SUPERSP

PURPOSE **SUPERSP** draws a superscript of "special" characters.

SYNTAX CALL SUPERSP (NSET, NFNT, ITITLE, NCHR, HEIGHT, ANG)

<u>*ARGUMENTS*</u>

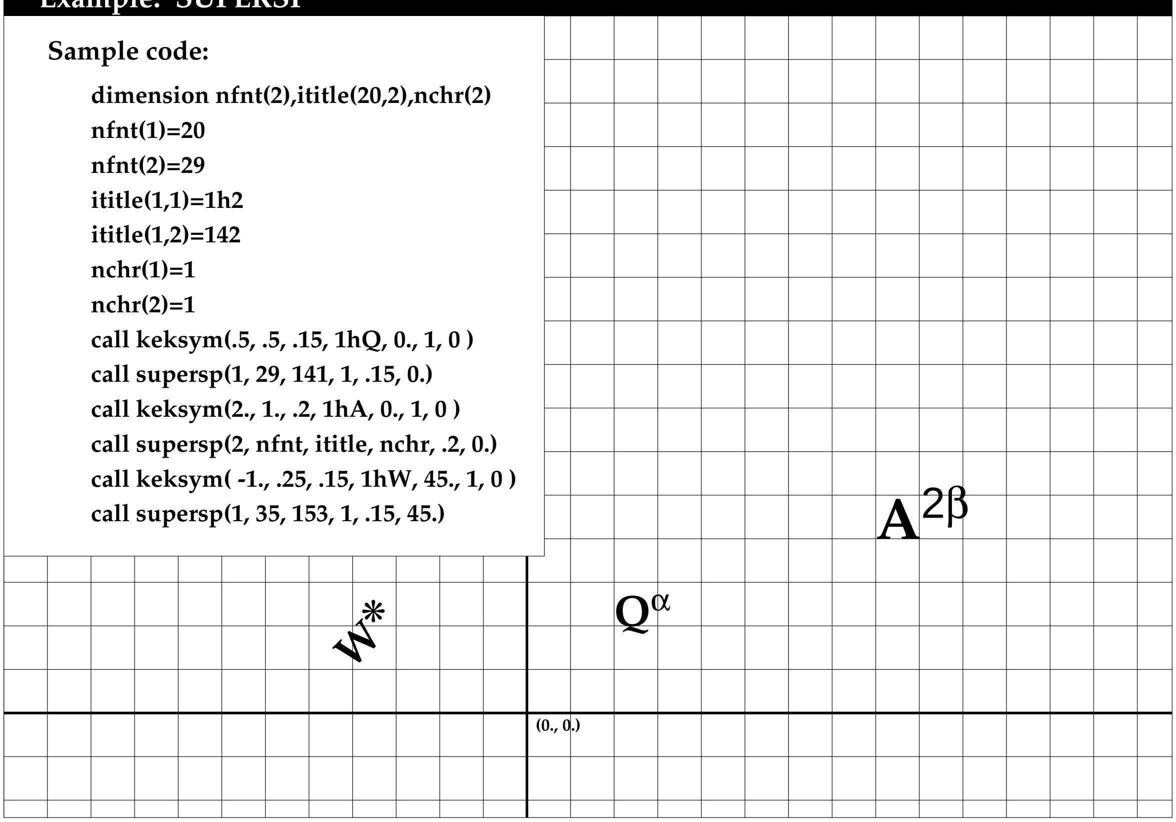
NSET	Number of different font sets needed.
NFNT	Array holding the font numbers 1 to NSET.
ITITLE	ITITLE holds the octal codes for characters of fonts 29 or 35; otherwise, it hold the text characters themselves.
NCHR	Number of characters of a given font. Usually one, but can be greater than one for fonts other than 29 and 35.
HEIGHT	Height of the **variable to be superscripted**. It is **not** the height of the superscript characters themselves.
ANG	Angle of the superscripted variable.

Note: SUPERSP must be called immediately after the call to create the superscripted variable.

Example: SUPERSP

Sample code:

```
dimension nfnt(2),ititle(20,2),nchr(2)
nfnt(1)=20
nfnt(2)=29
ititle(1,1)=1h2
ititle(1,2)=142
nchr(1)=1
nchr(2)=1
call keksym(.5, .5, .15, 1hQ, 0., 1, 0 )
call supersp(1, 29, 141, 1, .15, 0.)
call keksym(2., 1., .2, 1hA, 0., 1, 0 )
call supersp(2, nfnt, ititle, nchr, .2, 0.)
call keksym( -1., .25, .15, 1hW, 45., 1, 0 )
call supersp(1, 35, 153, 1, .15, 45.)
```



SUBSUPSP

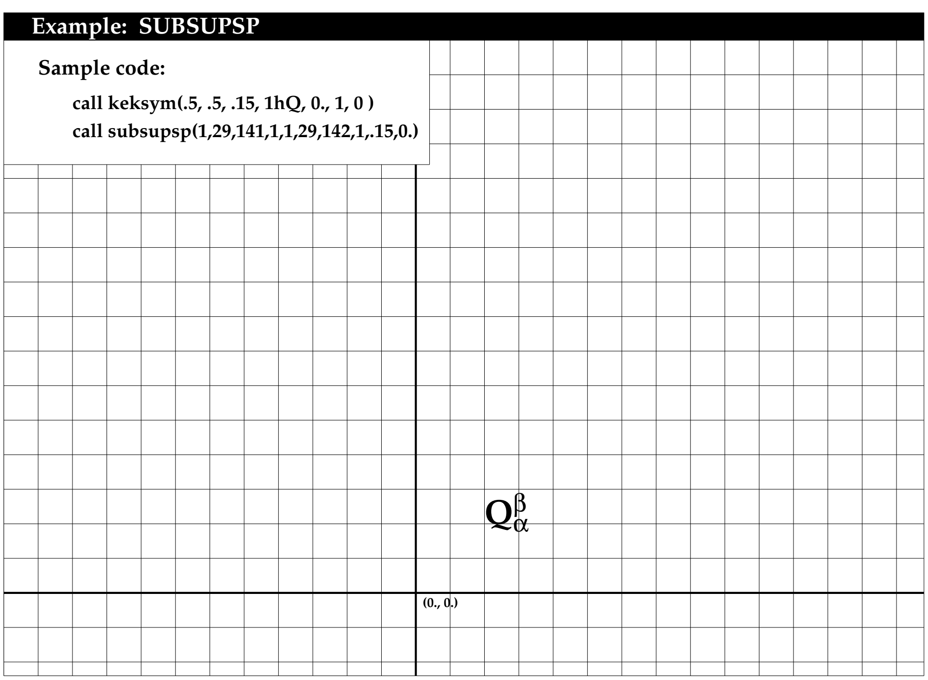
PURPOSE **SUBSUPSP** draws subscripts and superscripts of “special” characters.

SYNTAX CALL SUBSUPSP (NSUB, NFNTSB, ITITLESB, NCHRSB, NSUP, NFNTSP,ITITLESP, NCHRSP, HEIGHT, ANG)

<u>*ARGUMENTS*</u>

NSUB, NSUP	Number of different font sets needed for subscripts and superscripts, respectively.
NFNTSB, NFNTSP	Arrays holding the font numbers 1 to NSET for subscripts and superscripts, respectively.
ITITLESB, ITITLESP	Arrays holding the octal codes for characters of fonts 29 or 35; otherwise, holding text characters themselves.
NCHRSB, NCHRSP	Number of characters of a given font for subscripts and superscripts, respectively. Usually one, but can be greater than one for fonts other than 29 and 35.
HEIGHT	Height of the **variable to be subscripted**. It is **not** the height of the subscript characters themselves.
ANG	Angle of the subscripted variable.

Note: SUBSUPSP must be called immediately after the call to create the subscripted variable.



SUBBERSP

PURPOSE **SUBBERSP** draws a subscript of “special” characters.

SYNTAX CALL SUBBERSP (NSET, NFNT, ITITLE, NCHR, HEIGHT, ANG

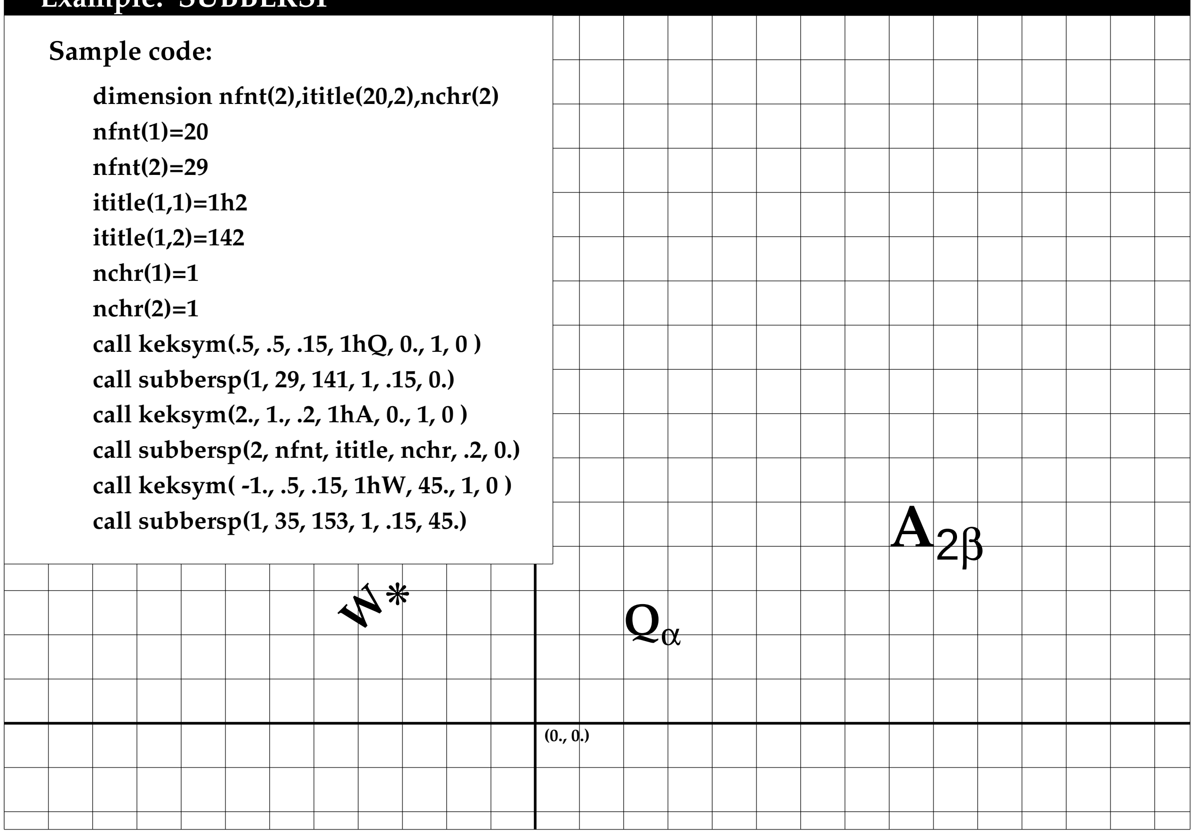
ARGUMENTS

NSET	Number of different font sets needed.
NFNT	Array holding the font numbers 1 to NSET.
ITITLE	ITITLE holds the octal codes for characters of fonts 29 or 35; otherwise, it hold the text characters themselves.
NCHR	Number of characters of a given font. Usually one, but can be greater than one for fonts other than 29 and 35.
HEIGHT	Height of the **variable to be subscripted**. It is **not** the height of the subscript characters themselves.
ANG	Angle of the subscripted variable.
Note:	SUBBERSP must be called immediately after the call to create the subscripted variable.

Example: SUBBERSP

Sample code:

```
dimension nfnt(2),ititle(20,2),nchr(2)
nfnt(1)=20
nfnt(2)=29
ititle(1,1)=1h2
ititle(1,2)=142
nchr(1)=1
nchr(2)=1
call keksym(.5, .5, .15, 1hQ, 0., 1, 0 )
call subbersp(1, 29, 141, 1, .15, 0.)
call keksym(2., 1., .2, 1hA, 0., 1, 0 )
call subbersp(2, nfnt, ititle, nchr, .2, 0.)
call keksym( -1., .5, .15, 1hW, 45., 1, 0 )
call subbersp(1, 35, 153, 1, .15, 45.)
```



STROKE

PURPOSE **STROKE** calls the PostScript operator *stroke* (which paints or draws the current path).

