

PC - 9 8 0 1 MS / F O R T R A N ( M S - D O S ) 用

## C R T 画 面 制 御 ラ イ ブ ラ リ

### 【 注 意 】

数値演算プロセッサが必要である。

MS-DOSはVer.3.0以上を使用のこと。

拡張グラフィックモードで動作する。

1988年 4月 15日

1990年 3月 10日

1990年 12月 30日

X線撮影系々解析研究会

## はじめに

科学技術計算というのは、理論的な計算や実験の結果をまとめるときに行なう。したがって、計算機に求められる機能というのは、基本的には

1. データの入力ができる。
2. 数式に基づいて計算ができる。
3. 計算結果を出力できる。

の3つである。

データの入力方法には、キーボードからの入力だけでなく、GP-IB等のインターフェイスを通して、外部の計測器から入力する必要がある場合もある。また、計算結果の出力形式にもいろいろある。ただ単に数値を羅列する方法。視覚的に理解しやすいように、結果を曲線としてグラフ化し、CRT上に描いたりX-Yプロッタに出力することも必要になってくる。また、画像を得るのが目的であるなら処理画像なりを表示することも必要となってくる。

このライブラリは、測定値など数値データの入力をキーボードから行なう方法と、測定とか計算結果の数値を表示したり、曲線というグラフで表わすための基本的な道具を提供するものである。

世間一般では、グラフというと円グラフや棒グラフを想像するが、科学技術計算の結果としてのグラフは主に曲線である。つまり、このライブラリでは

「画像を表示、制御する」

のではなく、

「ペンを使って線を引く」

という考え方でサブルーチンを構成した。つまりPC-9801のCRT画面をグラフ用紙の代わりに利用しようというわけである。

このライブラリは後発のプロッタ制御のための基本的なものとして開発した。

著 者

MS-DOSはマイクロソフト社の登録商標です。  
MS/FORTRANはマイクロソフト社の商標です。

## C R T 画 面 制 御 ラ イ ブ ラ リ で 扱 う 座 標

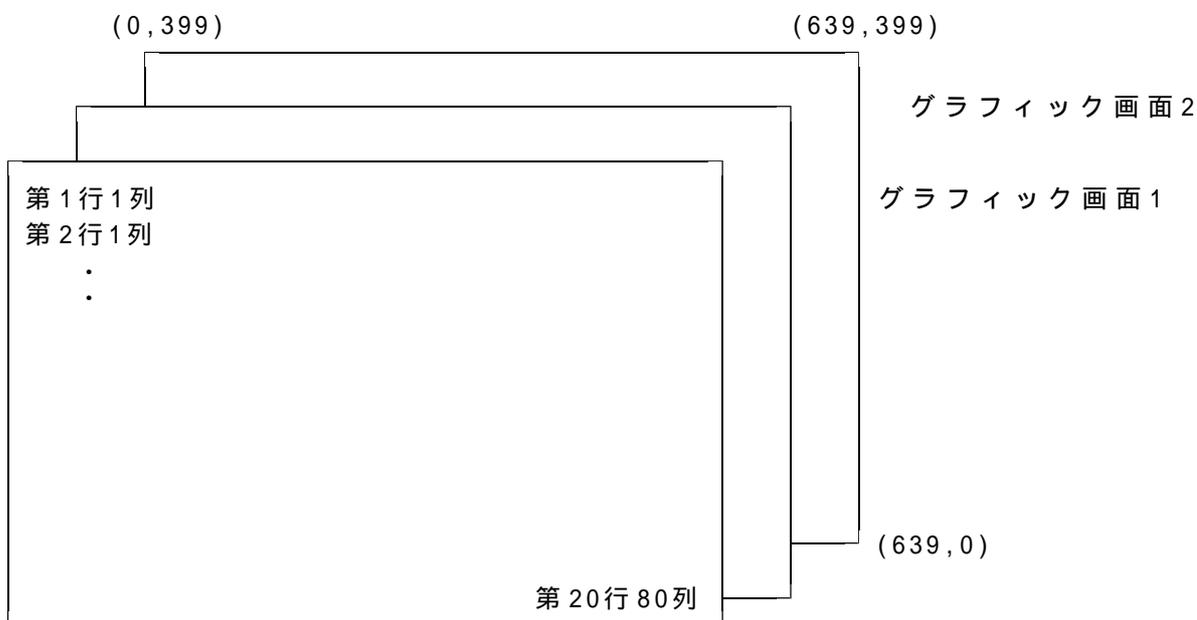
本ライブラリの初期化サブルーチンTGINITを実行すると、PC-9801の次の3つの画面が使えるようになる。

テキスト画面	20行80列	8色表示
グラフィックス画面1	X座標0～639,Y座標0～399	16色表示
グラフィックス画面2	X座標0～639,Y座標0～399	16色表示

グラフィック画面の座標値は画面のドットの位置であって、整数値に限る。中間の座標は存在しない。

漢字を表示した際の画面の見やすさを考慮して、テキスト画面は20行モードに固定してある。

CRT画面制御ライブラリでは、テキスト画面は第何行、何列という言い方を使用しているので注意(一般にはX座標、Y座標の用語を使用している)。



テキスト画面

ファンクションキーは表示しない

名 称	内 容	頁
CHDIR	カレント・ディレクトリの変更	8
CHILD	指定パス名のプログラムを実行	9
CHK232C	R S - 2 3 2 C のステータスを取得	10
CLEN	文字列の実際の長さ	11
CLOSEH	ファイルのクローズ	12
CREATH	ファイルの新規作成	13
DEGIMAL2	1 6 進数文字を 2 バイトの 1 0 進数に変換	15
DEGIMAL4	1 6 進数文字を 4 バイトの 1 0 進数に変換	16
DELET	ファイルの削除	17
DISKSPC	ディスクの空き容量を取得	18
EXEDIR	プログラムの存在ディレクトリを取得	19
FLIST	指定パス名のファイルを検索	20
GBOXF	矩形の塗りつぶし	22
GBOXFFST	高速矩形塗りつぶし	23
GBOXL	矩形の描画	24
GBOXLFST	高速矩形描画	25
GCIRCLE	円の描画	26
GCIRCLFN	扇形の描画	27
GCLR	画面の消去	29
GCURSW	十字カーソルの表示制御	30
GDRAW	直線の描画	31
GDRAWFST	高速直線描画	32
GETADR	変数の格納アドレスの取得	33
GETCD	カレント・ディスク番号の取得	34
GETDIR	カレント・ディレクトリの取得	39
GETDRV	接続ドライブ数の取得	36
GETLIN	ファイルからの文字列読み出し	37
GETOPTION	コマンドラインの文字列を取得	39
GETSEC	システム時刻を秒数で取得	40

名 称	内 容	頁
GGET	文字列配列へ図形情報を記憶	41
GLINSTL	線の種類の設定	43
GLPSETF	高速ライン P S E T	45
GMOVE	ペンを指定ドット座標値へ移動	47
GPAINT	図形内部の塗りつぶし	48
GPEN	ペン（パレット番号）の選択	49
GPENPOS	ペンの現在位置	51
GPSET	任意のドット座標値に点を打つ	52
GPSETF	高速 P S E T	54
GPUT	文字列配列内の図形情報を描画	55
GSETRGB	ペンの色を設定	56
GSYMBL	グラフィックス画面に文字を表示	60
GVIEW	ビューポートの設定	62
GVIEWCLR	ビューポート内を消去	63
INL232C	R S - 2 3 2 C からの文字列データ受信	64
INS232C	R S - 2 3 2 C からのデータ受信	65
KEYBUF	キーコードバッファの検査	66
KEYZERO	キーコードバッファを空にする	67
KINPUT	押下キーのコードを取得	68
KREAL	押下キーのコードをリアルタイムで取得	70
LPRINT	プリンタへの文字列出力	72
LSEEK	ファイル・ポインタの移動	73
MKDIR	ディレクトリの新規作成	75
MSMOVE	マウスカーソルを指定座標へ移動	77
MSPOS	マウスカーソルの現在位置を取得	78
MSSPEED	マウスの移動感度を設定	79
MSVIEW	マウスカーソルの移動範囲を設定	80
OPENH	ファイルのオープン	81
OUTL232C	R S - 2 3 2 C への文字列データ送信	82
OUTS232C	R S - 2 3 2 C へのデータ送信	83
PEEKB	メモリから 1 バイトデータを読み出す	84
PEEKW	メモリから 2 バイトデータを読み出す	85

名 称	内 容	頁
POKEB	メモリに1バイトデータを書き込む	86
POKEW	メモリに2バイトデータを書き込む	87
PORTIN	I/Oポートから1バイトデータを読み出す	88
PORTOUTB	I/Oポートに1バイトデータを書き込む	89
PORTOUTW	I/Oポートに2バイトデータを書き込む	90
PRINTER	文字列の出力先を指定	91
PUTLIN	ファイルへの文字列書き込み	92
PUTZERO	数文字における1桁目の処理	94
RENAME	ファイル名の変更	95
RMDIR	ディレクトリの削除	96
SCREEN	グラフィックス画面の指定	97
SET232C	R S - 2 3 2 C インターフェイスの初期化	98
STOPSW	S T O P キーの無効/有効スイッチ	99
TBEEP	指定時間だけブザーを鳴らす	100
TCHMOD	テキスト画面上の文字の色と属性を変更	101
TCOLOR	テキスト画面の文字の色と属性を設定	103
TCURMOV	テキストカーソルに関する各種制御	105
TCURPOS	テキストカーソルの現在位置を取得	106
TCURSW	テキストカーソルの消去/表示	107
TDATA1	1次元データの入力	108
TDATAN	任意次元データの入力	112
TDISPM	テキスト画面の指定座標から文字列表示	116
TDISPV	テキスト画面の指定座標から数値を表示	117
TESTEND	プログラム終了用ユーティリティ	118
TGCOPY	C R T 画面のハードコピー	119
TGCOPIES	C R T 画面ハードコピーの出力先を指定	120
TGEND	ライブラリの使用終了	121
TGETCOLR	テキスト画面の文字の色と属性を取得	122
TGETFORM	F O R M A T 文字列から桁数を算出	123
TGETPRT	文字列の出力先を取得	124
TGETRANG	テキスト画面のスクロール範囲を取得	125
TGINIT	ライブラリの初期化	126

名 称	内 容	頁
TLOCATE	テキスト画カーソルを指定位置へ移動	127
TPRINT	文字列の表示	128
TRANGE	テキスト画面のスクロール範囲を設定	132
TREAD	テキスト画面の任意位置から文字列入力	134
TREADI2	テキスト画面の任意位置から2バイト整数入力	137
TREADI4	テキスト画面の任意位置から4バイト整数入力	138
TREADR4	テキスト画面の任意位置から4バイト実数入力	139
TREADR8	テキスト画面の任意位置から8バイト実数入力	141
TSETCHR	指定コードの文字をテキスト画面に表示	142
TSETPRT	外部プリンタの初期化と漢字ピッチ設定	145
TWAIT	指定時間だけプログラムの実行を中断	146
TWINDSCR	文字列配列のスクロール表示	147
TXTGET	テキスト画面情報を文字列配列に格納	149
TXTPUT	文字列配列の情報をテキスト画面に表示	150
TYESNO	Y E S / N O 返答用ユーティリティ	151

CALL CHDIR ( PATH, IER )

[ 機能説明 ]

PATH名で指定するディレクトリを、カレント・ディレクトリとする。

[ 引 数 ]

PATH	文字列	ディレクトリ名。50字以内とする。 A:¥ABC¥TESTのように指定する。	入 力
IER	4B整数	エラー番号。 0.正常終了。 3.ディレクトリ名が無効。	出 力

[ 備 考 ]

サブルーチンGETDIRのテストプログラムTEST009でこのCHDIRを用いている。

CALL CHILD ( PATH, IER )

[ 機能説明 ]

指定パス名のプログラムを磁気ディスクからメモリにロードして実行する。プログラム終了後、このプログラムはメモリ上には残らない。

[ 引 数 ]

PATH	文字列	プログラム名。拡張子も付けること。 ディレクトリ名も含めて50字以内とする。	入 力
IER	4B整数	エラー番号。 0.読み込み成功 2.ファイルが存在しない 8.メモリ不足	出 力

[ 注 意 ]

プログラムをロードして実行するのに十分な空きメモリがあること。

市販のプログラムの中には、このサブルーチンを使用して実行出来ないものもあるので注意。

CALL CHK232C ( ICH,IST )

## [ 機能説明 ]

指定したチャンネルのRS232Cのステータスを取得する。

## [ 引 数 ]

ICH	4B整数	チャンネル番号。1,2または3。	入 力
IST	4B整数	ISTの各ビットは次の意味をもつ。 ビット0 0のとき受信キャリア検出。 ビット1 RS232CのCTS線。0のとき送信可能。 ビット2 送信バッファ空。1のとき送信可能。 ビット3 1のとき受信パリティエラー。 ビット4 1のときオーバーランエラー。 ビット5 1のときフレーミングエラー。 ビット6 1のブ레이크状態検出。 ビット7 RS232CのDSRの信号線。	出 力

## [ 注 意 ]

RS232Cを利用するには、あらかじめサブルーチンSET232Cを用いてインターフェイスの初期設定を行なっておくこと。

チャンネル2,3のRS232Cを使用するときはメモリスイッチ4のビット4をONにしておく必要がある。

CLEN ( CHR )

[ 機能説明 ]

4バイト関数。文字列CHRの実際の長さを返す。

FORTRAN77の規格には文字列の定義長さを求める内蔵関数LENがあるが、こちらのCLENは、実際に文字が代入されている長さを求める。NULL(まだ文字が代入されていない状態)のときは長さ0となる。

[ 引 数 ]

CHR	文字列	長さを求めたい文字列。
-----	-----	-------------

[ テストプログラム TEST001 ]

最初はNULL文字、次に空白を代入したとき、文字列'ABC'を代入したときの文字列の定義長さと実長さを順に求めて表示する。

```

PROGRAM TEST001

INTEGER*4 CLEN,LEN
CHARACTER CHR*50

WRITE (*,'(15)') CLEN(CHR)
WRITE (*,'(15)') LEN(CHR)
CHR=' '
WRITE (*,'(15)') CLEN(CHR)
WRITE (*,'(15)') LEN(CHR)
CHR=' ABC '
WRITE (*,'(15)') CLEN(CHR)
WRITE (*,'(15)') LEN(CHR)
END

```

実長さは0  
定義長は50

空文字なので実長さは0  
定義長は50

実長さは4となる  
定義長は不変

CALL CLOSEH ( HANDLE, IER )

[ 機能説明 ]

現在オープンしているファイルをクローズする。

[ 引 数 ]

HANDLE	4B整数	ハンドル番号。サブルーチンOPENHで得た値。	入 力
IER	4B整数	エラー番号。 0.正常終了。 6.ハンドル番号が無効。	出 力

[ 備 考 ]

通常の用途ではこのサブルーチンは必要ない。FORTRANのCLOSE文を使用すればよい。このサブルーチンはディスク関係のユティリティプログラムを作成しなければならない場合のために用意した。

CALL CREATH ( FILE,HANDLE,IER )

[ 機能説明 ]

FILEで指定する名前のファイルを新たに作成し、ファイルハンドル（ファイルの番号）値を返す。

もし同名のファイルが既にあれば、そのファイルの中味は空にされる。

[ 引 数 ]

FILE	文字列	ファイル名。50字以内のこと。	入 力
HANDLE	4B整数	ハンドルの値を返す。	出 力
IER	4B整数	エラー番号。 0.正常終了。 3.パス名が無効。 4.オープンファイル数が多すぎる。	出 力

[ 備 考 ]

HANDLEの値は5以上が返される。0～4はMS-DOSの標準ファイル・ハンドルとして、システム起動時に自動的にオープンされる。

ハンドル番号	機 能
0	標準入力。通常はキーボード。デバイス名CON。
1	標準出力。通常はスクリーン。デバイス名はCON。
2	標準エラー出力。通常はスクリーン。
3	標準補助装置。RS-232Cなど。デバイス名AUX。
4	標準プリンタ。デバイス名PRN。

通常の使用ではこのサブルーチンは必要ない。FORTRANのOPEN文において、

```
STATUS='NEW'
```

```
STATUS='UNKNOWN'
```

などを使用してファイルをオープンすればよい。このサブルーチンはディスク関係のユーティリティプログラムを作成しなければならない場合のために用意した。

FORTRANのオープン文では

```
STATUS='UNKNOWN'
```

を使用すると、指定したファイルが既に存在していれば、それをオープン（アクセス可能にするが空にはならない）し、なければ新たにその名前のファイルを作成してくれる。

DEGIMAL2 ( HEX )

## [ 機能説明 ]

2バイト整数関数。4桁までの16進数文字列を2バイトの10進数に変換する関数。

## [ 引 数 ]

HEX	文字列	有効文字が4文字以内の文字列。 4文字を越える場合の結果は保証しない。 また0~9、A~F以外の文字を含まないこと。これ以外の文字が含まれていると、0を出力する(TGINITが予め実行してある場合)。
-----	-----	--

## [ 備 考 ]

計算機内の数値は補数表現を用いている。そこで、例えば2バイト整数の場合、

16進数の7FFFは、10進数32767

16進数の8000は、10進数の-32768

16進数のFFFFは、-1

ということになる。

## DEGIMAL4 ( HEX )

## [ 機能説明 ]

4バイト整数関数。8桁までの16進数文字列を4バイトの10進数に変換する関数。

## [ 引 数 ]

HEX	文字列	有効文字が8文字以内の文字列。 8文字を越える場合の結果については保証しない。 また0~9、A~F以外の文字を含まないこと。これ以外の文字が含まれていると、0を出力する(TGINITが予め実行してある場合)。
-----	-----	--

## [ 備 考 ]

計算機内の数値は補数表現を用いている。そこで、例えば4バイト整数の場合、

16進数の7FFFFFFFは、10進数の2147483647

16進数の80000000は、10進数の-2147483648

16進数のFFFFFFFFは、-1

ということになる。

**ファイルの削除****D E L E T**

CALL DELET ( PATH, IER )

**[ 機能説明 ]**

PATHで指定するファイルを削除する。パス名にはワイルド・カードは使用できない。

**[ 引 数 ]**

PATH	文字列	削除するファイル名。50字以内。 ワイルド・カードは使用不可。	入 力
IER	4B整数	エラー番号。 0. 正常終了。 2. 指定したパス名が無効か存在しない。 5. 指定したパス名がディレクトリ名である。	出 力

ディスクの空き容量を取得

D I S K S P C

CALL DISKSPC ( IDR,IBYTES,IER )

[ 機能説明 ]

指定ドライブにある磁気ディスクの空き容量をバイト数で返す。

[ 引 数 ]

IDRV	4B整数	ドライブAのとき1を与える。 ドライブBのとき2を与える。 . .	入 力
IBYTES	4B整数	使用可能なバイト数。	出 力
IER	4B整数	0.正常終了 0以外 ドライブ番号が無効	出 力

[ テストプログラム TEST002 ]

ドライブ2、つまりドライブBの磁気ディスクの空き容量をバイト数で表示するプログラム。

```

PROGRAM TEST002

INTEGER*4  IDR,IER
INTEGER*4  IBYTES

CALL TGINIT

IDRV=2
CALL DISKSPC(IDRV,IBYTES,IER)
WRITE (*,*) ' IER = ',IER
WRITE (*,*) ' BYTES = ',IBYTES

CALL TESTEND(IRET)
END
    
```

CALL EXEDIR ( DIR, ILEN )

[ 機能説明 ]

EXE形式のプログラムを起動した直後，そのプログラムが存在したディレクトリとプログラム名を返す。例えば，

A:¥>B:¥ABC¥TEST

のように起動した場合は，B:¥ABC¥TESTという文字列をDIRに返す。この機能を利用すると，データファイルとかヘルプファイルをプログラムと同じディレクトリ，またはその下のディレクトリから検索することが可能となる。ハードディスクにアプリケーションを組み込む際に便利。

[ 引 数 ]

DIR	文字列	ディレクトリ名。拡張子を除いたプログラム名も付く。	出 力
ILEN	4B整数	取得した文字列DIRの実際の長さ。	出 力

[ 注 意 ]

このサブルーチンは，呼びだしたプログラムにおいて，キー入力がないうちに実行すること。具体的には，プログラムの最初に実行するとよい。なお，このサブルーチンは，TGINITを実行しなくても使用できる。

CALL FLIST ( MAX,DIR,FILES,N,IER )

