

ツインテール de エンジェルモード !!

システム変数とシステム関数

(こすちゅーむ220~対応版)

目次

0 . はじめに.....	3
(a) 用語の説明 :	3
(b) 記号の説明 :	3
(c) 共通の性質 :	3
1 . システム変数の一覧.....	4
(a) 基本定数.....	4
(b) コマンドライン引数.....	4
(c) ファイルポインタ.....	4
(d) デフォルト変換書式.....	5
(e) 角度の単位.....	5
(f) タイムゾーンと時差.....	5
(g) 最大値と最小値.....	5
(h) 変数のビット数.....	6
(i) 数学定数.....	6
(j) プロセスと割り込み.....	6
2 . システム関数の一覧.....	7
(a) コマンドライン引数.....	7
(b) データ表示.....	7
(c) ファイルI/O.....	8
(d) 文字列操作.....	9
(e) データ操作.....	10
(f) 日付と時刻.....	11
(g) 数学関連.....	14
(h) プロセスと割り込み.....	16

0 . はじめに

(a) 用語の説明 :

- システム変数とは、システムによって事前に定義された **global変数** のことです。
- システム関数とは、システムによって事前に定義された **global関数** のことです。
→ システム変数とシステム関数は、その値 (内容) を自由に変更することができます。

(b) 記号の説明 :

- **S,I,D,P,X,A** → データ型を表します。
- **FILE** → ファイル名を表す文字列、又は、ファイルポインタ を表します。
- **/RE/** → 正規表現 を表す文字列、又は、コンパイル済み正規表現を表します。
- ***** → 任意のデータ型を表します。
- **[x]** → x が省略可能であることを表します。(配列記号とは文脈で区別します。)
- **a|b** → a 又は b が選択可能であることを表します。

(c) 共通の性質 :

全てのシステム関数には、次の共通した性質があります。

- 常に非破壊的に動作します。(入力データを破壊しません。)
- 常に全自動で動作します。(メモリ管理やファイル管理は、自動で行います。)
- エラー戻り値は **NULL** で統一されています。

1 . システム変数の一覧

(a) 基本定数

名前	型	意味	備考
NULL	P	ポインタ NULL	関数のエラー戻り値
TRUE	I	真の値 (実体は整数 1)	
FALS	I	偽の値 (実体は整数 0)	
VER	I	インタプリタのバージョン番号	= こすちゅーむ番号
PID	I	自プロセスIDの値	= \$\$

(b) コマンドライン引数

名前	型	意味	備考
argc	I	コマンドライン引数の数 (コマンド名も含む)	
argv	A	コマンドライン引数の配列 (コマンド名も含む)	
argr	I	現時点でshift()可能なコマンドライン引数の残数	
args	S	argv[1] からargv[argc-1]まで連結した文字列	
cmd	S	コマンド名の文字列	= argv[0]
env	A	環境変数の配列	

(c) ファイルポインタ

名前	型	意味	備考
stdin	P	標準入力	
stdout	P	標準出力	
stderr	P	標準エラー出力	
:code	P	コード領域アクセス用	= __code__
:data	P	データ領域アクセス用	= __data__

【NOTE】 :code 及び :data は、通常ファイルに記述されたスクリプト内でのみアクセス可能です。

(d) デフォルト変換書式

名前	型	意味	備考
FMT_S2S	S	文字列 →文字列 への変換書式	初期値 = "%s"
FMT_I2S	S	整数型 →文字列 への変換書式	初期値 = "%d"
FMT_D2S	S	実数型 →文字列 への変換書式	初期値 = "%+f"
FMT_P2S	S	ポインタ→文字列 への変換書式 (NULL以外)	初期値 = "%p"
FMT_N2S	S	NULL →文字列 への変換書式 (NULLのみ)	初期値 = "NULL"
FMT_SDATE	S	年月日の変換書式 [C言語 strftime(3) 互換]	初期値 = "%Y-%m-%d"
FMT_STIME	S	時分秒の変換書式 [C言語 strftime(3) 互換]	初期値 = "%H:%M:%S"

