリ	クリア	クリア
スタック上昇	Enabled	Enabled
スタックレジスタ	全てゼロ	全てゼロ
変数	全てゼロ	全てゼロ
間接変数	未定義	未定義
入力方式	変更なし	RPN

電卓を落下させたときや、電源が完全に無くなったときは自動的にメモリがクリアされることがあります。

スタック上昇の設定

4 個のスタックレジスタは常にメモリに存在し、スタックには「スタック上昇の状態」が定義されています。つまり、X レジスタに新しい数値が入力されるとき、スタック上昇の設定によって、スタックが上昇するかどうかが判断されます。（第 2 章「メモリスタックの自動処理」もご参照下さい）

次の2つの操作ではスタック上昇が行われません。その他の操作ではスタックが上昇します。

スタック上昇が行われなくなる操作

[ENTER]、[Σ+]、[Σ-]、[↗][CLEAR][1] (1𝕏)、ならびに [↗][CLEAR][5] (5STK) の 5 つの操作では、その次の操作のスタック上昇が行われなくなります。これらのキーの後の操作では、Xレジスタに数値が上書きされます。Yレジスタ、Zレジスタ、ならびにTレジスタはそのままで、変更されません。

さらに、[C] と [←] を CLx の代わりに用いたときも同様に次の操作でスタック上昇が行われなくなります。

プログラムの INPUT 機能は、スタック上昇を「不可」にし、入力プロンプトを表示してプログラムの実行を一時停止します（よって、プロンプトに入力された全ての数値はXレジスタに上書きされます）。ただし、プログラムが再開するときにスタック上昇は元通り「可」に設定されます。

通常の操作

次の操作はスタック上昇の状態に対して影響を及ぼしません。

DEG, RAD, GRAD	FIX, SCI, ENG, ALL	DEC, HEX, OCT, BIN	CLVARS
PSE	SHOW	RADIX . RADIX ,	CLΣ
[OFF][RCL][+]	[R/S] と STOP	[^] と [v]	[C]* と [←]*
[MEM][1] (1VAR)**	[MEM][2] (2PGM)**	[GTO][.][.]	[GTO][.] label nnn
EQN	FDISP	エラー	[PRGM] とプログラム入力
2 進数への切替	数値入力	xiy r θ a	[UNDO]

＊　CLx のような操作を除きます。

＊＊　カタログ表示後の操作を全て含みます（ただし、{VAR} [ENTER] と {PGM} [XEQ] は除きます）。

LAST X レジスタの状態

RPN モードにおける次の操作により、x の値が LAST X レジスタに保存されます。

+, –, × , ÷	$\sqrt{x}$, x^2,	e^x, 10^x
LN, LOG	y^x, $\sqrt[x]{y}$	1/x, INT÷, Rmdr
SIN, COS, TAN	ASIN, ACOS, ATAN	→ →
SINH, COSH, TANH	ASINH, ACOSH, ATANH	IP, FP, SGN, INTG, RND, ABS
%, %CHG	Σ+, Σ–	RCL+, –, ×, ÷
	HMS→, →HMS	→DEG, →RAD
nCr nPr	!	ARG
CMPLX +, –, × ,÷	CMPLX e^x, LN, y^x, 1/x	CMPLX SIN, COS, TAN
→kg, →lb →l, →gal	→°C, →°F →KM →MILE	→cm, →in

/c は LAST X レジスタに影響を及ぼさないことにご注意下さい。

「[X] [RCL] [+] **変数**」という操作では x が LAST X レジスタに保存されます。これに対し、「[X] [RCL] **変数** [+]」では再呼び出しされた変数が LAST X レジスタに保存されます。

ALG モードでの LAST X レジスタには、式の最後の計算結果が保存されます。これにより、最後の結果を再利用できます。

スタックレジスタへのアクセス

X、Y、Z、T の 4 つのスタックレジスタには変数が保存されており、RPN モードの式や、プログラムの REGX、REGY、REGZ、REGT コマンドで呼び出すことができます。

この機能を使うには、まず [EQN] を押し、次に [R↓] で X、Y、Z、T レジスタのメニューを表示させます。[>] と [<] でレジスタ記号を選択し、どのレジスタにするか指定します。さらに [ENTER] で、選択したスタックレジスタをプログラムや式から呼び出すことができるようになります。この操作により、REGX、REGY、REGZ、REGT と表示されます。

たとえば、プログラムの最初に EQN を押し、REGX x REGY x REGZ x REGT と入力した場合は、4 つのスタックレジスタの積を計算し、その結果を X レジスタに保存します。この計算後は、X、Y、Z レジスタの数値がそれぞれ、Y、Z、T レジスタに保存されます。

HP35s のスタックにはこれ以外の法則はありません。これまで解説したスタックの法則をふまえて、効率的にスタックを利用して下さい。

C

ALG: まとめ

ALG について

この章では ALG モードの機能についてまとめます。次の項目も含みます。

- 2 つの引数をとる演算
- 指数と対数 (◀ 10^x, ◀ LOG, ▶ e^x, ▶ LN)
- 三角関数
- 数値の成分
- スタックの確認
- 複素数の計算
- 式の積分
- 2 進数、8 進数、16 進数の計算
- 2 変数の統計データを入力

ALG モードにするには MODE 4 (4ALG) と操作します。ALG モードのときは表示記号 ALG がオンになります。

ALG モードでは次の順番で計算が実行されます。

1. カッコの中の式。
2. 階乗の ！。これは ! を押す前に数値を入力して下さい。
3. COS, SIN, TAN, ACOS, ASIN, ATAN, LOG, LN, x^2, 1/x, $\sqrt{x}$, π, $\sqrt[3]{x}$, %, RND, RAND, IP, FP, INTG, SGN, nPr, nCr, %CHG, INT÷, Rmdr, ABS, e^x,10^x, 単位変換。これらは関数キーの後に数値を入力します。
4. $\sqrt[x]{y}$ と y^x
5. 符号変換（+/-）
6. ×, ÷
7. +, –
8. =

ALG における二項演算子

ここでは、ALG モードが影響を与える演算を解説します。次に示す二項演算子は、RPN と ALG では入力方法が異なります。

- **四則演算**
- **指数関数** ([y^x], [$\sqrt[x]{y}$])
- **百分率の計算** ([%] と [→] [%CHG])
- **順列と組合せ**([←] [nCr], [→] [nPr])
- **商と余り** ([←] [INTG] [2](2INTG÷), [←] [INTG] [3](3Rmdr))

四則演算

簡単な四則演算の例を示します。

ALG モードでは最初の数値を入力し、次に演算子 ([+], [−], [×], [÷])、続けて第 2 の数値、最後に [ENTER] を入力します。

例	操作	表示
12+3	[1] [2] [+] [3] [ENTER]	12+3 15.0000
12 – 3	[1] [2] [−] [3] [ENTER]	12-3 9.0000
12 × 3	[1] [2] [×] [3] [ENTER]	12×3 36.0000
12 ÷ 3	[1] [2] [÷] [3] [ENTER]	12÷3 4.0000

指数関数

ALG モードで y の x 乗を求めるには、y [y^x] x [ENTER] と操作します。

例	操作	表示
12^3	1 2 y^x 3 ENTER	12^3 1,728.0000
$64^{1/3}$ (立方根)	◀ $\sqrt[x]{y}$ 3 ▶ 6 4 ENTER	XROOT(3,64) 4.0000

百分率の計算

百分率は % キーを使って次のように計算します。

例	操作	表示
200 の 27%	▶ % 2 0 0 ▶ 2 7 ENTER	%(200,27) 54.0000
200 を 27%減少	2 0 0 − ▶ % 2 0 0 ▶ 2 7 ENTER	200-%(200,27) 146.0000
25 を 12%増加	2 5 + ▶ % 2 5 ▶ 1 2 ENTER	25+%(25,12) 28.0000

例	操作
y の *x*% *y* から *x* への変化率 ($y \neq 0$)	▶ % *y* ▶ *x* ENTER ◀ %CHG *y* ▶ *x* ENTER

例

昨年 $16.12 だった商品が今年は $15.76 になったとします。昨年と今年の変化率を求めて下さい。

キー	表示	説明
◀ %CHG 1 6 · 1 2 ▶ 1 5 · 7 6 ENTER	%CHG(16.12,15.76) -2.2333	昨年に比べて約 2.2%安くなりました。

順列と組合せ

例: 人の組合せ

ある会社に 14 人の女性と 10 人の男性がいます。この中で 6 人組をつくるとして、その組合せは何通りありますか？

キー	表示	説明
◀ nCr 2 4 ▶	nCr(24,6)	組合せの場合の数
6 ENTER	134,596.0000	

商と余り

商と余りを求めるには、◀ INTG 2 (2INTG÷) と ◀ INTG 3 (3Rmdr) を使います。どちらも 2 つの引数が必要です。

◀ INTG 2 (2INTG÷) **整数** $_1$ ▶ **整数** $_2$ ENTER

◀ INTG 3 (3Rmdr) **整数** $_1$ ▶ **整数** $_2$ ENTER

例:

58 ÷ 9 の商と余りを求めます。

キー	表示	説明
◀ INTG 2 (2INTG÷)	IDIV(58,9)	商
5 8 ▶ 9 ENTER	6.0000	
◀ INTG 3 (3Rmdr)	RMDR(58,9)	余り
5 8 ▶ 9 ENTER	4.0000	

カッコを含む計算

カッコを使うのは、中間計算を後回しにして、あらかじめ数値を入力する必要があるときです。たとえば、次の計算を考えます。

$$\frac{30}{85-12}\times 9$$

3 0 ÷ 8 5 − 1 2 × 9 と入力すると、その結果は－107.6471 となりますが、この結果は計算間違いです。除算の実行を 85－12 の後にするには、次のようにカッコを用います。

キー	表示	説明
3 0 ÷ ( ) 8 5 −	30÷(85-)	除算の実行を後回しにします
1 2 ›	30÷(85-12)_	85 – 12 の計算
× 9	30÷(85-12)×9_	30/73 の計算
ENTER	30÷(85-12)×9 3.6986	30/(85 – 12) × 9 の計算

左カッコの前の乗算記号 (×) は省略可能です。ただし、式モードでは省略できません。たとえば 2 × (5 – 4) は × キーを使わずに 2 ( ) 5 − 4 と入力することができます。この場合は「2」と「(」の間の乗算記号を省略しています。

指数と対数

例	操作	表示
自然対数 (底 e)	↱ LN 1 ENTER	LN(1) 0.0000
常用対数 (基数 10)	↰ LOG 1 0 ENTER	LOG(10) 1.0000
e を底とする指数	↱ e^x 2 ENTER	EXP(2) 7.3891
10 を底とする指数	↰ 10^x 2 ENTER	ALOG(2) 100.0000

三角関数

次の例では、角度の単位系が MODE 1 (1DEG) であるとします。

例	操作	表示
x の正弦	SIN 3 0 ENTER	SIN(30) 0.5000
x の余弦	COS 6 0 ENTER	COS(60) 0.5000
x の正接	TAN 4 5 ENTER	TAN(45) 1.0000
x の逆正弦	↷ ASIN 1 ENTER	ASIN(1) 90.0000
x の逆余弦	↷ ACOS 0 ENTER	ACOS(0) 90.0000
x の逆正接	↷ ATAN 0 ENTER	ATAN(0) 0.0000