## [ 機能説明 ]

DIRで指定したファイルを検索し、該当するファイルを作成年月日を付けた文字列で返す。

## [ 引 数 ]

MAX	4B実数	最大検索ファイル数。文字列配列FILESの大きさを越えないこと。	入 力
DIR	文字列	パス名で50字以内とする。 ワイルドカードの使用も可能。 ただし、拡張子は必ず指定すること。	入 力
FILES	文字列配列	検索したファイル名を収める文字列。1要素の大きさは18文字以上とする。 文字構成は 1文字目 識別文字 +.ディレクトリ名 -.拡張子が一致するファイル名 2~9文字目 ファイル名 10文字目 空白文字 11~18文字目 年月日	出 力
N	4B整数	見つかったファイル数。サブディレクトリも数に入れる。0なら該当パス名のファイルもサブディレクトリもなかったことを意味する。	出 力
IER	4B整数	0.1つ以上のファイルがあった。 2.指定パス名が無効かファイルがなかった。 18.指定したパス名の検索終了(検索ファイル数がMAX以下の場合)。	出 力

## [ 備 考 ]

ファイル名の先頭に識別文字を付けたのは、ファイル名の並び替え(ソート)を容易にするためである。文字列配列FILESをそのままソートすれば、まずディレクトリ名を、次に拡張子が一致するファイル名をソートしてくれる。

[ テストプログラム TEST003 ]

ディレクトリB:¥TEST¥の拡張子FORのファイルを検索し、反転スクロール表示するプログラム。[RET]キーによりファイル名を選択できる。

データファイルの一覧表示と選択の参考プログラム。つまり磁気ディスクから記録済みのデータを入力するプログラム作成の参考となる。キーボードからのデータ入力の参考となるものは、サブルーチンTDATA1,TDATAN,TREAD,TREADI2,TREADI4,TREADR4,TREADR8で説明してある。

```
PROGRAM TEST003

IMPLICIT INTEGER*4 (I-N)
CHARACTER DIR*40,LIST(300)*30

CALL TGINIT
CALL TRANGE(2,16,11,28)

IROW=1
ITOP=1
MAX=300
DIR='B:¥TEST¥*.FOR'
100 CALL FLIST(MAX,DIR,LIST,N,IER)
    IF (N.NE.0) THEN
        DO 110 I=1,N
110  LIST(I)=LIST(I)(2:)
        CALL TWINDSCR(N,LIST,ITOP,4,7,IROW)
    ELSE
        CALL TDISPM(18,1,'ファイルがありません。',2,0)
        CALL TBEEP(1200)
        CALL TDISPM(18,1,'',7,0)
    END IF

CALL TESTEND(IRET)
IF (IRET.EQ.1) GOTO 100
END
```

2～16行にスクロール表示する

反転カーソルは最上行に  
最上行は最初のファイルを表示  
300までのファイルを検索  
拡張子FORのファイルを検索

先頭の識別文字を削除  
TWINDSCRはスクロール表示  
[RET]または[ESC]で終了

YES/NOの表示と選択  
NOを選択したら再実行

このテストプログラム中、TWINDSCRでは、マウスドライバが組み込まれていれば、マウスによるスクロール制御が可能である。その際、左ボタンクリックが[RET]キーの、右ボタンクリックが[ESC]キーの代用となる。

CALL GBOXF ( IX1,IX2,IY1,IY2 )

[ 機能説明 ]

現在のペンを用いて、グラフィックス画面上に四角形を実線で描き、内部をそのときのペンの色で塗りつぶす。終了後のペン位置は(IX1,IY1)。

サブルーチンGVIEWで設定したビューポートを越えた部分は描画しない。

[ 引 数 ]

IX1	4B整数	描画始点のX軸ドット座標値。	入 力
IX2	4B整数	描画終点のX軸ドット座標値。	入 力
IY1	4B整数	描画始点のY軸ドット座標値。	入 力
IY2	4B整数	描画終点のY軸ドット座標値。	入 力

[ 注 意 ]

C R T 画面制御ライブラリのグラフィックスドット座標は次の様になっている。

X軸は0(左端) ~ 639(右端)

Y軸は0(下端) ~ 399(上端)

[ テストプログラム TEST004 ]

ペンを変えて3つの矩形を描く。

```

PROGRAM TEST004

IMPLICIT INTEGER*4(I-N)

CALL TGINIT

CALL GPEN(4)
CALL GBOXF(250,500,100,350)
CALL GPEN(5)
CALL GBOXF(200,500,50,300)
CALL GPEN(6)
CALL GBOXF(400,600,200,300)

CALL TDISPM(1,1,'四角形の塗りつぶし',7,0)
CALL TESTEND(IRET)
END
    
```

ペン4はデフォルトでは緑

ペン5はデフォルトでは水色

ペン6はデフォルトでは黄色

CALL GBOXFFST ( IX1,IX2,IY1,IY2,IP,MODE )

[ 機能説明 ]

GDCの直接制御による塗りつぶし矩形の描画。ペンの位置は変化しない。また、ビューポートの設定は無視される。

[ 引 数 ]

IX1	4B整数	描画始点のX軸ドット座標値。	入 力
IX2	4B整数	描画終点のX軸ドット座標値。	入 力
IY1	4B整数	描画始点のY軸ドット座標値。	入 力
IY2	4B整数	描画終点のY軸ドット座標値。	入 力
IP	4B整数	描画プレーンの指定。0は描画しない 1.青                    6.赤,緑                    11.青,赤,輝度 2.赤                    7.青,赤,緑                    12.緑,輝度 3.青,赤                    8.輝度                    13.青,緑,輝度 4.緑                    9.青,輝度                    14.赤,緑,輝度 5.青,緑                    10.赤,輝度                    15.青,赤,緑,輝度	入 力
MODE	4B整数	描画モード 0.リプレイス 1.コンプリメント(XOR) 2.クリア(AND) 3.セット(OR)	入 力

[ 注 意 ]

C R T 画面制御ライブラリのグラフィックスドット座標は次の様になっている。

X軸は0(左端) ~ 639(右端)

Y軸は0(下端) ~ 399(上端)

実際に表示される色は、描画プレーンと描画モードによって異なる。もし、グラフィックス画面に何も描画していなければ、描画プレーンで指定する色にすることができる。

CALL GBOXL ( IX1,IX2,IY1,IY2 )

[ 機能説明 ]

現在のペンと線の種類を用いて、グラフィックス画面に四角形を描く。描画後のペン位置は(IX1,IY1)となる。

サブルーチンGVIEWで設定したビューポートを越えた部分は描画しない。

[ 引 数 ]

IX1	4B整数	描画始点のX軸ドット座標値。	入 力
IX2	4B整数	描画終点のX軸ドット座標値。	入 力
IY1	4B整数	描画始点のY軸ドット座標値。	入 力
IY2	4B整数	描画終点のY軸ドット座標値。	入 力

[ 注 意 ]

C R T画面制御ライブラリのグラフィックスドット座標は次の様になっている。

X軸は0(左端) ~ 639(右端)

Y軸は0(下端) ~ 399(上端)

[ テストプログラム TEST005 ]

ペンを変更して3つの矩形を描く。

```

PROGRAM TEST005

IMPLICIT INTEGER*4(I-N)

CALL TGINIT

CALL GPEN(4)
CALL GBOXL(250,500,100,350)
CALL GPEN(5)
CALL GBOXL(200,500,50,300)
CALL GPEN(6)
CALL GBOXL(400,600,200,300)

CALL TESTEND(IRET)
END
    
```

ペン4のデフォルトは緑

ペン5のデフォルトは水色

ペン6のデフォルトは黄色

CALL GBOXLFST ( IX1,IX2,IY1,IY2,IP,MODE )

[ 機能説明 ]

GDCの直接制御による塗りつぶし矩形の描画。ペンの位置は変化しない。また、ビューポートの設定は無視される。

[ 引 数 ]

IX1	4B整数	描画始点のX軸ドット座標値。	入 力
IX2	4B整数	描画終点のX軸ドット座標値。	入 力
IY1	4B整数	描画始点のY軸ドット座標値。	入 力
IY2	4B整数	描画終点のY軸ドット座標値。	入 力
IP	4B整数	描画プレーンの指定。0は描画しない 1.青                    6.赤,緑                11.青,赤,輝度 2.赤                    7.青,赤,緑            12.緑,輝度 3.青,赤                8.輝度                 13.青,緑,輝度 4.緑                    9.青,輝度             14.赤,緑,輝度 5.青,緑                10.赤,輝度            15.青,赤,緑,輝度	入 力
MODE	4B整数	描画モード 0.リプレイス 1.コンプリメント(XOR) 2.クリア(AND) 3.セット(OR)	入 力

[ 注 意 ]

C R T 画面制御ライブラリのグラフィックスドット座標は次の様になっている。

X軸は0(左端) ~ 639(右端)

Y軸は0(下端) ~ 399(上端)

実際に表示される色は、描画プレーンと描画モードによって異なる。もし、グラフィックス画面に何も描画していなければ、描画プレーンで指定する色にすることができる。

CALL GCIRCLE ( IRX,IRY )

[ 機能説明 ]

現在のペンを用いて、現在ペンの位置を中心とした円をグラフィックス画面に描く。線は常に実線が使用される。円の内部を塗りつぶしたいときは、サブルーチンGPAINTを使用するか、GCIRCLFNにおいて扇型の角度を0～360にし、塗り潰しを指定する。

X軸方向とY軸方向の半径が異なると楕円になる。

[ 引 数 ]

IRX	4B整数	X軸方向半径のドット値。	入 力
IRY	4B整数	Y軸方向半径のドット値。	入 力

[ 注 意 ]

CRT画面制御ライブラリのグラフィックスドット座標は次の様になっている。

X軸は0(左端)～639(右端)

Y軸は0(下端)～399(上端)

[ テストプログラム TEST006 ]

半径40ドットの円を、色と中心座標を変えて7つ描く

```

PROGRAM TEST006

IMPLICIT INTEGER*4(I-N)

CALL TGINIT

IX=200
IY=250
DO 100 I=1,7
  CALL GPEN(I)
  CALL GMOVE(IX,IY)
  CALL GCIRCLE(40,40)
  IX=IX+40
100 CONTINUE

CALL TESTEND(IRET)
END

```

ペンを指定  
 ペンを中心座標へ移動  
 半径40ドットの円を描く  
 円中心のX座標を40ドット右へ

CALL GCIRCLFN ( IRX,IRY,ISX,ISY,IEX,IEY,ISW )

[ 機能説明 ]

そのときのペンを用いて、そのときのペンの位置を中心とした円の一部としての扇形を描く。線は常に実線が使用される。扇形の内部の塗りつぶしも可能。

[ 引 数 ]

IRX	4B整数	X軸方向の半径のドット値。	入 力
IRY	4B整数	Y軸方向の半径のドット値。	入 力
ISX	4B整数	描画開始X軸ドット座標値。	入 力
ISY	4B整数	描画開始Y軸ドット座標値。	入 力
IEX	4B整数	描画終了X軸ドット座標値。	入 力
IEY	4B整数	描画終了Y軸ドット座標値。	入 力
ISW	4B整数	0.輪郭線のみ 1.内部を塗りつぶす	入 力

[ 注 意 ]

C R T 画面制御ライブラリのグラフィックドット座標は次の様になっている。

X軸は0(左端) ~ 639(右端)

Y軸は0(下端) ~ 399(上端)

[ テストプログラム TEST007 ]

中心座標を(300,200)とした、0 ~ 120度と130 ~ 160度の扇型を、半径120ドットで描く。

与えられた角度から引数に必要な座標値を計算している。

```
PROGRAM TEST007
IMPLICIT INTEGER*4(I-N)
CALL TGINIT
IR=120
```

```
IX=300
IY=200
PI2=3.14156*2.0/360.0
CALL GMOVE(IX,IY)
```

度からラジアンへの換算係数

```
CALL GPEN(4)
ISX=IX+IR*COS(0.0*PI2)
ISY=IY+IR*SIN(0.0*PI2)
IEX=IX+IR*COS(120.0*PI2)
IEY=IY+IR*SIN(120.0*PI2)
CALL GCIRCLFN(IR,IR,ISX,ISY,IEX,IEY,0)
```

```
CALL GPEN(6)
ISX=IX+IR*COS(130.0*PI2)
ISY=IY+IR*SIN(130.0*PI2)
IEX=IX+IR*COS(160.0*PI2)
IEY=IY+IR*SIN(160.0*PI2)
CALL GCIRCLFN(IR,IR,ISX,ISY,IEX,IEY,1)
```

```
CALL TESTEND(IRET)
END
```

CALL GCLR ( ICOD )

[ 機能説明 ]

画面をクリアする。クリアする画面は引数で指定する。グラフィックス画面については、現在命令が有効となっている画面のみをクリアする（正確には0番のペンで画面を塗りつぶす）。

[ 引 数 ]

ICOD	4B整数	1.テキスト画面を消去 2.グラフィックス画面のビューポート内を消去 3.テキスト画面とグラフィックス画面のビューポート内を消去	入 力
------	------	--	-----

[ 備 考 ]

ビューポート内の消去はサブルーチンGVEWCLRでも実行可能である。

CALL GCURSW ( IX,IY,IC )

[ 機能説明 ]

CRT画面制御ライブラリ内蔵の十字カーソルの表示/非表示の制御。カーソルの色は白に固定してある。

[ 引数 ]

IX	4B整数	十字カーソルのX軸ドット座標。	入 力
IY	4B整数	十字カーソルのY軸ドット座標。	入 力
IC	4B整数	0.座標(IX,IY)に十字カーソルがあれば消去し、 無ければ表示する(XOR動作)。 1.CRT画面に十字カーソルを表示する(OR動作)。	入 力

[ 備 考 ]

座標(IX,IY)は十字の中心座標を示す。十字のいずれかの端がドット座標値を越えると、十字カーソルは表示されなくなる。

CRT画面制御ライブラリのグラフィックドット座標は次の様になっている。

X軸0は(左端)~639(右端)

Y軸0は(下端)~399(上端)

CALL GDRAW ( IX,IY )

[ 機能説明 ]

現在のペンと線の種類を用いて、ペンの現在位置からここで指定する座標まで線を引く。サブルーチンGVIEWで設定したビューポートの外側には線は引かない。

[ 引 数 ]

IX	4B整数	描画終点のX軸ドット座標値。	入 力
IY	4B整数	描画終点のY軸ドット座標値。	入 力

[ 注 意 ]

C R T画面制御ライブラリのグラフィックストット座標は次の様になっている。

X軸は0(左端) ~ 639(右端)

Y軸は0(下端) ~ 399(上端)

CALL GDRAWFST ( IX2, IY2, IP, MODE )

[ 機能説明 ]

GDCの直接制御による直線の描画。現在のペン位置から、現在の線種を用いて、座標(IX2, IY2)まで線を引く。描画後のペンの位置は変化しない。また、ビューポートの設定は無視される。

[ 引 数 ]

IX2	4B整数	描画終点のX軸ドット座標値。	入 力
IY2	4B整数	描画終点のY軸ドット座標値。	入 力
IP	4B整数	描画プレーンの指定。0は描画しない 1.青                    6.赤, 緑                11.青, 赤, 輝度 2.赤                    7.青, 赤, 緑            12.緑, 輝度 3.青, 赤                8.輝度                    13.青, 緑, 輝度 4.緑                    9.青, 輝度                14.赤, 緑, 輝度 5.青, 緑                10.赤, 輝度              15.青, 赤, 緑, 輝度	入 力
MODE	4B整数	描画モード 0.リプレイス 1.コンプリメント(XOR) 2.クリア(AND) 3.セット(OR)	入 力

[ 注 意 ]

CRT画面制御ライブラリのグラフィックスドット座標は次の様になっている。

X軸は0(左端) ~ 639(右端)

Y軸は0(下端) ~ 399(上端)

実際に表示される色は、描画プレーンと描画モードによって異なる。もし、グラフィックス画面に何も描画していなければ、描画プレーンで指定する色にすることができる。

```
CALL GETADR ( IVAL, ISEG, IOFF )
```

## [ 機能説明 ]

変数 IVAL の値が書き込まれて (記録されて) いるメモリ番地を調べて返す。配列変数の場合は、先頭の番地を返す。

## [ 引 数 ]

引数	型	説明	入出力
IVAL	4B整数	変数名	入 力
ISEG	4B整数	セグメントアドレス	出 力
IOFF	4B整数	オフセットアドレス	出 力

## [ テストプログラム TEST008 ]

MS/FORTRANには1バイト整数の宣言文があるが、ここではテストプログラムとして、4バイト整数配列の各要素を1バイトずつ分割して扱ってみる。1バイトなので0~255までの数値しか扱えない。

このプログラムでは、まず0~255までの整数をPOKEBによってメモリに記録し、確認のために、PEEKBを用いて再び読み出して表示している。

```

PROGRAM TEST008
  INTEGER*4  I, II, ISEG, IOFF, IADR, IA(64)  1バイトデータなので64*4=256
  CALL GETADR(IA, ISEG, IOFF)                配列変数IAの先頭アドレスを取得
  DO 100 I=0, 255
    IADR=IOFF+I
    CALL POKEB( ISEG, IADR, I)               0~255の数値をメモリへ書き込む
                                           IADRは書き込むメモリの番地
                                           1バイトデータをメモリへ書き込む
  100 CONTINUE

  DO 200 I=0, 255
    IADR=IOFF+I
    CALL PEEKB( ISEG, IADR, II)              256回数値を読み出す
    WRITE (*, *) II                          1バイトデータをメモリから読み出す
                                           それを表示する
  200 CONTINUE

  END

```

CALL GETCD ( IDR )

[ 機能説明 ]

現在設定されているドライブの番号を取得する。

[ 引 数 ]

IDRV	4B整数	ドライブAのときは1を返す。 ドライブBのときは2を返す。 ：	出 力
------	------	---------------------------------------	-----

CALL GETDIR ( IDR, PATH, IER )

[ 機能説明 ]

指定したドライブのカレント・ディレクトリ名を取得する。

[ 引 数 ]

IDRV	4B整数	ドライブ番号。 1. ドライブA 2. ドライブB 3. ドライブC . .	入 力
PATH	文字列	ディレクトリ名を返す。 先頭にドライブ名がつく。	出 力
IER	4B整数	エラー番号。 0. 正常終了。	出 力

[ テストプログラム TEST009 ]

カレント・ディレクトリをドライブBのABC¥TESTに移動した後、現在のディレクトリ名を調べて表示する。ディレクトリが本当に変更されているかを確認するプログラム。ただし、このプログラムはTGINITを実行していないので、実際にはIERの値は検出できない。

```
PROGRAM TEST009
```

```
IMPLICIT INTEGER*4 (I-N)  
CHARACTER PATH*40, CHR*40
```

```
PATH='B:¥ABC¥TEST'  
CALL CHDIR(PATH, IER)  
CALL GETDIR(2, CHR, IER)  
WRITE(*, *) IER  
WRITE(*, *) CHR
```

```
END
```

カレント・ディレクトリの変更  
ドライブBなのでIDRVは2

CALL GETDRV ( N )

[ 機能説明 ]

現在接続してあるドライブの数を調べて返す。

[ 引 数 ]

N	4B整数	接続ドライブ数。有効なドライブかどうかはチェックしない。たとえば、ディスクが挿入されていないドライブも数に含む。	出力
---	------	--	----

CALL GETLIN ( HDL,CHR,N )

[ 機能説明 ]

サブルーチンOPENHでオープンしたディスク・ファイルから、ファイルの中味を1行分だけ文字列CHRに代入する。読み出すと、ファイル・ポインタの位置は、自動的に次のデータを示すようにセットされる。

[ 引 数 ]

HDL	4B整数	ハンドル番号。サブルーチンOPENHで得た値。	入 力
CHR	文字列	大きさ255字以内の文字列。	出 力
N	4B整数	読みだしたバイト数。 ファイルの終了は0バイト。	出 力

[ 備 考 ]

文字列CHRには、CR(16進文字で0D),LF(16進文字で0A)がついてので、バイト数Nにはこの2文字分も含まれている。

したがって、何も文字がない行でも2バイトの出力がある。

通常の用途ではFORTRANのREAD文を使用すればよい。

[ テストプログラム TEST010 ]

ドライブB:のディスクサブディレクトリTESTのTEST010を読み取り、画面に表示するプログラム。表示の際は、実際の長さだけを表示するようにした。

```
PROGRAM TEST010
```

```
INTEGER*4 I,N,IER,HANDLE,ICLEN,LBYTES
```

```
INTEGER*4 CLEN
```

```
CHARACTER FILE*50,CHR(300)*79
```

```
CALL TGINIT
```

```
CALL TRANGE(1,18,1,80)
```

```
FILE='B:¥TEST¥TEST010.FOR'
```

```
CALL OPENH(FILE,HANDLE,IER)
```

```
IF (IER.NE.0) GOTO 999
```