SYNTAX CALL STROKE

<u>*ARGUMENTS*</u>

None

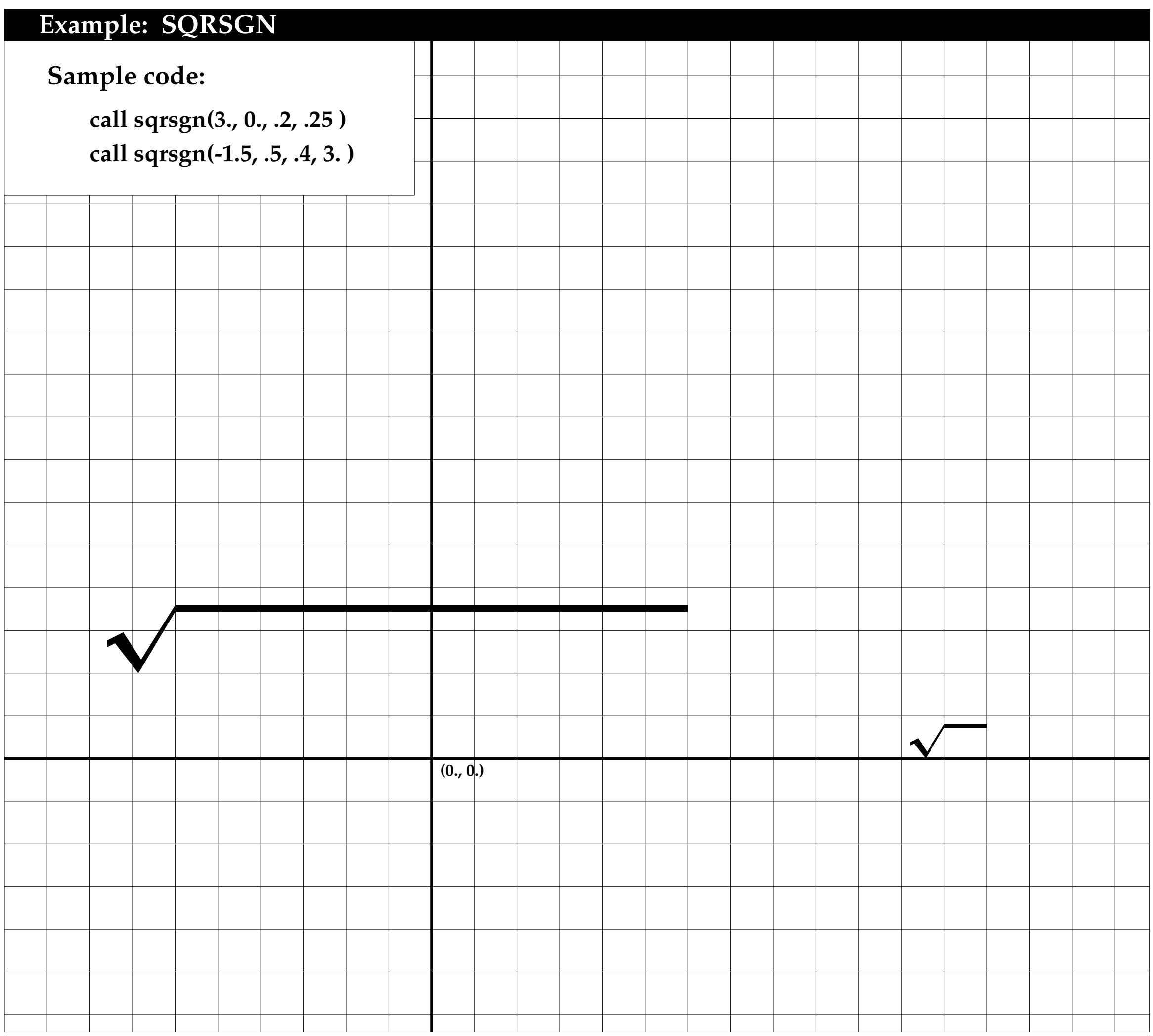
SQRSGN

PURPOSE **SQRSGN** draws a radical (square root) sign.

SYNTAX CALL SQRSGN (XPP, YPP, HEIGHT, RLEN)

ARGUMENTS

XPP,YPP	X,Y coordinates of lower left corner of the radical.
HEIGHT	Height of the radical.
RLEN	Length of the top of the radical.



SLDCRV

PURPOSE **SLDCRV** connects points in two data arrays using a line with a specified thickness.

SYNTAX CALL SLDCRV (XARR, YARR, NPTS, THK)

<u>*ARGUMENTS*</u>

XARR Array containing the X coordinates of the data points to be plotted.

YARR Array containing the Y coordinates of the data points to be plotted.

NPTS Number of points in the curve.

THK Specifies the thickness of the curve. If THK = 0., the current linewidth is used.

Example: SLDCRV

Sample code:

```
dimension xp1(6), yp1(6)
dimension xp2(5), yp2(5)
data xp1/ .5, 1., 1.5, 2., 2.5, 3. /
data yp1/ 0., 1., .5, 1.75, 1.5, .5 /
data xp2/ 3.5, 4., 4.5, 5., 5.5 /
data yp2/ .5, 1., 2., .5, 1. /
call sldcrv(xp1, yp1, 6, .02)
call sldcrv(xp2, yp2, 5, .01)
```

Code to draw axes not shown.

SETLW

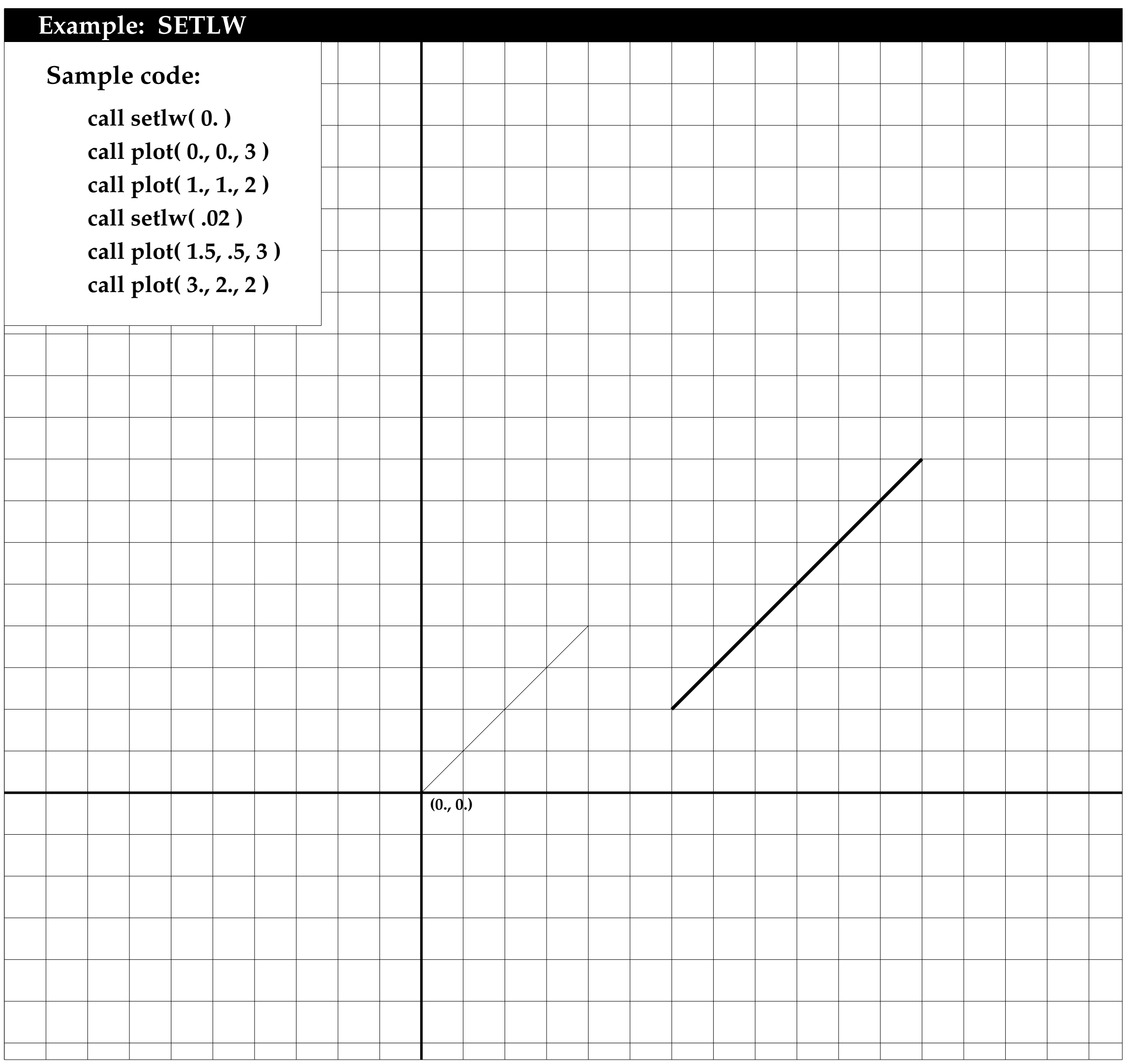
PURPOSE **SETLW** sets the current linewidth.

SYNTAX CALL SETLW (RLW)

<u>*ARGUMENTS*</u>

RLW	Line width.

After calling SETLW, RLW remains the current linewidth until SETLW is called again.



Characters and octal codes for Font ZapfDingbats

	✁	✂	✃	✄	☎	✆	✇	✈	✉	☛	☞	✌	✍	✎	✏
40	**41**	**42**	**43**	**44**	**45**	**46**	**47**	**50**	**51**	**52**	**53**	**54**	**55**	**56**	**57**
✐	✑	✒	✓	✔	✕	✖	✗	✘	✙	✚	✛	✜	✝	✞	✟
60	**61**	**62**	**63**	**64**	**65**	**66**	**67**	**70**	**71**	**72**	**73**	**74**	**75**	**76**	**77**
✠	✡	✢	✣	✤	✥	✦	✧	★	✩	✪	✫	✬	✭	✮	✯
100	**101**	**102**	**103**	**104**	**105**	**106**	**107**	**110**	**111**	**112**	**113**	**114**	**115**	**116**	**117**
✰	✱	✲	✳	✴	✵	✶	✷	✸	✹	✺	✻	✼	✽	✾	✿
120	**121**	**122**	**123**	**124**	**125**	**126**	**127**	**130**	**131**	**132**	**133**	**134**	**135**	**136**	**137**
❀	❁	❂	❃	❄	❅	❆	❇	❈	❉	❊	❋	●	❍	■	❏
140	**141**	**142**	**143**	**144**	**145**	**146**	**147**	**150**	**151**	**152**	**153**	**154**	**155**	**156**	**157**
❐	❑	❒	▲	▼	◆	❖	◗	❘	❙	❚	❛	❜	❝	❞	
160	**161**	**162**	**163**	**164**	**165**	**166**	**167**	**170**	**171**	**172**	**173**	**174**	**175**	**176**	**177**
	❡	❢	❣	❤	❥	❦	❧	♣	♦	♥	♠	①	②	③	④
240	**241**	**242**	**243**	**244**	**245**	**246**	**247**	**250**	**251**	**252**	**253**	**254**	**255**	**256**	**257**
⑤	⑥	⑦	⑧	⑨	⑩	❶	❷	❸	❹	❺	❻	❼	❽	❾	❿
260	**261**	**262**	**263**	**264**	**265**	**266**	**267**	**270**	**271**	**272**	**273**	**274**	**275**	**276**	**277**
➀	➁	➂	➃	➄	➅	➆	➇	➈	➉	➊	➋	➌	➍	➎	➏
300	**301**	**302**	**303**	**304**	**305**	**306**	**307**	**310**	**311**	**312**	**313**	**314**	**315**	**316**	**317**
➐	➑	➒	➓	➔	→	↔	↕	➘	➙	➚	➛	➜	➝	➞	➟
320	**321**	**322**	**323**	**324**	**325**	**326**	**327**	**330**	**331**	**332**	**333**	**334**	**335**	**336**	**337**
➠	➡	➢	➣	➤	➥	➦	➧	➨	➩	➪	➫	➬	➭	➮	➯
340	**341**	**342**	**343**	**344**	**345**	**346**	**347**	**350**	**351**	**352**	**353**	**354**	**355**	**356**	**357**
	➱	➲	➳	➴	➵	➶	➷	➸	➹	➺	➻	➼	➽	➾	
360	**361**	**362**	**363**	**364**	**365**	**366**	**367**	**370**	**371**	**372**	**373**	**374**	**375**	**376**	

SETFNT

PURPOSE **SETFNT** sets the current font.

SYNTAX CALL SETFNT (NFONT)

ARGUMENTS

NFONT Number of desired font between 1 and 35 (see table below).

The fonts sets 29 (Symbol) and 35 (ZapfDingbats) require the octal code of the character to be plotted. See the tables on the following pages for the character sets and corresponding octal code values for these two fonts.

Fonts for Subroutine SETFNT

1	AvantGarde-Book	*19*	*Helvetica-Oblique*
2	*AvantGarde-BookOblique*	20	Helvetica
3	**AvantGarde-Demi**	**21**	**NewCenturySchlbk-Bold**
4	***AvantGarde-DemiOblique***	***22***	***NewCenturySchlbk-BoldItalic***
5	**Bookman-Demi**	*23*	*NewCenturySchlbk-Italic*
6	***Bookman-DemiItalic***	24	NewCenturySchlbk-Roman
7	Bookman-Light	**25**	**Palatino-Bold**
8	*Bookman-LightItalic*	***26***	***Palatino-BoldItalic***
9	**Courier-Bold**	*27*	*Palatino-Italic*
10	***Courier-BoldOblique***	28	Palatino-Roman
11	*Courier-Oblique*	29	Symbol Σψμβολ
12	Courier	**30**	**Times-Bold**
13	**Helvetica-Bold**	***31***	***Times-BoldItalic***
14	***Helvetica-BoldOblique***	*32*	*Times-Italic*
15	**Helvetica-Narrow-Bold**	33	Times-Roman
16	***Helvetica-Narrow-BoldOblique***	*34*	*ZapfChancery-MediumItalic*
17	*Helvetica-Narrow-Oblique*	35	ZapfDingbats ✺❁❐❆✤❉■❇✲❁▼▲
18	Helvetica-Narrow		

RRECT

PURPOSE **RRECT** draws a rectangle (or square).with rounded corners. It can then fill the rectangle with the current graylevel or RGB values.