双曲線関数

例	操作
x の双曲線正弦 (SINH)	↶ HYP SIN, 数値入力, ENTER
x の双曲線余弦 (COSH)	↶ HYP COS, 数値入力, ENTER
x の双曲線正接 (SINH)	↶ HYP TAN, 数値入力, ENTER
x の双曲線逆正弦 (ASINH)	↶ HYP ↷ ASIN, 数値入力, ENTER
x の双曲線逆余弦 (ACOSH)	↶ HYP ↷ ACOS, 数値入力, ENTER
x の双曲線逆正接 (ASINH)	↶ HYP ↷ ATAN, 数値入力, ENTER

数値の成分

例	操作	表示
2.47 の整数部	◀ INTG 6 (6IP) 2 . 4 7 ENTER	IP(2.47) 2.0000
2.47 の小数部	◀ INTG 5 (5FP) 2 . 4 7 ENTER	FP(2.47) 0.4700
–7 の絶対値	▶ ABS +/– 7 ENTER	ABS(-7) 7.0000
9 の符号	◀ INTG 1 (1SGN) 9 ENTER	SGN(9) 1.0000
–5.3 以下の最も大きな整数	◀ INTG 4 (4INTG) +/– 5 . 3 ENTER	INTG(-5.3) -6.0000

スタックの確認

R↓ や ▶ R↓ キーを押すとメニューが表示され、X、Y、Z、T レジスタに保存された数値を確認できます。R↓ と ▶ R↓ では、最初に選択されるレジスタが異なります。▶ R↓ は T レジスタが選択され、R↓ は Y レジスタが選択されます。

R↓ では次のメニューが表示されます:

X Y Z T

数値

▶ R↓ では次のメニューが表示されます:

X Y Z T

数値

▷ と ◁ キーの代わりに R↓ と ▶ R↓ を使うこともできます。この場合は選択スタックが順番に切り替わります。ENTER で選択を確定すると、REGX、REGY、REGZ、REGT のいずれかが表示され、次の計算に利用できます。

ALG モードにおける X、Y、Z、T レジスタに保存された数値は、RPN モードと同じです。通常の計算、SOLVE、プログラム実行、積分実行の後のスタックの挙動も、ALG と RPN で同じです。また、RPN と ALG のモードを切り替えるときもスタックに変化はありません。

式の積分

1. 式を入力 (第 6 章「式リストに式を入力」をご参照下さい) し、式モードを終了します。
2. 積分区間を入力します。最初に**下限**を入力し、[x◂▸y] キー、次に**上限**を入力します。
3. 目的の式を表示させます。[EQN] を押し、必要であれば式リストを [▲] と [▼] でスクロールさせて目的の式を選択します。
4. 被積分変数を選択するため、「[◀] [/] 変数名」と操作します。すると計算を開始します。

複素数の計算

複素数の入力は次のように行います。

形式: x i y

1. 実数部を入力します。
2. [i] を押します。
3. 虚数部を入力します。

形式: x + y i

1. 実数部を入力します。
2. [+] を押します。
3. 虚数部を入力します。
4. [i] を押します。

形式: r θ α

1. r の値を入力します。
2. [▶] [θ] を押します。
3. θ の値を入力します。

複素数の関数による計算は次のように行います。

1. 関数を選択します。
2. 複素数 z を入力します。
3. ENTER を押して計算します。
4. 第 2 行に計算結果が表示されます。表示形式は MODE で設定されているものになります。

複素数の四則演算は次のように行います。

1. 最初の複素数 z_1 を入力します。
2. 演算子を入力します。
3. 第 2 の複素数 z_2 を入力します。
4. ENTER を押して計算します。
5. 第 2 行に計算結果が表示されます。表示形式は MODE で設定されているものになります。

次に、複素数の計算例を示します。

例

sin (2＋3*i*) を計算します。

キー	表示	説明
↶ DISPLAY 9 (9×i,v)		表示形式の設定
SIN 2 ＋ 3 i	SIN(2＋3i)	
ENTER	SIN(2＋3i) 9.1545i−4.1689	結果: 9.1545 *i*−4.1689

例

次の式を計算します。

$$z_1 \div (z_2 + z_3),$$

ただし、$z_1 = 23 + 13\,i$, $z_2 = -2 + i$　$z_3 = 4 - 3\,i$ とします。

キー	表示	説明
↺ DISPLAY ・ 1		表示形式の設定
(10x+yi)		
( )	⬅i÷(-2+i+4-3i)	
2 3 + 1 3 i		
› ÷ ( ) +/- 2 +		
i + 4 − 3 i		
ENTER	(23+13i)÷(-2+...	結果:
	2.5000+9.0000i	2.5000 + 9.0000 *i*

例

(4−2/5 *i*)×(3−2/3 *i*)　を計算します。

キー	表示	説明
( ) 4 − ・ 2 ・	⬅5i)×(3-0.2/3i)	
5 i › × ( ) 3		
− ・ 2 ・ 3 i		
ENTER	(4-0.2/5i)×(3...	結果:
	11.7333i-3.8667	11.7333 *i*–3.8667

2 進数、8 進数、16 進数の計算

2 進数、8 進数、16 進数モードの例題を示します。

例

$12F_{16} + E9A_{16} = ?$

キー	表示	説明
↷ BASE 2 (2HEX)		16 進数モードにします。表示記号 **HEX** が表示されます。
1 2 RCL F ↷	12Fh+E9Ah	計算結果。
BASE 6 (6h) + RCL	FC9h	

E 9 RCL A ↷
BASE 6 (6h) ENTER

$7760_8 - 4326_8$=?

↷ BASE 3 (3OCT)	12Fh+E9Ah 7711o	8 進数モードにします。表示記号 **OCT** が表示されます。
7 7 6 0 ↷ BASE 7 (7o) − 4 3 2 6 ↷ BASE 7 (7o) ENTER	7760o-4326o 3432o	計算結果。

$100_8 \div 5_8$=?

1 0 0 ↷ BASE 7 (7o) ÷ 5 ↷ BASE 7 (7o) ENTER	100o÷5o 14o	計算結果の整数部。

$5A0_{16} + 10011000_2$ =?

↷ BASE 2 (2HEX) 5 RCL A 0 ↷ BASE 6 (6h) + 1 0 0 1 1 0 0 0 ↷ 0 ↷ BASE 8 (8b)	5A0h+ ◀A0h+10011000b	16 進数モードにします。表示記号 **HEX** が表示されます。
ENTER	5A0h+10011000b 638h	計算結果（16 進数）
↷ BASE 1 (1DEC)	5A0h+b10011000b 1,592.0000	10 進数表示に変更

2 変数の統計データを入力

ALG モードでは (x, y) のセットを**逆順**に入力します。y x⇄y x または y ENTER x となります。これは y を Y レジスタに、x を X レジスタに保存するためです。

1. 既存の統計データを削除するために ↷ CLEAR 4 (4Σ) を押します。
2. **先に** y の値を入力し、x⇄y を押します。

3. 対応する z の値を入力し、Σ+を押します。
4. すると、統計データのセット数 *n* がディスプレイに表示されます。
5. 次の *x*, *y* のセットを入力します。入力するたびに *n* が更新されます。

入力直後のデータを削除するには↩ Σ−と操作します。統計データを削除すると、削除したデータがディスプレイ 1 行目に、残りのデータ数 *n* が 2 行目に表示されます。全ての統計データを削除するとディスプレイ 2 行目には 0 が表示されます。

例

表の左側のデータを入力した後、右側のデータに訂正してみましょう。

最初の (x, y)	訂正 (x, y)
20, 4 400, 6	20, 5 40, 6

キー	表示	説明
↪ CLEAR 4 (4Σ)		既存の統計データをクリア
4 x⇄y 2 0 Σ+	20 Σ+ 1.0000	最初のデータセットを入力
6 x⇄y 4 0 0 Σ+	400 Σ+ 2.0000	統計データのセット数 *n* がディスプレイに表示されます。
↪ LAST*x*	LAST*x* 400.0000	最後の *x* の値を呼び出します。 最後の *y* の値はまだ Y レジスタにあります。
↩ Σ−	400 Σ− 1.0000	最後のデータセットを削除
6 x⇄y 4 0 Σ+	40 Σ+ 2.0000	最後のデータセットを再入力
4 x⇄y 2 0 ↩ Σ−	20 Σ− 1.0000	最初のデータセットを削除
5 x⇄y 2 0 Σ+	20 Σ+ 2.0000	最初のデータセットを再入力。 2 つのデータセットが統計レジスタに保存されました。

線形回帰

線形回帰（L.R.、Linear Regression）は、与えられたデータ (x, y) にフィットする直線を求める統計手法です。

- y（または x ）に対応する x（または y）の予測値を求めるには、y（または x ）を入力後に ENTER を押し、 L.R. (ŷ) （または L.R. (x̂)）と操作します。
- 線形回帰で得られた直線を確認するには L.R. と操作し、(r), (m), (b) をカーソルで選択します。

D

SOLVE の詳細

この章では第 7 章で登場した SOLVE について詳しく解説します。

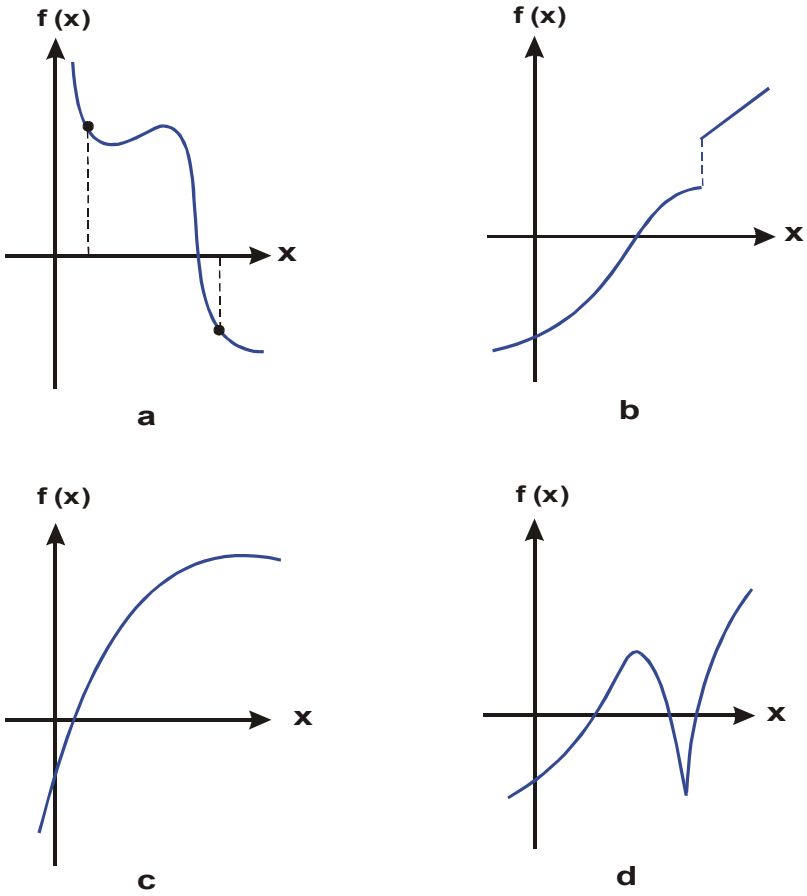
SOLVE が根を見つける方法

SOLVE はまず、未知変数について直接に SOLVE の実行を試みます。この試行が失敗したときは反復計算に移行します。この反復計算は式に対して繰り返し実行されます。式が返すのは関数 *f(x)*の未知数 x の値です。(*f(x)*は未知数 x を使って数学的に簡素化して表現された関数です。) SOLVE は、未知数 x の推定値を関数 *f(x)* に当てはめ、その推定値を洗練させていくという方法をとります。