表示範囲は1～18行まで

ファイルを開く

17-なら終了

```

      I=1
100 CALL GETLIN(HANDLE,CHR(I),LBYTES)
      IF (LBYTES.EQ.0) GOTO 200
      I=I+1
      GOTO 100

200 N=I-2
      CALL CLOSEH(HANDLE,IER)
      DO 300 I=1,N
          ICLEN=CLEN(CHR(I))
          CALL TPRINT(CHR(I)(1:ICLEN-2),2)
300 CONTINUE

999 CALL TESTEND(IRET)
      END

```

0文字ならファイルは終了

後半の2バイト以下の行は削除  
ファイルを閉じておく

文字列の実長さを求める  
CR,LFの2バイトは非表示

#### [ テストプログラム TEST011 ]

次のプログラムのようにすると、復帰、改行の制御文字を、文字列とは別の色で表示することができる。

```

PROGRAM TEST011

INTEGER*4  I,N,IER,HANDLE,ICLEN,LBYTES
INTEGER*4  CLEN
CHARACTER  FILE*50,CHR(300)*79

CALL TGINIT
CALL TRANGE(1,18,1,80)
FILE='B:¥TEST¥TEST010.FOR'
CALL OPENH(FILE,HANDLE,IER)
IF (IER.NE.0) GOTO 999

      I=1
100 CALL GETLIN(HANDLE,CHR(I),LBYTES)
      IF (LBYTES.EQ.0) GOTO 200
      I=I+1
      GOTO 100

200 N=I-2
      CALL CLOSEH(HANDLE,IER)
      DO 300 I=1,N
          ICLEN=CLEN(CHR(I))
          CALL TCOLOR(7,0)
          CALL TPRINT(CHR(I)(1:ICLEN-2),0)
          CALL TCOLOR(4,0)
          CALL TPRINT(CHR(I)(ICLEN-1:ICLEN),2)
300 CONTINUE

999 CALL TESTEND(IRET)
      END

```

表示範囲は1～18行まで

ファイルを開く  
エラーなら終了

0文字ならファイルは終了

後半の2バイト以下の行は削除  
ファイルを閉じておく

文字列の実長さを求める  
文字の表示色は7  
CR,LFの2バイトは非表示  
表示色を4にして  
CR,LFの制御文字を表示

CALL GETOPTION ( N,CHR )

[ 機能説明 ]

プログラムをオプション付きで起動する際、コマンドライン（起動するプログラム名の後に、空白で区切って与える文字列のこと）の文字列を取得する。

[ 引 数 ]

N	4B整数	文字列CHRの実際の長さ。	出 力
CHR	文字列	コマンドラインに与えた文字列。最初の空白も含む。空白、'-','/'のいずれかをコマンドの区切りと見なし、コマンド間を空白1文字で区切って出力する。空白は2文字分以上あっても1文字分に切り詰めて出力する。	出 力

システム時刻を秒数で取得

GETSEC

CALL GETSEC ( ISEC )

[ 機能説明 ]

システムの時刻を秒数に換算して返す。

[ 引 数 ]

ISEC	4B整数	換算した秒数。	出 力
------	------	---------	-----

[ 備 考 ]

プログラムの実行時間の計測用に作成した。ただし、システム時刻が00:00:00にまたがる場合には要注意。

```
CALL GGET ( IX1,IX2,IY1,IY2,LEN,MOJI )
```

## [ 機能説明 ]

2点(X1,Y1)と(X2,Y2)を対角とする領域に描かれている図形を、グラフィックスデータとして文字列配列MOJIに記憶する。

このサブルーチンで記憶した情報は、サブルーチンGPUTを用いて表示する。

## [ 引 数 ]

IX1	4B整数	領域左端のX軸ドット座標値。	入 力
IX2	4B整数	領域右端のX軸ドット座標値。	入 力
IY1	4B整数	領域下端のY軸ドット座標値。	入 力
IY2	4B整数	領域上端のY軸ドット座標値。	入 力
LEN	4B整数	情報を記憶するのに必要なバイト数。 以下の数値以上であること。 $INT((X2-X1+8)/8)*((Y2-Y1)+1)*4+4$	入 力
MOJI	文字列配列	画面情報を記憶しておく文字列配列。 大きさはLENバイト以上であること。	出 力

## [ 注 意 ]

X1<X2,Y1<Y2のこと。

C R T画面制御ライブラリのグラフィックスドット座標は次の様になっている。

X軸は0(左端)～639(右端)

Y軸は0(下端)～399(上端)

## [ テストプログラム TEST012 ]

画面中央付近に棒状の図形を描き、それを文字列配列MJに記憶し、再び表示するプログラム。ついでに円筒上の画像も描き、色もサブルーチンGSETRGBで変更し、いかにもコンピュータで表示したという画像らしく仕上げてみた。

```
PROGRAM TEST012
IMPLICIT INTEGER*4(I-N)
```

```
CHARACTER MJ(3800)*1,EXP*50
LN=3800
```

```
CALL TGINIT
```

まずはライブラリの初期化

```
白黒の濃淡の設定
```

```
CALL GSETRGB(7,14,15,15)
CALL GSETRGB(6,13,14,14)
CALL GSETRGB(5,11,12,12)
CALL GSETRGB(4,9,10,10)
CALL GSETRGB(3,7,8,8)
CALL GSETRGB(2,6,7,7)
CALL GSETRGB(1,5,6,6)
CALL GSETRGB(0,0,0,0)
```

ペンの色をグレイ系統に

```
棒状の画像を描く
```

```
IX1=240
IX2=390
```

```
IYU=237
DO 100 IPEN=7,1,-1
CALL GPEN(IPEN)
CALL GBOXF(IX1-150,IX2+150,IYU,IYU-2)
IYU=IYU-2
```

```
100 CONTINUE
```

```
棒状の画像を文字列配列MJに記憶する
```

```
CALL GGET(IX1-150,IX2+150,222,237,LN,MJ)
```

```
円筒状にみせる画像の作成
```

```
IYU=300
IYB=IYU-20
```

```
DO 200 IPEN=7,1,-1
DO 210 IY=IYU,IYB,-1
DO 220 IX=IX1,IX2
CALL GPSET(IX,IY,IPEN)
```

7番のペンから逆順に  
上の方から1ドットずつ下げて  
横にIX1~IX2まで  
1ドットずつ画面に点を打つ

```
220 CONTINUE
```

```
210 CONTINUE
```

```
IYU=IYU-20
IYB=IYU-20
```

20ドット分下に移動

```
200 CONTINUE
```

```
棒状の画像を表示する
```

```
CALL GPUT(IX1-150,285,LN,MJ,0)
```

```
CALL TESTEND(IRET)
END
```

CALL GLINSTL ( ISTL )

[ 機能説明 ]

線を引くときに使用する線の種類（スタイル）の指定。ここで指定する線のスタイルはグラフィックス画面上の16ドットを繰り返し周期としている。

[ 引 数 ]

ISTL	4B整数	線の種類の番号と、形状の関係は以下の通り。	入 力
		0.点 線1 (1ドットおき。体裁用の線に便利)	
		1.点 線2 (点の間隔が長い点線)	
		2.点 線3 (2ドット間隔の点線)	
		3.破 線1 (短間隔)	
		4.破 線2 (等間隔の破線)	
		5.破 線3 (実線部分が長い破線)	
		6.破 線4 (更に実線部分が長い破線)	
		7.一点鎖線1 (点が1ドット)	
		8.一点鎖線2 (点が2ドット分)	
		9.二点鎖線	
		10.実 線	

[ テストプログラム TEST013 ]

線種番号と線の形状を確認するためのプログラム。X軸方向に沿って直線を引いたあと、斜めに引くとどうなるか、左上に向かって描画する。

このテストプログラムにはグラフィック画面描画における基本的なサブルーチンの動作の理解にも役に立つ。

```
PROGRAM TEST013
```

```
IMPLICIT INTEGER*4(I-N)
CHARACTER CHR*2
```

```
CALL TGINIT
```

まずライブラリの使用宣言  
両端の座標を決める

IX1=200	X軸は200ドットの点から
IX2=550	550ドットの位置まで
IY1=390	Y軸は390ドットの位置
IY2=IY1	
DO 100 I=0,10	0から10の線種について
CALL GLINSTL(I)	ラインスタイルを指定
CALL GPEN(4)	ペンは4番の緑色
CALL GMOVE(IX1,IY1)	ペンを(200,IY1)に移動
CALL GDRAW(IX2,IY2)	(550,IY1)まで線を引く
WRITE (CHR, '(I2)') I	線種番号を文字に変換
CALL GPEN(7)	ペンを7番の白にして
CALL GSYMBL(IX2+10,IY1+8,CHR(2:2),2,2)	線種番号を表示する
IY1=IY1-20	20ドット下に移動する準備
IY2=IY1	右端のY座標は左端と同じ
100 CONTINUE	次の線種描画へ
IX1=200	X座標200ドットの位置から
IX2=450	450ドット目の位置まで
IY1=20	Y座標は20ドット位置から
IY2=190	右上の190ドットの位置まで
CALL GPEN(6)	ペンは6番の黄色で
DO 200 I=10,0,-1	10番の線種から1番まで
CALL GLINSTL(I)	線種を変えて
CALL GMOVE(IX1,IY1)	(200,IY1)にペンを移動し
CALL GDRAW(IX2,IY2)	(450,IY2)まで線を引く
IY1=IY1+20	20ドット上にペン移動の準備
IY2=IY2+20	右端のY座標も20ドット上に
200 CONTINUE	
CALL TESTEND(IRET)	
END	

[ 注 意 ]

C R T 画面制御ライブラリのグラフィックスポット座標は次の様になっている。

X軸は0(左端) ~ 639(右端)

Y軸は0(下端) ~ 399(上端)

CALL GLPSETF ( IX,IY,IARY,IC )

[ 機能説明 ]

V R A Mへの直接アクセスにより高速にドット描画を行なう。

整数配列 IARYに収めたパレットコード(ペン番号)で定義する色のドット配列を、グラフィックス画面X軸方向(右端方向)に連続描画する。サブルーチンGVIEWで設定したビューポートの範囲は無視される。

[ 引 数 ]

IX	4B整数	描画始点のX軸ドット座標値 描画終点のX軸ドット座標値は、IX+IC-1。	入 力
IY	4B整数	描画始点のY軸ドット座標値。	入 力
IARY	1B整数配列	パレットコード(描画に使用するペン番号)を収めておく整数配列。	入 力
IC	4B整数	描画するドット数。	入 力

[ 注 意 ]

IARYは1バイトの整数配列とすること。

C R T画面制御ライブラリのグラフィックスドット座標は次の様になっている。

X軸は0(左端)~639(右端)

Y軸は0(下端)~399(上端)

[ テストプログラム TEST014 ]

サブルーチンGGETのテストプログラムTEST012において、円筒状に見せる画像を、このGLPSETFで描くようにしたプログラム。

```

PROGRAM TEST014

IMPLICIT INTEGER*4(I-N)
INTEGER*1 IP(151)

CALL TGINIT
CALL SCREEN(1,1)
    
```

まずはライブラリの使用宣言

```
CALL GSETRGB(7,14,15,15)
CALL GSETRGB(6,13,14,14)
CALL GSETRGB(5,11,12,12)
CALL GSETRGB(4,9,10,10)
CALL GSETRGB(3,7,8,8)
CALL GSETRGB(2,6,7,7)
CALL GSETRGB(1,5,6,6)
CALL GSETRGB(0,0,0,0)
```

ペン(ハレット)の色を変更

```
IX1=240
IX2=390
```

```
IYU=237
DO 100 IPEN=7,1,-1
CALL GPEN(IPEN)
CALL GBOXF(IX1-150,IX2+150,IYU,IYU-2)
IYU=IYU-2
```

```
100 CONTINUE
```

```
IYU=300
IYB=IYU-20
```

```
DO 200 IPEN=7,1,-1
DO 201 I=1,151
```

```
201 IP(I)=IPEN
DO 210 IY=IYU,IYB,-1
210 CALL GLPSETF(IX1,IY,IP,151)
```

```
IYU=IYU-20
IYB=IYU-20
```

```
200 CONTINUE
```

```
CALL TESTEND(IRET)
END
```

7番のペンから逆順に  
Iはハレット配列の要素番号  
ペン番号を配列に代入  
Y座標のIYUから下のIYBまで  
151ドットの描画を繰り返す  
20ドット下に移動

CALL GMOVE ( IX, IY )

[ 機能説明 ]

ペンを指定座標まで移動する。移動するだけで線は引かない。サブルーチンGVIEWで設定したビューポートの外にも移動できる。

[ 引 数 ]

IX	4B整数	移動先のX軸ドット座標値。	入 力
IY	4B整数	移動先のY軸ドット座標値。	入 力

[ 注 意 ]

C R T画面制御ライブラリのグラフィックドット座標は次の様になっている。

X軸は0(左端) ~ 639(右端)

Y軸は0(下端) ~ 399(上端)

CALL GPAINT ( IC1,IC2 )

[ 機能説明 ]

現在ペンのある位置を囲んでいる図形の内部を、ペンIC2で塗りつぶす。この場合の図形とは、ペンIC1で描かれている閉曲線のことである。

[ 引 数 ]

IC1	4B整数	閉曲線を描いたペンの番号(0~15)。	入 力
IC2	4B整数	塗りつぶしに使用するペン番号(0~15)。	入 力

[ テストプログラム TEST015 ]

3つの矩形を描き、ペンを移動してGPAINTを実行した場合の動作を確認する。

	PROGRAM TEST015	
	IMPLICIT INTEGER*4(I-N)	
	CALL TGINIT	まずはライブラリの使用宣言
C		
C	7番のペンで描いた四角の外側の点から塗りつぶす	
C		
	CALL GPEN(7)	7番のペンで
	CALL GBOXL(250,550,100,350)	矩形を描画
	CALL GMOVE(100,50)	ペンを矩形の外側に移動
	CALL GPAINT(4,7)	4番のペンで7番の外側を塗る
C		
C	5番と6番のペンで四角を描く	
C		
	CALL GPEN(5)	5番のペンで
	CALL GBOXL(200,500,50,300)	矩形を描く
	CALL GPEN(6)	6番のペンで
	CALL GBOXL(400,600,200,300)	矩形を描く
C		
C	6番のペンで描いた四角の内側の点から塗りつぶす	
C		
	CALL GMOVE(580,250)	2つの矩形の間にペンを移動
	CALL GPAINT(2,6)	そこを塗りつぶす
	CALL TDISPM(1,1,'図形の塗りつぶし',0,0)	テキスト画面の1行1列目から説明文を表示する
	CALL TESTEND(IRET)	[RET]キーを押すと終了
	END	

CALL GPEN ( I )

## [ 機能説明 ]

線を引いたり図形を塗りつぶす際に使用するときのペンを指定する。CRT画面制御ライブラリでは「何番目のペンを使用する」という考え方を使用している。

ここで指定したペンで描画した後、別のペンで同じ描画を実行すると、線または図形の色は後の方のペンの色になる。

## [ 引 数 ]

IC	4B整数	ペン番号。	入 力
		番号と色の関係は以下のようになっている。 この関係はGSETRGBで変更できる。	
		1. 青	
		2. 赤	
		3. 紫	
		4. 緑	
		5. 水色	
		0. 黒	
		6. 黄色	
		7. 白	
		8. 灰	
		9. 暗い青	
		10. 暗い赤	
		11. 暗い紫	
		12. 暗い緑	
		13. 暗い水色	
		14. 暗い黄色	
		15. 暗い白	

## [ テストプログラム TEST016 ]

デフォルトのペンの色を確認するために、0～15番までのペンを使用して同じ長さの直線を引く。

PROGRAM TEST016

IMPLICIT INTEGER\*4(I-N)  
CHARACTER CHR\*1

CALL TGINIT

まずはライブラリの使用宣言

IX=200

IY=350

DO 100 I=0,7

CALL GPEN(I)

CALL GMOVE(IX,IY)

CALL GDRAW(IX+400,IY)

ペンを選択

ドット座標(IX,IY)まで移動

400ドット右へ線を引く

```
WRITE (CHR, '(I1)') I
CALL GPEN(7)
CALL GSYMBL(IX-18, IY+8, CHR, 2, 2)
IY=IY-20
100 CONTINUE
```

```
CALL TESTEND(IRET)
END
```

ハッシュ番号を数文字に変換  
その数文字を7番のハッシュで  
線の18ビット左側に描く  
次の直線は20ビット下に

```
CALL GPENPOS ( IX,IY )
```

## [ 機能説明 ]

現在ペンがあるドット座標値を調べてその値を返す。

## [ 引 数 ]

IX	4B整数	現在ペンがあるX軸のドット座標値	出力
IY	4B整数	現在ペンがあるY軸のドット座標値	出力

## [ 注 意 ]

C R T 画面制御ライブラリのグラフィックドット座標は次の様になっている。

X軸は0(左端) ~ 639(右端)

Y軸は0(下端) ~ 399(上端)

CALL GPSET ( IX,IY,IPEN )

[ 機能説明 ]

IX,IYで指定するグラフィックストット座標値に指定したペンで点を打つ。サブルーチンGVIEWで設定したビューポートの外側には打たない。

[ 引 数 ]

IX	4B整数	X軸ドット座標値。	入 力
IY	4B整数	Y軸ドット座標値。	入 力
IPEN	4B整数	ペン番号。 番号と色の関係は以下のようになっている。 この関係はGSETRGBで変更できる。 1.青                    6.黄色                    11.暗い紫 2.赤                    7.白                      12.暗い緑 3.紫                    8.灰                      13.暗い水色 4.緑                    9.暗い青                14.暗い黄色 5.水色                10.暗い赤               15.暗い白 0.黒	入 力

[ 注 意 ]

C R T画面制御ライブラリのグラフィックストット座標は次の様になっている。

X軸は0(左端) ~ 639(右端)

Y軸は0(下端) ~ 399(上端)

[ テストプログラム TEST017 ]

乱数を10000個発生させて、点を打つ。

```
PROGRAM TEST017

IMPLICIT INTEGER*4(I-N)

CALL TGINIT
CALL GVVIEW(150,550,100,300)
```

```

CALL GBOXL(150,550,100,300)

DO 100 I=1,10000
  IX=640*RNDD()
  IY=400*RNDD()
  IC=8*RNDD()
  CALL GPSET(IX,IY,IC)
100 CONTINUE

CALL TESTEND(IRET)
END

```

```

C -----
C 乱数の発生
C
REAL*4 FUNCTION RNDD( )

REAL*4 R,P,X
SAVE R,P,X
DATA R,P,X /223.0,4194304.0,3141597/

X=AMOD(R*X,P)
RNDD=X/P
RETURN
END

```

[ 注 意 ]

このテストプログラムで用いている乱数発生用サブルーチンは、良質の乱数発生器とはいえない。

CALL GPSETF ( IX,IY,IPEN )

[ 機能説明 ]

V R A Mへの直接アクセスにより高速にドット描画を行なう。サブルーチンGVIEWで設定したビューポートの範囲は無視される。

[ 引 数 ]

IX	4B整数	X軸ドット座標値。	入 力
IY	4B整数	Y軸ドット座標値。	入 力
IPEN	4B整数	ペン番号。 番号と色の関係は以下のようになっている。 この関係はGSETRGBで変更できる。 1.青                    6.黄色                    11.暗い紫 2.赤                    7.白                      12.暗い緑 3.紫                    8.灰                      13.暗い水色 4.緑                    9.暗い青                14.暗い黄色 5.水色                10.暗い赤               15.暗い白 0.黒	入 力

[ 注 意 ]

C R T画面制御ライブラリのグラフィックドット座標は次の様になっている。

X軸は0(左端) ~ 639(右端)

Y軸は0(下端) ~ 399(上端)

CALL GPUT ( IX,IY,LEN,MOJI,MODE )

[ 機能説明 ]

サブルーチンGGETで文字列配列MOJIに読み込んだグラフィックデータを、指定座標位置から表示する。

[ 引 数 ]

IX	4B整数	描画始点左端のX軸ドット座標値。	入 力
IY	4B整数	描画始点上端のY軸ドット座標値。	入 力
LEN	4B整数	サブルーチンGGETで使⽤した文字列配列のバイト数。	入 力
MOJI	文字列配列	サブルーチンGGETで使⽤したもの。	入 力
MODE	4B整数	描画（表示）モード 0.PSET 1.PRESET 2.OR 3.AND 4.XOR	入 力

[ 注 意 ]