SYNTAX CALL RECT (XX1, YY1, WIDTH, HEIGHT, RAD, ANG, FILL)

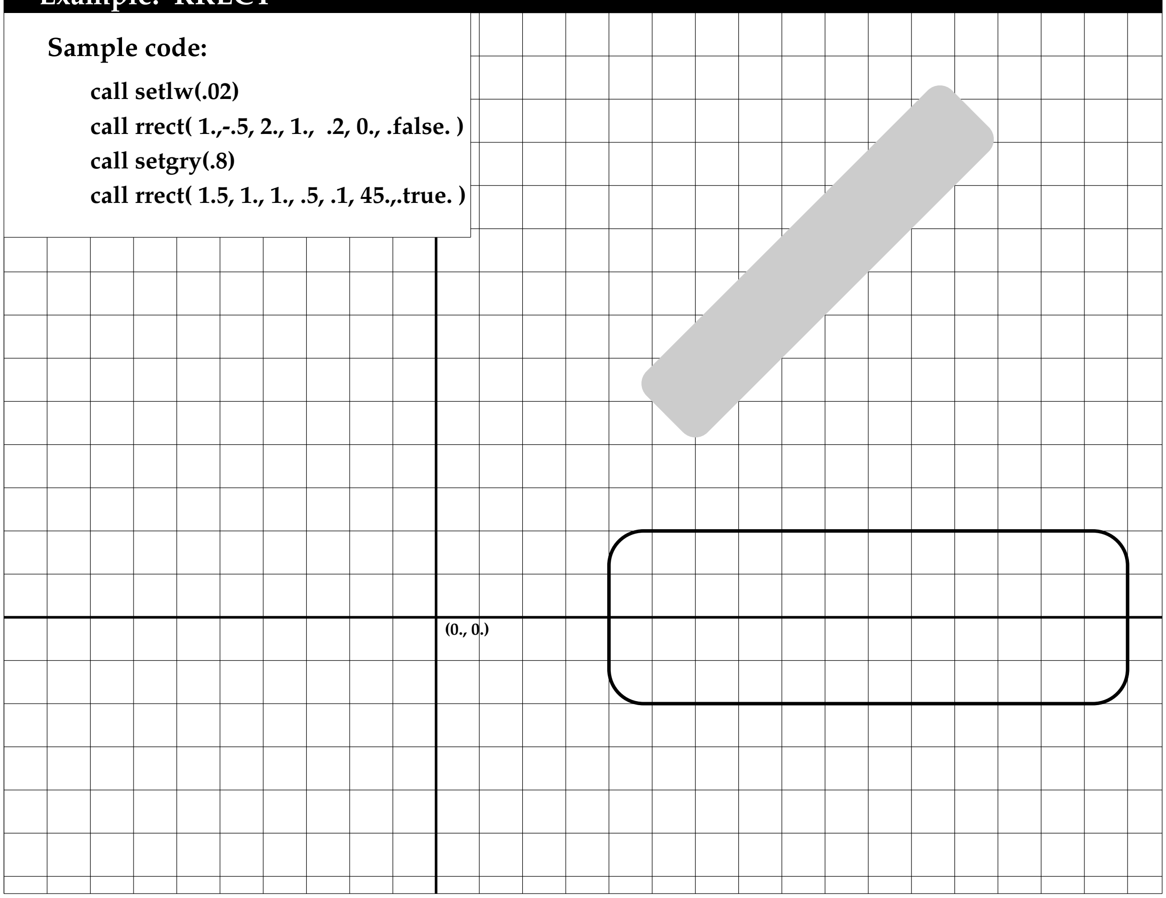
ARGUMENTS

XX1,YY1	X,Y coordinates of lower left corner of rectangle.
WIDTH	Width of rectangle.
HEIGHT	Height of rectangle.
RAD	Radius of rectangle corner curves.
ANG	Angle of rectangle rotation.
FILL	Logical value. If FILL=.true., then the rectangle is filled with the current graylevel or RGB values; otherwise, the rectangle is outlined.

Example: RRECT

Sample code:

```
call setlw(.02)
call rrect( 1.,-.5, 2., 1.,  .2, 0., .false. )
call setgry(.8)
call rrect( 1.5, 1., 1., .5, .1, 45.,.true. )
```



RECTFILG

PURPOSE **RECTFILG** draws a rectangle (or square) and fills it with a specified graylevel.

SYNTAX CALL RECTFILG (XX1, YY1, XX2, YY2, HEIGHT, GRYLEV)

ARGUMENTS

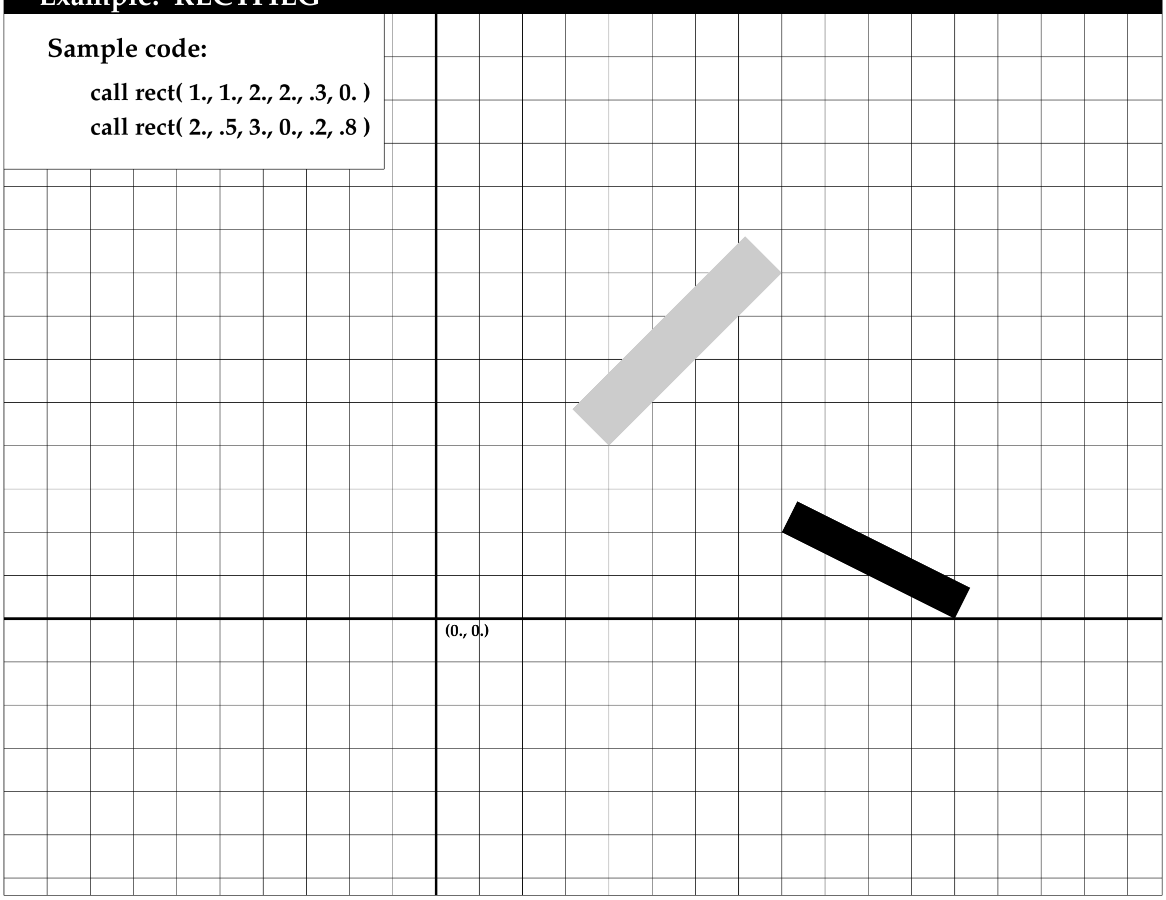
XX1,YY1 X,Y coordinates of lower left corner of rectangle.

XX2,YY2 X,Y coordinates of lower right corner of rectangle.

HEIGHT Height of rectangle.

GRYLEV Graylevel to use for filling rectangle. GRYLEV must lie beween 0. (black) and 1. (white), inclusive.

Example: RECTFILG



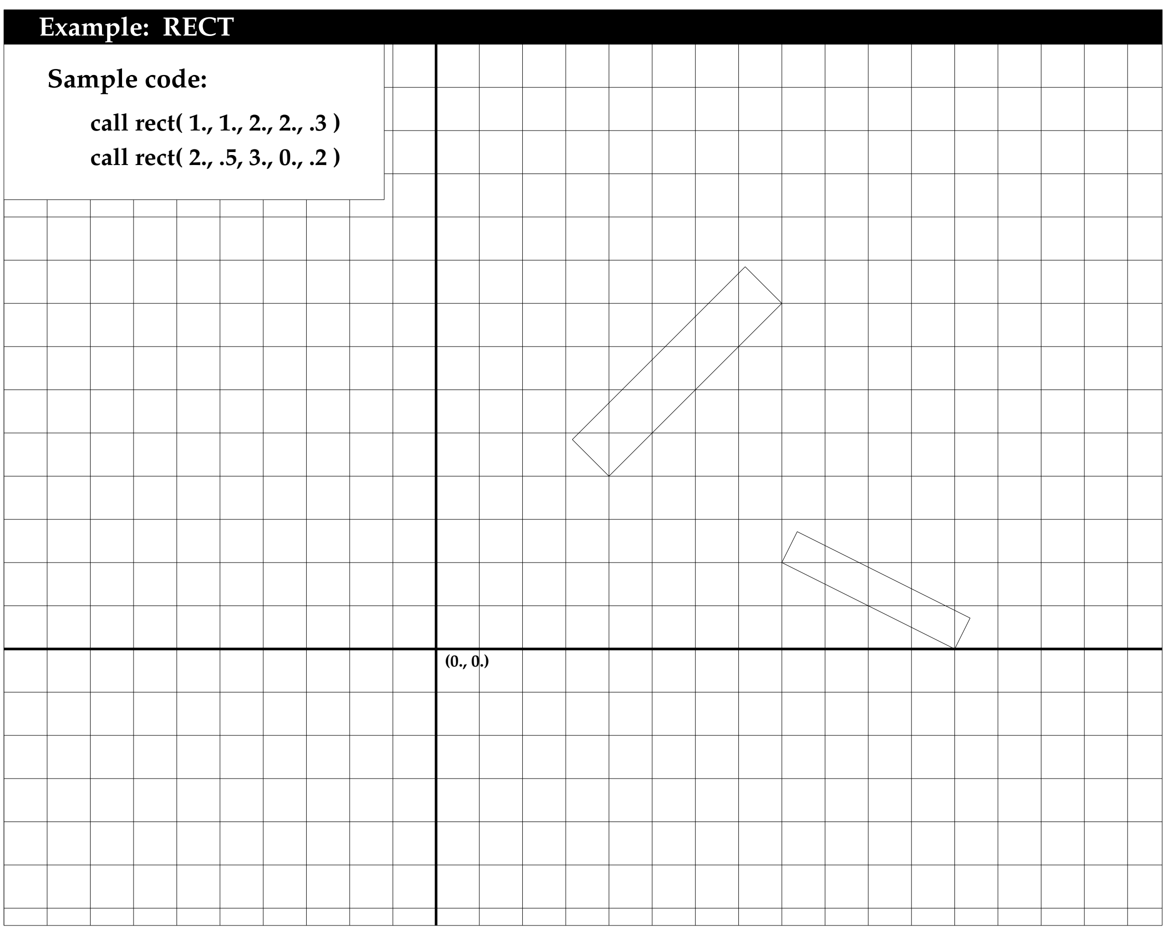
RECT

PURPOSE **RECT** draws a rectangle (or square).

SYNTAX CALL RECT (XX1, YY1, XX2, YY2, HEIGHT)

<u>ARGUMENTS</u>

XX1,YY1	X,Y coordinates of lower left corner of rectangle.
XX2,YY2	X,Y coordinates of lower right corner of rectangle.
HEIGHT	Height of rectangle.



PRIME

PURPOSE **PRIME** plots the symbol ′ .

SYNTAX CALL PRIME (XPP, YPP, HEIGHT, ANG, MJUS)

<u>*ARGUMENTS*</u>

XPP,YPP X,Y coordinates of the symbol to be plotted. Plotting may be continued from the end of a previously plotted character when used in conjunction with any of the subroutines which support continuation.

HEIGHT Height of symbol to be plotted.

ANG Angle, measured counterclockwise from the X-axis, at which the symbol is to be plotted.

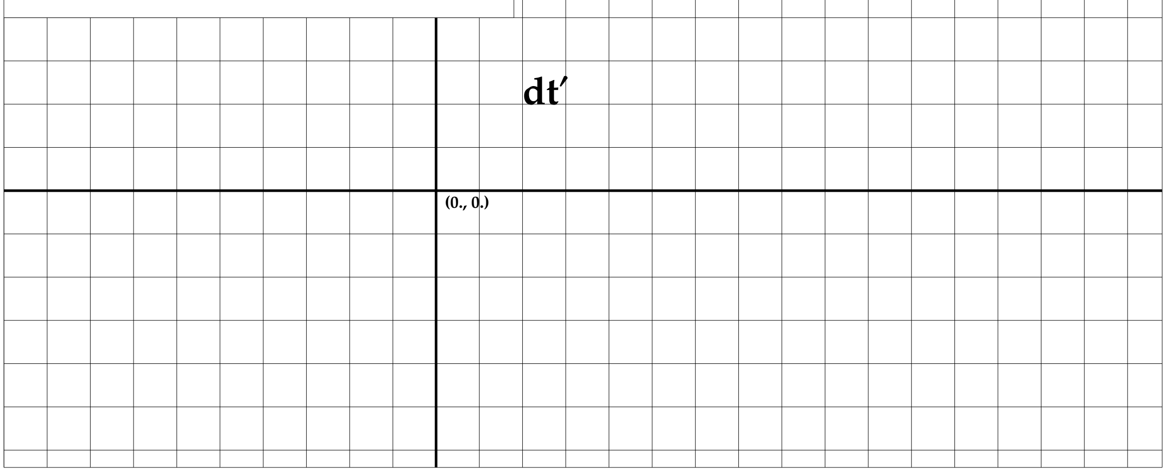
MJUS Controls the justification of the symbol to be plotted.

If MJUS = 0, (XPP,YPP) denotes the position of the lower left corner of the character.
If MJUS = 1, (XPP,YPP) denotes the position of the center of the character.
If MJUS = 2, (XPP,YPP) denotes the position of the lower right corner of the character.