2 つの連続な推定値を代入した *f(x)* の値がそれぞれ異なる符号になったとき、*f(x)* と x 軸がそれらの推定値の区間で少なくとも 1 回は交差すると、SOLVE は推定します。根が発見できるまで、推定値の間隔を規則的に狭くしていきます。

SOLVE が根を発見するためには、電卓が認識できる数値の範囲(オーバーフロー境界の内側)に根は存在している必要があります。また、繰り返し計算を実行するので、関数は数式的に定義されていなければなりません。根がオーバーフロー境界の内側にひとつでも存在するときは、次の条件のひとつ以上に合致するのであれば、SOLVE は必ず根を発見します。

- **2 つの推定値を代入した *f(x)* の値が異なる符号を持ち、これらの推定値の間で関数のグラフが x 軸に少なくとも 1 回交差する場合(次の図 a)**
- ***f(x)* が常に増加または減少する場合(図 b)**
- ***f(x)* のグラフがすべての領域で凸型か凹型の場合(図 c)**
- ***f(x)* に 1 つ以上の極小値または極大値があるときは、極小値または極大値がそれぞれ単独で *f(x)* の根に近接している場合(図 d)**



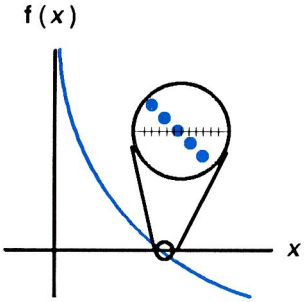
根を発見可能な式

数式処理の計算では *f(x)* = 0 ですが、電卓の 12 桁精度の制限により、ゼロではなく非常に小さい値が残ってしまうことが多いです。ただしほとんどの場合、計算で求めた根は精密な推定値であり、理論的な式の根に対しても十分な精度になります。

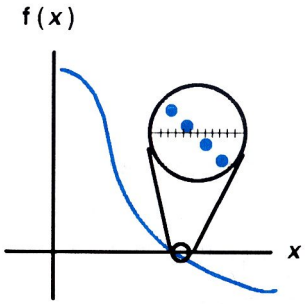
結果の調査

SOLVE は次のどちらかの状態のとき、結果を出力します。

- *f(x)* = 0 のとき（図 a）
- *f(x)* = 0 ではないが、その関数のグラフと x 軸との交点に対し、12 桁精度で計算された根が隣接するとき（図 b）。これは 2 つの最終推定値が隣り合っていて（12 桁目の数値の差が 1 である状態）、関数にそれぞれの推定値を代入した値のうち、一方が正でもう一方が負であるとき、あるいは、一方が $(0, 10^{-499})$ でもう一方が $(0, -10^{-499})$ であるときに発生します。ほとんどの場合、*f(x)* は相対的にゼロに近い値になります。



a



b

✓ SOLVE の結果をチェックするため、R↓ で Y レジスタに保存された、根の直前の推定値を呼び出すことができます。続けてもう一度 R↓ を押すと、Z レジスタに保存されていた、*f(x)* の値を確認できます。*f(x)* の値がゼロまたは相対的に十分小さいものであれば、無事に根を発見したとみなすことができます。これに対し、*f(x)* の値が相対的に大きかった場合は、結果の評価には注意を要します。

例：根が 1 つの式

次の式の根を計算します。

$$-2x^3 + 4x^2 - 6x + 8 = 0$$

この式を等号のない式（Expression）として登録します。

キー	表示	説明
EQN		式モードを選択
+/− 2 ×		式を入力
RCL X y^x 3		
+ 4 ×		

RCL X y^x 2		
− 6 × RCL X		
+ 8 ENTER	-2xX^3+4xX^2-6➡	
↺ SHOW	CK=B9AD	チェックサムと長さ.
	LN=18	
C		式モードをキャンセル

入力した式の根を SOLVE で求めます。

キー	表示	説明
0 ↻ STO X	10_	根の初期推定値
ENTER 1 0		
EQN	-2xX^3+4xX^2-6➡	式モードを選択。 式の左端が表示されます。
↻ SOLVE X	SOLVING	*X* を未知数として SOLVE を
	X=	実行。
	1.6506	結果が表示されます。
✓ R↓	1.6506	最後の 2 つの推定値が小数点以下 4 位まで一致。
✓ R↓	-4.0000E-11	*f(x)* の値が非常に小さいので、この近似値は良い結果であると言えます。

例: 根が 2 つの式

次の二次方程式の 2 つの根を計算します。

$$x^2 + x - 6 = 0.$$

まず、この式を等号のない式として登録します。

キー	表示	説明
EQN		式モードを選択
RCL X y^x 2 +		式を入力
RCL X − 6	X^2+X-6	
ENTER		
↺ SHOW	CK=3971	チェックサムと長さ.
	LN=7	

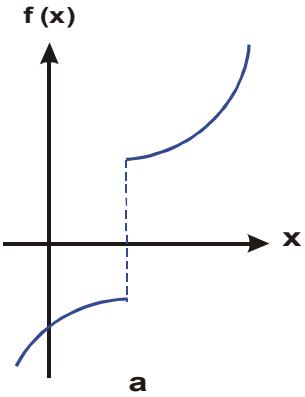
C		式モードをキャンセル

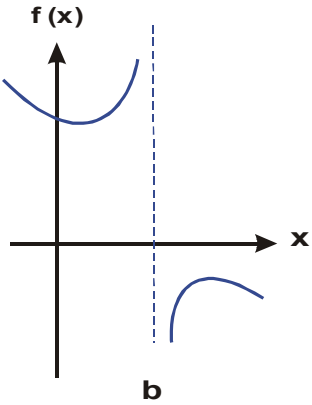
次に、入力した式の正の根ならびに負の根を SOLVE で求めます。

	キー	表示	説明
	0 ↗ STO X ENTER 1 0	10_	正の符号を持つ根の初期推定値を入力
	EQN	X^2+X-6	式モードを選択。 式の左端が表示されます。
	↗ SOLVE X	SOLVING X= 2.0000	推定値 0 から 10 で正の根を計算します。
✓	R↓	2.0000	最後の 2 つの推定値が一致
	R↓ ↖ SHOW	0.00000000000	$f(x) = 0$
✓	0 ↗ STO X ENTER 1 0 +/−	−10_	負の符号を持つ根の初期推定値を入力
	EQN	X^2+X-6	式を再表示
	↗ SOLVE X	SOLVING X= −3.0000	推定値 0 から−10 で負の根を計算します。
✓	R↓ R↓ ↖ SHOW	0.00000000000	$f(x) = 0$

次のような場合は注意して下さい。

- 次の図 a のように、関数のグラフが x 軸に交差する部分で不連続のとき。この場合、SOLVE は不連続な点に近接した値を返します。この場合、SOLVE は不連続な点に近接した値を返し、*f(x)* の値は相対的に大きくなります。
- 図 b のように、*f(x)*のグラフの符号が変化する点で、*f(x)* の値が無限大に向かっている可能性があるとき。この状態は極（*pole*）と呼ばれています。SOLVE は隣り合う 2 つの x を代入した *f(x)* の値に符号の変化があると、根の可能性があると判断します。しかし、このときの *f(x)*の値は相対的に大きなものになります。もし極が 12 桁精度で表現できる x で発生しているときは、エラーメッセージを出力して計算が停止します。





特別な場合 :不連続な式と、極がある式

例 : 不連続な式

次の式の根を計算します。

IP(x) = 1.5

まず、式を入力します。

キー	表示	説明
EQN		式モードを選択
↰ INTG 6 (6IP)		式を入力
RCL X › ↰ =		
1 . 5 ENTER	IP(X)=1.5	
↰ SHOW	CK=D2C1 LN=9	チェックサムと長さ.
C		式モードをキャンセル

入力した式の根を SOLVE で求めます。

キー	表示	説明
0 ↱ STO X		根の初期推定値
ENTER 5	5_	
EQN	IP(X)=1.5	式モードを選択。

			式の左端が表示されます。
[→] [SOLVE] [X]		SOLVING X= 2.0000	推定値 0 から 5 の間で根を計算します。
[◁] [SHOW]		1.99999999999	根を有効数字 12 桁で表示。
✓	[R↓] [◁] [SHOW]	2.00000000000	直前の推定値がやや大きい。
✓	[R↓]	-0.5000	*f(x)* は相対的に大きいです。

最終の 2 つの推定値に差がある点と、*f(x)* の値が相対的に大きい点に注目して下さい。ここでの問題は、推定値として得られた x を *f(x)* に代入してもゼロにならないということです。その一方で、得られた根に隣接する x = 1.99999999999 のとき、*f(x)* は反対の符号になります。

例 :

次式の根を計算します。

$$\frac{x}{x^2-6}-1=0$$

x が $\sqrt{6}$ に近づくと、*f(x)* が無限大または無限小に近づきます。

この式を等号のない式として登録します。

キー	表示	説明
[EQN]		式モードを選択
[RCL] [X] [÷] [( )]		式を入力
[RCL] [X] [y^x] [2]		
[−] [6] [▷]		
[−] [1] [ENTER]	X÷(X^2-6)-1	
[◁] [SHOW]	CK=7358 LN=11	チェックサムと長さ.
[C]		式モードをキャンセル

入力した式の根を SOLVE で求めます。

キー	表示	説明
[2] [.] [3] [→]		根の初期推定値

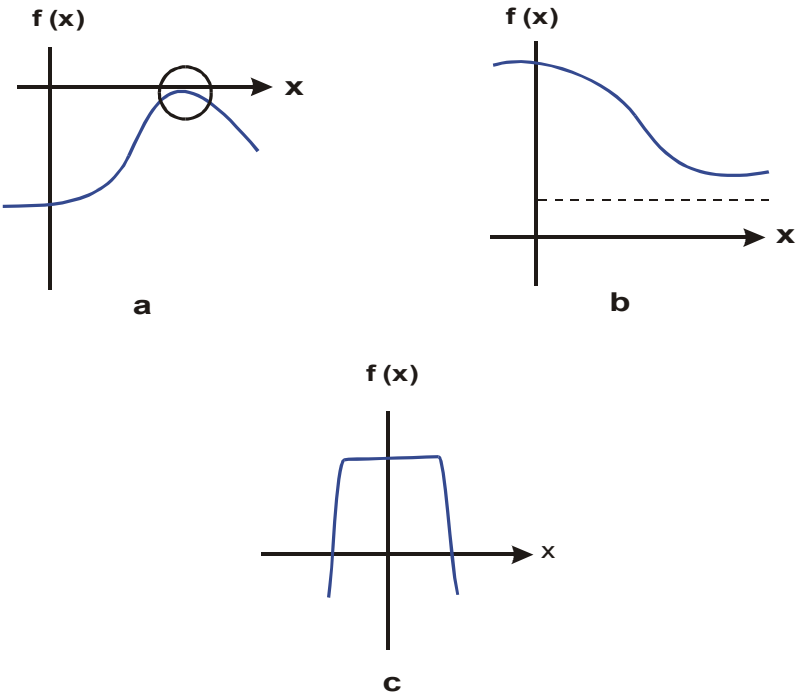
STO X ENTER 2 . 7	2.7_	
EQN	X÷(X^2-6)-1	式モードを選択。 式の左端が表示されます。
↗ SOLVE X	NO ROOT FND	*f(x)* の根は見つかりませんでした。