(e) 角度の単位

名前	型	意味	備考
SYS_ANGLE	D	システムの角度単位 (RAD 又は DEG)	初期値 = RAD

(f) タイムゾーンと時差

名前	型	意味	備考
UTC	S	協定世界時を表す "UTC" の文字列	
LTZ	S	ローカルタイムゾーンを表す文字列	日本の例: "JST"
LTZ_OFFSET	I	UTCを基準としたLTZ時刻の時差 (秒)	日本の例: +32400

【NOTE】LTZ と LTZ_OFFSET の初期値は、インタプリタ起動時に OS より自動設定します。LTZ_OFFSET の値を変更することにより、スクリプト内の時差を自由に設定することが出来ます。なお、この値を変更したとしても OS には影響を与えません。

(g) 最大値と最小値

名前	型	意味	備考
INT_MAX INT_MIN	I	整数の最大値 と 整数の最小値	正の値 と 負の値
DBL_MAX DBL_MIN	D	実数の最大値 と 実数の最小値	正の値 と 負の値
INT_QUANT	I	整数の正の最小値	
INT_EPSILON	I	整数のうち $X! = X+1$ となる正の最小値	イプシロン値
DBL_QUANT	D	実数の正の最小値	
DBL_EPSILON	D	実数のうち $X! = X+1.0$ となる正の最小値	イプシロン値

(h) 変数のビット数

名前	型	意味	備考
INT_SIZE	I	整数のビット数	例: 64 [bit]
DBL_SIZE	I	実数のビット数 [= DBL_MANTSIZE+DBL_EXPOSIZE+1]	例: 64 [bit]
PTR_SIZE	I	ポインタのビット数	例: 64 [bit]
DBL_MANTSIZE	I	実数の仮数部のビット数	例: 52 [bit]
DBL_EXPOSIZE	I	実数の指数部のビット数	例: 11 [bit]

(i) 数学定数

名前	型	意味	備考
INF	D	無限大	isinf() で検査可能
NAN	D	非数 (Not a Number)	isnan() で検査可能
M_E	D	自然対数の底	
M_PI	D	円周率	

(j) プロセスと割り込み

名前	型	意味	備考
\$\$	I	自プロセスIDの値	= PID
\$?	I	(直近) 外部コマンドの終了値	
SIG_DFL	P	割り込み動作設定用 (既定動作)	rxsig() で使用
SIG_IGN	P	割り込み動作設定用 (受信無視)	rxsig() で使用
SIG_ERR	P	割り込み設定時のエラー戻り値 [= NULL]	rxsig() で使用
NSIG	I	シグナル番号の定義数	0 ~ (NSIG-1) が有効

シグナル番号 (I) = SIGHUP / SIGINT / SIGQUIT / SIGILL / SIGABRT / SIGFPE / SIGKILL / SIGSEGV / SIGPIPE / SIGALRM / SIGTERM / SIGUSR1 / SIGUSR2 / SIGCHLD / SIGCONT / SIGSTOP / SIGTSTP / SIGTTIN / SIGTTOU /

2. システム関数の一覧

(a) コマンドライン引数

取得と返却

システム関数	説明
S = shift ([*])	コマンドライン引数を、呼び出し毎に argv[1] から順に取得する
S = opshift([*])	(//) ただし、対応する引数が option の場合にのみ取得する
I = unshift()	次の取得位置を 1 つ前に戻す (戻り値=取得位置)
取得するコマンドライン引数がない場合は、指定されたパラメータ * (又は、指定が無い場合 NULL) が戻り値となります。又、オプションとは '-' 文字で始まるコマンドライン引数のことです。	

(b) データ表示

簡易表示と詳細表示

システム関数	説明
void = p(*, ...)	変数、又は、式の値を簡易表示します。
void = d(*, ...)	変数、又は、式の値を詳細表示します。 ← ダンプ表示
x ← 変数名のみを記述した場合。	変数 x の値を簡易表示します。 ← p(x) に同じ
詳細表示では、識別 ID {スコープ種別(G=Global S=Static L=Local)+識別番号}、名前、データ型、データ属性、データ値、といった内部情報を表示します。	