Example: PRIME

Sample code:

```
call keksym( .5, .5, .14, 2hdt, 0., 2, 0 )
call prime( 999., 999., .14, 0., 0)
```



PLOTND

PURPOSE PLOTND closes the output PostScript plot file to terminate the current plot defined by the user application program.

SYNTAX CALL PLOTND

<u>*ARGUMENTS*</u>

None

You ***must*** call PLOTND (or PLOT (0., 0., 999)) as the last plotting call in the application program.

OVRGRK

PURPOSE **OVRGRK** plots a Greek symbol with an overbar.

SYNTAX CALL OVRGRK (XPP, YPP, HEIGHT, ICH, ANG, NCHAR, MJUS)

ARGUMENTS

XPP,YPP — X,Y coordinates of the Greek symbol to be plotted. Plotting may be continued from the end of a previously plotted character when used in conjunction with any of the subroutines which support continuation.

HEIGHT — Height of symbol to be plotted.

ICH — Integer value between 1 and 50 corresponding to the desired Greek symbol (see table below Subroutine GRKSYM).

ANG — Angle, measured counterclockwise from the X-axis, at which the symbol is to be plotted.

NCHAR — Number of characters in the string to plot (should be set to 1).

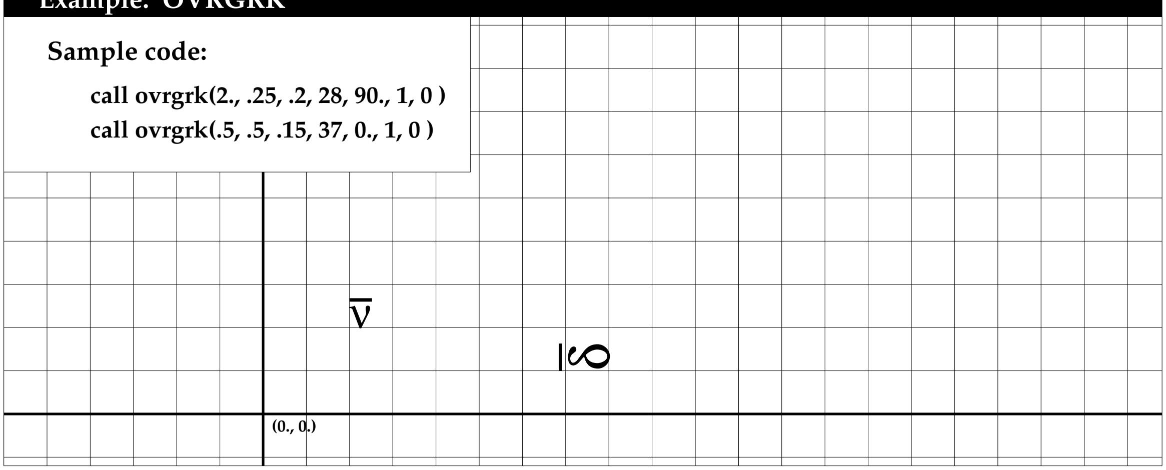
MJUS — Controls the justification of the symbol to be plotted.

If MJUS = 0, (XPP,YPP) denotes the position of the lower left corner of the symbol.
If MJUS = 1, (XPP,YPP) denotes the position of the center of the symbol.
If MJUS = 2, (XPP,YPP) denotes the position of the lower right corner of the symbol.

Example: OVRGRK

Sample code:

```
call ovrgrk(2., .25, .2, 28, 90., 1, 0 )
call ovrgrk(.5, .5, .15, 37, 0., 1, 0 )
```



OVERSUP

PURPOSE **OVERSUP** plots a superscripted character string with an overbar. The overbar extends to cover the subscript.

SYNTAX CALL OVERSUP (XPP, YPP, HEIGHT, ICH, ANG, NCHAR, MJUS,ISUP, NSUP)

<u>*ARGUMENTS*</u>

XPP,YPP X,Y coordinates of the character string to be plotted. Plotting may be continued from the end of a previously plotted character when used in conjunction with any of the subroutines which support continuation.

HEIGHT Height of character string to be plotted.

IBCD Character string to be plotted (Hollerith).

ANG Angle, measured counterclockwise from the X-axis, at which the character string is to be plotted.

NCHAR Number of characters in the string to plot.

MJUS Controls the justification of the character string to be plotted.

If MJUS = 0, (XPP,YPP) denotes the position of the lower left corner of the character string.
If MJUS = 1, (XPP,YPP) denotes the position of the center of the character string.
If MJUS = 2, (XPP,YPP) denotes the position of the lower right corner of the character string.

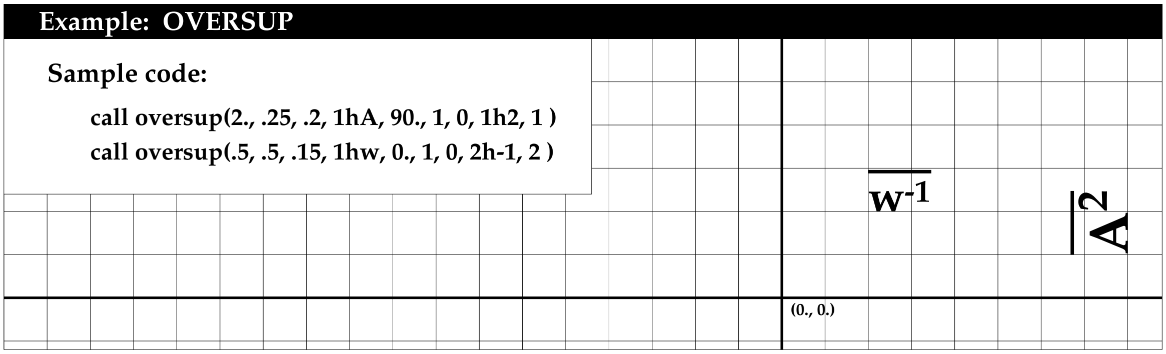
ISUP Hollerith character(s) comprising the superscript.

NSUP Number of characters in the superscript string.

Example: OVERSUP

Sample code:

```
call oversup(2., .25, .2, 1hA, 90., 1, 0, 1h2, 1 )
call oversup(.5, .5, .15, 1hw, 0., 1, 0, 2h-1, 2 )
```



OVERSUB

PURPOSE **OVERSUB** plots a subscripted character string with an overbar. The overbar extends to cover the subscripts.

SYNTAX CALL OVERSUB (XPP, YPP, HEIGHT, IBCD, ANG, NCHAR, MJUS, ISUB, NSUB)

<u>*ARGUMENTS*</u>

XPP,YPP X,Y coordinates of the character string to be plotted. Plotting may be continued from the end of a previously plotted character when used in conjunction with any of the subroutines which support continuation.

HEIGHT Height of character string to be plotted.

IBCD Character string to be plotted (Hollerith).

ANG Angle, measured counterclockwise from the X-axis, at which the character string is to be plotted.

NCHAR Number of characters in the string to plot.

MJUS Controls the justification of the character string to be plotted.

If MJUS = 0, (XPP,YPP) denotes the position of the lower left corner of the character string.
If MJUS = 1, (XPP,YPP) denotes the position of the center of the character string.
If MJUS = 2, (XPP,YPP) denotes the position of the lower right corner of the character string.

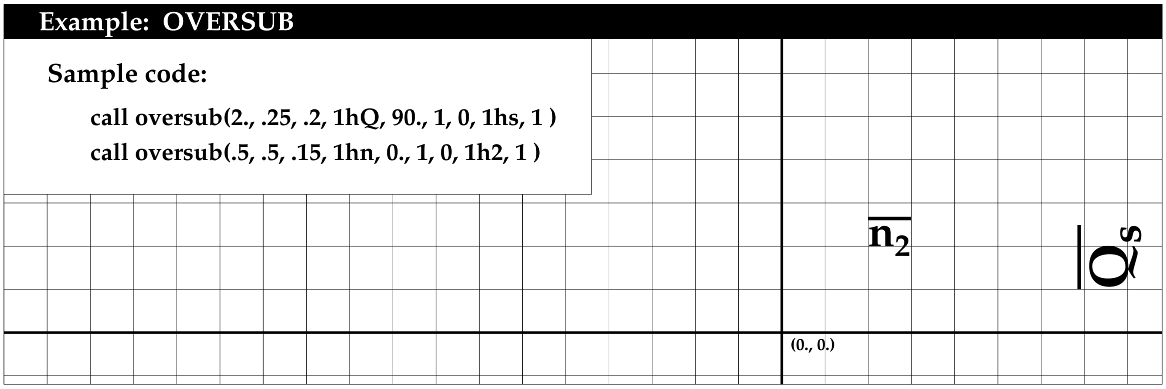
ISUB Hollerith character(s) comprising the subscript.

NSUB Number of characters in the subscript string.

Example: OVERSUB

Sample code:

```
call oversub(2., .25, .2, 1hQ, 90., 1, 0, 1hs, 1 )
call oversub(.5, .5, .15, 1hn, 0., 1, 0, 1h2, 1 )
```



OVERSBSP

PURPOSE **OVERSBSP** plots a subscripted and superscripted character string with an overbar. The overbar extends to cover both the subscripts and superscripts.

SYNTAX CALL OVERSBSP (XPP, YPP, HEIGHT, IBCD, ANG, NCHAR, MJUS,ISUB, NSUB, ISUP, NSUP)

<u>*ARGUMENTS*</u>

XPP,YPP X,Y coordinates of the character string to be plotted. Plotting may be continued from the end of a previously plotted character when used in conjunction with any of the subroutines which support continuation.

HEIGHT Height of character string to be plotted.

IBCD Character string to be plotted (Hollerith).

ANG Angle, measured counterclockwise from the X-axis, at which the character string is to be plotted.

NCHAR Number of characters in the string to plot.

MJUS Controls the justification of the character string to be plotted.

If MJUS = 0, (XPP,YPP) denotes the position of the lower left corner of the character string.
If MJUS = 1, (XPP,YPP) denotes the position of the center of the character string.
If MJUS = 2, (XPP,YPP) denotes the position of the lower right corner of the character string.

ISUB Hollerith character(s) comprising the subscript.

NSUB Number of characters in the subscript string.

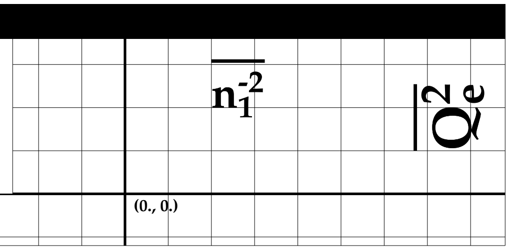
ISUP Hollerith character(s) comprising the superscript.

NSUP Number of characters in the superscript string.

Example: OVERSBSP

Sample code:

```
call oversbsp(2., .25, .2, 1hQ, 90., 1, 0, 1he, 1, 1h2, 1 )
call oversbsp(.5, .5, .15, 1hn, 0., 1, 0, 1h1, 1, 2h-2, 2 )
```



ONEHLF

PURPOSE **ONEHLF** plots the symbol ½ .

SYNTAX CALL ONEHLF (XPP, YPP, HEIGHT, ANG, MJUS)

ARGUMENTS

XPP,YPP X,Y coordinates of the symbol to be plotted.

HEIGHT Height of symbol to be plotted.

ANG Angle, measured counterclockwise from the X-axis, at which the symbol is to be plotted.

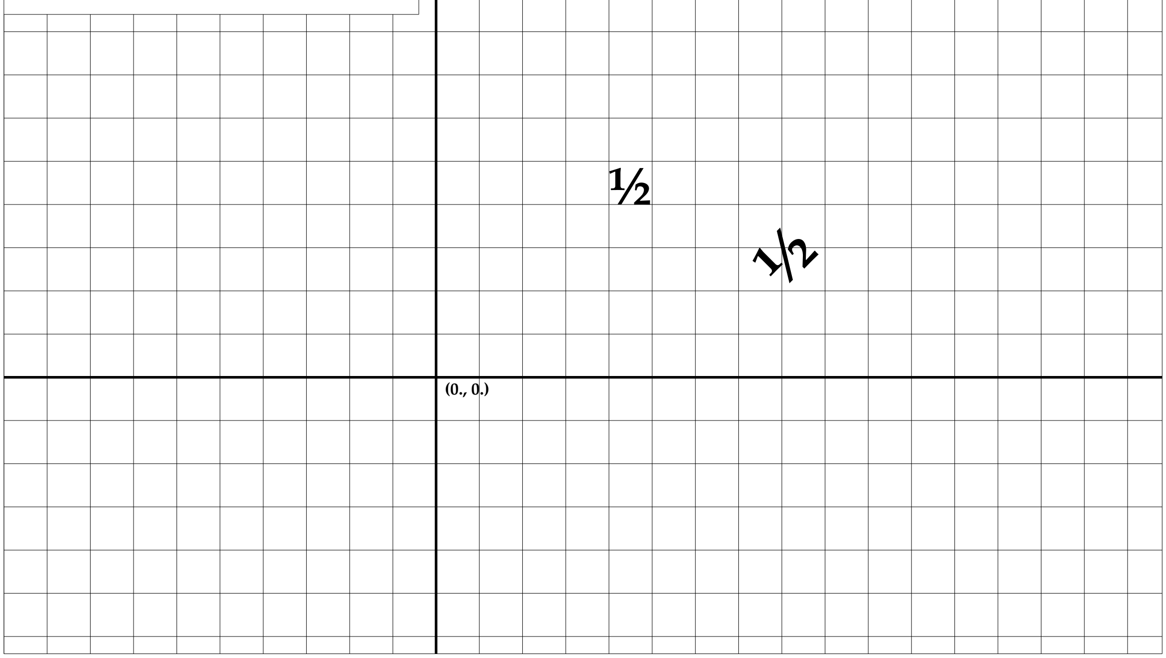
MJUS Controls the justification of the character string to be plotted.