SOLVE が根を発見できなかったときは

SOLVE は常に根を発見できるわけではありません。次のような場合には NO ROOT FND というメッセージを出力し、計算を停止します。

- 次の図 a のように、極小値または極大値の付近で根の検索が終了してしまう場合。
- 図 b のように、*f(x)* が広範囲の x に対して基本的に一定で、水平線に漸近する場合も根の検索が途中終了します。
- 図 c のように局所で平坦な領域がある場合は、根の検索がその領域に集中してしまいます。

これらの場合、スタックの値は SOLVE の前後で変化しません。



根が見つからない例

例: 再小値

次の二次方程式の根を計算します。

$$x^2 - 6x + 13 = 0.$$

x = 3 で最小値になります。この式を等号のない式として登録します。

キー	表示	説明
EQN		式モードを選択
RCL X y^x 2		式を入力
− 6 × RCL X		
+ 1 3 ENTER	X^2-6xX+13	
↶ SHOW	CK=EC74 LN=10	チェックサムと長さ.

C		式モードをキャンセル

入力した式の根を SOLVE で求めます。

キー	表示	説明
0 ↷ STO X		根の初期推定値
ENTER 1 0	10_	
EQN	X^2-6×X+13	式モードを選択。 式の左端が表示されます。
↷ SOLVE X	NO ROOT FND	推定値 0 から 10 の間で根を計算します。

例: 漸近線

次式の根を計算します。

$$10-\frac{1}{X}=0$$

この式を等号のない式として登録します。

キー	表示	説明
EQN		式モードを選択
1 0 − 1/x		式を入力
RCL X ENTER	10-INV(X)	
↶ SHOW	CK=6EAB LN=9	チェックサムと長さ.
C		式モードをキャンセル
. 0 0 5 ↷		根の正の推定値
STO X ENTER 5	5_	
EQN	10-INV(X)	式モードを選択。 式の左端が表示されます。
↷ SOLVE X	X= 0.1000	推定値 0.005 から 5 の間で根を計算します。
✓ R↓	0.1000	最後の 2 つの推定値が一致
✓ R↓ ↶ SHOW	0.00000000000	*f* (x) = 0

推定値に負の値を与えるとどうなるか確認します。

キー	表示	説明
[+/−] [1] [→] [STO] [X] [ENTER]	-1.0000	根の負の推定値
[+/−] [2] [EQN]	10-INV(X)	式モードを選択。 式の左端が表示されます。
[→] [SOLVE] [X]	X= 0.1000	SOLVE を実行。 計算結果が表示されます。

例: 次式の根を計算します。

$$\sqrt{[x \div (x + 0.3)]} - 0.5 = 0$$

この式を等号のない式として登録します。

キー	表示	説明
[EQN]		式モードを選択
[√x] [RCL] [X] [÷] [( )] [RCL] [X] [+] [.] [3] [>] [>] [−] [.] [5]		式を入力
[ENTER]	SQRT(X÷(X+0.3))➡	
[←] [SHOW]	CK=9F3B LN=19	チェックサムと長さ.
[C]		式モードをキャンセル

最初に正の根を探します。

キー	表示	説明
[0] [→] [STO] [X]		根の正の推定値
[ENTER] [1] [0]	10_	
[EQN]	SQRT(X÷(X+0.3))➡	式モードを選択。 式の左端が表示されます。
[→] [SOLVE] [X]	X= 0.1000	推定値 0 から 10 で正の根を計算します。

次に、推定値を0および－10として、負の根を計算します。xが0から–0.3の間のとき、平方根の中の分子が負、分母が正になり、この関数は複素数になるという点にご注意下さい。

キー	表示	説明
0 ↷ STO X		
ENTER +/− 1 0	-10_	
EQN	SQRT(X÷(X+0.3))➡	式モードを選択。 式の左端が表示されます。
↷ SOLVE X	NO ROOT FND	*f(x)* の根は見つかりませんでした。

例: 局所で平坦な関数

次式の根を計算します。

$x < -1$ のとき　$f(x) = x + 2$

$-1 \le x \le 1$ のとき　$f(x) = 1$（平坦な領域）

$x > 1$ のとき　$f(x) = -x + 2$

RPN モードでは、この関数をプログラムとして次のように定義します。

```
J001 LBL J
J002 1
J003 2
J004 RCL+ X
J005 x<y?
J006 RTN
J007 4
J008 -
J009 +/-
J010 x>y?
J011 R↓
J012 RTN
```

チェックサムと長さ: 9412 39

推定値を 10^{-8} と -10^{-8} として根を計算します。

キー (RPN モード)	表示	詳細
1 E 8		推定値を入力
+/− ↻ STO X 1 +/− E 8 +/−	-1E-8_	
↺ FN= J	-1.0000E-8	プログラム「J」を関数として選択します。
↻ SOLVE X	X= -2.0000	SOLVE を実行。 結果を表示します。

丸め誤差

有効桁が 12 桁で有限なので、丸めによって誤差が発生することがあります。また、丸めは、SOLVE と積分の繰り返し計算で増幅されることもあります。たとえば次の式は *f(x)* が常にゼロより大きいため、根がありません。

$$[(|x|+1)+10^{15}]^2 - 10^{30} = 0$$

しかし、初期推定値に1と2を与えると、丸めによってSOLVEは1.0000を出力します。

丸め誤差により、根が発見できなくなることもあります。たとえば、次式の根は $\sqrt{7}$ です。

$$|x^2 - 7| = 0$$

しかし 12 桁精度では $\sqrt{7}$ を正確に表すことができませんから、関数をゼロにすることができません。さらに、関数の符号が変化することもありません。よって、SOLVE は根を発見できず、NO ROOT FND が出力されます。

E

積分の詳細

この章では第 8 章で登場した積分について詳しく解説します。

積分の方法

∫FN dx のアルゴリズムでは、関数 *f(x)* に対してその積分領域内で多数の値を x のサンプル点として与え、その加重平均を求めることにより、*f(x)* の積分の近似値を計算します。

この手法では、計算の精度はサンプル点の数にほぼ依存します。もしサンプル数が無限であり、かつ、*f(x)* の計算における誤差が無いとすれば、数学的に正確な積分値が計算できます。しかし、無限のサンプル点の計算には無限の時間がかかってしまいます。

一方で、積分の最良な精度は関数の計算精度に依存しますから、無限のサンプル数は必要ありません。有限のサンプル点のみでも、実用的な精度による計算が可能です。

このアルゴリズムでは最初に、2～3 個のサンプル点のみを使った近似値を求めます。もしそれが十分な正確度ではないとき、サンプル点を増やして繰り返し計算を行います。*f(x)*に内包される不確定値を考慮した場合と、その計算で得られた近似値が同等の正確度になるまで、サンプル点を約 2 倍ずつ増加させて再計算が行われます。

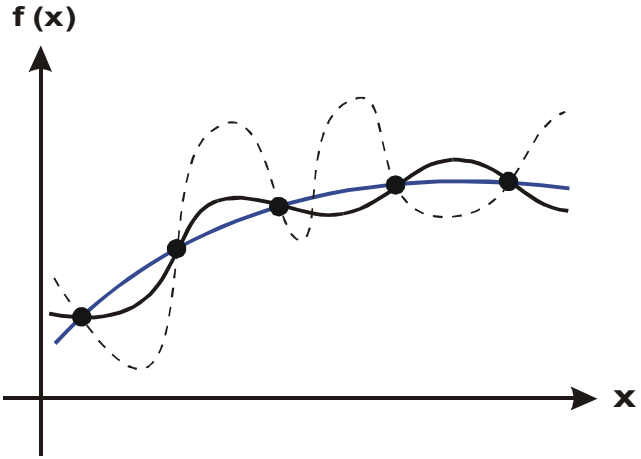
第 8 章で解説のとおり、最終的な近似値の不確定値は表示形式の設定に依存します。それぞれの繰り返し計算において、最後に得られた近似値と、その直前の 2 つの近似値を比較します。
これら 3 つの近似値でそれぞれ、残りの 2 つの近似値との差を求めます。その差が最後の近似値に対する不確定値の許容誤差よりも小さいとき、積分計算を終了します。このとき、最後の近似値が X レジスタに、不確定値が Y レジスタに保存されます。

この計算における 3 つの近似値と厳密解との誤差が、3 つの近似値のそれぞれの差より大きくなることは、ほとんどありません。したがって、最後の近似値の誤差は不確定値よりも小さくなります（ *f(x)* には急激な変化がないものとします）。最後の近似値の誤差を計算することはできませんが、その誤差は不確定値を越えないと見なすことができます。言い換えれば、Y レジストリに保存された不確定値は、得られた近似値と厳密解との差の「最大値」と考えることができます。

不正確な結果を引き起こす条件

数値積分のアルゴリズムの中でも、HP 35s に搭載されているのは最良のもののひとつですが、特定の条件で不正確な結果を出力します。**ただし、その条件はほとんど発生しません。**このアルゴリズムは、スムーズな関数で精度の良い結果を出力するように設計されています。急激に変化する関数の場合のみ、不正確な結果を出力するリスクが高いです。ただし、そのような関数の数は、自然科学で登場する関数に比べて非常に少ないですし、問題になる関数は計算前に認識でき、簡単な方法で対処できます。

アルゴリズムが *f(x)* について認識できるのは各サンプル点における関数値のみであり、*f(x)* とそれに似た関数が各サンプル点で交差するときは、それぞれの関数を区別することができません。このような状態は次の図のときに発生します。ここで示す積分範囲において、3 つの関数が共通のサンプル点で交差しています。



図のサンプル点では、全ての関数で同じ近似値を出力します。青の実線に対する積分の厳密解は出力された近似値に近いものになり、*f(x)*がこれだったとき、その推定値は精度が良いと言えます。しかし、破線の関数の厳密解は、その他の関数の厳密解と大きく異なります。よってその場合の近似値は、より不正確なものとなります。

このアルゴリズムでは、サンプル点をもっと増やすことによって、被積分関数の特性をより正確に計算できるようになります。ある積分区間における関数値の変動が、それ以外の区間における変動に比べて大きく異なるとき、このアルゴリズムの繰り返し計算ではその変動を予想して検出します。これが発生すると、その関数のほとんどの区間における変動（特性ではありません）を表現できる近似値が得られるまで、サンプル点の数が増加されます。