C R T画面制御ライブラリのグラフィックストット座標は次の様になっている。

X軸は0(左端) ~ 639(右端)

Y軸は0(下端) ~ 399(上端)

実際に表示される色は描画モードによって異なる。もし、グラフィックス画面に何も描画していなければ、GGETで取得したままとなる。

CALL GSETRGB ( IPEN,IR,IG,IB )

[ 機能説明 ]

グラフィックドット画面に線を引いたり、図形を描く際のペンの色を設定する。設定は赤、緑、青の輝度（相対値）を16段階で指定する。0はその色を含まないことであり、15とはその色の輝度が一番高いことである。

[ 引 数 ]

IPEN	4B整数	ペン番号。0から15まで。	入 力
IR	4B整数	赤色成分の輝度。0から15まで。	入 力
IG	4B整数	緑色成分の輝度。0から15まで。	入 力
IB	4B整数	青色成分の輝度。0から15まで。	入 力

[ 注意 ]

この命令でペンの色を変更すると、いままでその番号のペンで描かれていた図形、線の色も変わるので注意。

[ テストプログラム TEST018 ]

このライブラリは実験結果などのグラフ作成用であるが、試みに画像表示や処理に利用する場合に利用出来るような階調処理の例をいくつか示す。

まず最初は16階調の白黒濃淡である。画面上の方に1番明るい白を表示する。

```

PROGRAM TEST018

CALL TGINIT

DO 100 I=1,15
    CALL GSETRGB(I,I,I,I)
100 CONTINUE

IX=300
IY=380
DO 200 I=15,0,-1
    CALL GPEN(I)
    CALL GBOXF(IX,IX+50,IY,IY-20)
    IY=IY-20
200 CONTINUE
    
```

```
CALL TESTEND(IRET)
END
```

[ テストプログラム TEST019 ]

3原色（赤、緑、青）の濃淡による16段階の階調。ただし白だけは残しておくことにする。量や質の違いを表現する場合に利用できる。

```
PROGRAM TEST019

CALL TGINIT

CALL GSETRGB( 0, 0, 0, 0)
CALL GSETRGB( 1, 0, 0, 7)
CALL GSETRGB( 2, 0, 0, 9)
CALL GSETRGB( 3, 0, 0,12)
CALL GSETRGB( 4, 0, 0,15)
CALL GSETRGB( 5, 0, 7, 0)
CALL GSETRGB( 6, 0,10, 0)
CALL GSETRGB( 7, 0,12, 0)
CALL GSETRGB( 8, 0,15, 0)
CALL GSETRGB( 9, 7,15, 0)
CALL GSETRGB(10,10,15, 0)
CALL GSETRGB(11,15, 0, 0)
CALL GSETRGB(12,12, 0, 0)
CALL GSETRGB(13,10, 0, 0)
CALL GSETRGB(14, 7, 0, 0)
CALL GSETRGB(15,15,15,15)

IX=300
IY=380
DO 100 I=15,0,-1
  CALL GPEN(I)
  CALL GBOXF(IX,IX+50,IY,IY-20)
  IY=IY-20
100 CONTINUE

CALL TESTEND(IRET)
END
```

[ テストプログラム TEST020 ]

次は温度の分布などに利用できそうな階調表示である。

```
PROGRAM TEST020

CALL TGINIT

CALL GSETRGB( 0, 0, 0, 0)
CALL GSETRGB( 1, 4, 0, 0)
CALL GSETRGB( 2, 6, 0, 0)
CALL GSETRGB( 3, 8, 0, 0)
CALL GSETRGB( 4,10, 0, 0)
```

```

CALL GSETRGB( 5,12, 1, 0)
CALL GSETRGB( 6,14, 2, 0)
CALL GSETRGB( 7,15, 4, 0)
CALL GSETRGB( 8,15, 6, 0)
CALL GSETRGB( 9,15, 8, 0)
CALL GSETRGB(10,15,10, 0)
CALL GSETRGB(11,15,12, 0)
CALL GSETRGB(12,15,13, 6)
CALL GSETRGB(13,15,14, 8)
CALL GSETRGB(14,15,15,11)
CALL GSETRGB(15,15,15,15)

IX=300
IY=380
DO 100 I=15,0,-1
  CALL GPEN(I)
  CALL GBOXF(IX,IX+50,IY,IY-20)
  IY=IY-20
100 CONTINUE

CALL TESTEND(IRET)
END

```

[ テストプログラム TEST021 ]

今度は緑色で16階調を表示する例。15番目のペンの色は白にしておいた方が便利である。

```

PROGRAM TEST021

CALL TGINIT

CALL GSETRGB( 0, 0, 0, 0)
CALL GSETRGB( 1, 0, 3, 0)
CALL GSETRGB( 2, 0, 4, 0)
CALL GSETRGB( 3, 0, 5, 0)
CALL GSETRGB( 4, 0, 6, 0)
CALL GSETRGB( 5, 0, 7, 0)
CALL GSETRGB( 6, 1, 8, 0)
CALL GSETRGB( 7, 2,10, 0)
CALL GSETRGB( 8, 4,12, 0)
CALL GSETRGB( 9, 6,13, 0)
CALL GSETRGB(10, 7,14, 0)
CALL GSETRGB(11, 8,15, 0)
CALL GSETRGB(12,11,15, 3)
CALL GSETRGB(13,13,15, 6)
CALL GSETRGB(14,15,15,10)
CALL GSETRGB(15,15,15,15)

IX=300
IY=380
DO 200 I=15,0,-1
  CALL GPEN(I)
  CALL GBOXF(IX,IX+50,IY,IY-20)

```

```
      IY=IY-20  
200 CONTINUE
```

```
      CALL TESTEND(IRET)  
      END
```

[ テストプログラム TEST022 ]

いろいろな物質の濃度分布（放射線の量や強度にも）に利用できそうな青色系統の16階調表示。

```
PROGRAM TEST022
```

```
CALL TGINIT
```

```
CALL GSETRGB( 0, 0, 0, 0)  
CALL GSETRGB( 1, 0, 0, 6)  
CALL GSETRGB( 2, 0, 0, 8)  
CALL GSETRGB( 3, 0, 0, 10)  
CALL GSETRGB( 4, 0, 1, 12)  
CALL GSETRGB( 5, 0, 2, 14)  
CALL GSETRGB( 6, 0, 4, 15)  
CALL GSETRGB( 7, 0, 6, 15)  
CALL GSETRGB( 8, 0, 8, 15)  
CALL GSETRGB( 9, 0, 10, 15)  
CALL GSETRGB(10, 2, 11, 15)  
CALL GSETRGB(11, 3, 13, 15)  
CALL GSETRGB(12, 5, 15, 15)  
CALL GSETRGB(13, 9, 15, 15)  
CALL GSETRGB(14, 12, 15, 15)  
CALL GSETRGB(15, 15, 15, 15)
```

```
IX=300
```

```
IY=380
```

```
DO 200 I=15,0,-1
```

```
  CALL GPEN(I)
```

```
  CALL GBOXF(IX,IX+50,IY,IY-20)
```

```
  IY=IY-20
```

```
200 CONTINUE
```

```
      CALL TESTEND(IRET)  
      END
```

CALL GSYMBL ( IX,IY,CHR,KIND,MOD )

[ 機能説明 ]

現在のペン位置を基準として、グラフィックス画面に文字を描画する。ペンは現在のものを使用する。このサブルーチンの呼出で描画できるのは1文字だけである。

[ 引 数 ]

IX	4B整数	表示始点のX軸ドット座標値。 文字はこの右側に表示される。	入 力
IY	4B整数	表示始点のY軸ドット座標値。 文字はこの下側に表示される。	入 力
CHR	4B整数	文字。1文字に限る。	入 力
KIND	4B整数	文字の種類。 1.1/4角文字 2.ANK文字 3.漢字	入 力
MOD	4B整数	描画モード 0.PSET 1.PRESET 2.OR 3.AND 4.XOR	入 力

[ 注 意 ]

CRT画面制御ライブラリのグラフィックスドット座標は次の様になっている。

X軸は0(左端)～639(右端)

Y軸は0(下端)～399(上端)

実際に表示される色は描画モードによって異なる。

[ テストプログラム TEST023 ]

漢字文字列を表示する。表示位置の確認のために十字の線を引くようにしてある。  
この十字の交点が引数で指定した座標値である。

```
PROGRAM TEST023

IMPLICIT INTEGER*4 (ION)
INTEGER*4 CLEN
CHARACTER CHR*20,CHR1*2

CALL TGINIT

CHR='医用画像情報学会'
ILEN=CLEN(CHR)

IX=250
IY=200
ID=20
CALL GMOVE(0,IY)
CALL GDRAW(639,IY)
CALL GMOVE(IX,0)
CALL GDRAW(IX,399)

CALL GPEN(5)
DO 100 I=1,ILEN,2
  CHR2=CHR(I:I+1)
  CALL GSYMBL(IX,IY,CHR2,3,2)
  IX=IX+ID
100 CONTINUE

CALL TESTEND(IRET)
END
```

まずはライブラリの使用宣言

CRTに十字の線を描画

Xドット座標の0から  
639まで線を引く  
Yドット座標の0から  
399まで線を引く

ペンを5にして  
LENは文字列の長さ  
漢字なので2バイト分取り出す  
このサブルーチンで表示する  
次のX軸ドット座標に移動

[ 備考 ]

数値をグラフィックス画面に表示したい場合、サブルーチンGLINSTL,GPENのテストプログラムTEST013,TEST016で示すように、FORTRANのWRITE文を使用して、数値を数文字に変換しておく。そうすれば、この数文字を本サブルーチンGSYMBLを用いて、任意のドット座標値に表示可能となる。

このとき、10進1桁目の処理には、サブルーチンPUTZEROを用いるとよい。

CALL GVIEW ( IX1,IX2,IY1,IY2 )

[ 機能説明 ]

ビューポートを設定する。ここで設定した範囲の外側には線を引いたりできなくなる。ペンの移動は可能である。

C R T画面制御ライブラリ使用開始時点のビューポートはX軸方向が0から639、Y軸方向は0から399となっている。新たにビューポートを設定すると、それまでの設定は無効となる。

[ 引 数 ]

IX1	4B整数	ビューポートのX軸左端ドット座標値。	入 力
IX2	4B整数	ビューポートのX軸右端ドット座標値。	入 力
IY1	4B整数	ビューポートのY軸下端ドット座標値。	入 力
IY2	4B整数	ビューポートのY軸上端ドット座標値。	入 力

[ 注 意 ]

C R T画面制御ライブラリのグラフィックストット座標は次の様になっている。

X軸は0(左端) ~ 639(右端)

Y軸は0(下端) ~ 399(上端)

ビューポート内を消去

G V I E W C L R

CALL GVIEWCLR

[ 機能説明 ]

グラフィックス画面のビューポート内をクリアする。正確には、0番のペンで画面を塗りつぶしている。したがって、現在のペン0の色が黒以外であると、ビューポート内の領域全体がその色になってしまうので注意。

[ 備 考 ]

このサブルーチンはGCLR(2)と同じ動作をする。

CALL INL232C ( ICH,CHR,ICNT )

[ 機能説明 ]

RS232Cインターフェイスから受信したデータを文字列CHRに代入する。

CR(復帰・キャリッジリターン)またはLF(改行・ラインフィード)のコードを受信すると動作終了する。

[ 引 数 ]

ICH	4B整数	チャンネル番号(1,2または3)。	入 力
CHR	文字列	データを格納する文字列。	出 力
ICNT	4B整数	受信したバイト数。CR,LFは受信バイト数に含まない。	出 力

[ 注 意 ]

受信した文字列から4バイト整数や実数への変換は、利用者が行なう。

サブルーチンINS232Cの方は、受信バイト数をあらかじめ指定しておくが、INL232Cでは受信したバイト数を取得するようになっている。

RS232Cを利用するには、あらかじめサブルーチンSET232Cを用いてインターフェイスの初期設定を行なうこと。

チャンネル2,3のRS232Cを使用するときはメモリスイッチ4のビット4をONしておく必要がある。

CALL INS232C ( ICH, ISEG, IOFF, ICNT)

[ 機能説明 ]

RS232Cインターフェイスからデータを受信する。

ICNTバイトのデータを受信するまでサブルーチンは終了しない。

[ 引 数 ]

ICH	4B整数	チャンネル番号(1,2または3)。	入 力
ISEG	4B整数	データを格納する変数のセグメント・アドレス。	入 力
IOFF	4B整数	データを格納する変数のオフセット・アドレス。	入 力
ICNT	4B整数	受信するバイト数。	入 力

[ 注 意 ]

受信したデータIDATから整数、実数または文字列への変換は、利用者が行なう。

受信したデータを格納する変数のセグメントとオフセットを調べるには、サブルーチンGETADRを利用する。

サブルーチンINL232Cでは受信バイト数を取得するようになっているが、INS232Cでは、あらかじめ受信バイト数を指定するようになっている。

RS232Cを利用するには、あらかじめサブルーチンSET232Cを用いてインターフェイスの初期設定を行なっておくこと。

チャンネル2,3のRS232Cを使用するときはメモリスイッチ4のビット4をONにしておく必要がある。

CALL KEYBUF ( IBUF )

[ 機能説明 ]

PC-9801のキーボードBIOSには、キーコードデータを16個まで格納できるバッファがある。このバッファにデータがあるかどうかを検査する。

[ 引 数 ]

IBUF	4B整数	0. バッファにデータはない。 1. バッファにデータがある。	出 力
------	------	------------------------------------	-----

[ テストプログラム TEST024 ]

漢字入力の方法例。

日本語入力フロントエンド・プロセッサでは、漢字変換の確定信号によってこの入力した文字列の漢字コードをキーボードバッファに移すので、バッファの内容を読み出せば変換後の文字列を得ることができる。

PROGRAM TEST024	
IMPLICIT INTEGER*4 (I-N)	
CHARACTER STR*255	
CALL TGINIT	まずライブラリの初期化
CALL TLOCATE(18,1)	入力位置はテキスト画面の18行1列目から
J=1	文字列の代入位置は1番目から
STR=' '	文字列を空にしておく
100 CALL KINPUT(KIND, ISHIFT, ICODE)	キー入力用サブルーチン
STR(J:J)=CHAR(ICODE)	取得した文字コードを文字列に代入
CALL KEYBUF(IBUF)	バッファを検査
IF (IBUF.EQ.0) GOTO 200	空なら終わり(変換確定文字はない)
J=J+1	次の文字コードへ
GOTO 100	バッファが空になるまで繰り返す
200 CALL TLOCATE(3,1)	テキスト画面の第3行1列目から
CALL TPRINT(STR,1)	文字列STRを表示する
900 CALL TESTEND(IRET)	
END	

キーコードバッファを空にする

KEYZERO

CALL KEYZERO

[ 機能説明 ]

キーコードバッファを空にする。連続キー入力時のキーボードバッファのオーバーフローを防止に利用する。

CALL KINPUT ( KIND,ISHIFT,ICODE )

[ 機能説明 ]

キーの押下（またはマウスボタンのクリック等）があるまで待ち、該当するキーコードを出力するサブルーチン。ただし、ファンクションキーの押下は検出しない。また [COPY], [ESC], [RET], [DEL], [HELP] キーのオートリピート機能は無効にしてある。

[ 引 数 ]

KIND	4B整数	ICODEの種類を示す変数 0.機能キーのコード 1.ASCIIコード 2.シフトJISコードの1バイト目 3.シフトJISコードの2バイト目	出 力
ISHIFT	4B整数	シフトとコントロールキーの押下状態 0.[SHIFT],[CTRL]キー共に押されていない 1.[SHIFT]キーが押されている 2.[CTRL]キーが押されている	出 力
ICODE	4B整数	KIND=0：最下位バイト0,2バイト目キーコード 備考参照 KIND=1：最下位バイトASCIIコード,2バイト目0 KIND=2：シフトJISコード	出 力

[ 備 考 ]

CRT画面制御ライブラリでは、PC-9801の各キーに付けられたキー番号を機能コードと呼んでいる。それらは次の様になっている。

キーの種類	ICODEの値	マウス
[ESC]	16#1B00	右ボタンクリック
[STOP]	16#6000	
[COPY]	16#6100	
[BS]	16#0E00	

[TAB]	16#0F00	
[RET]	16#1C00	左ボタンクリック
[ROLL UP]	16#3600	
[ROLL DOWN]	16#3700	
[INS]	16#3800	
[DEL]	16#3900	
[ ]	16#3A00	上方移動
[ ]	16#3B00	
[ ]	16#3C00	
[ ]	16#3D00	下方移動
[HOME CLR]	16#3E00	
[HELP]	16#3F00	

[ テストプログラム TEST025 ]

キーを押すとそのキーコードを表示し、アスキー文字であれば表示する。[ESC]キーで終了。

```

PROGRAM TEST025

IMPLICIT INTEGER*4 (I-N)
CHARACTER C*2,LIST*79

CALL TGINIT
CALL TRANGE(1,18,1,80)
CALL TCOLOR(4,0)
100 CALL KINPUT(KIND,ISHIFT,ICODE)
   IF (KIND.EQ.1) THEN
     C=CHAR(ICODE)
     WRITE (LIST,'(I10,I10,6X,Z4,5X,A)') KIND,ISHIFT,ICODE,C
   ELSE
     WRITE (LIST,'(I10,I10,6X,Z4,5X)') KIND,ISHIFT,ICODE
   END IF
   CALL TPRINT(LIST,2)
   IF (ICODE.EQ.16#1B00) GOTO 999
   GOTO 100

999 CALL TESTEND(IRET)
END

```

CALL KREAL ( ISHIFT,ICODE )

[ 機能説明 ]

現在押下されているキーのコード（またはマウスボタンのクリックの状態）を出力するサブルーチン。ただし、本サブルーチンは十字カーソルの移動制御用に作成したため、カーソル移動キー等の特殊なキーしか検出しない。

[ 引 数 ]

ISHIFT	4B整数	シフトとコントロールキーの押下状態 0.[SHIFT],[CTRL]キー共に押されていない 1.[SHIFT]キーが押されている 2.[CTRL]キーが押されている	出 力
ICODE	4B整数	押下キーにより次の値を返す [ESC] 16#1B00を返す [RET] 16#1C00を返す [COPY] 16#6100を返す [HOME CLR] 16#3E00を返す [ ] ICODEのビット2が1となる [ ] ICODEのビット3が1となる [ ] ICODEのビット4が1となる [ ] ICODEのビット5が1となる マウス右ボタンクリック 16#1B00を返す マウス左ボタンクリック 16#1C00を返す 上記以外は16#0000を返す	出 力

[ テストプログラム TEST026 ]

カーソル移動キーを押すと、十字カーソルが移動するプログラム。[ESC]キーで終了。十字カーソルは、その一部分でもビューポートの外側にでると表示しなくなる。そこで画面9ドット分内側だけ移動するようにしている。斜め方向への移動も可能。

```

PROGRAM TEST026

IMPLICIT INTEGER*4 (I-N)

CALL TGINIT
CALL TCURSW(0)
IX1=9
IX2=630
IY1=20
IY2=390
CALL GBOXL(IX1,IX2,IY1,IY2)
IX=50
IY=50
100 CALL GCURSW(IX,IY,0)
CALL KREAL(ISHIFT,ICODE)
IF (ICODE.EQ.16#1B00) THEN
  CALL GCURSW(IX,IY,0)
  GOTO 999
ELSE
  CALL GCURSW(IX,IY,0)
  IF (BTEST(ICODE,2)) THEN
    IY=IY+1
    IF (IY.GE.IY2) IY=IY2
  END IF
  IF (BTEST(ICODE,3)) THEN
    IX=IX-1
    IF (IX.LE.IX1) IX=IX1
  END IF
  IF (BTEST(ICODE,4)) THEN
    IX=IX+1
    IF (IX.GE.IX2) IX=IX2
  END IF
  IF (BTEST(ICODE,5)) THEN
    IY=IY-1
    IF (IY.LE.IY1) IY=IY1
  END IF
END IF
CALL KEYZERO
GOTO 100

999 CALL TESTEND(IRET)
END

```

+ 字カーソルを表示

+ 字カーソルを一旦消去  
[ ]キ-の検出

[ ]キ-の検出

[ ]キ-の検出

[ ]キ-の検出

CALL LPRINT ( CHR )

[ 機能説明 ]

文字列をセントロニクス・インターフェイスを通じて、外部プリンタへ出力する。  
文字列の後に、CR(復帰)とLF(改行)のコードも出力する。

[ 引 数 ]

CHR	文字列	プリンタへ出力する文字列。	入 力
-----	-----	---------------	-----

[ 備 考 ]