If MJUS = 0, (XPP,YPP) denotes the position of the lower left corner of the character string.
If MJUS = 1, (XPP,YPP) denotes the position of the center of the character string.
If MJUS = 2, (XPP,YPP) denotes the position of the lower right corner of the character string.

Example: ONEHLF

Sample code:

```
call onehlf(1., 1., .2, 0., 0 )
call onehlf(2., .5, .25, 45., 0 )
```



NEWDEV

PURPOSE **NEWDEV** specifies a non-default filename which will contain the application program's output PostScript plot file.

SYNTAX CALL NEWDEV (FLNAME, NCHAR)

ARGUMENTS

FLNAME Character string or FORTRAN character variable which contains the name of the output Postscript file.

NCHAR Number of characters in the filename.

For example,

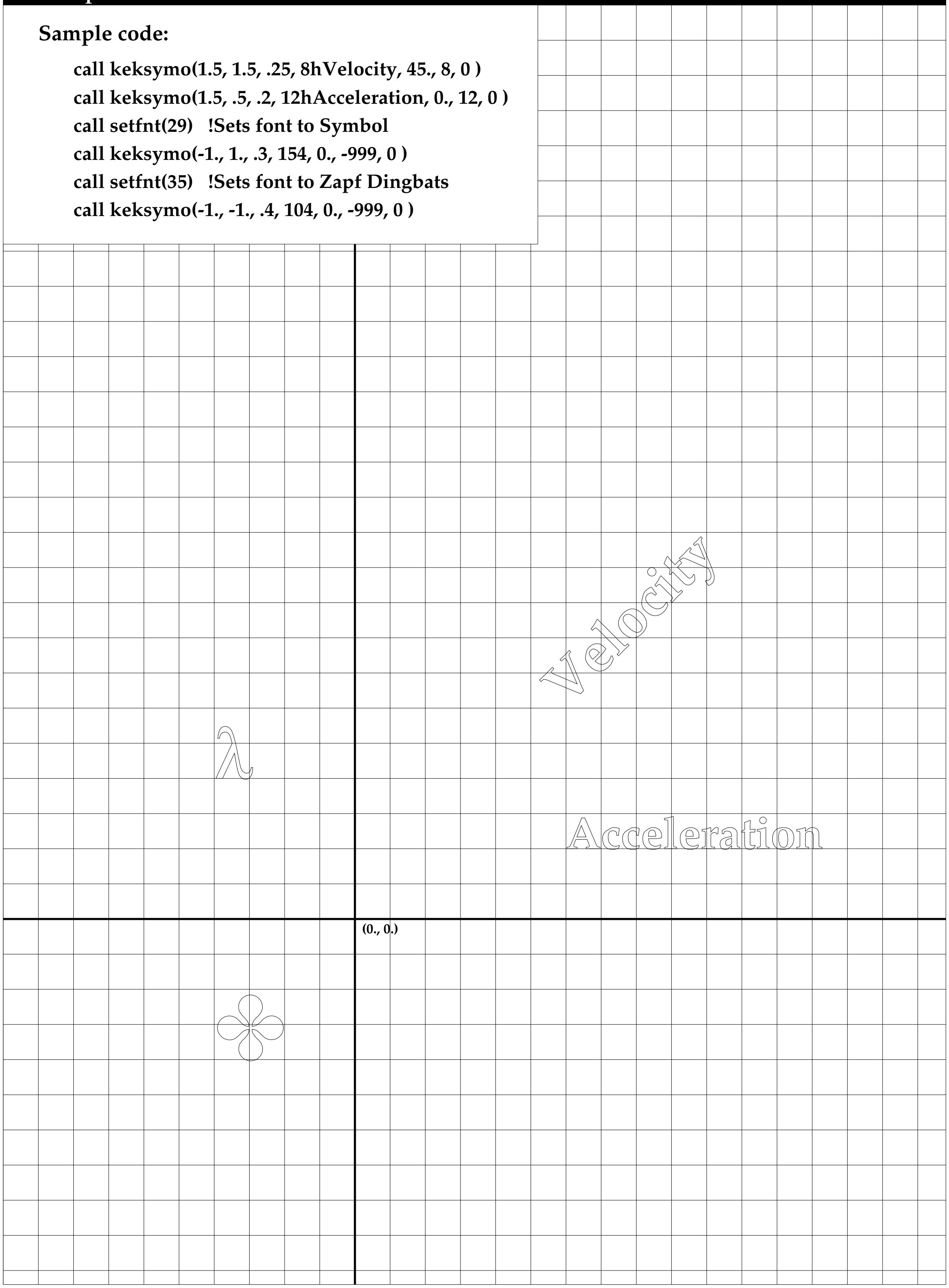
```
CALL NEWDEV ('MYPLOT.DAT', 10)
```

NEWDEV is called only once per plotting session and must be called prior to PSINIT. If NEWDEV is not called, the name of the output PostScript file is **psplot.ps**.

Example: KEKSYMO

Sample code:

```
call keksymo(1.5, 1.5, .25, 8hVelocity, 45., 8, 0 )
call keksymo(1.5, .5, .2, 12hAcceleration, 0., 12, 0 )
call setfnt(29)   !Sets font to Symbol
call keksymo(-1., 1., .3, 154, 0., -999, 0 )
call setfnt(35)   !Sets font to Zapf Dingbats
call keksymo(-1., -1., .4, 104, 0., -999, 0 )
```



KEKSYMC

PURPOSE **KEKSYMC** plots a character string. Same as **KEKSYM**, but text is input as a character string rather than Hollerith.

SYNTAX CALL KEKSYMC (XPP, YPP, HEIGHT, CSTR, ANG, NCHAR, MJUS)

<u>*ARGUMENTS*</u>

XPP,YPP	X,Y coordinates of the character string to be plotted. Plotting may be continued from the end of a previously plotted character when used in conjunction with any of the subroutines which support continuation.
HEIGHT	Height of character string to be plotted.
CSTR	Character string to be plotted (Hollerith).
ANG	Angle, measured counterclockwise from the X-axis, at which the character string is to be plotted.
NCHAR	Number of characters in the string to plot.
MJUS	Controls the justification of the character string to be plotted. If MJUS = 0, (XPP,YPP) denotes the position of the lower left corner of the character string. If MJUS = 1, (XPP,YPP) denotes the position of the center of the character string. If MJUS = 2, (XPP,YPP) denotes the position of the lower right corner of the character string.

<u>Special feature of KEKSYMC:</u>

You can plot special characters, for example the characters in fonts Symbol and Zapf Dingbats, by setting NCHAR=-999 and CSTR equal to the octal code of the character you want to plot. CSTR must be in single quotes.

KEKSYM

PURPOSE **KEKSYM** plots a character string.

SYNTAX CALL KEKSYM (XPP, YPP, HEIGHT, IBCD, ANG, NCHAR, MJUS)

<u>*ARGUMENTS*</u>

XPP,YPP X,Y coordinates of the character string to be plotted. Plotting may be continued from the end of a previously plotted character when used in conjunction with any of the subroutines which support continuation.

HEIGHT Height of character string to be plotted.

IBCD Character string to be plotted (Hollerith).

ANG Angle, measured counterclockwise from the X-axis, at which the character string is to be plotted.

NCHAR Number of characters in the string to plot.

MJUS Controls the justification of the character string to be plotted.

If MJUS = 0, (XPP,YPP) denotes the position of the lower left corner of the character string.
If MJUS = 1, (XPP,YPP) denotes the position of the center of the character string.
If MJUS = 2, (XPP,YPP) denotes the position of the lower right corner of the character string.

<u>***Special feature of KEKSYM:***</u>

You can plot special characters, for example the characters in fonts Symbol and Zapf Dingbats, by setting NCHAR=-999 and IBCD equal to the octal code of the character you want to plot.

KEKFLT

PURPOSE **KEKFLT** is the same as KEKNUM.

SYNTAX CALL KEKFLT (XPP, YPP, HEIGHT, FNUM, ANG, NDEC, MJUS)

ARGUMENTS

Same as KEKNUM.

INTEGRAL

PURPOSE **INTEGRAL** plots an integral with upper and lower limits.

SYNTAX CALL INTEGRAL (XPP, YPP, HEIGHT, ANG, LOWER, NL, LUPPER, NU)

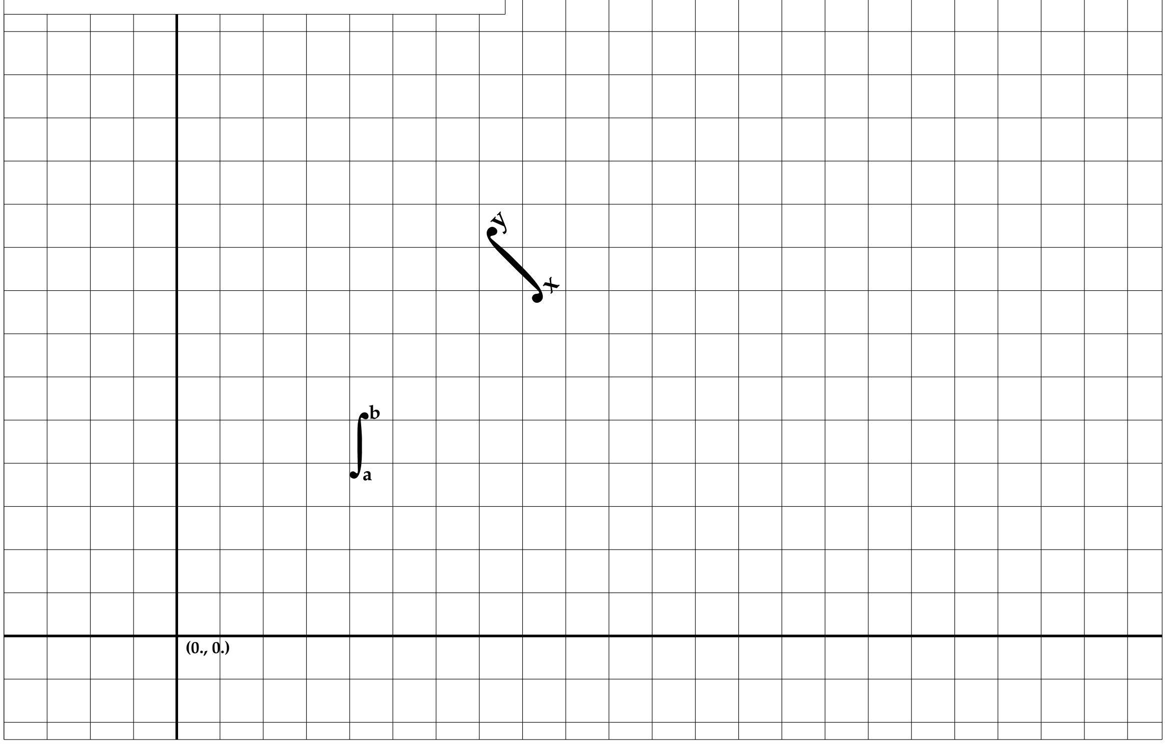
ARGUMENTS

XPP,YPP	X,Y coordinates of the lower left edge of the integral.
HEIGHT	Height of the integral.
ANG	Angle, measured counterclockwise from the X-axis, at which the integral is plotted.
LOWER	Character string of the lower limit (Hollerith).
NL	Number of characters in the lower limit text.
LUPPER	Character string of the upper limit (Hollerith).
NU	Number of characters in the upper limit text.

Example: INTEGRAL

Sample code:

```
call integral(1.,1.,.15,0.,1ha,1,1hb,1)
call integral(2.,2.,.2,45.,1hx,1,1hy,1)
```



HILITEG

PURPOSE **HILITEG** prints text surrounded by a rectangular box (grayscale).

SYNTAX CALL HILITEG (XPP, YPP, HEIGHT, CSTR, ANG, EDG, JUSX, JUSY, FGRY, BGRY)

<u>*ARGUMENTS*</u>

XPP,YPP	X,Y coordinates of the box surrounding the text.
HEIGHT	Height of text to be plotted.
CSTR	Character string text to be plotted.
ANG	Angle, measured counterclockwise from the X-axis, at which the text is to be plotted.
EDG	Fraction of the text height to use as an edge border.
JUSX	Justification in the x-direction. JUSX=0 is left justification. JUSX=1 is centered. JUSX=2 is right justification.
JUSY	Similar to JUSX, but in the y-direction. JUSY=0 is bottom justification. JUSY=1 is centered. JUSY=2 is top justification.
FGRY	Graylevel of the (foreground) text. FGRY must lie between 0. and 1.0, inclusive.
BGRY	Graylevel of the (background) surrounding box. BGRY must lie between 0. and 1.0, inclusive.

GSAV

PURPOSE **GSAV** calls the PostScript operator *gsave*. It is useful before establishing and using a clipping region.

SYNTAX CALL GSAV

<u>*ARGUMENTS*</u>

NONE

GSAV is used to save the graphics state of the PostScript page prior to defining a clipping region. After the clipping region is used, GREST is called to restore the original graphics state.

GREST

PURPOSE **GREST** calls the PostScript operator *grestore*. It is useful after establishing and using a clipping region when it is no longer needed.

SYNTAX CALL GREST

ARGUMENTS

NONE

GREST is used to restore the graphics state of a PostScript page. It is called after a clipping region has been established and is no longer needed. GREST is used in conjunction with GSAV.

FILRGN

PURPOSE **FILRGN** is the same as FILLBOX, except it restores the current gray level after filling.

SYNTAX CALL FILRGN (XARR, YARR, NPTS, GRY)

ARGUMENTS

XARR — Array containing the X coordinates of the points forming the perimeter of the shape to fill.

YARR — Array containing the Y coordinates of the points forming the perimeter of the shape to fill.

NPTS — Number of points comprising the shape perimeter.