たとえば次の近似値を考えます。

$$\int_0^{\infty} xe^{-x}dx$$

数値積分を行うので、積分の上限は無限大ではなく、この電卓で取り扱い可能な最大値である 10^{499} とします。

入力して何が起こるか試してみましょう。まず関数 $f(x) = xe^{-x}$ を入力します。

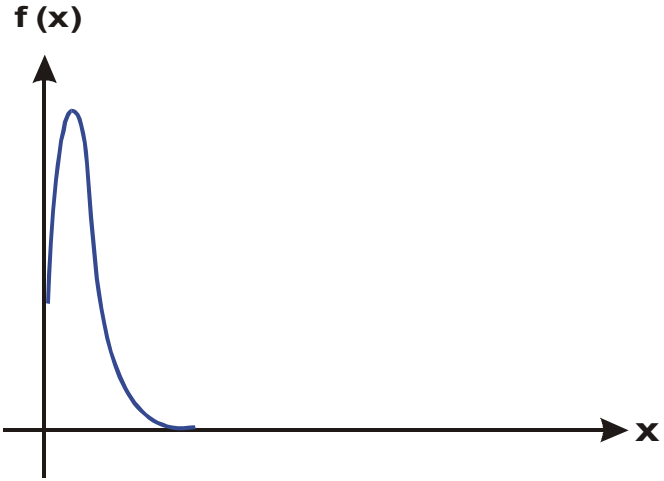
キー	表示	説明
EQN		式モードを選択
RCL X × ↷ e^x	X×EXP(	式を入力
+/− RCL X ENTER	X×EXP(−X)	式を入力終了
↶ SHOW	CK=2FE6 LN=9	チェックサムと長さ.
C		式モードをキャンセル

表示形式を SCI 3 に、下限と上限をそれぞれゼロと 10^{499} とし、積分を実行します。

キー	表示	説明
✓ ↶ DISPLAY 2 (2SCI)		精度、積分の上限・下限を入力します。
3 ENTER 1 E 4 9 9	1E499_	
EQN	X×EXP(−X)	式モードを選択。 式の左端が表示されます。

[↵] [/] [X]	INTEGRATING ∫ = 0.000E0	積分の近似値

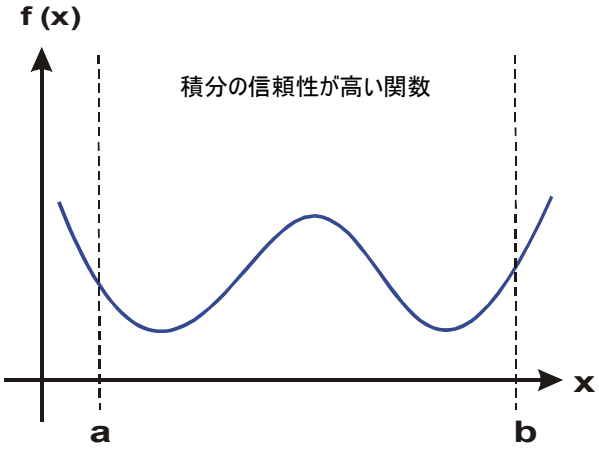
$f(x) = xe^{-x}$ のゼロから無限大における積分の厳密解は 1 ですから、得られた近似値は不正確です。ただし、ゼロから 10^{499} を積分範囲とした場合の厳密解は 1 に非常に近いものになりますから、無限大を 10^{499} で表現したのが問題なのではありません。不正確になった理由は、$f(x)$ の積分区間におけるグラフではっきりします。

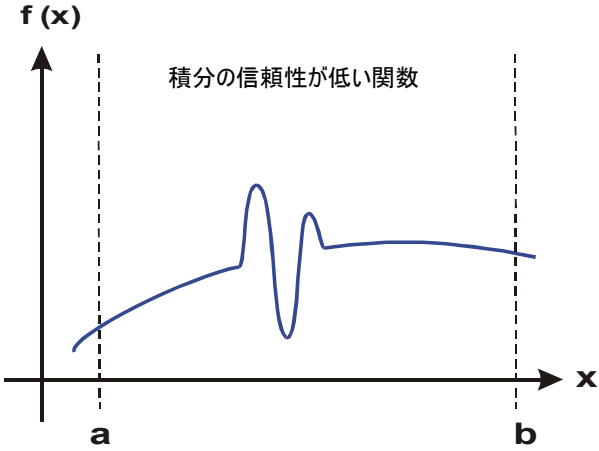


このグラフは原点に非常に近い位置で大きく変動しています。この変動をカバーするサンプル点が無いため、アルゴリズムは $f(x)$ が積分区間においてずっとゼロであるとみなします。表示形式を SCI 11 や ALL にしてサンプル点を増やしても、この積分区間と被積分関数では変動をとらえることができません。(このような場合の対処法は次節の「計算時間が長くなる条件」をご参照下さい。)

このような、ある区間が他の区間に比べて急激に変化する特性を持つ関数は、ほとんど存在しません。不正確な結果を引き起こす可能性のある関数は、その計算時間および、積分区間での低次の導関数を調べるという簡単な方法で特定できます。基本的に、関数値および低次の導関数の変化が急激であればあるほど、計算時間がかかるようになります。それにも関わらず計算時間が短いときは、その結果の信頼性は低いと判断できます。

関数値や低次の導関数が急激に変化するときは、積分区間の幅に注意しなければなりません。たとえば、ある関数 *f(x)* が積分区間で 3 回の変動を持つとします。次図のように、その積分区間の狭い領域のみで 3 回の変動があるのか、あるいは、全体に渡って変動するのかにより、関数をうまく表現できるサンプル数が異なってきます。被積分関数の変動を振動としてとらえると、その振動の周波数に対する積分区間の幅の比を、信頼性の判断基準にすることができます。この比が大きければ、積分区間で変動が少ないと考えられるため、計算速度が速くなって計算結果の信頼性が増します。





被積分関数が積分区間のどの位置で大きく変動するのか、あらかじめ分かっていることが多いでしょう。変動の情報が無く、また、その変動が問題になると予想されるときは、プログラムや式の機能を使って、いくつかのサンプル点を与えて関数値を評価し、その関数のグラフ形状を予想してください。

どんな場合でも、積分を計算したらその結果の正当性を評価して下さい。正当性を簡単に評価するには、積分区間を複数に分割して、それぞれを新たな積分区間として積分し、得られた複数の積分値の総計を求めます。新たな積分区間により、サンプル点も新しく設定され、その前の計算では隠れてしまった変動が出てくるかもしれません。検証する積分の計算結果が妥当なものであれば、この手法で出力される値と同じになります。

計算時間が長くなる条件

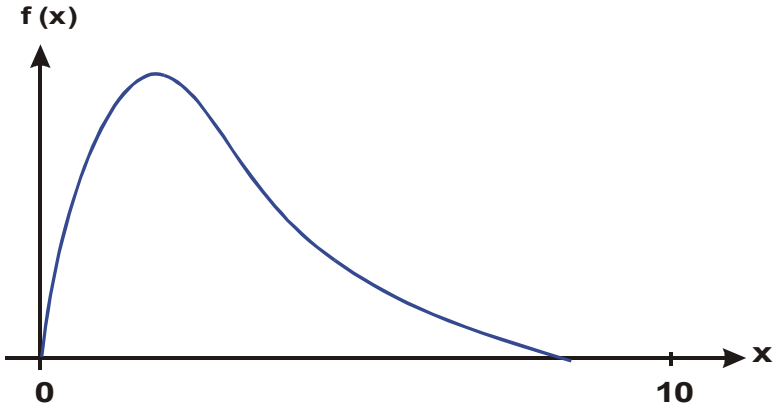
前の例では、関数の変動をとらえることができずに不正確な計算結果となっていました。これは関数値の変動が積分区間の幅に対して急激だったからです。もし積分区間の幅が狭ければ、妥当な計算結果になります。ただし、その幅がまだ大きいと長い計算時間が必要になります。

この例の積分区間はとても大きく、過大な計算時間を要しますが、計算結果は不正確でした。問題の関数 $f(x) = xe^{-x}$ では、x が ∞ に近づくと関数値が急速にゼロに近づきますから、大きな x の領域では、積分値に対する寄与度が無視できるほど小さくなります。よって、積分の上限の ∞ を 10^{499} ではなく、たとえば 10^3 で置き換えてみましょう。

前の例に戻って、新しい上限を設定します。

キー	表示	説明
0 ENTER 1 E 3	1E3_	新しい下限と上限
EQN	X×EXP(-X)	式モードを選択。 式の左端が表示されます。
↺ / X	INTEGRATING ∫ = 1.000E0	積分を実行 (計算には 1～2 分かかります)
x◂▸y	1.000E-3	不確定値

計算結果は妥当ですが、計算時間が長いです。この関数のグラフを、x = 0 と x = 10^3 の区間でプロットしたものと、x = 0 と x = 10 の区間でプロットしたもの（下図）とを比較してみると理解しやすいです。



グラフが示すとおり、この関数は、x が小さな値のときだけ変動します。x が大きな値になると関数値がゼロに漸近し、変動しなくなっていきます。

積分のアルゴリズムでは、計算で得られた複数の推定値の差が許容できるほど小さくなるまで、サンプル点を増加していきます。積分区域を、この関数が変動する領域だけにすることができれば、計算時間は少なくて済みます。

積分領域の広さに関わらずサンプル点の密度を同じにするためには、積分区域が広いときはサンプル点の数を増やさなければなりません。つまり積分区域が広い場合に同じ正確度の推定値を求めるときは、繰り返し計算の数が増えます。その結果、計算に時間がかかるようになります。

積分の計算時間は、関数値が変動する領域におけるサンプル点の密度をどれだけ素早く確定できるかに依存します。よって、積分区間に関数値の変動しない領域が多く含まれていれば、どんな関数でも計算時間が長くなります。そのような関数を計算するときは、計算時間がかからないように条件を調整することもできます。条件の調整には、積分区間を分割する手法と、変数を調整する手法があります。積分の上限と下限や式を調整することにより、積分区間における被積分関数の振る舞いを取り扱いしやすいものにする手法です。

F

メッセージ

特定の状態になったときや特定のキーを押したとき、メッセージが表示されます。メッセージに注目を促すため ⚠ 記号が表示されます。重要なメッセージはクリアされるまで表示され続けます。[C] や [←] キーでクリアすると、その前の表示に戻ります。その他のキーでもメッセージはクリアされますが、そのキーの機能は実行されません。

∫FN ACTIVE	実行中のプログラムが、積分の計算中の他のプログラムラベル(FN=ラベル名)を選択しようとしています。
∫(∫FN)	実行中のプログラムが、積分の計算中の他のプログラム(∫FN d 変数)をさらに積分しようとしています。
∫(SOLVE)	実行中のプログラムが、積分の計算中の他のプログラムをさらに選択しようとしています。
ALL VARS=0	変数カタログ ([↶] [MEM] [1] (1VAR)) に値が保存されていません。
BAD GUESS	ある式の変数について SOLVE を実行したとき、推定値に不正な数値(複素数やベクトル)が設定されています。
CALCULATING	計算時間がかかる機能を実行しています。
CLR ALL? Y N	メモリの全てをクリアするかどうかの確認
CLR EQN? Y N	編集中の式をクリアするかどうかの確認(式編集モードのみで登場します)
CLR PGMS? Y N	全てのプログラムをクリアするかどうかの確認(プログラム編集モードのみで登場します)
DIVIDE BY 0	ゼロで除算しています。(Y レジスタにゼロが保存されている状態で [↶] [%CHG] を押したときも表示されます)