PRINT関数： 書式付き出力

システム関数	説明
I = print (FMT,...)	標準 出力用の print 文です。
I = eprint(FMT,...)	標準エラー出力用の print 文です。
I = fprintf(FILE,FMT,...)	ファイル 出力用の print 文です。
S = sprintf(FMT,...)	文字列 出力用の print 文です。
void = dying(FMT,...)	この関数は { eprint(FMT,...); exit(1); } と等価です。
FMT は、C 言語 printf() 互換の出力書式文字列です。拡張機能として2進数出力 %b %B、及び、数値出力 (整数又は実数を、その型に応じて %d %f %e 又は %E を自動選択する) %v %V が指定できます。	

(c) ファイルI/O

GET/PUT/UNGET関数： 1文字or 1行の入出力

システム関数	説明
I = getc ([FILE]) S = gets ([FILE]) S = getn ([FILE, Max])	指定されたファイルから、1文字、又は、1行を読み込みます。 FILE 省略時は、標準入力を対象とします。
I = putc ([FILE,] Chr) I = puts ([FILE,] Str) I = putn ([FILE,] Str,Max)	指定されたファイルへ、 1文字、又は、1行を書き込みます。 FILE 省略時は、標準出力を対象とします。
I = ungetc([FILE,] Chr) I = ungets([FILE,] Str) I = ungetn([FILE,] Str,Max)	指定されたファイルへ、 1文字、又は、1行を戻します。 FILE 省略時は、標準入力を対象とします。
文字はChr(I)、文字列はStr(S)、最大文字数はMax(I)で指定します。	

READ/WRITE関数： バイナリーデータの入出力

システム関数	説明
(Ptr,Cnt) = read (FILE, Cnt)	指定バイト数の読み込み。
Cnt = write(FILE,Ptr,Cnt)	指定バイト数の書き込み。
Cnt(I)は、入出力の要求バイト数と入出力の実行バイト数です。 Ptr(P)は、入出力対象データの保存場所アドレスです。	

制御関数：

システム関数	説明
P = open (FILE,Mode)	ファイルの明示的オープンです。(通常は不要です。)
P = close (FILE)	ファイルの明示的クローズです。(通常は不要です。)
P = flush (FILE)	ファイルバッファのフラッシュです。(NULL=全ファイル)
I = getpos(FILE)	ファイル位置の取得です。
I = setpos(FILE,I)	ファイル位置の設定です。(絶対値指定) {先頭=00/末尾=-1}
I = movpos(FILE,I)	ファイル位置の移動です。(相対値指定) {進行=正/後退=負}
I = rewind(FILE)	ファイル先頭位置(00)への移動です。 [= setpos(FILE,0)]
Modelは、C 言語 fopen() 互換のモード文字列(S)です。	

(d) 文字列操作

文字列の比較

システム関数	説明
I = strcmp (S1,S2) I = strcmpi (S1,S2)	文字列 S1 と S2 の大小比較です。
I = strncmp (S1,S2,I) I = strncmpi (S1,S2,I)	文字列 S1 と S2 の大小比較です。(最大 I 文字まで)
戻り値は、C 言語互換です。又、i 付き関数は Ignore Case 版です。	

文字列の検索

システム関数	説明
I = strchr (S,Chr) strrchr (S,Chr)	文字 Chr の {順方向 逆方向} 検索です。
I = strstr (S,Str) strrstr (S,Str)	文字列 Str の {順方向 逆方向} 検索です。
(Is,Ie) = strreg (S,/RE/)	正規表現 /RE/ のマッチ位置検索です。{Is=始点、Ie=終点}
いずれの関数も、文字列 S 内の位置を戻り値とします。(先頭=0)	

文字列の置換

システム関数	説明
S = ssub (S,V1,V2) gsup (S,V1,V2)	文字列 S 内の V1 を V2 に {1回 全部} 置換します。
S = evalesc (S)	ESCシーケンスを順方向変換します。(例: "\n" → 0x0a)
S = rvalesc (S)	ESCシーケンスを逆方向変換します。(例: 0x0a → "\n")
* = uc (S I) lc (S I)	引数を {大文字化 小文字化} します。
パラメータ V1 には、文字、文字列、又は、正規表現が指定できます。 パラメータ V2 には、文字、文字列 が指定できます。	