サブルーチンTPRINTでは、出力先を指定することにより、CRT画面または外部プリンタへ出力するが、こちらのLPRINTは常に外部プリンタへ出力するようになっている。サブルーチンTSETPRT(プリンタの初期化と漢字ピッチの設定)の項も参照のこと。

CALL LSEEK ( HDL,BYTES,MODE,IER )

[ 機能説明 ]

サブルーチンOPENHでオープンしたファイルのファイル・ポインタ（ファイルの読み出し位置）を指定位置へ移動する。ただし、ファイルの先頭より先へは移動できない。

[ 引 数 ]

HDL	4B整数	ハンドル番号。サブルーチンOPENHで得た値。	入 力
BYTES	4B整数	移動バイト数。 BYTES<0 先頭方向に向かったのバイト数。 BYTES>0 終端方向に向かったのバイト数。 ファイルの先頭を越えることはできない。	入 力
MODE	4B整数	移動モード。 0.ファイルの先頭から。 1.ファイル・ポインタの現在位置から。 2.ファイルの終端から。	入 力
IER	4B整数	エラー番号。 0.正常終了。 1.移動モードが無効。 6.ハンドル番号が無効（ファイルがオープンされていない等）。	出 力

[ テストプログラム TEST027 ]

ファイルTEST030.DAT（サブルーチンTEST030で作成する）の先頭から64バイト目から1行分（1記録分）を読みだして表示する。64バイト目からは、空白2文字があり、そのあと

3 ..... C

となっているはずである。1記録分の文字は30文字であるが、0DH,0AHがついていて32文字分になっていることに注意。

PROGRAM TEST027

IMPLICIT INTEGER\*4 (I-N)  
INTEGER\*4 CLEN  
CHARACTER FILE\*50,CHR\*80

CALL TGINIT

FILE='B:TEST030.DAT'

CALL OPENH(FILE,HANDLE,IER)

IF (IER.NE.0) GOTO 999

IMODE=0

IBYTES=64

CALL LSEEK(HANDLE,IBYTES,MODE,IER)

CALL GETLIN(HANDLE,CHR,LBYTES)

WRITE (\*,\*) CHR(1:LBYTES-2)

999 WRITE (\*,'(A9,I4)') ' ERROR = ',IER

CALL TESTEND(IRET)

END

まずはライブラリの初期化。これ  
を実行しないと IER 検出不可  
ファイル名は B:¥TEST030.DAT  
ファイルを開く  
エラーなら 999 へ  
ファイル先頭から数えて  
64 バイトから読み出す  
ファイルポインタを移動  
ファイルの中味を CHR に取得  
CR, LF を除いて表示  
エラー番号を表示して終了

CALL MKDIR ( PATH, IER )

[ 機能説明 ]

指定した名前のサブ・ディレクトリを作成する。現在あるディレクトリの下に作成するので、例えばドライブBにABCというディレクトリが存在しない場合、

B:¥ABC¥TEST

というような指定はできない。このような場合は、まずABCというディレクトリをドライブBに作成しておく必要がある。

[ 引 数 ]

PATH	文字列	ディレクトリ名。50文字以内で指定する。 B:¥ABC¥TEST のように指定する。	入 力
IER	4B整数	エラー番号。 0.正常終了。 3.ディレクトリ名が無効。 5.ディスクフルか、同名のディレクトリがすでに ある。	出 力

[ テストプログラム TEST028 ]

ドライブBのディスクにディレクトリABCを作成する。

```

PROGRAM TEST028

INTEGER*4 IER
CHARACTER PATH*50

CALL TGINIT

PATH='B:¥ABC'
CALL MKDIR(PATH, IER)
WRITE(*,*) IER

CALL TESTEND(IRET)
END

```

[ テストプログラム TEST029 ]

ドライブBの磁気ディスクのディレクトリABCの下に、TESTというサブ・ディレクトリを作成する。もし、ABCというディレクトリがないとエラー。

このプログラムを実行する前にTEST028を実行していればエラーとはならない。

```
PROGRAM TEST029

INTEGER*4 IER
CHARACTER PATH*50

CALL TGINIT

PATH='B:¥ABC¥TEST'
CALL MKDIR(PATH, IER)
WRITE(*,*) IER

CALL TESTEND(IER)
END
```

CALL MSMOVE ( IX, IY )

[ 機能説明 ]

マウスカーソルを指定したグラフィックドット座標値へ移動する。

[ 引 数 ]

IX	4B整数	移動先のX軸ドット座標値。	入 力
IY	4B整数	移動先のY軸ドット座標値。	入 力

[ 注 意 ]

C R T 画面制御ライブラリのグラフィックドット座標は次の様になっている。

X軸は0(左端) ~ 639(右端)

Y軸は0(下端) ~ 399(上端)

C R T 画面制御ライブラリでは、マウスカーソルは表示しない。

このサブルーチンMSMOVEは、マウスを使用して、C R T 画面制御ライブラリ内蔵の十字カーソルを制御するために使用する。

CALL MSPOS ( IX,IY )

[ 機能説明 ]

マウスカーソルの現在位置を調べて返す。マウスが使用不可の状態の場合(マウス・ドライバがセットされていない等)は現在のペン位置を返す。

[ 引 数 ]

IX	4B整数	マウスカーソルのX軸ドット座標値。	出 力
IY	4B整数	マウスカーソルのY軸ドット座標値。	出 力

[ 注 意 ]

C R T画面制御ライブラリのグラフィックドット座標は次の様になっている。

X軸は0(左端)～639(右端)

Y軸は0(下端)～399(上端)

C R T画面制御ライブラリでは、マウスカーソルを表示しない。このサブルーチンMSPOSは、マウスを利用して、C R T画面制御ライブラリ内蔵の十字カーソルの表示位置を制御をするために使用する。

CALL MSSPEED ( ISX, ISY )

[ 機能説明 ]

マウスカーソルが8ドット移動するのに必要なマウスの移動距離を定義する。

[ 引 数 ]

ISX	4B整数	X軸方向の感度。 標準では8になっている。	入 力
ISY	4B整数	Y軸方向の感度。 標準では8になっている。	入 力

CALL MSVIEW ( IX1,IX2,IY1,IY2 )

[ 機能説明 ]

C R T 画面上においてマウスカーソルが移動可能な領域を設定する。マウスカーソルはここで設定した領域を越えて移動できなくなる。C R T 画面制御ライブラリ使用開始時点での移動可能範囲はX軸方向が0から639、Y軸方向は0から399となっている。新たに設定すると、それまでの設定は無効となる。

[ 引 数 ]

IX1	4B整数	移動限界左端のX軸ドット座標値。	入 力
IX2	4B整数	移動限界右端のX軸ドット座標値。	入 力
IY1	4B整数	移動限界下端のY軸ドット座標値。	入 力
IY2	4B整数	移動限界上端のY軸ドット座標値。	入 力

[ 注 意 ]

C R T 画面制御ライブラリのグラフィックストット座標は次の様になっている。

X軸は0(左端) ~ 639(右端)

Y軸は0(下端) ~ 399(上端)

CALL OPENH ( PATH,HDL,IER )

[ 機能説明 ]

PATH名で指定するファイルを、読み書き可能なモードでオープンする。

このサブルーチンを実行すると、ファイルポインタはファイルの一番先頭になる。

[ 引 数 ]

PATH	文字列	パス名。50字以内とする。	入 力
HDL	4B整数	ハンドルの値を返す。	出 力
IER	4B整数	エラー番号。 0.正常終了。 2.ファイル名が無効か存在しない。 3.パス名が無効か存在しない。 4.ディレクトリをオープンしようとした。	出 力

CALL OUTL232C ( ICH,CHR )

[ 機能説明 ]

RS232Cインターフェイスを通じて文字列CHRを送信する。

文字列の実長さ(後部の空白を除いた部分)を送信したあと、CR(復帰・キャリッジリターン)を自動的に送信して動作終了となる。

[ 引 数 ]

ICH	4B整数	チャンネル番号(1,2または3)。	入 力
CHR	文字列	送信する文字列。	入 力

[ 注 意 ]

整数や実数は、文字列に変換してから送信することもできる。

RS232Cを利用するには、あらかじめサブルーチンSET232Cを用いてインターフェイスの初期設定を行なうこと。

チャンネル2,3のRS232Cを使用するときはメモリスイッチ4のビット4をONにしておく必要がある。

CALL OUTS232C ( ICH, ISEG, IOFF, ICNT )

[ 機能説明 ]

RS232Cインターフェイスを通じてデータを送信する。

ICNTバイトのデータを送信するまでサブルーチンは終了しない。

[ 引 数 ]

ICH	4B整数	チャンネル番号(1,2または3)。	入 力
ISEG	4B整数	送信データを格納してある変数のセグメント。	入 力
IOFF	4B整数	送信データを格納してある変数のオフセット。	入 力
ICNT	4B整数	送信するバイト数。	入 力

[ 注 意 ]

送信するデータが格納してある変数のセグメントとオフセットを調べるには、サブルーチンGETADRを利用する。

サブルーチンOUTL232Cでは送信バイト数は自動的に文字列の実長さとなるが、本サブルーチンOUTS232Cでは、あらかじめ送信バイト数を指定するようになっている。

RS232Cを利用するには、あらかじめサブルーチンSET232Cを用いてインターフェイスの初期設定を行なうこと。

チャンネル2,3のRS232Cを使用するときはメモリスイッチ4のビット4をONにしておく必要がある。

メモリから1バイトデータを読み出す

P E E K B

CALL PEEKB ( ISEG, IOFF, IDAT )

[ 機能説明 ]

指定メモリ番地のデータを1バイト分読み出して、変数IDATに代入する。

[ 引 数 ]

ISEG	4B整数	セグメントアドレス	入 力
IOFF	4B整数	オフセットアドレス	入 力
IDAT	4B整数	読み出したバイトデータ	出 力

[ テストプログラム TEST008 ]

MS/FORTRANには1バイト整数の宣言文があるが、ここではテストプログラムとして、4バイト整数配列の各要素を1バイトずつ分割して扱ってみる。1バイトなので0~255までの数値しか扱えない。

このプログラムでは、まず0~255までの整数をPOKEBによってメモリに記録し、確認のために、PEEKBを用いて再び読み出して表示している。

```

PROGRAM TEST008

INTEGER*4  I, II, ISEG, IOFF, IADR, IA(64)  1バイトデータなので64*4=256

CALL GETADR(IA, ISEG, IOFF)                配列変数IAの先頭アドレスを取得

DO 100 I=0,255
  IADR=IOFF+I
  CALL POKEB( ISEG, IADR, I)               0~255の数値をメモリへ書き込む
  IADRは書き込むメモリの番地
  1バイトデータをメモリへ書き込む
100 CONTINUE

DO 200 I=0,255
  IADR=IOFF+I
  CALL PEEKB( ISEG, IADR, II)              256回数値を読み出す
  WRITE (*, *) II                          1バイトデータをメモリから読み出す
  それを表示する
200 CONTINUE

END

```

メモリから2バイトデータを読み出す

P E E K W

CALL PEEKW ( ISEG,IOFF,IDAT )

[ 機能説明 ]

指定メモリ番地のデータを1ワード(2バイト)分読み出して、変数IDATに代入する。

[ 引 数 ]

ISEG	4B整数	セグメントアドレス	入 力
IOFF	4B整数	オフセットアドレス	入 力
IDAT	4B整数	読み出したワードデータ	出 力

# メモリに1バイトデータを書き込む

POKEB

CALL POKEB ( ISEG, IOFF, IDAT )

## [ 機能説明 ]

指定メモリ番地にバイトデータを書き込む。

## [ 引 数 ]

ISEG	4B整数	セグメントアドレス	入 力
IOFF	4B整数	オフセットアドレス	入 力
IDAT	4B整数	書き込むバイトデータ	入 力

## [ テストプログラム TEST008 ]

MS/FORTRANには1バイト整数の宣言文があるが、ここではテストプログラムとして、4バイト整数配列の各要素を1バイトずつ分割して扱ってみる。1バイトなので0~255までの数値しか扱えない。

このプログラムでは、まず0~255までの整数をPOKEBによってメモリに記録し、確認のために、PEEKBを用いて再び読み出して表示している。

PROGRAM TEST008	
INTEGER*4 I, II, ISEG, IOFF, IADR, IA(64)	1バイトデータなので64*4=256
CALL GETADR(IA, ISEG, IOFF)	配列変数IAの先頭アドレスを取得
DO 100 I=0,255 IADR=IOFF+I CALL POKEB( ISEG, IADR, I)	0~255の数値をメモリへ書き込む IADRは書き込むメモリの番地 1バイトデータをメモリへ書き込む
100 CONTINUE	
DO 200 I=0,255 IADR=IOFF+I CALL PEEKB( ISEG, IADR, II)	256回数値を読み出す
WRITE (*,*) II	1バイトデータをメモリから読み出す それを表示する
200 CONTINUE	
END	

メモリに2バイトデータを書き込む

P O K E W

CALL POKEW ( ISEG,IOFF,IDAT )

[ 機能説明 ]

指定メモリ番地にワード(2バイト分)データを書き込む。

[ 引 数 ]

ISEG	4B整数	セグメントアドレス	入 力
IOFF	4B整数	オフセットアドレス	入 力
IDAT	4B整数	書き込むワードデータ	入 力

I / O ポートから 1 バイトデータを読み出す

P O R T I N

CALL PORTIN ( IPORT, IDATA )

[ 機能説明 ]

指定 I/O ポートからバイトデータを、変数 IDATA に読み出す。

[ 引 数 ]

I PORT	4B 整数	I/O ポート番号	入 力
I DATA	4B 整数	読み出したバイトデータ	出 力

I / O ポートに 1 バイトデータを書き込む

P O R T O U T B

CALL PORTOUTB ( IPORT, IDATA )

[ 機能説明 ]

指定 I/O ポートにバイトデータを送出する。

[ 引 数 ]

I PORT	4B 整数	I/O ポート番号	入 力
I DATA	4B 整数	書き込むバイトデータ	入 力

I / O ポートに 2 バイトデータを書き込む

P O R T O U T W

CALL PORTOUTW ( IPORT, IDATA )

[ 機能説明 ]

指定 I/Oポートにワード（ 2 バイト分 ）データを送出する。

[ 引 数 ]

I PORT	4B整数	I/Oポート番号	入 力
I DATA	4B整数	書き込むワードデータ	入 力

CALL PRINTER ( IPRT )

[ 機能説明 ]

サブルーチンTPRINTで出力する文字列の出力先をテキスト画面にするか、外部プリンタにするかを指定する。

このサブルーチンで出力先を指定しなければ、自動的にテキスト画面出力となる。

[ 引 数 ]

IPRT	4B整数	出力先の番号 1.テキスト画面 2.外部プリンタ	入 力
------	------	--------------------------------	-----

[ 備 考 ]

このサブルーチンで指定した出力先はサブルーチンTGETPRTで検査出来る。

CALL PUTLIN ( HDL,CHR,N )

[ 機能説明 ]

サブルーチン OPENHでオープンしたファイルに、文字列CHRを出力（記録）する。出力したあと、ファイル・ポインタは次の記録位置へ自動的に移動する。

[ 引 数 ]

HDL	4B整数	ハンドル番号。サブルーチンOPENHで得た値。	入 力
CHR	文字列	記録する文字列。255バイト以内。 文字列の後ろに0DH(CR),0AH(LF)が付く。	入 力
N	4B整数	記録した文字数をバイト値で返す。 ディスクに空き容量がないときは0を返す。 記録に失敗すると9999を返す。	出 力

[ テストプログラム TEST030 ]

ドライブ Bのディスク上に、新たにTEST030.DATというファイルを作成し、文字列配列CHRの中味を記録する。

```

PROGRAM TEST030

INTEGER*4  I,N,IN,IER,HANDLE
CHARACTER  FILE*50,CHR(50)*80

CALL TGINIT

FILE='B:TEST030.DAT'
CALL OPENH(FILE,HANDLE,IER)
CALL CREATH(FILE,HANDLE,IER)
IF (IER.NE.0) GOTO 999

N = 8
CHR(1)=' 1 ..... A'
CHR(2)=' 2 ..... B'
CHR(3)=' 3 ..... C'
CHR(4)=' 4 ..... D'
CHR(5)=' 5 ..... E'
CHR(6)=' 6 ..... F'
CHR(7)=' 7 ..... G'
CHR(8)=' 8 ..... H'
    
```

ディスクエラー検出可能にする  
データファイル名はB:TEST030.DAT  
ファイルを開く  
ファイルは新規作成である  
新規作成に失敗したら999へ  
データ数は8

DO 100 I=1,N	N個のデータを
CALL PUTLIN(HANDLE,CHR(I)(1:30),IN)	HANDLE番目のファイルに記録する
100 CONTINUE	
IF (IN.EQ.9999) GOTO 999	記録に失敗したら999へ
CALL CLOSEH(HANDLE,IER)	記録終了したらファイルを閉じる
999 WRITE (*,*) IN	記録バイト数(最後のデータ分)
CALL TESTEND(IRET)	プログラム終了
END	

[ 備 考 ]

サブルーチンLSEEKに、このテストプログラムで記録したデータを読み出すプログラムの例がある(TEST027)。

CALL PUTZERO ( CHR )

[ 機能説明 ]

MS/FORTRANでは数値Xが

$$-1.0 < X < 1.0$$

の範囲のとき、書式FまたはEで数値を出力すると、10進の1桁目が空白または省略されてしまい体裁がよくない。このサブルーチンは、内部ファイルに読み込んだ書式付数文字列FORMが上記の数値の範囲のものであれば、10進1桁目に0を置く。それ以外の場合は何もしない。

[ 引 数 ]

FORM	文字列	数文字列が代入されている文字列。 10進1桁目に0がなければ付けて返す。	入出力
------	-----	---	-----

[ テストプログラム TEST031 ]

数値-0.958を内部ファイルCHRに、数文字列として出力したあと、そのままの形で表示した場合と、このサブルーチンPUTZEROを用いて整えたものと比較する。

```

PROGRAM TEST031

REAL*4    X
CHARACTER CHR*30

X=-0.958
WRITE(CHR, '(F10.4)') X
WRITE(*,*) CHR
CALL PUTZERO(CHR)
WRITE(*,*) CHR

END

```

数値Xを数文字列に変換  
結果を表示する  
10進1桁目に0を付ける  
結果を表示する

[ 備 考 ]

このプログラムの手法と、サブルーチンGSYMBLを利用すると、座標軸の目盛数字を、書式を整えて、グラフィックス画面の任意位置に表示可能となる。

## ファイル名の変更

## R E N A M E

CALL RENAME ( OLD,NEW,IER )

### [ 機能説明 ]

OLDで指定する名前のファイルを、NEWで指定する名前に変更する。新しいファイル名は異なるディレクトリ内にも作れるが、異なるドライブ上には作れない。また、新しいファイルができると、古いファイルは削除される。ワイルド・カードは使用できない。

### [ 引 数 ]

OLD	文字列	現在のパス名。50字以内。	入 力
NEW	文字列	新しいパス名。50字以内。	入 力
IER	2B整数	エラー番号。 0.正常終了。 2.旧パス名のファイルがない。 5.アクセスが否定された。 17.旧パス名と新パス名に指定したドライブが同じでない。	出 力

CALL RMDIR ( PATH, IER )

[ 機能説明 ]

PATHで指定するディレクトリを削除する。指定したディレクトリの下に別のサブ・ディレクトリがある場合はエラーとなる（エラー番号の5を返す）。

[ 引 数 ]

PATH	文字列	ディレクトリ名。50字以内とする。たとえば B:¥ABC¥TEST のようにする。	入 力
IER	4B整数	エラー番号。 0.正常終了。 3.ディレクトリ名が無効。 5.ディレクトリが空でない。 ディレクトリがない。 ディレクトリがルート・ディレクトリである (ルートディレクトリは削除できない) 16.カレント・ディレクトリである。 (自分自身は削除できない)	出 力

CALL SCREEN ( IACT, IDSP )

[ 機能説明 ]

CRT画面制御ライブラリでは、16色表示可能なグラフィック画面を2枚利用できる。本サブルーチンは、グラフィックに関するの命令を実行する画面番号と実際にユーザが目で見ることのできる画面番号を指定する。

[ 引 数 ]

IACT	4B整数	命令が有効となる画面番号。 1か2。	入 力
IDISP	4B整数	実際に目で見ることの出来る画面番号。 0. どの画面も見えなくする。 1. 1番目の画面を表示する。 2. 2番目の画面を表示する。	入 力

CALL SET232C ( ICH,BPS,BITS,PARITY,STOP,IMS,IER )