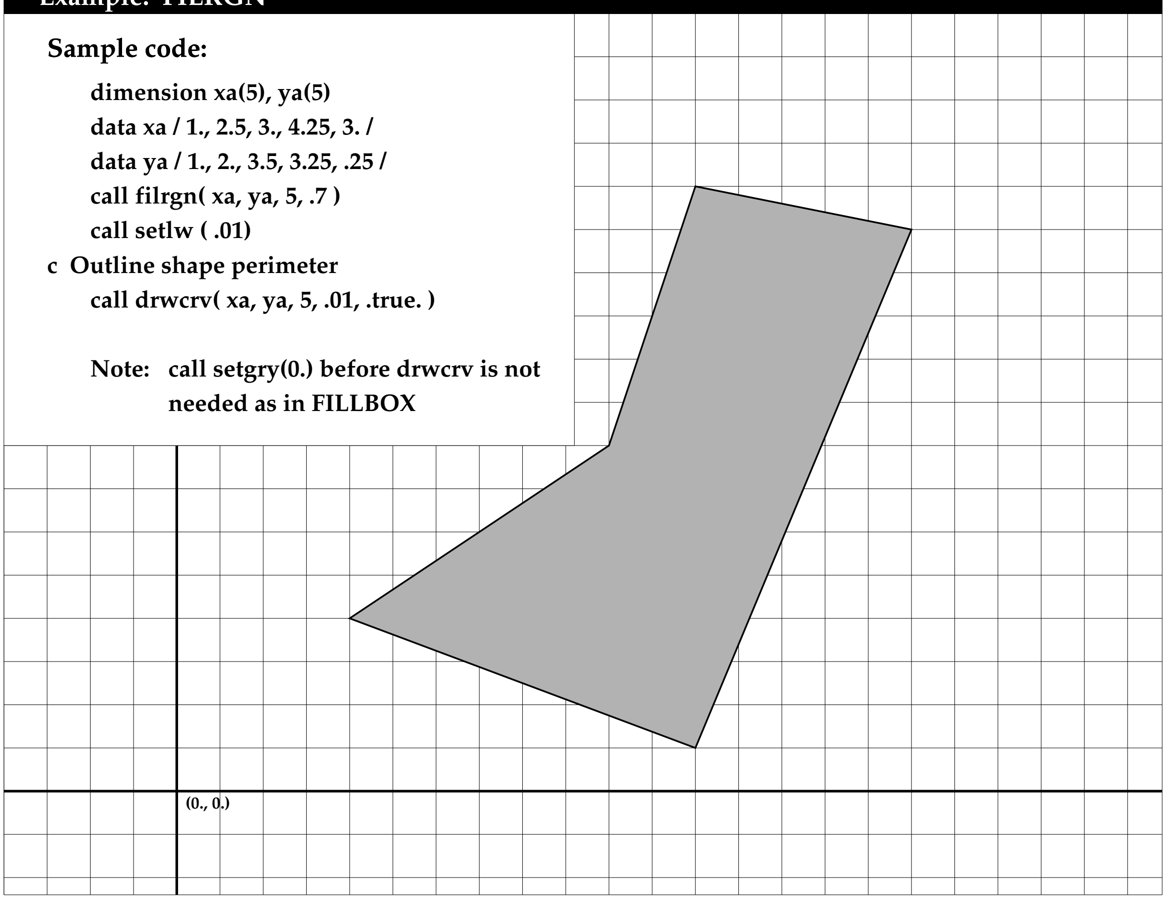
GRY — Gray level of filled shape. GRY must lie between 0.(black) and 1. (white), inclusive.

Example: FILRGN

Sample code:

```
      dimension xa(5), ya(5)
      data xa / 1., 2.5, 3., 4.25, 3. /
      data ya / 1., 2., 3.5, 3.25, .25 /
      call filrgn( xa, ya, 5, .7 )
      call setlw ( .01)
c  Outline shape perimeter
      call drwcrv( xa, ya, 5, .01, .true. )
```

Note: call setgry(0.) before drwcrv is not needed as in FILLBOX



FAROHED

PURPOSE **FAROHED** draws a "fancy" arrowhead at a specified location.

SYNTAX CALL FAROHED (XPP, YPP, DIR, AROLNP, SPRANG, LOCXY, FILL)

<u>*ARGUMENTS*</u>

XPP,YPP X,Y coordinates of the tip of the arrowhead.

DIR Direction of arrowhead, measured east from north.

AROLNP Length of arrowhead sides.

SPRANG Half the angular spread of arrowhead.

LOCXY Location of arrowhead point

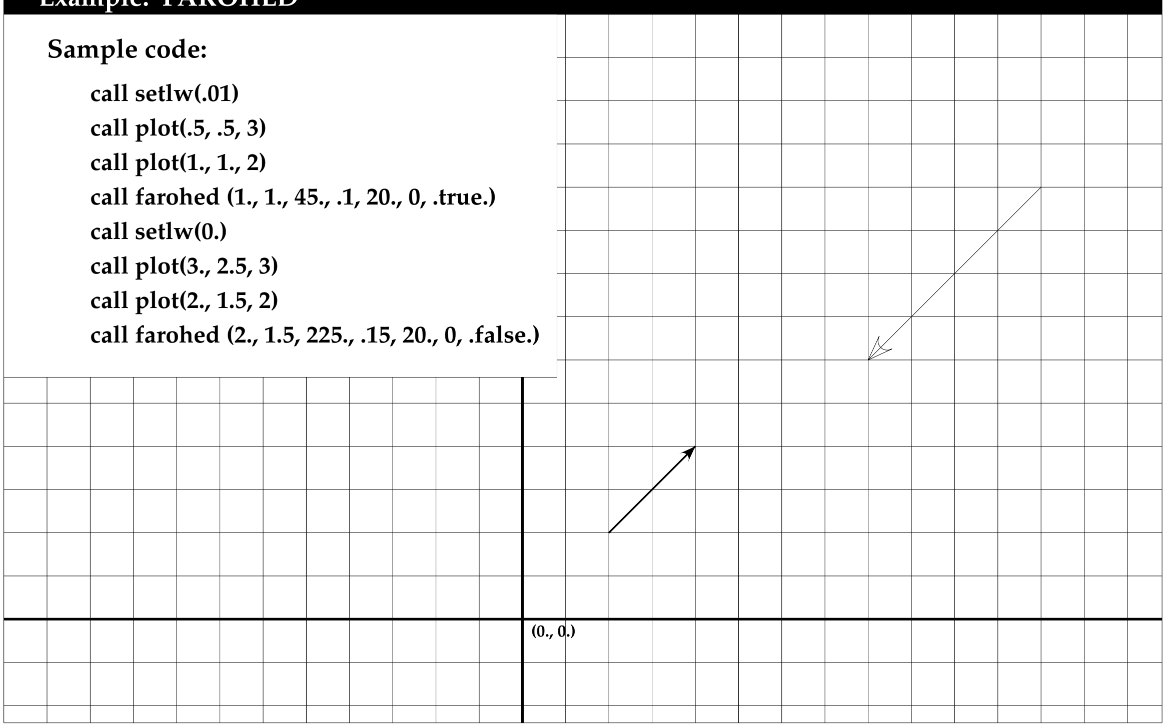
LOCXY=1	XPP,YPP at arrowhead point (most commonly used)
LOCXY=2	XPP, YPP at center of arrowhead.
LOCXY=3	XPP,YPP at tail of arrowhead.

FILL Logical variable. If FILL= .TRUE., the arrowhead is filled; otherwise it is outlined.

Example: FAROHED

Sample code:

```
call setlw(.01)
call plot(.5, .5, 3)
call plot(1., 1., 2)
call farohed (1., 1., 45., .1, 20., 0, .true.)
call setlw(0.)
call plot(3., 2.5, 3)
call plot(2., 1.5, 2)
call farohed (2., 1.5, 225., .15, 20., 0, .false.)
```



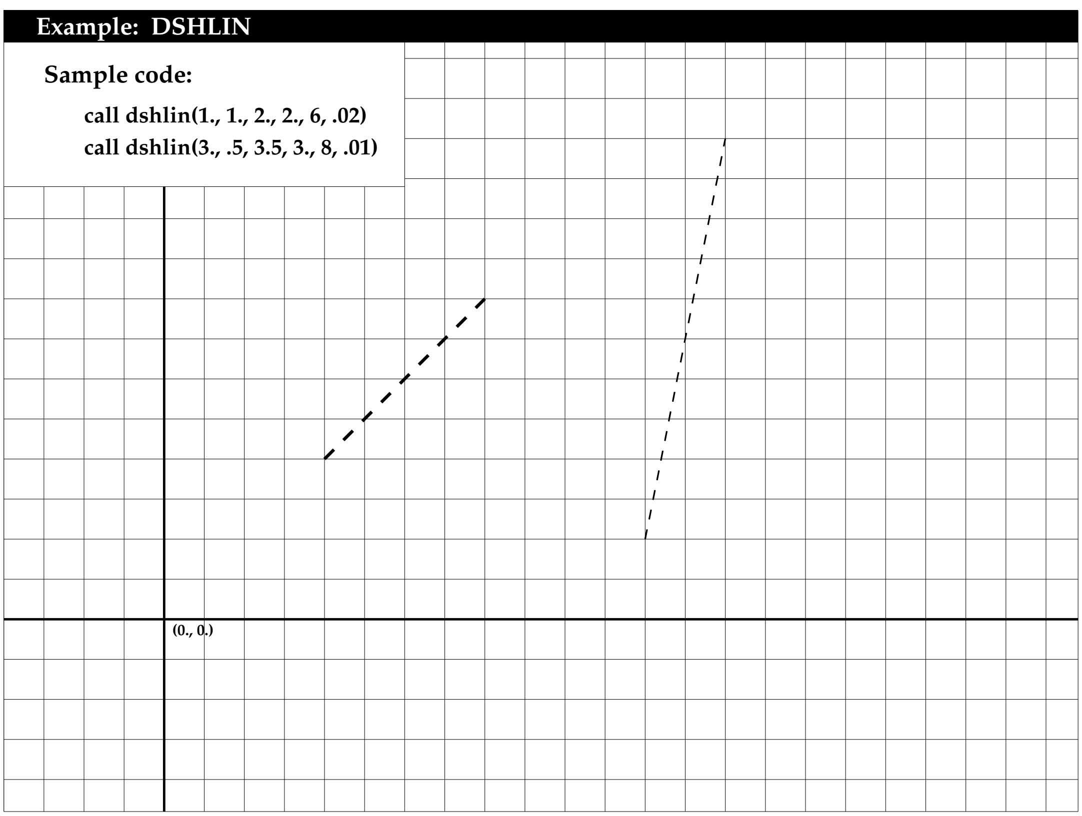
DSHLIN

PURPOSE **DSHLIN** connects two points using a line with a specified dash pattern.

SYNTAX CALL DSHLIN (X1, Y1, X2, Y2, IDSHPN, THK)

ARGUMENTS

X1,Y1	X,Y coordinates of first data point.
X2,Y2	X,Y coordinates of second data point.
IDSHPN	Dashed pattern of the line used to connect the two data points. IDSHPN is specified as the number of times the dashed pattern (solid line/blank space) is repeated per inch.
THK	Thickness of the dashes in the connecting line. If THK = 0., the current line width is used.



DRWTRI

PURPOSE **DRWTRI** draws an equilateral triangle.

SYNTAX CALL DRWTRI (XC, YC, SIDE, THK)

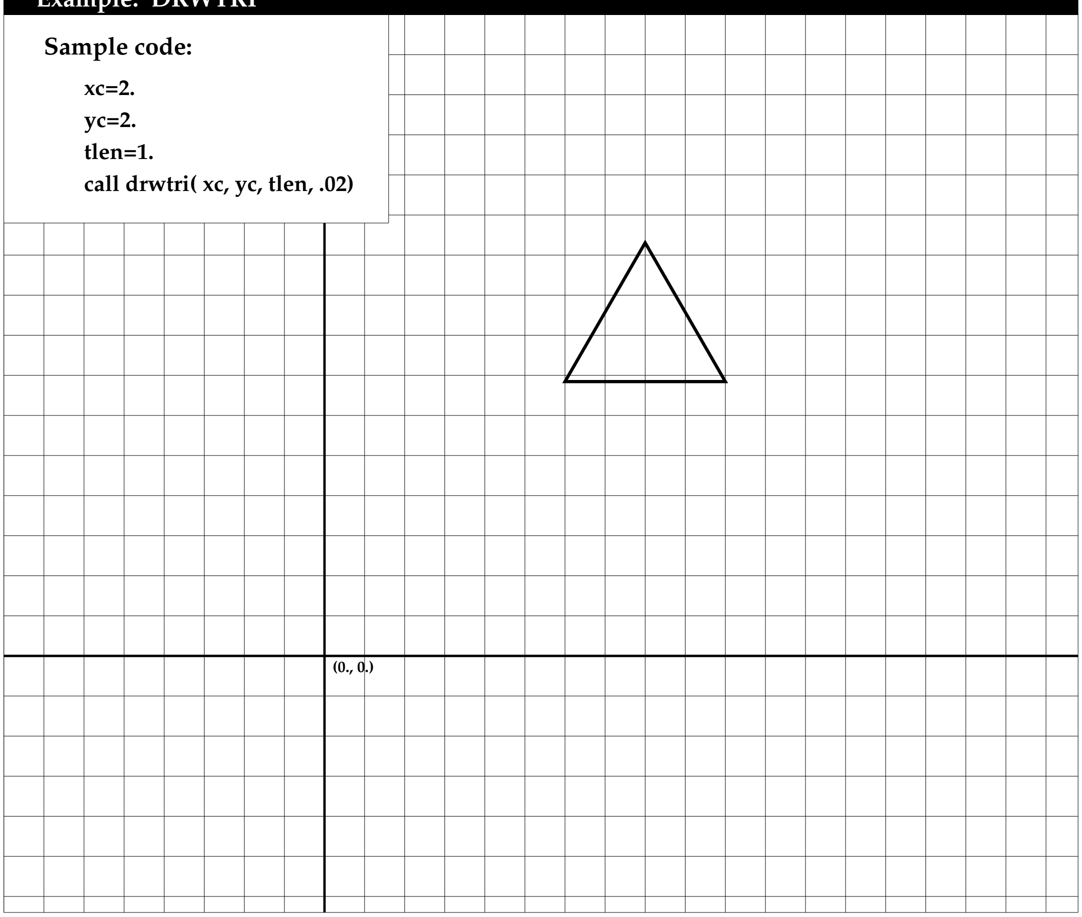
<u>*ARGUMENTS*</u>

XC, YC	X, Y coordinates of triangle **center**.
SIDE	Length of triangle side.
THK	Thickness of line used to draw the triangle. If THK = 0., the current line width is used.