改行等の削除

システム関数	説明
S = chop (S)	{ 行末 } の改行コードを 1組削除します。
S = hstrip (S) tstrip (S) strip (S)	{行頭 行末 両方} の空白コードを全部削除します。
改行コードは、Win/Mac/Unix 形式に対応します。空白コードは、isspace() で判定します。	

部分文字・部分文字列

システム関数	説明
I = subchr (S,Ic)	文字列 S の位置 Ic の文字を戻します。(= S[Ic])
I = substr (S,Is,Ie)	文字列 S の位置 Is~Ie の部分文字列を戻します。
位置の指定方法 (先頭から指定する場合) : 先頭 = 0, 1, 2, ... 位置の指定方法 (末尾から指定する場合) : 末尾 = -1,-2,-3, ...	

文字列の配列化

システム関数	説明
A = split (S,V [,I])	文字列 S を V で区切った各要素で配列化します。
A = psplit (S,V [,I])	(//) ただし、空要素はスキップします。(Packed形式)
A = scan (S,V [,I])	文字列 S を V でマッチさせた各要素で配列化します。
A = pscan (S,V [,I])	(//) ただし、空要素はスキップします。(Packed形式)
いずれの関数も、パラメータ V には、文字、文字列、又は、正規表現が指定できます。 戻り値 A は、新規に生成された 1次元配列です。(添字は 0~) 第 3パラメータ(I)指定時は、その添字値に対応する要素(S)のみを戻します。	

(e) データ操作

データ変換

システム関数	説明
I = int (*)	パラメータを整数値化します。【別名 atoi()】
D = dbl (*)	パラメータを実数値化します。【別名 atof()】
I D = atov (*)	パラメータを数値化します。(戻り値の型は自動判定)
S = str (*)	パラメータを文字列化します。
P = ptr (*)	パラメータをポインタ化します。
I = bin(S) oct(S) dec(S) hex(S)	文字列 S を {2 8 10 16}進数と見なして整数化します。
/RE/ = reg(S)	文字列 S の正規表現をコンパイルします。

データ判定

システム関数	説明
I = isalnum(I) isalpha(I) isascii(I) isblank(I) iscntrl(I) isdigit(I) isgraph(I) islower(I) isprint(I) ispunct(I) isspace(I) isupper(I) isxdigit(I)	C 言語互換の文字種別判定関数です。(TRUE/FALS)
I = isdef (*)	パラメータが定義済かどうかを判定します。(TRUE/FALS)
I = type (*)	パラメータのデータ型を取得します。(戻り値は、データ型を表す 1文字です。'U' 'S' 'I' 'D' 'P' 'X' 'A')

データ消去

システム関数	説明
U = erase (*)	定義済み変数を消去します。(未定義化)

データ複製

システム関数	説明
* = dup(*)	パラメータを複製します。なお、配列又は（静的変数を持つ）関数以外では、代入文と同じ効果となります。

サイズ計測

システム関数	説明
I = size(*)	パラメータ*のサイズを求めます。
I = strlen(S)	文字列 S の長さを求めます。(バイト数)
関数 size() の戻り値は、文字列→長さ(バイト数)、整数型&実数型&ポインタ→表現ビット数、関数型→演算ノード数、配列型→要素数となります。	

ソート実行

システム関数	説明
A = sort([X,] V, ...)	パラメータ V を昇順にソートします(戻り値=ソート済み配列)
A = rsort([X,] V, ...)	パラメータ V を降順にソートします(戻り値=ソート済み配列)
比較関数 X を指定しない場合は、パラメータの単純ソートとなります。この場合、パラメータ V は、{文字列 整数 実数} 又はそれらを含む配列が指定できます。比較関数 X を指定した場合は、比較関数による一般ソートとなります。この場合、パラメータ V は自由に設定出来ます。なお、比較関数 X は、2 つの引数を取り比較結果を数値の符号で戻す関数となります。(C言語 qsort(3) の比較関数互換。)	