[ 機能説明 ]

RS-232Cインターフェースの初期化。チャンネル1はメモリスイッチも書き換える。

[ 引 数 ]

ICH	4B整数	チャンネル番号。1,2または3。	入 力
BPS	4B整数	ボーレート。下記のいずれかの値。 150,300,600,1200,2400,4800,9600 ICH=2,3では拡張ボード上で行なうのでダミー。	入 力
BITS	4B整数	キャラクタ長。7か8。	入 力
PARITY	4B整数	パリティチェック。 0.パリティチェックをしない。 1.奇数パリティでチェックをする。 2.偶数パリティでチェックをする。	入 力
STOP	4B整数	ストップビット 1.ストップビットを1にする。 2.ストップビットを2にする。	入 力
IMS	4B整数	送受信待ち時間。ミリ秒で指定。10未満の値にすると送受信があるまで永久に待つ。	入 力
IER	4B整数	エラー番号。0なら正常終了。それ以外エラー。 エラーとなるのはインターフェースが未接続か外部機器の電源がOFFの場合だけである。	出 力

[ 注 意 ]

RS232Cインターフェースを利用する場合は、予めこのサブルーチンを用いて、インターフェースを初期化しておくこと。

チャンネル2,3のRS232Cを使用するときはメモリスイッチ4のビット4をONにしておく必要がある。

CALL STOPSW

[ 機能説明 ]

システムの[STOP]キーを無効にしたり元に戻したりする。このサブルーチンを実行すると[STOP]キーは無効になる。もう一度実行すると有効になる。

[ 備 考 ]

このサブルーチンでシステムの[STOP]を無効にすると、[STOP]キーによるハードディスクのリトラクト動作ができなくなる。

通常の用途で、このサブルーチンを使用する必要はない。このサブルーチンを使用しなくても、CRT画面制御ライブラリではTGINITを実行すると、自動的にMS-DOS上の[CTRL]+[C]と[STOP]キーは無効となる。

[STOP]キーを無効にしたままプログラムを終了してしまった場合は、ライブラリ供給時に提供するSTOPSWを実行するか、[CTRL]+[f・6],[CTRL]+[f・7]等を実行して正常なMS-DOSの状態に戻す。

指定時間だけブザーを鳴らす

T B E E P

CALL TBEEP( IMSEC )

[ 機能説明 ]

指定ミリ秒だけ内蔵スピーカを鳴らす。20ミリ秒以下の値は無視される（ブザーは鳴らない）。

[ 引 数 ]

IMSEC	4B整数	ブザーを鳴らす時間で、ミリ秒値。 最大約15秒まで可能。	入 力
-------	------	---------------------------------	-----

CALL TCHMOD ( IYT,IYB,IXL,IXR,IU,IR,IB,IS,ICOLOR )

[ 機能説明 ]

テキスト画面の第IYT行から第IYB行までの、IXL列からIXR列に囲まれた範囲の文字の表示色と属性を変更する。

[ 引 数 ]

IYT	4B整数	変更範囲の上の行番号。	入 力
IYB	4B整数	変更範囲の下の行番号。	入 力
IXL	4B整数	変更範囲の左側の列番号。	入 力
IXR	4B整数	変更範囲の右側の列番号。	入 力
IU	4B整数	1.アンダーライン 0.付けない。	入 力
IR	4B整数	1.リバーズ 0.しない。	入 力
IB	4B整数	1.ブリンク 0.させない。	入 力
IS	4B整数	1.シークレット 0.表示する。	入 力
ICOLOR	4B整数	文字の色。0~7の数値に応じて以下の値。 0.黒 1.青 2.赤 3.紫 4.緑 5.水色 6.黄 7.白	入 力

[ 注 意 ]

PC-9801RAには、文字列のブリンク(点滅)機能はない。常にIB=0で使用する。

[ テストプログラム TEST032 ]

5 ~ 10行をスクロール範囲とし、文字列CHRを緑色で50行表示する。その後、第7行から9行と14 ~ 31列に囲まれた文字を黄色のリバーズで表示しなおす。

PROGRAM TEST032	
INTEGER*4 I, IU, IL	
CHARACTER CHR*80	
CALL TGINIT	ライブラリの使用宣言(初期化)
CALL TRANGE(5, 10, 1, 80)	5 ~ 10行の間にスクロール表示
CHR='応用物理学会 医用画像情報学会 日本放射線技術学会'	
CALL TCOLOR(4, 0)	文字の色は4の緑
DO 100 I=1, 50	50行分表示
CALL TPRINT(CHR, 1)	CHRが80バイトなのでTPRINTの第2
100 CONTINUE	引数が1でも自動的にLFする。
CALL TCHMOD(7, 9, 14, 31, 0, 1, 0, 0, 6)	指定領域の色と属性を変更
CALL TWAIT(2000)	2秒間表示する
CALL TGEND	ライブラリ利用終了。MS-DOSに戻る
END	

CALL TCOLOR ( ICOLOR,MODE )

[ 機能説明 ]

これからCRT画面に表示する文字の色、属性を設定する。この設定は、サブルーチンTPRINTを用いて表示する文字に対して動作する。

表示済みの文字の属性を変更するには、TCHMODを使用する。

[ 引 数 ]

ICOLOR	4B整数	表示色。 0.黒    1.青    2.赤    3.紫 4.緑    5.水色   6.黄    7.白	入 力
MODE	4B整数	表示の属性。 0.ノーマル(通常表示) 1.ブリンク(点滅表示) 2.リバーズ(反転表示) 3.リバーズ+ブリンク 4.アンダーライン(下線を付ける)	入 力

[ 注 意 ]

PC-9801RAには、文字列のブリンク(点滅)機能はない。

[ テストプログラム TEST033 ]

文字列CHRを50回表示するが、表示に先だって文字の色を4の緑に設定しておく。

```

PROGRAM TEST033

INTEGER*4 I
CHARACTER CHR*40

CALL TGINIT
CALL TRANGE(5,10,1,80)

CHR='(社)日本放射線技師学会'
CALL TCOLOR(4,0)
DO 100 I=1,50
    CALL TPRINT(CHR,2)
100 CONTINUE
    
```

ライブラリ使用宣言(初期化)  
表示範囲を5~10行とする

4は緑色

数値2はCR,LF  
10行目は空白となる

CALL TWAIT(2000)

2秒間表示して終了

CALL TGEND

画面消去

END

CALL TCURMOV ( I )

[ 機能説明 ]

Iの値に応じて、カーソルの移動や文字の削除、スクロール範囲のクリアなどを行う。

[ 引 数 ]

I	4B整数	<p>0.カーソルから行末までを消去。</p> <p>1.8文字分のスペースを出力。</p> <p>2.復帰改行(CR,LF)。</p> <p>3.カーソルをスクロール範囲の左上隅に移動。</p> <p>4.スクロール範囲内のクリア。</p> <p>カーソルはスクロール範囲の左上に移動。</p> <p>5.カーソルを行先頭に移動(復帰)。</p> <p>6.カーソルのある行を削除。行は詰められる。</p> <p>7.カーソルを右に移動。漢字は2桁分移動。</p> <p>8.カーソルを左に移動。漢字は2桁分移動。</p> <p>9.カーソルを上を移動。現在のカーソル位置がスクロール最上行なら下に1行スクロールする</p> <p>10.カーソルを下を移動。現在のカーソル位置が再下行なら上に1行スクロールする。</p> <p>11.カーソル行に空白行を挿入</p>	入 力
---	------	--	-----

[ 備 考 ]

引数Iを4にする動作はよく使用する。これは現在のスクロール範囲内だけのテキスト画面情報を消去する。

CALL TCURPOS ( IY,IX )

[ 機能説明 ]

現在のテキストカーソルがテキスト画面の第何行目の何列目にあるかを調べて返す。

[ 引 数 ]

IY	4B整数	現在カーソルがある行番号。	出 力
IX	4B整数	現在カーソルがある列番号。	出 力

[ テストプログラム TEST034 ]

文字列を50回表示したあと、最終的にカーソルがどこにあるかを調べて表示する。  
内部ファイルCLISTを使用し、桁数を整えて表示する。

```

PROGRAM TEST034

IMPLICIT INTEGER*4 (I-N)
CHARACTER CHR*40,CLIST*80

CHR=' ( 社 ) 日本放射線技師学会 '
CALL TGINIT
CALL TRANGE(5,10,1,80)

CALL TCOLOR(4,0)
DO 100 I=1,50
    CALL TPRINT(CHR,2)
100 CONTINUE

CALL TPOS(IY,IX)
WRITE(CLIST,'(A10,I2,A2,I2,A4)') 'カーソルは',IY,'行',IX,'桁目'
CALL TPRINT(CLIST,1)
CALL TLOCATE(IY,IX)
CALL TWAIT(2000)
                                     文字列表示でカーソル位置が変更したので戻す

CALL TGEND
END
    
```

CALL TCURSW ( ISW )

[ 機能説明 ]

テキスト画面上にカーソルを表示しないようにしたり、再び表示したりする。

[ 引 数 ]

ISW	4B整数	0.カーソルを消去する。 1.カーソルを表示する。	入 力
-----	------	------------------------------	-----

[ 注 意 ]

カーソルを表示しないままプログラムを終了すると、MS-DOSに戻ってもカーソルは消えたままになる。サブルーチンTGENDまたはTESTENDでは、システムを初期状態に戻すようになっているので、プログラム最後にはこれらのサブルーチンを実行しておくといよい。

CALL TDATA1 ( X,N,MAX,LIST,LN,ILN,FORM,IEDIT,MODE )

[ 機能説明 ]

1次元のデータ入力用サブルーチン。入力する文字と入力データ箇所を示す反転カーソルの色は白に固定してある。既に入力済みのデータの修正にも使用する。

本サブルーチンは[ESC]キーで終了する。

[ 引 数 ]

X	8B実数配列	データ用1次元配列。要素数はMAX以上のこと。 空白入力の箇所には該当要素に次の数値を返す。 9.99999999D+99(9ナインの99乗) 整数型入力の場合は8バイト実数に変換した数値	入出力
N	4B整数	入力済みデータ数。 新規入力の場合には0にしておくこと。 空白入力を許可するモードではMAXを返す。	入出力
MAX	4B整数	入力可能とする最大データ数。	入 力
LIST	文字列配列	データのスクロール表示用文字列配列。要素数はMAX以上のこと。 文字入力の結果はこの文字列配列から取り出す。	入出力
LN	4B整数	データ番号の表示開始列位置。データ番号の表示には4桁を使用する。 この値が0のときはデータ番号を表示しない。	入 力
ILN	4B整数	データを表示開始するCRT上の列位置。	入 力
FORM	文字列	データの表示形式を指定する文字列。 可能なのは以下の様にE,F,I,Aの4つの型のみ。	入 力

書式の例	表示形式	入力桁数
(1PE10.3)	浮動小数点表示	実数 10桁
(F12.3)	固定小数点表示	実数 12桁
(I5)	整数型の表示	整数 5桁
(A30)	文字列	文字 30字

IEDIT	4B整数	データの入力、修正フラグ(文字列入力では無効) 0.新規入力または修正があった。 1.新規入力または修正あり。 再計算の必要性チェックに利用する。	出力
MODE	4B整数	次の数値で入力モードを指定する。 0.行の挿入、削除、空白入力などは不可。 1.行の挿入、削除だけを許可。 2.空白の入力だけを許可。 3.行の挿入、削除、空白の入力を許可。 空白データは欠損値データとして扱える。	入力

[ 注 意 ]

表示書式について

表示書式F,Iでは、指定した書式によって入力可能な整数部の桁数が決まる。この桁数を越えての入力はできない。

この桁数を越えて入力すると、表示桁全体が\*(アスタリスク)で埋まる。この状態では、入力位置を示す反転カーソルは移動しなくなる。すなわち正しい入力をしていない限り、入力位置は変更できないし、サブルーチンも終了不可能である。

行の挿入について

空白入力が不可能なモードの場合、新しい行を挿入すると、何らかの入力をするかその行を削除しない限り、入力位置の変更はできない。

数値データの修正について

データは、[RET]キーを押しただけでは修正されない場合がある。

このサブルーチンでは、実際にキーボードから入力した数値を8バイト実数配列に代入している。画面には表示書式に応じた四捨五入値を表示する。

書式'(F6.3)'の場合を例にして、このことを説明する。

手順	キー入力	表示	実際の値	備考
1	2.3005[RET]	2.301	2.3005	表示と実際値が違う
2	[RET]	2.301	2.3005	変化なし
3	[HOME CLR][RET]			結果はMODEの値による
4	2.301[RET]	2.301	2.3010	表示と実際の値が同じとなる

[ 編集用キー ]

データの入力編集には以下のキーを使用する。

押下キー	動作内容
[ESC]	サブルーチンを終了する。
[COPY]	CRT画面のハードコピー
[RET]	現在入力中のデータを確定。
[ROLL UP]	表示を上スクロール。
[ROLL DOWN]	表示を下スクロール。
[BS]	カーソルの左の文字を削除
[INS]	入力の挿入/上書きの切り替え。
[DEL]	カーソル位置の文字を削除
[↑]	入力位置を1行上に移動
[←]	カーソルを1桁左に移動
[→]	カーソルを1桁右に移動
[↓]	入力位置を1行下に移動
[HOME CLR]	入力位置のデータを消去(クリア)
[SHIFT]+[INS]	反転カーソル行に空白行を挿入
[SHIFT]+[DEL]	反転カーソル行を削除

[ テストプログラム TEST035 ]

入力終了後、確認のために入力したデータ数とデータそのもの、そして比較のために表示用の文字列配列も表示するようにした。

```

PROGRAM TEST035

IMPLICIT INTEGER*4 (I-N)
REAL*8      X(20)
CHARACTER LIST(20)*30,FORM*30,TITLE*26

CALL TGINIT                                ライブラリの初期化(使用宣言)

TITLE='          番号          測定値          '   タイトルを決める
CALL TDISPM(2,1,TITLE,6,4)                   そのタイトルを表示
CALL TRANGE(3,18,1,80)                       データ表示範囲を3~18行とする

MAX=20                                       20個のデータ入力を可能にする
ILN=15                                       データは画面15列目から表示する
FORM='(1PE10.3) '                            浮動小数点で10桁まで入力可
N=0                                           新規入力なのでデータ数は0
DO 100 I=1,MAX                               未入力箇所に仮データを代入
100 X(I)=9.99999999D+99
    
```

```
LN=5
IER=1
CALL TDATA1(X,N,MAX,LIST,LN,ILN,FORM,IEDIT,IER)

CALL TGEND
WRITE(*,*) ' 入力したデータ数 = ',N
DO 200 I=1,MAX
WRITE (*, '(1PE15.8,3X,A20)') X(I),LIST(I)(ILN:ILN+15)
200 CONTINUE
END
```

データ番号を画面5桁目から表示  
データの挿入と削除を可能にする

ライブラリ使用終了

CALL T DATAN ( X,N,MAX,NC,LIST,LN,ILN,FORM,IEDIT,MODE )

[ 機能説明 ]

任意次元のデータ入力用サブルーチン。入力する文字と入力データ箇所を示す反転カーソルの色は白に固定してある。既に入力済みのデータの修正にも使用する。

本サブルーチンは[ESC]キーで終了する。

[ 引 数 ]

X	8B実数配列	データ用NC次元配列。 大きさの定義はX(N,NC)としておくこと。 空白入力の箇所には該当要素に次の数値を返す。 9.99999999D+99(9ナインの99乗) 整数型入力の場合は8バイト実数に変換した数	入出力						
N	4B整数	入力済みデータ組数。 新規入力の場合には0にしておくこと。 空白入力を許可するモードではMAXを返す。	入出力						
MAX	4B整数	入力可能とする最大データ組数。	入 力						
NC	4B整数	次元数。							
LIST	文字列配列	データのスクロール表示用文字列配列。要素数はMAX以上のこと。 文字入力の結果はこの文字列配列から取り出す。	入出力						
LN	4B整数	データ番号の表示開始列位置。データ番号の表示には4桁を使用する。 この値が0のときはデータ番号を表示しない。	入 力						
ILN	4B整数配列	データを表示開始するCRT上の列位置。	入 力						
FORM	文字列配列	データの表示形式を指定する文字列配列。 可能なのは以下の様にE,F,I,Aの4つの型のみ。	入 力						
		<table border="1"> <thead> <tr> <th>書式の例</th> <th>表示形式</th> <th>入力桁数</th> </tr> </thead> <tbody> <tr> <td>(1PE10.3)</td> <td>浮動小数点表示</td> <td>実数10桁</td> </tr> </tbody> </table>	書式の例	表示形式	入力桁数	(1PE10.3)	浮動小数点表示	実数10桁	
書式の例	表示形式	入力桁数							
(1PE10.3)	浮動小数点表示	実数10桁							

		<table border="1"> <tr> <td>(F12.3)</td> <td>固定小数点表示</td> <td>実数 12桁</td> </tr> <tr> <td>(I5)</td> <td>整数型の表示</td> <td>整数 5桁</td> </tr> <tr> <td>(A30)</td> <td>文字列</td> <td>文字 30字</td> </tr> </table>	(F12.3)	固定小数点表示	実数 12桁	(I5)	整数型の表示	整数 5桁	(A30)	文字列	文字 30字	
(F12.3)	固定小数点表示	実数 12桁										
(I5)	整数型の表示	整数 5桁										
(A30)	文字列	文字 30字										
IEDIT	4B整数	<p>データの入力、修正フラグ(文字列入力では無効)</p> <p>0.新規入力または修正があった。</p> <p>1.新規入力または修正あり。</p> <p>再計算の必要性チェックに利用する。</p>	出力									
MODE	4B整数	<p>次の数値で入力モードを指定する。</p> <p>0.行の挿入、削除、空白入力などは不可。</p> <p>1.行の挿入、削除だけを許可。</p> <p>2.空白の入力だけを許可。</p> <p>3.行の挿入、削除、空白の入力を許可。</p> <p>空白データは欠損値データとして扱える。</p>	入力									

[ 注 意 ]

表示書式について

表示書式F,Iでは、指定した書式によって入力可能な整数部の桁数が決まる。この桁数を越えての入力はできない。この桁数を越えて入力すると、表示桁全体が\* (アスタリスク)で埋まる。この状態では、正しい入力をしないと、入力位置は変更できないし、サブルーチンも終了不可能となる。

行の挿入について

空白入力が不可能なモードの場合、新しい行を挿入すると、何らかの入力をするかその行を削除しない限り、入力位置の変更はできない。

数値データの修正について

データは、[RET]キーを押しただけでは修正されない場合がある。

このサブルーチンでは、実際にキーボードから入力した数値を8バイト実数配列に代入している。画面には表示書式に応じた四捨五入値を表示する。

書式'(F6.3)'の場合を例にして、このことを説明する。

手順	キー入力	表示	実際の値	備考
1	2.3005[RET]	2.301	2.3005	表示と実際値が違う
2	[RET]	2.301	2.3005	変化なし
3	[HOME CLR][RET]			結果はMODEの値による
4	2.301[RET]	2.301	2.3010	表示と実際の値が同じとなる。

[ 編集用キー ]

データの入力編集には以下のキーを使用する。

押下キー	動作内容
[ESC]	サブルーチンを終了する。
[COPY]	CRT画面のハードコピー
[RET]	現在入力中のデータを確定。
[ROLL UP]	表示を上スクロール。
[ROLL DOWN]	表示を下スクロール。
[BS]	カーソルの左の文字を削除
[INS]	入力の挿入/上書きの切り替え。
[DEL]	カーソル位置の文字を削除
[↑]	入力位置を1行上に移動
[←]	カーソルを1桁左に移動
[→]	カーソルを1桁右に移動
[↓]	入力位置を1行下に移動
[SHIFT]+[←]	入力位置を左の列へ移動
[SHIFT]+[→]	入力位置を右の列へ移動
[TAB]	入力位置を右の列へ移動
[HOME CLR]	入力位置のデータを消去(クリア)
[SHIFT]+[INS]	反転カーソル行に空白行を挿入
[SHIFT]+[DEL]	反転カーソル行を削除

[ テストプログラム TEST036 ]

いろいろな表示形式で数値入力を行なうプログラム。入力を終了してテキスト画面を消去した後、確認のため入力したデータを表示する。

```

PROGRAM TEST036

IMPLICIT INTEGER*4 (I-N)
INTEGER*4 LN(5)
REAL*8 X(10,4)
CHARACTER LIST(100)*70,LINE*79,FORM(5)*30

CALL TGINIT
CALL TGCOPYES(1)
CALL TRANGE(3,18,1,80)
CALL TCURSW(0)

CALL TDISPM(1,1,' TEST020.FOR ',7,2)
    
```