DUPLICAT.LBL　すでに存在するプログラムラベルが、別のプログラムルーチンに対するものとして入力されました。

EQN LIST TOP　式リストの「トップ」を示しています。式リストは循環型であり、EQN LIST TOP は式リストの最後の式にも隣接しています。

INTEGRATING　プログラムまたは式を積分しています。多くの場合、しばらく時間がかかります。

INTERRUPTED　ALG、RPN、EQN、PGM モードで [C] または [R/S] が押されたことにより、実行中の計算、SOLVE、積分が中断されました。

INVALID DATA　データエラー:

- エラーのあるデータの保存または計算が実行されました。
- 組み合わせまたは順列の計算のとき、$r > n$ であるか、r や n が整数ではない、あるいは、$n \geq 10^{16}$ という条件になっています。
- 複素数やベクトルを統計データとして保存しようとしています。
- 基数 n における最大の数を超える数値を保存しようとしています。
- [x⇄y] ボタンが押されたことにより、統計レジスタに不正なデータが保存されようとしています。
- 複素数またはベクトルを比較しようとしています。
- 三角関数または逆送曲線関数の引数が不正です:
  - [TAN] で x が 90°の奇数倍の角度
  - [→] [ACOS] または [→] [ASIN] で、$x < -1$ または $x > 1$
  - [←] [HYP] [→] [ATAN] で、$x \leq -1$ または $x \geq 1$
  - [←] [HYP] [→] [ACOS] で、$x < 1$

INVALID VAR　式を SOLVE するとき、不正な変数名が指定されました。

INVALID x!　階乗やガンマ関数の計算のとき、x に負の整数が指定されています。

INVALID y^x	べき乗のエラー: ■ 底が0で、指数が0または負の数値になっている。 ■ 底が負の数値であるのに、指数が整数以外のとき。 ■ 底が複素数（$0 + i0$）、指数の実数部が負のとき。
INVALID (I)	不正な間接変数を使って計算しようとしています。 ((I)が定義されていません)
INVALID (J)	不正な間接変数を使って計算しようとしています。 ((J)が定義されていません)
LOG(0)	ゼロまたは $(0 + i0)$ の対数を計算しようとしています。
LOG(NEG)	負の数値の対数を計算しようとしています。
MEMORY CLEAR	すべてのユーザメモリが消去されました。 (B-3 ページをご参照下さい)
MEMORY FULL	その操作を行うためのメモリが不足しています。 (付録 B をご参照下さい)
NO	セルフテストでチェックされた状態が、真ではありません。 (キーボードから実行された場合のみ)
NONEXISTENT	GTO, XEQ, FN で、存在しないプログラムラベル(または行番号)が参照されました。次のどれかの場合、エラーメッセージ NONEXISTENTが表示されます。 ■ 明示的に（キーボードから）存在しないプログラムラベルを指定した場合。 ■ 呼び出したプログラムが他のラベルを参照していて、それが存在しない場合。 ■ 積分の結果が存在しない場合。
NO LABELS	プログラムカタログ(↩ MEM 2 (2PGM))にプログラムラベルが保存されていません。
NO SOLUTION	線形回帰が計算できませんでした。
MULT SOLUTION	線形回帰で複数の解が見つかりました。

NO ROOT FND	SOLVE(式モードとプログラムモードを含む)が、指定された初期推定値では式の根を発見できませんでした(参考: D-8 ページ)。 推定値が不正の場合、根が発見できない場合、右辺と左辺が等しくない場合もこのメッセージが出力されます。 プログラムの中でSOLVEが呼び出された場合は、このメッセージは出力されず、「SOLVE 変数」に続く行の処理に移ります。
OVERFLOW	瞬間的に表示される警告で、計算結果の数値が電卓で取り扱い可能な範囲を超えています。警告に続いて、設定済みの表示形式で ±9.99999999999E499 が表示されます。(1-16 ページ「数値の範囲とオーバーフロー」をご参照下さい) この状態になるとフラグ 6 がセットされます。 フラグ 5 がセットされているときにオーバーフローが発生すると、キーを押すまで実行中のプログラムが中断され、このメッセージが表示され続けます。
PRGM TOP	プログラムの「トップ」を示します。 プログラムメモリは循環型であり、PRGM TOP はプログラムの最終行にも隣接しています。
RUNNING B	式の計算やプログラムを実行しています。(SOLVE と積分ルーチン以外)
SELECT FN	「SOLVE 変数」または「∫FN d 変数」が実行されましたが、プログラムラベルが指定されいません。これは MEMORY CLEAR のメッセージが出た後の最初に SOLVE や ∫FN が実行されたとき、あるいは最後に指定したラベルがすでに存在しないときに発生します。
SOLVE ACTIVE	実行中のプログラムが、SOLVE のためにプログラムラベル (FN=ラベル名) を選択しました。
SOLVE(SOLVE)	実行中のプログラムが他のプログラムを SOLVE しています。
SOLVE(∫FN)	実行中のプログラムが他のプログラムを積分しています。
SOLVING	式やプログラムの根を計算しています。 多くの場合、しばらく時間がかかります。
SQRT(NEG)	負の数の平方根を計算しています。

STAT ERROR	統計計算のエラー。次の場合に表示されます。 ■ $n = 0$ で統計計算が実行されたとき ■ $n = 1$ で s_x s_y, $\hat{x}$, $\hat{y}$, m, r, b のいずれかが計算されたとき ■ x のデータしかないとき（つまり、全ての y のデータがゼロのとき）、r, $\hat{x}$, $\bar{x}w$ のいずれかが計算されたとき ■ 全ての x のデータが等しい状態で $\hat{x}$, $\hat{y}$, r, m, b のいずれかが計算されたとき
SYNTAX ERROR	式(Equatoin)、等号のない式（Expression）、[SOLVE]、[∫] の評価において文法エラーが見つかりました。このエラーメッセージを [←] または [C] でクリアし、問題点を修正して下さい。
TOO BIG	基数を HEX、OCT、BIN にするには与えられた数値が大きすぎます。次の範囲にして下さい。 $-34{,}359{,}738{,}368 \le n \le 34{,}359{,}738{,}367$.
XEQ OVERFLOW	実行中のプログラムが 21 番目の入れ子になった「XEQ ラベル名」を実行しました。（サブルーチンの入れ子は 20 までです） SOLVE と ∫FN ではラベルを参照しますので、このエラーは発生しません。
YES	セルフテストによってチェックされた状態が真です。 （キーボードからテストを実行したときのみ表示されます）

セルフテストのメッセージ

35S-OK	セルフテストとキーボードテストを通過しました。
35S-FAIL n	セルフテストまたはキーボードテストに失敗しました。技術サポートにご連絡下さい。
© 2007 HP DEV CO. L. P.	セルフテストが無事に完了したときに表示される著作権メッセージ。

G

操作もくじ

この章では全ての関数、操作、および入力形式の一覧を ABC 順で示します。関数名はプログラムで使うものと同じです。たとえば FIX *n* という名前の関数は ↑ DISPLAY 1 (1FIX) *n* で実行します。

プログラムで利用できない機能は、名前の欄にキーを表示しています。たとえば ← です。

英字以外の関数とギリシア文字の関数は、英字に変換されています。その場合は英字の前に矢印を添えています(たとえば →DEG)。

表の最後の列(∗)は、この章の最後の注意点をご参照下さい。

名前	キーと詳細	ページ	＊
[▽]	一覧で直後のものを表示。 式リストで次の式に移動。 プログラム入力中にプログラムポインタを次のステップに移動。 プログラムを入力していないときに現在のプログラム行を実行。	1-24 6-2 13-10 13-19	
[◁] と [▷]	カーソルを移動(削除はしません)。	1-11	
[↩][◁] と [↩][▷]	表示しきれない桁があるとき、ディスプレイの表示を移動。 2 進数や式の表示しきれない部分を表示。 CONST や SUMS メニューで次のページに移動。	1-10 6-3 11-7	
[↩] [△]	プログラムモードにおいて、式の先頭行や、プログラムの最後のラベルにおける先頭行に移動。	6-2	
[↩] [▽]	プログラムモードにおいて、式の終端行や、プログラムの最後のラベルにおける最終行に移動。	6-2	
,	[↰][,] 関数に対する複数の引数を区切ります。	6-5	1
1/x	[1/x] 逆数。	1-16	1
10^x	[↩] [10^x] 常用指数。 10 の x 乗を出力します。	4-2	1
%	[↩][%] 百分率。 $(y \times x) \div 100$ を出力します。	4-5	1
%CHG	[↰] [%CHG] 変化率。 $(x - y)(100 \div y)$ を出力します。	4-5	1
π	[↰] [π] π の近似値。 3.14159265359 (12 桁)を出力。	4-3	1
Σ+	[Σ+] 統計レジスタに(y, x)を保存。	12-2	
Σ–	[↰] [Σ–] 統計レジスタから(y, x)を削除。	12-2	

名前	キーと詳細	ページ	＊
Σx	[SHIFT] [SUMS] [▷] (Σx) x の総和。	12-10	1
Σx^2	[SHIFT] [SUMS] [▷] [▷] [▷] (Σx^2) x の平方の総和。	12-10	1
Σxy	[SHIFT] [SUMS] [▷] [▷] [▷] [▷] [▷] (Σxy) x と y の積の総和。	12-10	1
Σy	[SHIFT] [SUMS] [▷] [▷] (Σy) y の総和。	12-10	1
Σy^2	[SHIFT] [SUMS] [▷] [▷] [▷] [▷] (Σy^2) y の平方の総和。	12-10	1
σx	[SHIFT] [S,σ] [▷] [▷] (σx) x の母標準偏差。 $\sqrt{\sum (x_i - \bar{x})^2 \div n}$	12-6	1
σy	[SHIFT] [S,σ] [▷] [▷] [▷] (σy) y の母標準偏差。 $\sqrt{\sum (y_i - \bar{y})^2 \div n}$	12-6	1
∫FN d *変数*	[ALPHA] [∫] (∫FN d _) *変数* 表示中の式またはプログラムの FN= で選択された式を積分します。Y レジスタの値を下限、X レジスタの値を上限とします。	8-1 15-7	
( )	[( )] カッコ。 カッコの外側に続けて何か入力するときは [▷] キーを使います。	6-5	1
[ ]	[SHIFT] [[ ]] ベクトルを入力するための記号。	10-1	1
θ	[SHIFT] [θ] 複素数を入力するための記号。	9-1	1
A から Z	[RCL] *変数*	6-4	1