配列関連

システム関数	説明
Ao = vals(Ai)	配列 Ai の各要素値を要素とする、1次元配列 Ao を生成します
Ao = keys(Ai)	配列 Ai の各添字値を要素とする、1次元配列 Ao を生成します
I = ndim(Ai)	(ハッシュ要素を含まない) 配列 Ai の次元数を求めます
Ao = shape(Ai)	(ハッシュ要素を含まない) 配列 Ai の形状を求めます

(f) 日付と時刻

【参考】
・UNIX時刻とは、1970-01-01 00:00:00 からの経過時刻 [単位=実数秒] のことです。
・UTCとは、協定世界時のことです。すなわち、 世界時間 のことです。
・LTZとは、ローカルタイムゾーン設定に基づく 地方時間 のことです。(例：日本標準時)

現在時刻 (=UNIX時刻) の取得

システム関数	説明
D = time()	現在の UNIX時刻 を取得します。【単位=実数秒】
UNIX時刻は、国や地域などに依存しない世界共通の時刻となります。(時差調整や夏時間調整が存在しません。) 又、単一の数値で表現できる利便性もあるため、 ツインテールdeエンジェルモード!! に含まれる【日付と時刻】の処理関数の多くは、このUNIX時刻を処理対象としています。	

プロセス時刻の取得

システム関数	説明
A = times()	Process時間, UsrCPU時間, SysCPU時間を取得します。
戻り値 A は構造体です。メンバー名と設定値の意味は以下の通りです。【単位=実数秒】 A.ptime = スクリプト開始 ~ 関数 times() 実行時点までの経過時間 (Process時間) A.utime = Process時間のうちユーザーモードの実行時間 (UsrCPU 時間) A.stime = Process時間のうちカーネルモードの実行時間 (SysCPU 時間)	

UNIX時刻の操作 (分解と合成)

システム関数	説明
(year, mon, day[, hour, min, sec]) = t2ymd(D)	UNIX時刻(D)から、UTC 年月日時分秒を求めます。
(year, mon, day[, hour, min, sec]) = t2lymd(D)	UNIX時刻(D)から、LTZ 年月日時分秒を求めます。
D = ymd2t(year, mon, day[, hour, min, sec])	UTC 年月日時分秒から、UNIX時刻(D)を求めます。
D = lymd2t(year, mon, day[, hour, min, sec])	LTZ 年月日時分秒から、UNIX時刻(D)を求めます。
上記4関数において、{year mon day hour min} は整数型、{sec}は実数型です。関数 ymd2t() 及び lymd2t() では、年月日[時分秒]を表す1つの文字列でも日時指定が可能です。	

時刻文字列の操作

システム関数	説明
S = sdate(D) ldate(D)	UNIX時刻(D) から "YYYY-MM-DD" {UTC LTZ} 文字列を生成します。(*1)
S = stime(D) ltime(D)	UNIX時刻(D) から "hh:mm:ss" {UTC LTZ} 文字列を生成します。(*1)
S = sftime(FMT,D) lsftime(FMT,D)	UNIX時刻(D) から {UTC LTZ}で表現された時刻文字列(S) を生成します。(*2)
D = sptime(FMT,S) lspitime(FMT,S)	{UTC LTZ}で表現された時刻文字列(S) から UNIX時刻(D) を生成します。(*3)
(*1) 生成される文字列は、システム変数 {FMT_SDATE FMT_STIME} で変更可能です。 (*2) FMT は、変換書式の文字列です。C言語 strftime(3) 関数の変換書式と同一です。 (*3) FMT は、変換書式の文字列です。C言語.strptime(3) 関数の変換書式と同一です。	

うるう年の判定

システム関数	説明
I = isleap(I)	西暦年(I)のうるう年判定をします。(戻り値=TRUE FALS)

時刻構造体の操作

システム関数	説明
$A = t2utc(D) \mid t2ltz(D)$	UNIX時刻(D) から、{UTC LTZ} 時刻構造体 を求めます。
$D = utc2t(A) \mid ltz2t(A)$	{UTC LTZ} 時刻構造体(A) から、UNIX時刻(D) を求めます。
$A = utc2ltz(A)$	UTC 時刻構造体(A) から、LTZ 時刻構造体(A) を求めます。
$A = ltz2utc(A)$	LTZ 時刻構造体(A) から、UTC 時刻構造体(A) を求めます。
$A = tm_norm(A)$	時刻構造体を正規化します。(例: 3時65分→4時05分)