ライブラリの初期化  
 ハードコピーはプリンタ  
 スクロール表示は3~18行  
 テキストカーソルは消去

以下タイトル等の表示

```
CALL TDISPM(1,62,'テストプログラム',7,0)
CALL TDISPM(20,1,'[ESC]入力終了',7,0)
CALL TCHMOD(1,1,15,80,0,1,0,0,5)
CALL TCHMOD(19,19,1,80,0,1,0,0,5)
```

MAX=10	10組のデータ入力を可能にする
NC=4	4変数(次元)データ入力
LN(1)=25	第1変数の表示は25桁目から
LN(2)=35	第2変数の表示は35桁目から
LN(3)=51	第3変数の表示は51桁目から
LN(4)=59	第4変数の表示は59桁目から
FORM(1)='(F7.3)'	第1変数は固定小数点形式で7桁までの入力
FORM(2)='(I10)'	第2変数は整数形式で10桁までの入力
FORM(3)='(F5.1)'	第3変数は固定小数点形式で5桁までの入力
FORM(4)='(1PE9.1)'	第4変数は浮動小数点形式で9桁までの入力

```
LINE=' 番号      第一変数      第二変数      第三変数      第四変数  '
CALL TDISPM(2,13,LINE(1:58),4,4)
```

N=0	新規入力なので0にする
NL=12	データ番号は12桁目から表示
IER=1	挿入削除を可能にする
CALL TDATAN(X,N,MAX,NC,LIST,NL,LN,FORM,IEDIT,IER)	
CALL TGEND	ライブラリ使用終了
DO 500 I=1,MAX	確認のためのデータ表示
WRITE(*,'(5(1PE16.4))') X(I,1),X(I,2),X(I,3),X(I,4)	
500 CONTINUE	
END	

CALL TDISPM (IY,IX,CHR,ICOLOR,MODE)

[ 機能説明 ]

テキスト画面の第IY行、第IX列から文字列CHRを定義長分だけ表示する。サブルーチンTPRINTとことなり、最後の文字がテキスト画面の80列目になっても改行しない。

また、このサブルーチンを実行しても、元の文字の表示色と属性は変化しない。

[ 引 数 ]

IY	4B整数	表示開始テキスト画面行番号。	入 力
IX	4B整数	表示開始テキスト画面列番号。	入 力
CHR	文字列	表示する文字列。80文字までとする。	入 力
ICOLOR	4B整数	表示色。 0.黒      1.青      2.赤      3.紫 4.緑      5.水色   6.黄      7.白	入 力
MODE	4B整数	表示の属性。 0.ノーマル(通常表示) 1.ブリンク(点滅表示) 2.リバーズ(反転表示) 3.リバーズ+ブリンク 4.アンダーライン(下線を付ける)	入 力

[ 注 意 ]

PC-9801RAには、文字列のブリンク(点滅)機能はない。MODE=2は使用しないこと。

[ 備 考 ]

ほとんどの用途ではTPRINTよりも、このTDISPMで十分と思われる。ただし、TPRINTでは外部のプリンタに文字列を出力できるが、このTDISPMでは不可能。

CALL TDISPMV ( IY,IX,FORM,R8,ICOLOR,MODE )

[ 機能説明 ]

8バイト実数R8を、テキスト画面の任意位置に、書式を整えて表示する。書式文字列にI型を使用すれば、小数点以下を省略した表示も可能。

また、このサブルーチンを実行しても、元の文字の表示色と属性は変化しない。

[ 引 数 ]

IY	4B整数	表示開始テキスト画面行番号。	入 力
IX	4B整数	表示開始テキスト画面列番号。	入 力
FORM	文字列	書式文字列。E,F,Iの書式が使用可能。	入 力
R8	8B実数	書式を整えて表示したい8B実数。	入 力
ICOLOR	4B整数	表示色。 0.黒      1.青      2.赤      3.紫 4.緑      5.水色   6.黄      7.白	入 力
MODE	4B整数	表示の属性。 0.ノーマル(通常表示) 1.ブリンク(点滅表示) 2.リバーズ(反転表示) 3.リバーズ+ブリンク 4.アンダーライン(下線を付ける)	入 力

[ 注 意 ]

サブルーチンTPRINTと同様に、文字列の最後の列が80だと1行上にスクロールしてしまうので注意。

CALL TESTEND(IRET)

[ 機能説明 ]

このサブルーチンを実行すると、テキスト画面の最下行(20行目)に

終了[YES/NO]

の表示が現われる。このときYESの文字が反転表示となっている。どちらかを選択するとサブルーチンが終了する。

[ 引 数 ]

IRET	4B整数	0.YESを選択。 YESの文字が反転表示のとき[RET]か[Y]キーを押した場合。 1.NOを選択。 NOの文字が反転表示のとき[RET]か[N]キーを押した場合。	出力
------	------	--	----

[ 備 考 ]

反転表示は[ ], [ ], [TAB]または[ESC]キーで移動する。

[ 注 意 ]

YESを選択すると、テキスト画面とグラフィック画面を消去し、テキスト画面を25行モード、ファンクションキー表示状態に戻す。そして、CRT画面制御ライブラリのグラフィックス関係の命令は使用できなくなる(正常動作をしなくなる)。

CALL TGCOPY

[ 機 能 説 明 ]

CRT画面のハードコピー(テキスト画面とグラフィックス画面の両方)を行なう。  
デフォルトではNECのPR-201系用となっているが、サブルーチンPGCOPYESを使用すると  
エプソンESC/P制御のプリンタにも出力できる。

[STOP]キーを押すと、ハードコピーを中止する。

CALL TGCOPYES(IPRT)

[ 機能説明 ]

サブルーチンTGCOPYを使用してCRT画面のハードコピーを行なうときの、プリンタの指定。

[ 引数 ]

IPRT	4B整数	0.NECのPR-201系(互換)のプリンタに出力する。 1.エプソンのESC/P制御プリンタに出力する。	入 力
------	------	--	-----

CALL TGEND

[ 機能説明 ]

テキスト画面とグラフィック画面を消去し、テキスト画面を25行モード、ファンクションキー表示状態に戻す。このサブルーチン実行後は、CRT画面制御ライブラリのグラフィックス関係の命令は使用できなくなる(正常動作をしなくなる)。

したがって、サブルーチンCHILDを用いて別のプログラムを実行する場合には、その終了の仕方に工夫が必要となる。

サブルーチンTESTENDではYESを選択すると、このTGENDを実行するようになっている。

[ 備 考 ]

このサブルーチン実行後、グラフィックスの命令が使用可能な状態にしていけないのは、作成したアプリケーション・プログラムをハードディスクに組み込んで利用する場合を考えてのことである。

このCRT画面制御ライブラリは、初期化サブルーチンTGINITを実行すると、グラフィックス関係の初期化を行うが、市販のアプリケーション・プログラムの中にはグラフィックスの初期化が完全でないものがある(例えば一太郎Ver.3.0花子Ver.1.0)。CRT画面制御ライブラリを利用したプログラムを実行したあと、そのようなプログラムを実行すると、正常動作が期待できない。そこで、このTGENDでは、そのようなプログラムが正常動作するような処置を行っている。

CALL TGETCOLR ( ICOLOR,MODE )

[ 機能説明 ]

サブルーチンTCOLORで設定した、テキスト文字の表示色とその属性を取得する。

[ 引 数 ]

ICOLOR	4B整数	表示色。 0.黒    1.青    2.赤    3.紫 4.緑    5.水色 6.黄    7.白	出力
MODE	4B整数	表示の属性。 0.ノーマル(通常表示) 1.ブリンク(点滅表示) 2.リバーズ(反転表示) 3.リバーズ+ブリンク 4.アンダーライン(下線を付ける)	出力

CALL TGETFORM ( FORM,IFIG,IFIX,ITYPE )

[ 機能説明 ]

FORTRANのFORMAT文字列から表示桁数を算出する。このサブルーチンは各種のデータ入力用のサブルーチンで使用している。

[ 引 数 ]

FORM	文字列	FORMATの書式文字列。並び書式は不可。	入 力
IFIG	4B整数	全桁数(小数点と符号分を含む)。	出 力
IFIX	4B整数	小数部の桁数(小数点は含まない)。	出 力
ITYPE	4B整数	書式の種類に応じた次の値。 0.書式を正しく読み取れなかった 2.書式がE(浮動小数点形式) 2.書式がF(固定小数点形式) 5.書式がA(文字列形式)	出 力

[ 備 考 ]

書式の値はサブルーチンTREADの引数の入力形式に合わせてある。

[ テストプログラム TEST037 ]

書式文字列が'(F10.2)'の場合に桁数と書式の種類を求めて表示する。

```

PROGRAM TEST037

INTEGER*4 IFIG1,IFIG2,ITYPE
CHARACTER FORM*30

FORM='(F10.2) '
CALL TGETFORM(FORM,IFIG1,IFIG2,ITYPE)
WRITE(*,*) '全表示桁数',IFIG1
WRITE(*,*) '小数点以下桁数',IFIG2
WRITE(*,*) '入力形式の種類',ITYPE

END
    
```

```
CALL TGETPRT ( IPRT )
```

## [ 機能説明 ]

TPRINTによる文字列の出力先が、テキスト画面になっているか、外部プリンタになっているかを検査する。

## [ 引 数 ]

IPRT	4B整数	出力先の番号 1.テキスト画面 2.外部プリンタ	出 力
------	------	--------------------------------	-----

## [ 備 考 ]

出力先の指定はサブルーチンPRINTERによって行なう。

**テキスト画面のスクロール範囲を取得****TGETRANG**

CALL TGETRANG ( IYT, IYB, IXL, IXR )

## [ 機能説明 ]

サブルーチンTRANGEで設定したテキスト画面のスクロール範囲を取得する。

## [ 引 数 ]

IYT	4B整数	スクロール範囲の上の行番号。	出力
IYB	4B整数	スクロール範囲の下の行番号。	出力
IXL	4B整数	スクロール範囲の左の桁番号。	出力
IXR	4B整数	スクロール範囲の右の桁番号。	出力

CALL TGINIT

[ 機能説明 ]

PC-9801を、本CRT画面制御ライブラリが利用可能な状態にする。

初期化の内容は以下の通り。

次の3つの画面が使えるようにし、テキスト画面だけを消去する。

テキスト画面	20行80列(20行表示に固定)	8色表示
グラフィックス画面1	X座標0~639,Y座標0~399	16色表示
グラフィックス画面2	X座標0~639,Y座標0~399	16色表示

テキスト画面のスクロール範囲を第1行~20行、横は第1列~80列までのCRT画面全体とする。ファンクションキーは消去する。

テキスト画面の表示色は7番の白とし、カーソルを第1行1列目にセットする。

グラフィックス画面の命令が有効な画面を1とし、この画面を表示する。

グラフィックス画面用のペンは7番の白にセットし、線種を10番の実線とする。

マウス・ドライバの有無を調べて、あれば利用可能状態にセットする。

CRT画面のハードコピーの出力先プリンタをNECのPR-201系とする(エプソンのESC/P制御プリンタ出力に変更するには、サブルーチンTGCOPIYESを使用する)。

ディスク操作時などの致命的エラーが発生した場合、MS-DOSには戻らないようにする(致命的エラーを無効にする)。これにより、FORTRANのREAD文,WRITE文におけるエラーの検出を可能としている。

MS-DOSにおける[STOP]と[CTRL]+[C]キーの機能を無効にする。

[ 備 考 ]

ライブラリにはこのTGINITによる初期化を行なわなくても、実行可能なサブルーチンもある。

**テキストカーソルを指定位置へ移動****T L O C A T E**

CALL TLOCATE ( IY,IX )

## [ 機能説明 ]

テキストカーソルを画面上の第IY行、IX列目へ移動する。

## [ 引 数 ]

IY	4B整数	画面上からの行番号(1~20)。	入 力
IX	4B整数	画面(左端から)の列番号(1~80)。	入 力

## [ 注 意 ]

C R T画面制御ライブラリでは、テキスト画面は20行モード固定となっている。

CALL TPRINT ( CHR,MODE )

[ 機能説明 ]

文字列CHRを現在のカーソル位置から表示する。FORTRANのWRITE文では、画面の任意の位置から文字や数値を表示できないので、このサブルーチンを使用する。数値は文字列に変換してからこのサブルーチンを利用すればよい。

このサブルーチンでは、文字列CHRの定義した長さ分だけ表示または印刷する。したがって、実際の文字数が、定義した文字数よりも少ない場合でも、残りの空白分も文字として出力する。なお、サブルーチンTSETPRT(プリンタの初期化と漢字ピッチの設定)の項も参照のこと。

[ 引 数 ]

CHR	文字列	半角で255字以内。	入 力
MODE	4B整数	文字列出力後のカーソル位置の指定。 0.カーソルは文字列のすぐ後ろ。 80桁を越えると改行。 1.カーソルは文字列の先頭に。復帰(CR)。 80桁を越えると改行してから復帰する。 2.カーソルは文字列の次の行の先頭に。 復帰、改行(CR,LF)。 80桁を越えると改行してから復帰、改行する。	入 力

[ テストプログラム TEST038 ]

ID=0,1,2の場合のカーソル位置を確認するプログラム。

```

PROGRAM TEST038

CHARACTER CHR*80

CALL TGINIT
CALL TRANGE(5,10,1,80)
CHR='応用物理学会 医用画像情報学会 日本放射線技術学会'
CALL TPRINT(CHR,0)          文字列の定義長さ80なので改行
CALL TWAIT(2000)

CALL TCURMOV(4)

```

```

CALL TPRINT(CHR,1)           80桁を超えるので改行
CALL TWAIT(2000)

CALL TCURMOV(4)
CALL TPRINT(CHR,2)         80桁を越えて改行し、更に改行
CALL TWAIT(2000)

CALL TGEND
END

```

[ テストプログラム TEST039 ]

引数に直接文字列を書くこともできる。

```

PROGRAM TEST039

CALL TGINIT
CALL TPRINT('応用物理学会 医用画像情報学会 ',0)
CALL TWAIT(2000)
CALL TGEND

END

```

[ テストプログラム TEST040 ]

引数の文字列に文字列の位置指定をしてもよい。

```

PROGRAM TEST040

CHARACTER CHR*70

CALL TGINIT
CHR='応用物理学会 医用画像情報学会 日本放射線技術学会 '
CALL TPRINT(CHR(1:50),0)
CALL TWAIT(2000)
CALL TGEND

END

```

[ テストプログラム TEST041 ]

C R T画面制御ライブラリで用意した関数CLENを利用すると、実際の文字数分だけを表示することができる。CLENは4バイト関数である。

```

PROGRAM TEST041

INTEGER*4 CLEN,ILEN
CHARACTER CHR*70

CALL TGINIT
CALL TRANGE(5,20,1,80)
CHR='応用物理学会 医用画像情報学会 日本放射線技術学会 '
ILEN=CLEN(CHR)
CALL TPRINT(CHR(1:ILEN),0)
CALL TWAIT(2000)

```

```
CALL TGEND  
END
```

[ テストプログラム TEST042 ]

数値を任意の位置に表示するには、次のようにFORTRANの内部ファイルを利用する。  
このプログラム例では文字列CLISTを内部ファイルとしている。

```
PROGRAM TEST042  
  
INTEGER*4 I,N  
REAL*8 PI,ANG,RAD  
CHARACTER CLIST(10)*79 自動改行防止のため文字列長は79に  
  
CALL TGINIT  
CALL TRANGE(1,18,1,80)  
  
PI=3.141592653589793D0  
N=(720/90)+1  
  
DO 100 I=1,N  
    ANG=90.0D0*(I-1)  
    RAD=2.0D0*PI*ANG/360.0D0  
    WRITE(CLIST(I),'(15X,1PE15.5,1PE15.5)') ANG,DSIN(RAD)  
100 CONTINUE  
  
CALL TPRINT('          角度          正弦関数值',2)  
DO 200 I=1,N  
    CALL TPRINT(CLIST(I),2)  
200 CONTINUE  
  
CALL TESTEND(IRET)  
END
```

[ テストプログラム TEST043 ]

FORTRANではプログラム文1行の長さが72列までとなっているため、CRT画面80列いっ  
ぱいの文字列はプログラム文1行に書けない。ここに示したように文字列の文字位置を  
指定した方法は、文字列の長さを80列としておくと、実際のCRT画面上の列位置との対  
応がつけやすい。

```
PROGRAM TEST043  
  
CHARACTER CHR*80  
  
CALL TGINIT  
CHR( 5:18)=' 応用物理学会 '  
CHR(27:44)=' 医用画像情報学会 '  
CHR(58:75)=' 日本放射線技術学会 '  
CALL TPRINT(CHR,0)  
CALL TWAIT(2000)  
CALL TGEND
```

```
END
```

[ テストプログラム TEST044 ]

アスキー制御文字を入力しても制御はしない。制御文字をそのまま表示してしまう。

```
PROGRAM TEST044

CHARACTER CHR*80,LF*1

LF=CHAR(16#0A)
CALL TGINIT
CHR='医用 '//LF//'画像 '//LF//'情報 '//LF//'学会 '
CALL TPRINT(CHR,0)
CALL TWAIT(2000)
CALL TGEND

END
```

[ テストプログラム TEST045 ]

サブルーチンPRINTERを用いて、文字列の出力先を外部プリンタにすることができる。  
テストプログラムTEST042において、文字列の出力先を外部プリンタとした場合。

```
PROGRAM TEST045

INTEGER*4 I,N
REAL*8 PI,ANG,RAD
CHARACTER CLIST(10)*79

CALL TGINIT
CALL TRANGE(1,18,1,80)

PI=3.141592653589793D0
N=(720/90)+1

DO 100 I=1,N
  ANG=90.0D0*(I-1)
  RAD=2.0D0*PI*ANG/360.0D0
  WRITE(CLIST(I),'(15X,1PE15.5,1PE15.5)') ANG,DSIN(RAD)
100 CONTINUE

CALL PRINTER(2)
CALL TPRINT('          TEST042に追加した部分
          角度          正弦関数値 ',2)
DO 200 I=1,N
  CALL TPRINT(CLIST(I),2)
200 CONTINUE

CALL TESTEND(IRET)
END
```

CALL TRANGE ( IYT,IYB,IXL,IXR )

[ 機能説明 ]

テキスト画面のスクロール範囲を設定する。新たにスクロール範囲を設定すると、それまでの設定は無効となる。

[ 引 数 ]

IYT	4B整数	スクロール範囲の上の行番号。	入 力
IYB	4B整数	スクロール範囲の下の行番号。	入 力
IXL	4B整数	スクロール範囲の左の桁番号。	入 力
IXR	4B整数	スクロール範囲の右の桁番号。	入 力

[ 備 考 ]

このサブルーチンの設定とは無関係に、文字列はテキスト画面の任意の位置から表示可能である。ただし、スクロール表示関係の制御は、ここで設定した範囲の表示に対して行なわれる。

[ テストプログラム TEST046 ]

文字列CHRを緑色で50回表示した後、スクロール範囲を再設定し、今度は水色で50回表示する。

PROGRAM TEST046	
INTEGER*4 I	
CHARACTER CHR*40	
CALL TGINIT	ライブラリの初期化
CALL TRANGE(5,10,1,80)	スクロール範囲を設定
CHR='(社)日本放射線技師学会'	表示する文字列
CALL TCOLOR(4,0)	表示色は4の緑色
DO 100 I=1,50	50回表示する
CALL TPRINT(CHR,2)	CR,LFなので自動改行
100 CONTINUE	したがって10行目は空白
CALL TRANGE(12,17,1,80)	スクロール範囲を再設定
CALL TCURMOV(4)	念のためスクロール範囲内をクリア

```
CALL TCOLOR(5,0)
CALL TWAIT(2000)
DO 200 I=1,50
    CALL TPRINT(CHR,2)
200 CONTINUE
```

```
CALL TESTEND(IRET)
END
```

```
5番の水色で
2000ミリ秒(2秒間)待つて
50回表示
自動改行なので上にスクロール表示
17行目は空白
```

CALL TREAD ( IY,IX,CHR,N,ITYPE,IER,ISHIFT )

[ 機能説明 ]

テキスト画面の第 IY 行 IX 列から文字列を入力するためのサブルーチン。

FORTTRANのREAD文では任意の位置からの文字や数値の入力ができないので、このサブルーチンを使用する。既入力の文字列の修正にも使用する。

[ 引 数 ]