名前	キーと詳細	ページ	＊
	指定された変数の値を出力。		
ABS	[→] [ABS] 絶対値。 $\|x\|$ を出力。	4-16	1
ACOS	[→] [ACOS] 逆余弦。 $\cos^{-1}x$ を出力。	4-4	1
ACOSH	[←] [HYP] [→] [ACOS] 双曲線逆余弦。$\cosh^{-1} x$ を出力.	4-5	1
[MODE] [4] (4ALG)	ALG モードをアクティブにします。	1-9	
ALOG	[←] [10^x] 常用指数。 10 を底、引数を指数とします。	6-14	1
ALL	[←] [DISPLAY] [4] (4ALL) 有効桁を全て表示。全ての桁を表示させるには [→] [>] で右にスクロールしてください。	1-20	
AND	[←] [LOGIC] [1] (1AND) 論理演算子。	11-4	1
ARG	[←] [ARG] 複素数の偏角 "θ" を出力。	4-16	1
ASIN	[→] [ASIN] 逆正弦。 $\sin^{-1} x$ を出力。	4-4	1
ASINH	[←] [HYP] [→] [ASIN] 双曲線逆正弦。 $\sinh^{-1} x$ を出力。	4-5	1
ATAN	[→] [ATAN] 逆正接。 $\tan^{-1} x$ を出力。	4-4	1
ATANH	[←] [HYP] [→] [ATAN] 双曲線逆正接。 $\tanh^{-1} x$ を出力。	4-5	1
b	[←] [L.R.] [>] [>] [>] [>] (b) 線形回帰の *y* 切片を出力。 $\bar{y} - m\bar{x}$.	12-10	1
b	[→] [BASE] [8] (8b) 2 進数を入力。	11-4	1
[→] [BASE]	基数の変換メニューを表示。	11-1	

名前	キーと詳細	ページ	＊
BIN	[↗] [BASE] [4] (4BIN) 2 進数(base 2)を選択。	11-1	
[C]	電源オン。x をクリア。 メッセージやプロンプトをクリア。 一覧表示をキャンセル。 式の入力をキャンセル。 プログラムの入力をキャンセル。 式の実行を中断。 プログラムの実行を中断。	1-1 1-4 1-8 1-25 6-2 13-5 13-18	
/c	[↖] [/c]　分母の設定。 x を分母の最大値に設定します。 x = 1 のときは現在の設定を表示。	5-4	
→°C	[↗] [→°C]　華氏を摂氏に変換	4-13	1
CF n	[↖] [FLAGS] [2] (2CF) *n* フラグ *n* をクリア。(n = 0 から 11)	14-11	
[↗] [CLEAR]	数値またはメモリの一部を削除するためのメニューを表示。 指定された変数やプログラムを MEM カタログから削除。 表示中の式を削除。	1-5 1-25	
[↗] [CLEAR] [3] (3ALL)	保存されたデータ、式、プログラムを全て削除。	1-25	
[↗] [CLEAR] [3] (3PGM)	プログラムモードにおいて、全てのプログラムを削除。.	13-21	
[↗] [CLEAR] [3] (3EQN)	式モードにおいて全ての式を削除。	13-5	
CLΣ	[↗] [CLEAR] [4] (4Σ) 統計レジスタをクリア。	12-1	
CLVARS	[↗] [CLEAR] [2] (2VARS) 全ての変数をクリアし、ゼロに置き換えます。	3-6	
CLx	[↗] [CLEAR] [1] (1X) x (X レジスタ) をクリアし、ゼロに置き換えます。	2-3 2-7 13-7	

名前	キーと詳細	ページ	*
CLVARx	[→] [CLEAR] [6] (6CLVARx) 間接変数のうち、アドレスが x 以上のものをクリアし、ゼロに置き換えます。	1-4	
CLSTK	[→] [CLEAR] [5] (5STK) 全てのスタックをクリアし、ゼロに置き換えます。	2-7	
→CM	[→] [→cm] インチからセンチメートルへの変換	4-13	1
nCr	[↶] [nCr] 異なる *n* 個のものから順序を問わずに *r* 個を取り出したときの組合せ。 n! ÷ (r! (n – r)!) を出力します。	4-14	1
COS	[COS] 余弦。 cos x を出力します。	4-3	1
COSH	[↶] [HYP] [COS] 双曲線余弦。 cosh x を出力。	4-5	1
[↶] [CONST]	41 個の物理定数にアクセス。	4-7	
d	[→] [BASE] [5] (5d) 10 進数の入力記号。	11-1	1
DEC	[→] [BASE] [1] (1DEC) 10 進数入力モードを選択。	11-1	
DEG	[MODE] [1] (1DEG) 角度モードを度分法(Degree)に指定。	4-3	
→DEG	[→] [→DEG] ラジアンから度分法に変換。 (360/2π) x を出力。	4-12	1
[↶] [DISPLAY]	表示形式、区切り文字(. と ,)、桁区切り、複素数の表示形式に関するメニューを表示。	1-19	
DSE *変数*	[→] [DSE] *変数* 減少; 以下であればスキップ (*Decrement*, *Skip if Equal* or less) ループの制御値が *ccccccc.fffii* のとき、ccccccc（カウンター値）から *ii*（増	14-18	

名前	キーと詳細	ページ	＊
	減値）を減少させ、fff（最終値）を比較します。ccccccc－*ii* ≤ fff のときは次行をスキップします。		
E	指数の入力を開始し、直前の数値に"E" を追加します。続いて入力した数値の 10 を底とした指数になります。	1-14	1
ENG n	↶ DISPLAY 3 (3ENG) *n* 最初の数値に *n* 桁を足した工学表示を選択 (*n* = 0 から 11)。	1-19	
←ENG と ENG→	工学表記の指数表示を 3 桁ずつ移動。	1-19	
ENTER	連続入力する 2 つの数値を分割。 式の入力を完了。 表示中の式を評価し、結果が得られたら保存。	1-17 6-3 6-10	
ENTER	ENTER x を Y レジスタにコピー。*y* は Z レジスタに、z は T レジスタに移動します。*t* は消滅します。	2-5	
EQN	式入力モードの切り替え。	6-2 13-5	
e^x	↷ e^x *指数* *e* の *x* 乗を出力します。	4-1	1
EXP	↷ e^x *指数* *e* の指数関数。	6-14	1
→°F	↶ →°F 摂氏を華氏へ変換。	4-13	1
↷ FDISP	分数表示モードの切り替え。	5-1	
FIX n	DISPLAY 1 (1FIX) *n* 小数点以下を *n* に固定して表示。ただし、0 ≤ n ≤ 11 とします。	1-19	
↶ FLAGS	フラグの設定、クリア、テストのためのメニューを表示。	14-11	
FN = *ラベル*	↶ FN= *ラベル* ラベル付きプログラムを関数として選択。(SOLVE と ∫ FN で使います)	15-1 15-5	

名前	キーと詳細	ページ	＊
FP	◀ INTG 5 (5FP) x の小数部。	4-16	1
FS? *n*	◀ FLAGS 3 (3FS?) *n* フラグ *n* (*n*=0 から 11)がセットのとき次のプログラム行を実行。 *n* がクリアのときは次のプログラム行をスキップ。	14-11	
→GAL	◀ →gal リットルをガロンに変換	4-13	1
GRAD	MODE 3 (3GRD) 角度モードをグラジアンに設定。	4-3	
GTO . *ラベル* nnn	指定されたプログラムラベルの *nnn* 行にポインタを移動。	13-20	
GTO . .	プログラムポインタを PRGM TOP に移動。	13-20	
h	▶ BASE 6 (6h) 16 進数の入力記号。	11-1	1
HEX	▶ BASE 2 (2HEX) 16 進数モードを選択。	11-1	
◀ HYP	双曲線関数の指定のため HYP_を入力。	4-5	
→HMS	▶ →HMS x に小数で定義された時間を HMS（時間、分、秒）形式に変換。	4-12	1
HMS→	◀ HMS→ x に HMS（時間、分、秒）形式で定義された時間を小数に変換。	4-12	1
i	複素数の入力に使います。	9-2	1
(I)/(J)	RCL (I) / (J) または STO (I) / (J) 変数 I / J に保存された数値に対応する変数の値。	6-4 14-21	1
→IN	◀ →in センチメートルをインチに変換。	4-13	1
IDIV	◀ INTG 2 (2INT÷)	6-14	1

名前	キーと詳細	ページ	＊
	引数に与えた 2 つの整数に対する除算の商を出力。		
INT÷	[↰] [INTG] [2] (2INT÷) 2 つの整数に対する除算の商を出力。	4-2	1
INTG	[↰] [INTG] [4] (4INTG) 与えられた数値以下でもっとも大きい整数を出力。	4-16	1
INPUT *変数*	[↰] [INPUT] *変数* 変数を X レジスタに呼び出します。変数名とその値を表示し、実行中のプログラム実行を停止します。 [R/S]でプログラム実行を再開するか、[⌄]で現在のプログラム行を実行して入力した数値を変数に保存します。(この機能はプログラムのみで利用できます)	13-12	
INV	[1/x] 引数の逆数。	6-14	1
IP	[↰] [INTG] [6] (6IP)	4-16	1
ISG *変数*	[↰] [ISG] *変数* 増加; より大きければスキップ (*Increment, Skip if Greater*) ループの制御値が *ccccccc.fffii* のとき、ccccccc (カウンター値)から *ii* (増減値)を増加させ、fff(最終値)を比較します。ccccccc + *ii* > fff のときは次行をスキップします。	14-18	
→KG	[↱] [→kg] ポンドをキログラムに変換。	4-13	1
→KM	[↱] [→KM] マイルをキロメートルに変換。	4-13	1
→L	[↱] [→l] ガロンをリットルに変換。	4-13	1
LASTx	[↱] [LAST x] LAST X レジスタに保存された値を出力。	2-8	

名前	キーと詳細	ページ	＊
→LB	◀ [→lb] キログラムをポンドに変換。	4-13	1
LBL *ラベル*	▶ [LBL] *ラベル* XEQ、GTO、FN= で利用される、英字一文字のプログラムラベルを定義します。 (プログラムのみで使用します)	13-3	
LN	▶ [LN] *自然対数* $\log_e x$ を出力。	4-1	1
LOG	◀ [LOG] *常用対数* $\log_{10} x$ を出力。	4-1	1
◀ [L.R.]	線形回帰のメニューを表示。	12-4	
m	◀ [L.R.] [›] [›] [›] (m) 線形回帰で得られた直線の傾きを出力。 $[\Sigma(x_j-\bar{x})(y_j-\bar{y})]\div\Sigma(x_j-\bar{x})^2$	12-7	1
→MILE	◀ [→MILE] キロメートルをマイルに変換。	4-13	1
◀ [MEM]	利用可能なメモリ量とカタログメニューを表示。	1-24	
◀ [MEM] [2] (2PGM)	プログラムカタログの開始。	13-21	
◀ [MEM] [1] (1VAR)	変数カタログの開始。	3-4	
[MODE]	ALG モード、RPN モード角度モードの選択メニューを表示。	1-7 4-3	
n	▶ [SUMS] (n) データ点セットの総数を出力。	12-10	1