時刻構造体の書式 (フォーマット)

時刻構造体のメンバー	有効性	型	備考
t.year = 西暦年 4桁	○	I	
t.mon = 月 (01~12)	○	I	
t.day = 日 (01~31)	○	I	
t.hour = 時 (00~23)	○	I	
t.min = 分 (00~59)	○	I	
t.sec = 秒 (00~60)	○	D	実数型
t.wday = 曜日 (00~06)	△	I	日曜=0、月曜=1 ~ 土曜=6
t.yday = 年日 (0~365)	△	I	正月=0、大晦日=364 又は 365
t.isdst = 夏時間 (TRUE/FALS)	○	I	

注意 1) ○の項目は、常に有効かつ常に必須です。
 注意 2) △の項目は、入力時には設定不要 (未利用) ですが、出力時には利用可能となります。

(g) 数学関連

実数の処理

システム関数	説明
$D = \text{ceil}(D) \mid \text{floor}(D) \mid \text{trunc}(D)$	実数の丸め込み {切り上げ 切り下げ 0の方向}
$D = \text{rint}(D) \mid \text{round}(D)$	実数の丸め込み {偶数丸め 四捨五入}
$D = \text{abs}(D)$	絶対値
$(Dman, Iexp) = \text{frexp}(D)$	実数 D から {仮数=Dman & 指数=Iexp} への分解
$D = \text{ldexp}(Dman, Iexp)$	{仮数=Dman & 指数=Iexp} から実数 D への合成

指数と対数

システム関数	説明
$D = \log(D) \mid \log_2(D) \mid \log_{10}(D)$	{底 e 底 2 底 10} の対数
$D = \exp(D) \mid \exp_2(D) \mid \exp_{10}(D)$	{ e 2 10 } の累乗
$D = \text{pow}(Dx, Dy) \mid \text{mod}(Dx, Dy)$	DxのDy乗 Dx / Dyの余剰
$D = \text{sqrt}(D) \mid \text{cbrt}(D)$	平方根 立方根
$D = \text{lin}(D) \mid \text{dbi}(D)$	リニア値 デシベル値

角度の単位

システム関数	説明
$D = \text{rad2deg}(D) \mid \text{deg}(D)$	Rad → Deg 変換
$D = \text{deg2rad}(D) \mid \text{rad}(D)$	Deg → Rad 変換
$(\text{Deg}, \text{Min}, \text{Sec}) = \text{rad2dms}(D)$	Rad → 度分秒 変換
$(\text{Deg}, \text{Min}, \text{Sec}) = \text{deg2dms}(D)$	Deg → 度分秒 変換
$D = \text{dms2rad}(\text{Deg}, \text{Min}, \text{Sec})$	度分秒 → Rad 変換
$D = \text{dms2deg}(\text{Deg}, \text{Min}, \text{Sec})$	度分秒 → Deg 変換

三角関数

システム関数	説明
$D = \sin(D) \mid \cos(D) \mid \tan(D)$	{正弦 余弦 正接} 関数
$D = \text{asin}(D) \mid \text{acos}(D) \mid \text{atan}(D)$	{正弦 余弦 正接} 逆関数
$D = \text{atan2}(Y, X)$	{正接} 逆関数 (2変数指定)

角度単位の初期値は rad です。(システム変数 SYS_ANGLE で変更できます。)

双曲関数

システム関数	説明
D = sinh(D) cosh(D) tanh(D)	双曲線 { 正弦 余弦 正接 } 関数
D = asinh(D) acosh(D) atanh(D)	双曲線 { 正弦 余弦 正接 } 逆関数