IY	4B整数	入力開始のテキスト行番号。	入 力
IX	4B整数	入力開始のテキスト列番号。	入 力
CHR	文字列	入力修正する文字列。 終了時にはそれまでの入力結果を返す。	入出力
N	4B整数	入力文字数。最大、半角で255字以内。	入 力
ITYPE	4B整数	入力可能な文字に制限をつける以下の値。 1. 整数入力用 +, -, 0, 1, 2, …, 9 2. 実数入力用 +, -, 0, 1, 2, …, 9, . (小数点), E (指数部) 3. MS-DOSファイル名入力用 英大文字と数字、-^¥@:;!#\$%&()_{} 4. ASCIIコードの20H~7EHまでの値。 5. 上記以外の文字。 各々11,12,13,14,15にすると、呼び出し時にカーソルが先頭にくる。 負の値にすると、入力可能桁全体が反転カーソル表示となる。	入 力
IER	4B整数	終了時に返す数値。備考参照。	出 力
ISHIFT	4B整数	終了時の[SHIFT]キー押下状態を返す。	出 力

[ 備 考 ]

文字の表示色は現在の設定色を使用する。

データの入力編集には以下のキーを使用する。

押下キー	動作内容
[RET]	データを確定して終了
[ESC]	データ確定せず強制終了
[COPY]	CRT画面のハードコピー
[BS]	カーソルの左の文字を削除
[INS]	挿入/上書切り替え
[DEL]	カーソル位置の文字を削除
[ ]	カーソルを1桁左に移動
[ ]	カーソルを1桁右に移動
[HOME CLR]	入力位置のデータを消去(クリア)

本サブルーチン以下キーの押下により終了する。

終了時に IER, ISHIFT の各引数に返す数値も示す。[RET] キー以外の終了でも、それまでに入力、修正した文字列が CHR に代入されている。

押下キー	IER	ISHIFT	備 考
[RET]	16#0000	0	空文字出力
[RET]	16#1C00	0	修正あり出力
[RET]	-16#1C00	0	修正無し出力
[ESC]	16#1B00	0	強制終了
[ROLL UP]	16#3600	0	
[ROLL DOWN]	16#3700	0	
[ ]	16#3A00	0	
[ ]	16#3D00	0	
[SHIFT]+[ ]	16#3A00	1	
[SHIFT]+[ ]	16#3B00	1	
[SHIFT]+[ ]	16#3C00	1	
[SHIFT]+[ ]	16#3D00	1	
[TAB]	16#0F00	0	
[SHIFT]+[INS]	16#3800	1	
[SHIFT]+[DEL]	16#3900	1	

[ テストプログラム TEST047 ]

入力または修正した文字列を、スクロール範囲に順番に表示するようにした。

```
PROGRAM TEST047

IMPLICIT INTEGER*4 (I-N)
INTEGER*4 CLEN
CHARACTER STR*80

CALL TGINIT
CALL TRANGE(1,17,1,80)

IT=5
CALL TLOCATE(19,1)
CALL TPRINT('入力 '//
+ '['                                     ]',0)
IF=0
IC=1
STR=' '
N=40
100 CALL TREAD(19,7,STR,40,IT,IER,ISHIFT)
IER=IABS(IER)
IF (IER.EQ.16#1B00) GOTO 999
IF ((IER.EQ.16#1C00).OR.(IER.EQ.0)) THEN
  IF (IC.GT.17) IC=17
  ILEN=CLEN(STR)
  CALL TCOLOR(4,0)
  CALL TLOCATE(IC,1)
  CALL TPRINT(STR(1:ILEN),2 )
  IC=IC+1
END IF
GOTO 100

999 CALL TESTEND
END
```

[ESC]キ-なら終了  
[RET]キ-なら文字列表示  
スクロール最下行のとき

TDISPMはスクロールしないの  
で使用不可、  
TPRINTを使用して表示

CALL TREADI2 ( IY,IX,IV2,FORM,ICODE )

[ 機能説明 ]

テキスト画面の第IY行IX列目から2バイト整数IV2に数値を入力する。

[ 引 数 ]

IY	4B整数	入力開始のテキスト画面行番号。	入 力
IX	4B整数	入力開始のテキスト画面列番号。	入 力
IV2	2B整数	呼び出し時には初期値を代入しておく。	入出力
FORM	文字列	表示形式の書式。入力桁数の決定にも使用する。	入 力
ICODE	4B整数	16#1C00 [RET]キー押下終了時。 16#1B00 [ESC]キー押下終了時。 16#3A00 [ ]キー押下終了時。 16#3D00 [ ],[TAB]キー押下終了時。	出 力

[ 備 考 ]

文字の表示色は現在の設定色を使用する。

[RET]キー以外の終了では、IV2の値は、本サブルーチン呼び出し時の値に戻る。

データの入力編集には以下のキーを使用する。

押下キー	動 作 内 容
[RET]	データを確定して終了
[ESC],[ ],[ ],[TAB]	データ確定せず強制終了
[COPY]	CRT画面のハードコピー
[BS]	カーソルの左の1桁を削除
[INS]	挿入/上書切り替え
[DEL]	カーソル位置の1桁を削除
[ ]	カーソルを1桁左に移動
[ ]	カーソルを1桁右に移動
[HOME CLR]	入力位置のデータを消去(クリア)

CALL TREADI4 ( IY,IX,IV4,FORM,ICODE )

[ 機能説明 ]

テキスト画面の第IY行IX列目から4バイト整数IV4に数値を入力する。

[ 引 数 ]

IY	4B整数	入力開始のテキスト画面行番号。	入 力
IX	4B整数	入力開始のテキスト画面列番号。	入 力
IV4	4B整数	呼び出し時には初期値を代入しておく。	入出力
FORM	文字列	表示形式の書式。入力桁数の決定にも使用する。	入 力
ICODE	4B整数	16#1C00 [RET]キー押下終了時。 16#1B00 [ESC]キー押下終了時。 16#3A00 [ ]キー押下終了時。 16#3D00 [ ],[TAB]キー押下終了時。	出 力

[ 備 考 ]

文字の表示色は現在の設定色を使用する。

[RET]キー以外の終了では、IV4の値は、本サブルーチン呼び出し時の値に戻る。

データの入力編集には以下のキーを使用する。

押下キー	動 作 内 容
[RET]	データを確定して終了
[ESC],[ ],[ ],[TAB]	データ確定せず強制終了
[COPY]	CRT画面のハードコピー
[BS]	カーソルの左の1桁を削除
[INS]	挿入/上書切り替え
[DEL]	カーソル位置の1桁を削除
[ ]	カーソルを1桁左に移動
[ ]	カーソルを1桁右に移動
[HOME CLR]	入力位置のデータを消去(クリア)

[ テストプログラム TEST048 ]

テキスト画面第5行10列目から4バイト整数を入力する。初期値は100。8桁までの入力を可能にしている。入力位置には入力桁の範囲を示すために[ ]を表示している。

```
PROGRAM TEST048

IMPLICIT INTEGER*4 (I-N)
CHARACTER FORM*10

CALL TGINIT
IY=5
IX=10
IV4=100
FORM='(I8) '
CALL TDISPM(IY,IX,'整数[      ]',7,0)
CALL TCHMOD(IY,IY,IX,IX+3,0,0,0,0,6)
100 CALL TREADI4(IY,IX+5,IV4,FORM,ICODE)
200 CALL TESTEND(IRET)
IF (IRET.EQ.1) GOTO 100

END
```

CALL TREADR4 ( IY,IX,RV4,FORM,ICODE )

[ 機能説明 ]

テキスト画面の第IY行IX列目から4バイト実数RV4に数値を入力する。

[ 引 数 ]

IY	4B整数	入力開始のテキスト画面行番号。	入 力
IX	4B整数	入力開始のテキスト画面列番号。	入 力
RV4	4B実数	呼び出し時には初期値を代入しておく。	入出力
FORM	文字列	表示形式の書式。入力桁数の決定にも使用する。	入 力
ICODE	4B整数	16#1C00 [RET]キー押下終了時。 16#1B00 [ESC]キー押下終了時。 16#3A00 [ ]キー押下終了時。 16#3D00 [ ],[TAB]キー押下終了時。	出 力

[ 備 考 ]

文字の表示色は現在の設定色を使用する。

[RET]キー以外の終了では、RV4の値は、本サブルーチン呼び出し時の値に戻る。

データの入力編集には以下のキーを使用する。

押下キー	動 作 内 容
[RET]	データを確定して終了
[ESC],[ ],[ ],[TAB]	データ確定せず強制終了
[COPY]	CRT画面のハードコピー
[BS]	カーソルの左の1桁を削除
[INS]	挿入/上書切り替え
[DEL]	カーソル位置の1桁を削除
[ ]	カーソルを1桁左に移動
[ ]	カーソルを1桁右に移動
[HOME CLR]	入力位置のデータを消去(クリア)

CALL TREADR8 ( IY,IX,RV8,FORM,ICODE )

[ 機能説明 ]

テキスト画面の第IY行IX列目から8バイト実数RV8に数値を入力する。

[ 引 数 ]

IY	4B整数	入力開始のテキスト画面行番号。	入 力
IX	4B整数	入力開始のテキスト画面列番号。	入 力
RV8	8B実数	呼び出し時には初期値を代入しておく。	入出力
FORM	文字列	表示形式の書式。入力桁数の決定にも使用する。	入 力
ICODE	4B整数	16#1C00 [RET]キー押下終了時。 16#1B00 [ESC]キー押下終了時。 16#3A00 [ ]キー押下終了時。 16#3D00 [ ],[TAB]キー押下終了時。	出 力

[ 注 意 ]

CRT画面制御ライブラリのデータ入力関係のサブルーチンでは、8バイト実数を0.0及び1.0E-100を越えて1.0E+100未満(絶対値で)に制限している。

[ 備 考 ]

文字の表示色は現在の設定色を使用する。

[RET]キー以外の終了では、RV8の値は、本サブルーチン呼び出し時の値に戻る。

データの入力編集には以下のキーを使用する。

押下キー	動 作 内 容
[RET]	データを確定して終了
[ESC],[ ],[ ],[TAB]	データ確定せず強制終了
[COPY]	CRT画面のハードコピー
[BS]	カーソルの左の1桁を削除
[INS]	挿入/上書切り替え
[DEL]	カーソル位置の1桁を削除

[ ]	カーソルを1桁左に移動
[ ]	カーソルを1桁右に移動
[HOME CLR]	入力位置のデータを消去(クリア)

[ テストプログラム TEST049 ]

テキスト画面第5行10列目から8バイト実数を入力する。初期値は100.0。10桁までの入力を可能にしている。入力位置には入力桁の範囲を示すために[ ]を表示している。

```

PROGRAM TEST049

IMPLICIT INTEGER*4 (I-N)
REAL*8    RV8
CHARACTER FORM*10

CALL TGINIT
IY=5
IX=10
RV8=100.0
FORM='(1PE10.3) '
CALL TDISPM(IY,IX,'実数 [          ]',7,0)
CALL TCHMOD(IY,IY,IX,IX+3,0,0,0,0,6)
100 CALL TREADR8(IY,IX+5,RV8,FORM,ICODE)
200 CALL TESTEND(IRET)
    IF (IRET.EQ.1) GOTO 100

END

```

CALL TSETCHR ( IY,IX,ICODE,ICOLOR )

[ 機能説明 ]

ICODEで指定するコードの文字をテキスト画面に表示する。

サブルーチンTPRINTでは文字と漢字しか出力できない。PC-9801に内蔵されているグラフィック文字等を表示する場合はこのサブルーチンを使用する。

[ 引 数 ]

IY	4B整数	表示位置のテキスト画面行番号。	入 力
IX	4B整数	表示位置のテキスト画面列番号。	入 力
ICODE	4B整数	表示する文字のコード。00H~F7Hの値。	入 力
ICOLOR	4B整数	表示色。 0.黒    1.青    2.赤    3.紫 4.緑    5.水色 6.黄    7.白	入 力

[ テストプログラム TEST050 ]

画面中央付近に、四角い枠を表示する。ただし、縦方向の線はつながらない(CRT画面制御ライブラリではテキスト画面を20行モードで使用している)。

```

PROGRAM TEST050

CALL TGINIT
IXL=20
IXR=60
IYT=7
IYB=12
ICOLOR=4

C
C   さきに四隅のグラフ文字
C
CALL TSETCHR(IYT-1,IXL-1,16#98,ICOLOR)
CALL TSETCHR(IYT-1,IXR+1,16#99,ICOLOR)
CALL TSETCHR(IYB+1,IXL-1,16#9A,ICOLOR)
CALL TSETCHR(IYB+1,IXR+1,16#9B,ICOLOR)

C
C   上下の横の線と左右の線
C
DO 100 IX=IXL,IXR
    
```

```
100 CALL TSETCHR(IYT-1,IX,16#95,ICOLOR)
    DO 110 IX=IXL,IXR
110 CALL TSETCHR(IYB+1,IX,16#95,ICOLOR)
    DO 120 IY=IYT,IYB
120 CALL TSETCHR(IY,IXL-1,16#96,ICOLOR)
    DO 130 IY=IYT,IYB
130 CALL TSETCHR(IY,IXR+1,16#96,ICOLOR)

    CALL TESTEND(IRET)
    END
```

```
CALL TSETPRT( IPRT )
```

## [ 機能説明 ]

外部プリンタの初期化（電源投入時と同じ状態にする）を行い、漢字ピッチをANK文字2文字分の5CPIに設定する。

## [ 引数 ]

IPRT	4B整数	0.NECのPR-201系（互換）プリンタの初期化。 1.EPSOONのESP/C制御系プリンタの初期化。	入 力
------	------	--	-----

## [ 注 意 ]

このサブルーチン実行時には、プリンタの電源が投入されていること。  
エプソンのプリンタは、ESC/Pスーパー（通称PCセット）にする必要はない。

指定時間だけプログラムの実行を中断

T W A I T

CALL TWAIT ( IMSEC )

[ 機能説明 ]

IMSECで指定した時間だけプログラムの実行を中断する。20ミリ秒以下の値は無視される。

[ 引 数 ]

IMSEC	4B整数	中断時間で、ミリ秒値。 最大約15秒まで可能。	入 力
-------	------	----------------------------	-----

CALL TWINDSCR ( N,LIST,IFRST,IC1,IC2,IER )

[ 機能説明 ]

現在設定してあるスクロール範囲内に、文字列配列LISTをスクロール表示する。メニュー選択や計算結果のデータ表示などに使用するためのサブルーチン。

また、サブルーチンPRINTERで文字列の出力先を外部プリンタに指定しておけば外部プリンタに印刷することもできる(この場合、テキスト画面表示は行なわない)。

計算結果をスクロール表示する場合は、あらかじめ数値を文字列に変換して、文字列配列LISTの各要素に代入しておく。それには、FORTRANのWRITE文を使用する。

[ 引 数 ]

N	4B整数	スクロール表示する文字列配列の要素数。	入 力
LIST	文字列配列	スクロール表示する文字列配列。 表示桁がスクロール範囲の横幅を越えないこと。	入 力
IFRST	4B整数	スクロール範囲の最上行に表示する文字列配列要素番号。サブルーチン終了後は終了時、最上行に表示している要素番号を返す。	入出力
IC1	4B整数	文字の表示色。	入 力
IC2	4B整数	反転カーソルの色。負にすると反転表示なしとなる(計算結果のスクロール表示等の場合)。	入 力
IER	4B整数	反転カーソルを表示するスクロール範囲内の行番号を与える。16#3F00を入力すると前回と同じ。終了時の値については備考参照。	入出力

[ 備 考 ]

[RET]キーで終了すると、反転カーソルの色が(引数IC1で指定する)文字の色と同じとなる。[ESC]キーによる終了の場合は反転カーソルはなくなる。

Nを負にすると、スクロール範囲に表示してある文字列要素だけの選択動作となる。この場合は、[ROLL UP],[ROLL DOWN]キーは使用不可となる。ただし、反転カーソルが最下行のとき、[ ]キーで最上行に移動するようになる([ ]キーも同様)。

スクロール表示制御には以下のキーを使用する。

押下キー	動作内容	備考
[RET]	サブルーチン終了	IERにスクロール上端からの行番号を返す(上端が1) マウス左ボタンクリックも同じ
[ESC]	サブルーチン終了	IERに16#1B00を返す マウス右ボタンクリックも同じ
[HELP]	サブルーチン終了	IERに16#3F00を返す
[COPY]	CRT画面のハードコピー	
[ROLL UP]	上にスクロール	
[ROLL DOWN]	下にスクロール	
[ ]	反転カーソル1行上移動	IC2<0なら1行上にスクロール
[ ]	反転カーソル1行下移動	IC2<0なら1行下にスクロール

[ テストプログラム TEST051 ]

0から100までの整数の2乗を計算し、結果をCRT画面にスクロール表示する。再表示の際は、終了時の表示が現われる。

```

PROGRAM TEST051

IMPLICIT INTEGER*4 (I-N)
CHARACTER LIST(150)*20

CALL TGINIT
CALL TRANGE(5,15,10,30)
NO=100
N=NO+2
LIST(1)='  整数値      2乗 '
DO 100 I=0,NO
100 WRITE(LIST(I+2)(4:20),'(I5,I10)') I,I*I

IGX1=(10-1)*8-4
IGX2=(30+1)*8-4
IGY1=400-(5-1)*20+8
IGY2=400-(15+1)*20+8
CALL GBOXL(IGX1,IGX2,IGY1,IGY2)
IC1=4
IC2=-7
IDISP=1
200 CALL TWINDSCR(N,LIST,IDISP,IC1,IC2,IER)
CALL TESTEND(IRET)
IF (IRET.EQ.1) GOTO 200
WRITE(*,'(5X,I4,5X,I4,5X,I4)') N,IDISP,IER

END

```

ライブラリの初期化  
スクロール範囲は5～15行(11行)  
100までの2乗を計算  
0タイトル分を加える  
タイトル文字  
2乗の計算  
結果を文字列配列に代入

スクロール範囲を囲む体裁用の枠  
の各座標値を計算  
これはグラフィックス画面に描く

矩形の描画  
文字の色は4の緑  
反転カーソルは表示しない  
1番上は第1要素を表示

[RET]または[ESC]で終了  
NOなら再スクロール表示

確認用の表示

CALL TXTGET ( IYT, IYB, IXL, IXR, IP )

[ 機能説明 ]

テキスト画面の第IYT行から第IYB行までの、IXL列からIXR列に囲まれた範囲の文字情報を文字列配列Pに格納る。ここで記憶した情報はサブルーチンTXTPUTで任意の位置から表示できる。

[ 引 数 ]

IYT	4B整数	記憶する上の行番号。	入 力
IYB	4B整数	記憶する下の行番号。	入 力
IXL	4B整数	記憶する左側の列番号。	入 力
IXR	4B整数	記憶する右側の列番号。	入 力
P	文字列配列	情報を格納する文字列配列。 大きさが以下のバイト以上であること。 $(IYB - IYT + 1) * (IXR - IXL + 1) * 4 + 2$	出 力

```
CALL TXTPUT ( IY,IX,IP )
```

## [ 機能説明 ]

サブルーチンTXTGETを用いて格納したテキスト画面情報を、指定位置から再表示する。

## [ 引 数 ]

IY	4B整数	表示を開始する上の行番号。	入 力
IX	4B整数	表示を開始する左側の列番号。	入 力
P	文字列配列	サブルーチンTXTGETで用いた情報が格納してある文字列配列。	入 力

## [ 注 意 ]

サブルーチンTXTGETで格納したテキスト画面情報が全て表示出来るような範囲とすること。このサブルーチンではこの範囲のチェックはしていない。

CALL TYESNO ( IY,IX,ICODE )

[ 機 能 説 明 ]

このサブルーチンを実行すると、テキスト画面の第IY行IX列目に

[YES/NO]

の表示が現われる。ICODEの値によってYESまたはNOのどちらかの文字列を反転表示する。どちらかを選択するとサブルーチンが終了し、押下キーに応じた値をICODEに返す。

[ 引 数 ]

IY	4B整数	表示開始テキスト画面行番号。	入 力
IX	4B整数	表示開始テキスト画面列番号。	入 力
ICODE	4B整数	16#0059を入力: YESを反転表示 16#004Eを入力: NOを反転表示	入 力
		16#0059を出力 YESの文字が反転表示のとき[RET]か[Y]キーを押した場合。 16#004Eを出力 NOの文字が反転表示のとき[RET]か[N]キーを押した場合。 16#1B00を出力 NOの文字が反転表示のとき[ESC]キーを押下。	出 力

[ 備 考 ]

反転表示は[ ], [ ]または[TAB]キーで移動できる。

YESの文字が反転表示のとき, [ESC]キーを押すと反転表示がNOの文字に移動する。