Example: DRWTRI

Sample code:

```
xc=2.
yc=2.
tlen=1.
call drwtri( xc, yc, tlen, .02)
```



CURVE

PURPOSE **CURVE** draws a curve between two points, given the beginning and ending curve slopes.

SYNTAX CALL CURVE (X1, Y1, X2, Y2, X3, Y3, X4, Y4, CONTIN)

<u>*ARGUMENTS*</u>

X1,Y1 X,Y coordinates of the beginning of the curve.

X2,Y2 X, Y coordinates specifying the beginning curve slope. The initial slope of the curve is given by (Y2-Y1)/(X2-X1).

X3,Y3 X,Y coordinates specifying the ending curve slope. The end slope of the curve is given by (Y4-Y3)/(X4-X3).

X4,Y4 X,Y coordinates of the endpoint of the curve.

CONTIN Logical value. If CONTIN=.false. then the curve drawn starts a newpath; otherwise, the curve is appended to the current path.

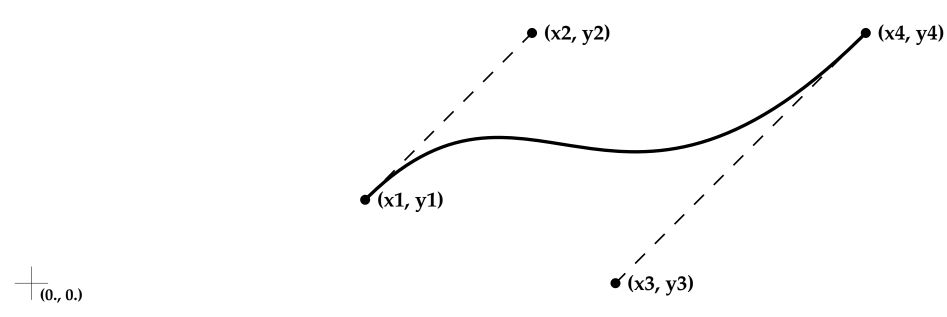
Note: The curve produced by CURVE **is not** automatically stroked.

Example: CURVE

Sample code:

```
data x1, x2, x3, x4/2., 3., 3.5, 5./
data y1, y2, y3, y4/.5, 1.5, 0., 1.5/
call setlw(.02)
call curve(x1,y1,x2,y2,x3,y3,x4,y4,.false.)
call stroke
```

Code to indicate points and dashed lines not shown.



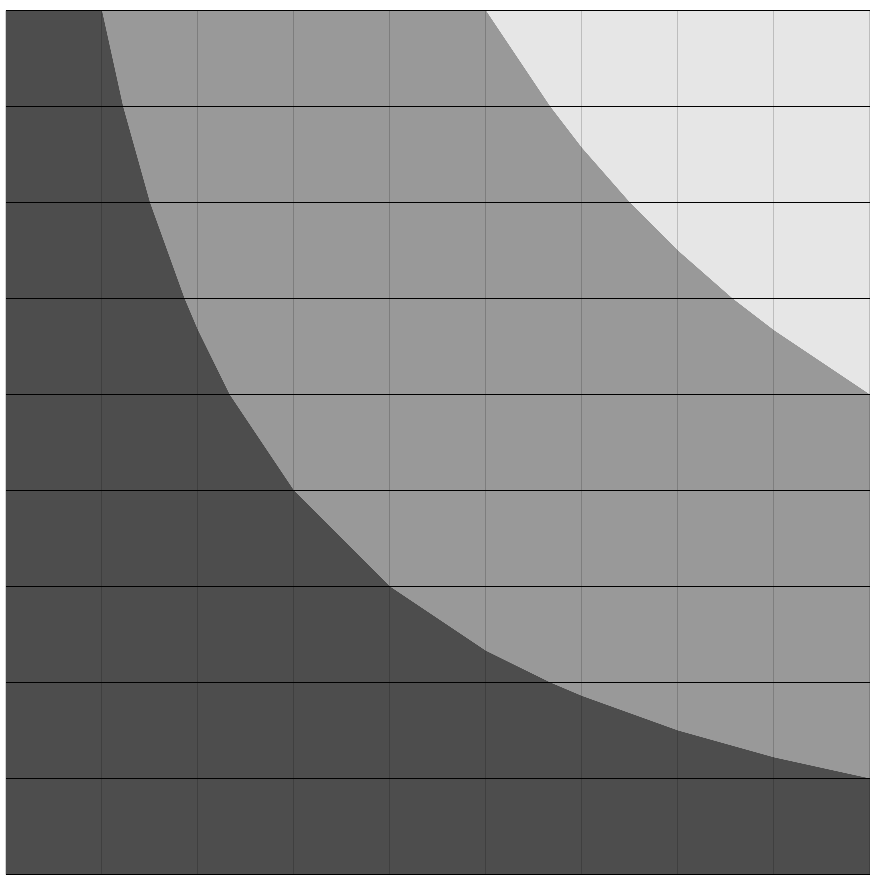
IOFFP	If 1, you can specify regions not to contour by using SPVAL. Grid boxes with any corner values equal to SPVAL are not contoured. Default = 0.
SPVAL	Special value to denote areas not to contour (usually set to 999.). Default = 0.
ILEGG	If 0, legend below contour plot showing contour values, scale factors, etc. will not be printed. Default = 1.
ILABB	If 0, contour labels are not plotted. Default = 1.
NHII	If 0, H and L mark relative highs and lows, with the data value plotted underneath the symbol. If > 0, values are plotted at each array point. If < 0, neither of the above is done. Default = 0.
NDECCN	Number of digits to the right of the decimal point in the contour labels. Default = 1.
NLBLL	Number of unlabeled contours between labelled contours. Default = 3.
LSCAL	If LSCAL = 0, the contours are scaled such that 0 < labeled values < 100. If LSCAL = 1, the contours are not scaled, i.e. the contour values are the actual data values. Default = 0.
LDASH	Specifies whether contours are solid (LDASH = 0) or dashed (LDASH NE 0). If LDASH is not equal to 0, then LDASH specifies the type of dash line to use to draw the contours. The numeric value of LDASH is the number of times the dashed pattern (solid line/blank space) is repeated per inch. Default= 0.
HGTLAB	Specifies the height of contour labels. If HGTLAB=0, the height of the contours is .11 inches.

Example: CONFILL

Sample code:

```
      do 10 j=1,10
      do 10 i=1,10
10    f(i,j)=i*j
      xlen=5.
      ylen=5.
      nval=6
      do 20 n=1,nval
      cval(n)=(n-1)*20
      if(cval(n).le.20.) then
       grylev(n)=.3
      elseif(cval(n).le.60.) then
       grylev(n)=.6
      else
       grylev(n)=.9
      endif
      continue
20    call confill(f,10,10,10,xlen,ylen,cval,grylev,nval,ioffp,spval)
```

Code to draw grid boxes not shown.



CONCOLR

PURPOSE **CONCOLR** draws color contour plots of a two-dimensional array of regularly spaced data.

SYNTAX CALL CONCOLR (ARR, IMAX, IEXT, JEXT, XLEN, YLEN, CVAL, COLOR, NVAL, IOFFP, SPVAL)

<u>*ARGUMENTS*</u>

ARR — Two-dimensional array containing regularly spaced data to be contoured. CONCOLR assumes ARR(1,1) is located at the lower left corner of the plot. Data points ARR(1,1) through ARR(IEXT,JEXT) are contoured in an area XLEN x YLEN.

IMAX — The first dimension of ARR in the calling program.

IEXT — Number of points in x-direction of ARR to be contoured.

JEXT — Number of points in y-direction of ARR to be contoured.

XLEN — X-direction length of plotting area.

YLEN — Y-direction length of plotting area.

CVAL — Array containing the values to be used for contour intervals (1 to NVAL). CVAL must be dimensioned with a dimension of at least 1. Unlike in subroutine CONREC, NVAL cannot equal 0.

COLOR — Array dimensioned (3,NVAL) containing the red, green, and blue values for each of the contour levels. The red value is stored in (1,n), the green value in (2,n) and the blue value in (3,n), where n specifies the contour index corresponding to CVAL. Regions less than or equal to the corresponding contour value CVAL are filled with the corresponding RBG values.

NVAL — Number of contour intervals. NVAL must be less than or equal to 100, and unlike in CONREC, NVAL cannot equal 0.

IOFFP — Flag indicating that grid boxes whose vertices have the value SPVAL are to be ignored during contouring.

SPVAL — Special value denoting which grid boxes are to be ignored during contouring.

See coding example for CONFILL.

CLIPBOX

PURPOSE **CLIPBOX** defines a clipping region.

SYNTAX CALL CLIPBOX (XPTS, YPTS, NPTS)

<u>ARGUMENTS</u>

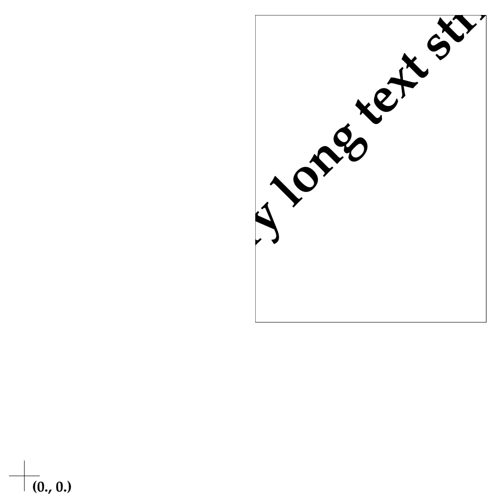
XPTS	Array containing x-coordinates of region path.
YPTS	Array containing y-coordinates of region path.
NPTS	Number of points in region path.

Example: CLIPBOX

Sample code:

```
dimension xa(4), ya(4)
data xa/1.5,3.,3.,1.5/
data ya/1.,1.,3.,3./
call clipbox(xa, ya, 4)
call keksym(.5, .5, .2, 23hA very long text string, 45., 23,0)
```

Code to outline clipping region not shown.



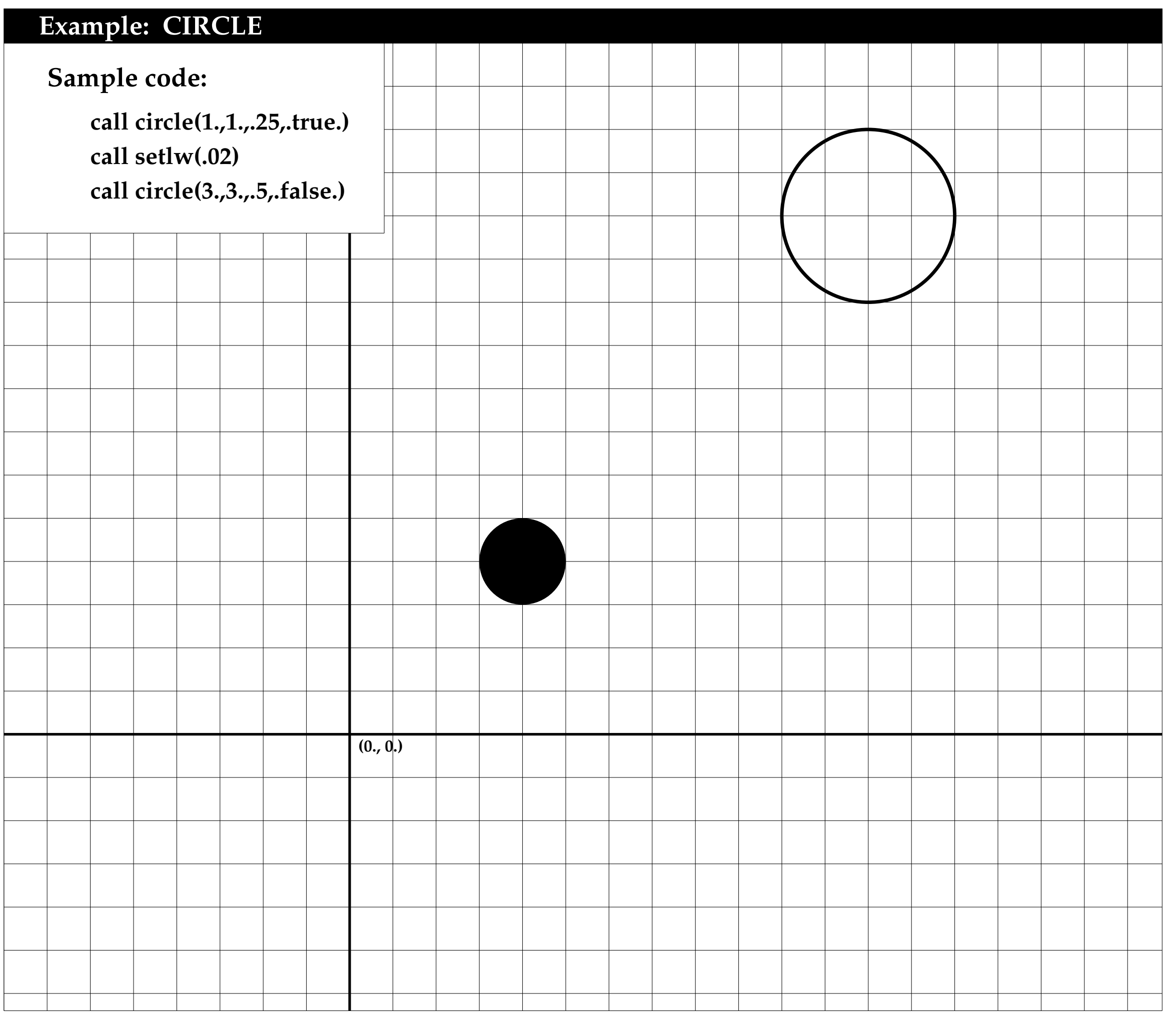
CIRCLE

PURPOSE **CIRCLE** draws a circle.