ベッセル関数

システム関数	説明
D = j0(D) j1(D) jn(D,D)	第一種ベッセル関数
D = y0(D) y1(D) yn(D,D)	第二種ベッセル関数

座標変換

システム関数	説明
(X,Y,Z) = xrot(x,y,z,θ)	X座標軸を +θ 回転した後の座標を求める
(X,Y,Z) = yrot(x,y,z,θ)	Y座標軸を +θ 回転した後の座標を求める
(X,Y,Z) = zrot(x,y,z,θ)	Z座標軸を +θ 回転した後の座標を求める
角度単位の初期値は rad です。(システム変数 SYS_ANGLE で変更できます。)	

統計関数

システム関数	説明
D = max(V) med(V) min(V)	{ 最大値 中央値 最小値 } を求める
I = argmax(V) argmin(V)	{ 最大値 最小値 } のインデックスを求める
(Imin,Imax) = argmed(V)	中央値のインデックスを求める (戻り値は2つ。)
D = sum (V) ave (V)	{ 合計値 平均値 } を求める
D = svar(V) uvar(V)	標本分散 (Sample) 不偏分散 (Unbias) を求める
D = sdev(V) udev(V)	標本標準偏差(Sample) 不偏標準偏差(Unbias) を求める
D = lgamma(D) tgamma(D)	{ loge True } ガンマ関数
D = erf(D) erfc(D)	{ 誤差 余誤差 } 関数

パラメーター V は { 整数、実数、又は、それらの配列 } の任意並びです。
関数 argmed() の戻り値は、引数が奇数個の時 Imin=Imax、偶数の時 Imin!=Imax となります。
標本分散と標本標準偏差は 1/N の式、不偏分散と不偏標準偏差は 1/(N-1) の式です。

乱数発生

システム関数	説明
I = rand(MAX)	0 <= x < MAX(I) の整数乱数を発生します。(0以上MAX未満)
D = urand()	0.0 < x < 1.0 の一様乱数を発生します。
D = nrand([AVE,DEV])	正規乱数を発生します。{ 平均値=AVE(D) / 標準偏差=DEV(D) }
void = srand(I)	乱数のシードを手動設定します。(デフォルトでは、自動設定)
関数 nrand() の引数を省略した場合は、標準正規分布 (AVE=0.0 / DEV=1.0) となります。	

無限大と非数

システム関数	説明
I = isinf(D)	無限大かどうか判定します。(戻り値=TRUE/FALS)
I = isnan(D)	非数かどうか判定します。(戻り値=TRUE/FALS)

(h) プロセスと割り込み

プロセス関数

システム関数	説明
I = pid() ppid()	{ 自 親 } プロセス ID を取得します。
I = sleep (I)	スリープ状態に I [秒] 入ります。(戻り値=残時間[秒])
void = pause()	ポーズ 状態に入ります。
void = exit(I)	スクリプトを終了値 I で終了します。

外部コマンド

システム関数	説明
I = system(S)	外部コマンド S を実行します。(戻り値=コマンドの終了値)
So = `S` [< Si]	外部コマンド S を実行します。(戻り値=標準出力)
記号 ` ` にて実行する場合は、オプションで標準入力 Si(S) の指定が出来ます。又、標準出力は戻り値 So(S) としてキャプチャされます。(標準エラー出力はキャプチャされません。)なお、終了値は \$? にセットされます。	

割り込み関数

システム関数	説明
I = txsig(Isig,Ipid)	シグナル (Isig番) をプロセス (Ipid番) に送信します。
I = rxsig(Isig,Func)	シグナル (Isig番) 受信時の処理関数を登録します。
Func() は、1つの引数を持つ任意のユーザー関数です。シグナル受信時に割り込み実行されます。仮引数には受信したシグナル番号がセットされます。(仮引数名は任意です。)なお、Func() の代わりに {SIG_DFL SIG_IGN} を指定すると、シグナル受信時の動作が、それぞれ {既定動作 受信無視} となります。	

ガーベージコレクション

システム関数	説明
void = gc_collect(void)	ガーベージコレクションを明示的に実行します。
通常は、自動でメモリ管理がされていますので、当関数を明示的に実行する必要はありません。しかし、大量のデータを使用 & 破棄する場合などにおいて、適切なタイミングで当関数を実行することによって、メモリ使用量を削減させることができます。	