名前	キーと詳細	ページ	＊
NAND	[↰] [LOGIC] [5] (5NAND) 論理演算子。	11-4	1
NOR	[↰] [LOGIC] [6] (6XOR) 論理演算子。	11-4	1
NOT	[↰] [LOGIC] [4] (4NOT) 論理演算子。	11-4	1
o	[↱] [BASE] [7] (7o) 8 進数の入力記号。	11-4	1
OCT	[↱] [BASE] [3] (3OCT) 8 進数モードを選択。	11-1	
OR	[↰] [LOGIC] [3] (3OR) 論理演算子。	11-4	1
[↱] [OFF]	電源オフ。	1-1	
nPr	[↱] [nPr] 異なる *n* 個のものから順に *r* 個を取り出したときの順列。 *n*! ÷ (*n* – *r*)! を出力します。	4-14	1
[↱] [PRGM]	プログラム入力モードの切り替え。	13-5	
PSE	[↱] [PSE] 中断。 プログラムの実行を中断したうえで x や変数、または式を表示し、実行を再開します。 (プログラムでのみ使われます)	13-17 13-18	
r	[↰] [L.R.] [›] [›] (r) 次式に示す x と *y* の間の相関係数を出力。 $\frac{\sum(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum(x_i-\bar{x})^2\times(y_i-\bar{y})^2}}$	12-7	1
rθa	[↰] [DISPLAY] [1] [0] (10rθa) 複素数の表示形式を変更。	1-22	
RAD	[MODE] [1] (1RAD) 角度モードをラジアンに変更。	4-3	
→RAD	[↰] [→RAD]	4-12	1

名前	キーと詳細	ページ	＊
	度分法 (Degree) からラジアンに変換。(2π/360) x を出力。		
RADIX ,	[◱] [DISPLAY] [6] (6,) コンマ (,) を小数点記号に指定。	1-20	
RADIX .	[◱] [DISPLAY] [5] (5.) ピリオド (.) を小数点記号に指定。	1-20	
RANDOM	[◲] [RAND] RANDOM 関数を実行。 0 から 1 の間の乱数を出力します。	4-14	1
RCL *変数*	[RCL] *変数* 再呼び出し。 指定された変数を X レジスタにコピーします。	3-6	
RCL+ *変数*	[RCL] [+] *変数* 「x + 変数」を出力。	3-6	
RCL– *変数*	[RCL] [−] *変数* 「x – 変数」を出力。	3-6	
RCLx *変数*	[RCL] [×] *変数* 「x × 変数」を出力。	3-6	
RCL÷ *変数*	[RCL] [÷] *変数* 「x ÷ 変数」を出力。	3-6	
RMDR	[◱] [INTG] [3] (3Rmdr) 2 つの整数に対する除算の剰余を出力	6-14	1
RND	[◲] [RND] 丸め。 FIX *n* 表示モードのとき *x* を小数点以下 *n* 桁に丸めます。 SCI *n* または ENG *n* 表示モードのときは *n* + 1 を有効桁として丸めます。 分数表示モードでは表示中の分数に丸めます。	4-16 5-8	1
[RPN]	[MODE] [5] (5RPN) 逆ポーランド法への切り替え。	1-9	
RTN	[◱] [RTN] リターン。 プログラムの終了を定義します。プロ	13-3 14-1	

名前	キーと詳細	ページ	＊
	グラムポインタがトップに戻るかコールされたルーチンに戻ります。		
R↓	R↓ ロールダウン。 RPN モードにおいて *t* を Z レジスタに移動。z は Y レジスタに、*y* は X レジスタに、x は T レジスタに移動します。 ALG モードではスタックの X,Y,Z,T を閲覧するメニューを表示します。	2-3 C-7	
R↑	▶ R↓ ロールアップ。 RPN モードにおいて *t* を X レジスタに移動。z は T レジスタに、*y* は Z レジスタに、x は Y レジスタに移動します。 ALG モードではスタックの X,Y,Z,T を閲覧するメニューを表示します。	2-3 C-7	
▶ S,σ	標準偏差メニューを表示。	12-4	
SCI n	◀ DISPLAY 2 (2SCI) *n* 小数点以下 *n* 桁の科学表記モードを選択。(*n* = 0 から 11)	1-19	
SEED	◀ SEED シード値を X として乱数生成器をリセット。	4-14	
SF *n*	◀ FLAGS 1 (1SF) *n* フラグ *n* をセット。(*n* = 0 から 11)	14-11	
SGN	◀ INTG 1 (1SGN) x の符号を出力。	4-16	1
◀ SHOW	x（または現在のプログラム行の数値）の仮数（12 桁）を全て表示。 式やプログラムのチェックサムとバイト数を表示。	6-17 13-22	
SIN	SIN 正弦。sin x を出力。	4-4	1
SINH	◀ HYP SIN 双曲線正弦。sinh x を出力。	4-5	1
▶SOLVE *変数*	▶ SOLVE *変数*	7-1	

名前	キーと詳細	ページ	＊
	表示中の式または FN=で選択されたプログラムを SOLVE します。初期推定値は変数に指定された値と x を用います。	15-1	
[↗] [SPACE]	式中に空白文字を挿入。	14-8	1
SQ	[↗] [x^2]　引数の 2 乗。	6-14	1
SQRT	[$\sqrt{x}$]　x の平方根。	6-14	1
STO *変数*	[↗] [STO] *変数* x を変数に保存。	3-2	
STO +*変数*	[↗] [STO] [+] *変数* 変数 + x を変数に保存。	3-6	
STO –*変数*	[↗] [STO] [−] *変数* 変数 – x を変数に保存。	3-6	
STO ×*変数*	[↗] [STO] [×] *変数* 変数 × x を変数に保存。	3-6	
STO ÷*変数*	[↗] [STO] [÷] *変数* 変数 ÷ x を変数に保存。	3-6	
STOP	[R/S] 実行 / 停止 (*Run*/stop)。 現在のプログラム行からプログラムの実行を開始。 実行中のプログラムを停止し X レジスタを表示。	13-18	
[↗] [SUMS]	総和(summation)メニューを表示。	12-4 12-6	
sx	[↗] [S,σ] (Σ×) x の標本標準偏差を出力。 $\sqrt{\sum(x_i-\bar{x})^2 \div (n-1)}$	12-6	1
sy	[↗] [S,σ] [›] (Σy) y の標本標準偏差を出力。 $\sqrt{\sum(y_i-\bar{y})^2 \div (n-1)}$	12-6	1
TAN	[TAN]	4-4	1

名前	キーと詳細	ページ	＊
	正接。tan x を出力。		
TANH	[◀] [HYP] [TAN] 双曲線正接。 tanh x を出力。	4-5	1
VIEW *変数*	[◀] [VIEW] *変数* 変数のラベルと保存された数値をスタックに呼び出すことなく表示。	3-4 13-14	
[XEQ]	表示中の式を評価。	6-11	
XEQ *ラベル*	[XEQ] *ラベル* 指定されたラベルのプログラムを実行。	14-1	
x^2	[▶] [x^2] x の 2 乗。	4-2	1
$\sqrt{x}$	[$\sqrt{x}$] x の平方根。	4-2	1
$\sqrt[x]{y}$	[◀] [$\sqrt[x]{y}$] y の x 乗根。	4-2	1
$\bar{x}$	[◀] [$\bar{x},\bar{y}$] ($\bar{x}$) x の平均を出力。 $\Sigma x_i \div n$	12-4	1
$\hat{x}$	[◀] [L.R.] ($\hat{x}$) 線形回帰で得られた直線における、与えられた y（X レジスタから読み取ります）に対応する x の推定値。 $\hat{x} = (y - b) \div m$	12-7	1
!	[▶] [!] 階乗（ガンマ関数）。 (x)(x − 1) ... (2)(1) または $\Gamma(x + 1)$ を出力。	4-14	1
XROOT	[◀] [$\sqrt[x]{y}$] 引数 2 の引数 1 乗根。	6-14	1
$\bar{x}$ w	[◀] [$\bar{x},\bar{y}$] [▷] [▷] ($\bar{x}$ w) x の加重平均を出力。 $(\Sigma y_i x_i) \div \Sigma y_i$	12-4	1
[◀] [$\bar{x},\bar{y}$]	平均メニューを表示。	12-4	

名前	キーと詳細	ページ	＊
x<> *変数*	[↺] [x≶] x を交換。 x を指定された変数と交換。	3-8	
x<>y	[x◂▸y] x と y の交換。 x を Y レジスタに、y を X レジスタにそれぞれ移動。	2-4	
[↺] [x?y]	比較判定 "x?y" のメニューを表示。	14-7	
x≠y	[↺] [x?y] (≠) x≠y のとき次のプログラム行を実行。 x≠y のとき次のプログラム行をスキップ。	14-7	
x≤y?	[↺] [x?y] [›] (≤) x≤y のとき次のプログラム行を実行。 x>y のとき次のプログラム行をスキップ。	14-7	
x<y?	[↺] [x?y] [›] [›] (<) x<y のとき次のプログラム行を実行。 x≥y のとき次のプログラム行をスキップ。.	14-7	
x>y?	[↺] [x?y] [›] [›] [›] (>) x>y のとき次のプログラム行を実行。 x≤y のとき次のプログラム行をスキップ。	14-7	
x≥y?	[↺] [x?y] [›] [›] [›] [›] (≥) x≥y のとき次のプログラム行を実行。 x<y のとき次のプログラム行をスキップ。	14-7	
x=y?	[↺] [x?y] [›] [›] [›] [›] [›] (=) x=y のとき次のプログラム行を実行。 x≠y のとき次のプログラム行をスキップ。	14-7	
[↻] [x?0]	比較判定 "x?0" のメニューを表示。	14-7	

名前	キーと詳細	ページ	＊
x≠0?	◀ [x?0] (≠) x≠0 のとき次のプログラム行を実行。 x=0 のとき次のプログラム行をスキップ。	14-7	
x≤0?	◀ [x?0] [▶] (≤) x≤0 のとき次のプログラム行を実行。 x>0 のとき次のプログラム行をスキップ。	14-7	
x<0?	◀ [x?0] [▶] [▶] (<) x<0 のとき次のプログラム行を実行。 x≥0 のとき次のプログラム行をスキップ。	14-7	
x>0?	◀ [x?0] [▶] [▶] [▶] (>) x>0 のとき次のプログラム行を実行。 x≤0 のとき次のプログラム行をスキップ。	14-7	
x≥0?	◀ [x?0] [▶] [▶] [▶] [▶] (≥) x≥0 のとき次のプログラム行を実行。 x<0 のとき次のプログラム行をスキップ。	14-7	
x=0?	◀ [x?0] [▶] [▶] [▶] [▶] [▶] (=) x=0 のとき次のプログラム行を実行。 x≠0 のとき次のプログラム行をスキップ。	14-7	
XOR	◀ [LOGIC] [2] (2XOR) 論理演算子。	11-4	1
xiy	◀ [DISPLAY] [9] (9×i·y) 複素数の表示形式を変更。	4-10	
x+yi	◀ [DISPLAY] [·] [1] (11×+y·i) 複素数の表示形式を変更。 （ALG モードのみ）	1-22	
$\bar{y}$	◀ [x̄,ȳ] [▶] ($\bar{y}$) y の平均を出力。 $\Sigma y_i \div n$	12-4	1

名前	キーと詳細	ページ	＊
$\hat{y}$	[↩] [L.R.] [›] ($\hat{y}$) 線形回帰で得られた直線における、与えられた x(X レジスタから読み取ります)に対応する y の推定値。 $\hat{y} = m\,x + b$	12-7	1
y^x	[y^x] べき乗。 y の x 乗を出力します。	4-2	1

注意:

1. 式でも利用可能な機能

さくいん

M

O

P

R

S

T

V

X

あ

い

う

え

お

か

き

く

け

こ

さ

し

す

せ

そ

た

ち

て

と

な

に

は

ひ

ふ

へ

ほ

ま

め

も

よ

ら

り

る

れ

ろ

わ