SYNTAX CALL CIRCLE (XC, YC, RAD, FILL)

ARGUMENTS

XC,YC	X,Y coordinates of the center of the circle.
RAD	Radius of the circle.
FILL	Logical value. If FILL=.TRUE., circle is filled with the current color or gray scale; otherwise it is outlined.



BORDER

PURPOSE **BORDER** draws a rectangular border with tick marks. The lower left corner of the border is located at the current plot origin.

SYNTAX CALL BORDER (XLEN, YLEN, ITIC, IBRD, MAJX, MINX, MAJY, MINY)

ARGUMENTS

XLEN Length of x-direction side of border.

YLEN Length of y-direction side of border.

ITIC Four digit number which determines which of the border sides will contain tick marks. Each of the digits is either 0 or 1, and if set to 1, that side will contain tick marks. The border sides are ordered as follows:

Left-vertical Bottom Right-vertical Top

For example, if ITIC = 1011, all sides except the bottom would contain tick marks. Additionally, if ITIC<0, the tick marks will be drawn on the outside of the border than on the inside (default).

IBRD IBRD is similar to ITIC except that it determines which sides of the border will be drawn. Hence, you can have tick marks with no border, and vice versa.

MAJX Number of major divisions in the x-direction. A longer tick mark is drawn for these divisions.

MINX Number of minor divisions in the x-direction, i.e. the number of divisions per major division. A shorter tick mark is drawn for these divisions.

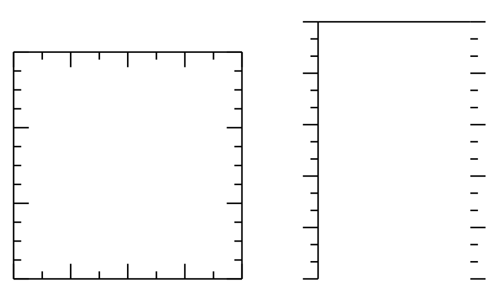
MAJY Same as MAJX in y-direction.

MINY Same as MINX in y-direction.

Example: BORDER

Sample code:

```
call setlw(.01)
call border(1.5, 1.5, 1111, 1111, 4, 2, 3, 4)
call plot(2., 0., -3)
call border(1., 1.7, -1010, 1001, 4, 1, 5, 3)
```



AXIS

PURPOSE **AXIS** draws an axis of a graph, with user-specified axis titles. Tick marks are drawn with a spacing of one inch.

SYNTAX CALL AXIS (XPP, YPP, IBCD, NC, AXLEN, THETA, RMIN, DEL)

<u>*ARGUMENTS*</u>

XPP,YPP X,Y coordinates of axis origin.

IBCD Axis title (Hollerith).

NC Number of characters in title. Its sign is used to specify on which side of the axis the title is to appear: positive for the counterclockwise side of the axis, negative for the clockwise side. Positive labeling is generally used for Y axes, negative for X axes.

AXLEN Length of axis.

THETA Angle of axis. Generally, 0 for X axes, 90 for Y axes.

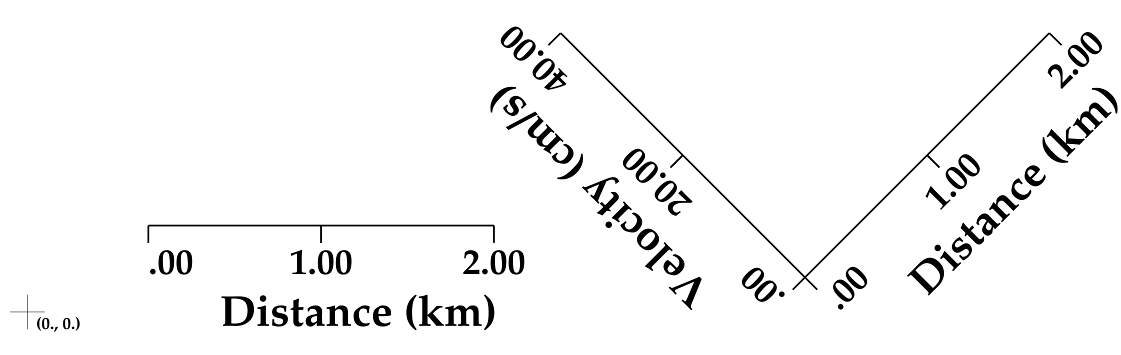
RMIN The starting value and annotation of the first tick mark.

DEL The data value increment between tick marks.

Example: AXIS

Sample code:

```
xlen=2.
ylen=2.
call setlw(.01)
call axis(.7, .5, 13HDistance (km), -13, xlen , 0., 0., 1.)
call axis(4.5, .2, 13HDistance (km), -13, xlen , 45., 0., 1.)
call axis(4.5, .2, 15HVelocity (cm/s), 15, ylen, 135., 0., 20.)
```



AROHED

PURPOSE **AROHED** draws an arrowhead at a specified location.

SYNTAX CALL AROHED (XPP, YPP, DIR, AROLNP, SPRANG, LOCXY)

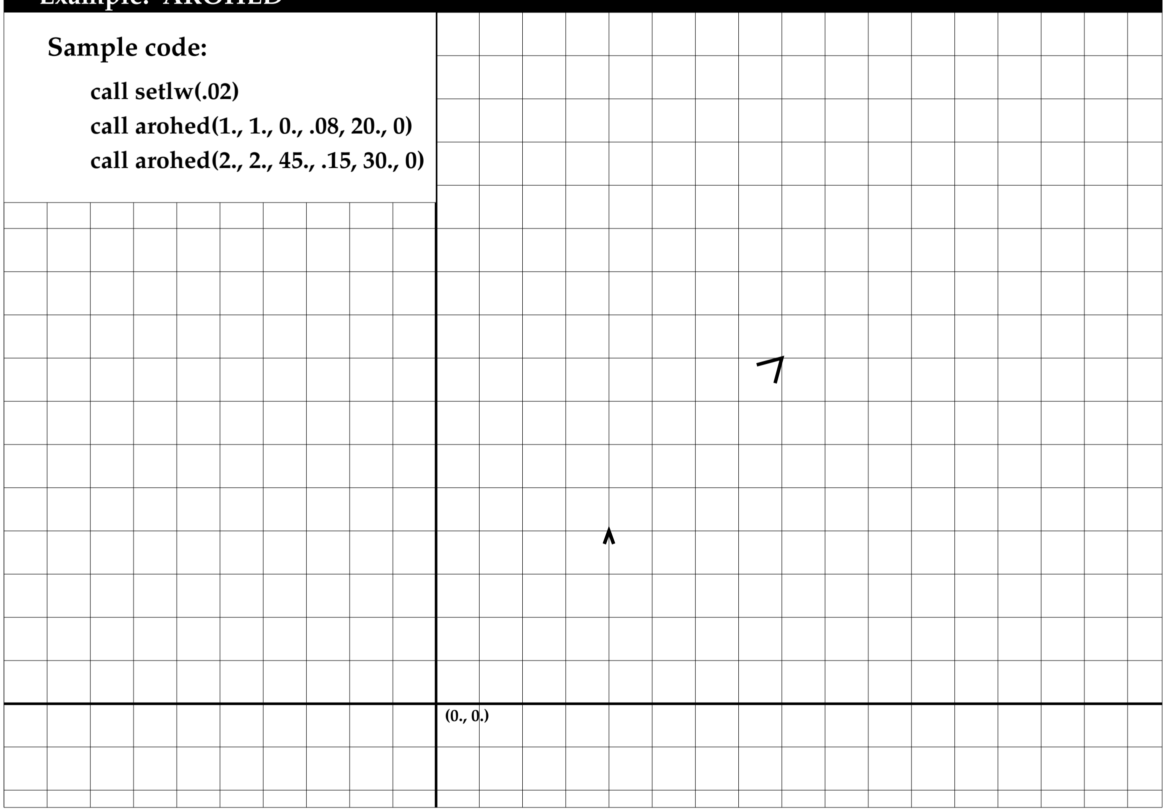
ARGUMENTS

XPP,YPP	X,Y coordinates of the tip of the arrowhead.
DIR	Direction of arrowhead, measured east from north.
AROLNP	Length of arrowhead sides.
SPRANG	Half the angular spread of arrowhead.
LOCXY	Location of arrowhead point LOCXY=1 XPP,YPP at arrowhead point (most commonly used) LOCXY=2 XPP, YPP at center of arrowhead. LOCXY=3 XPP,YPP at tail of arrowhead.

Example: AROHED

Sample code:

```
call setlw(.02)
call arohed(1., 1., 0., .08, 20., 0)
call arohed(2., 2., 45., .15, 30., 0)
```



SETFNT	Sets the current font.
SETGRY	Sets the current gray level value.
SETLW	Sets the current line width.
SIGMA	Draws the symbol Σ, with upper and lower limits.
SLDCRV	Connects data points with a solid curve.
SLDLIN	Draws a solid line between two points.
SQRSGN	Draws the radical (square root) sign.
SQUARE	Draws a square.
STROKE	Calls the PostScript operator *stroke*.
SUBBER	Draws a subscript.
SUBBERSP	Draws a subscript of "special" characters.
SUBSUP	Draws a subscript and superscript.
SUBSUPSP	Draws subscripts and superscripts of "special" characters.
SUPER	Draws a superscript.
SUPERSP	Draws a superscript of "special" characters.
SYMBOL	Plots a character string. Similar to KEKSYM, but justification and continuation are not supported.

The plotting subroutines in the PSPLOT plotting library are written in Fortran and are called by user-written application programs. In the following subroutine descriptions, integer variables are denoted by arguments beginning with the letters I-N, as per Fortran convention. All other variables are real (4 byte) floating point Fortran and are called by user-written application programs. All of the subroutines are written in single precision. Hence, if your program generates data in double precision, any arguments sent to the plotting library subroutines must first be converted to single precision.

DSHCRV	Connects data points with a dashed curve.
DSHLIN	Draws a dashed line between two points.
FACTOR	Enlarges or reduces a plot by a specified ratio.
FAROHED	Draws a "fancy" arrowhead.
FILLBOX	Fills a shape with a specified grayscale level.
FILRGN	Same as FILLBOX, except it restores the current gray level after filling.
FILRGNC	Fills a shape with the current gray level or color.
GREST	Calls the PostScript operator *grestore*.
GRKSYM	Plots a Greek symbol.
GSAV	Calls the PostScript operator *gsave*.
HILITEC	Prints text surrounded by a rectangular box (color).
HILITEG	Prints text surrounded by a rectangular box (grayscale).
INTEGRAL	Plots an integral with upper and lower limits.
KEKEXP	Plots a floating point number in exponential format.
KEKFLT	Plots a floating point number in floating point format.
KEKNUM	Plots a floating point number.
KEKSYM	Plots a character string.
KEKSYMC	Plots a character string, with the text string being chararacter rather than Hollerith.
KEKSYMO	Plots a character string with the characters outlined instead of filled.
LENSTR	Computes the number of characters in a character string.
NEWDEV	Specifies the name of the output PostScript file.
NUMBER	Plots a floating point number. Similar to KEKNUM, but justification and continuation are not supported.

All plotting commands use an *absolute* plot coordinate system. This means that all coordinates passed to plotting subroutines are expected to be in terms of distance from the current plot origin. This is contrast to a *relative* plot coordinate system, in which coordinates are assumed to be in terms of distance from the current pen position.

In all plotting subroutines, plot coordinates, character heights, distances, etc. are measured in *inches*.

Unless otherwise noted, any subroutine argument which specifies an angle is stated in *degrees relative to the X axis, with positive angles measured counterclockwise from the X axis*.

Most character variables are specified as Hollerith, rather than character strings. The reason for this is primarily historical, combined with the fact that the enormous amount of code already written and in use at my site precluded a comprehensive revision. This should not cause any problems, however.

The initial font of a plot session is Helvetica, with a size of 12 points. This can be changed permanently in subroutine PSINIT. Of course, the current font can always be changed with subroutine SETFNT. PSPLOT supports the standard 35 fonts found on most PostScript printers.

PSPLOT supports color, although all examples in this manual are shown in grayscale for the purposes of reproduction. Color is specified in the relevant subroutines as red, green, and blue (RGB) values.

Continuation allows you to append a character string or number to the end of a previously plotted string or number. The coordinates of the appended string are automatically calculated. Whether or not a subroutine supports continuation is stated in subroutine description in the next section.

Continuation is specified by setting the X and/or Y coordinate argument in calls to the subroutines listed above to 999, and may be applied to X and Y coordinates separately. A subroutine call with continuation must immediately follow the previous plotting call. Continuation is useful when plotting strings containing variable values, such that the resultant string length is not known beforehand.

For example,

```
ND=10
CALL KEKNUM (1., 2., .15, FLOAT(ND), 0., -1, 0)
CALL KEKSYM (999., 999., .15, 5H DAYS, 0. ,5, 0)
```

produces "10 DAYS".

1. INTRODUCTION

This manual is intended for users interested in generating two-dimensional technical drawings or graphics for technical journals in PostScript format. The manual describes a library of Fortran-callable subroutines which can be combined in a calling program to produce PostScript plot files. The purpose, syntax, and calling convention of each subroutine is presented , along with plotting examples.

PostScript is a very robust graphics and typesetting language with wide-ranging capabilities. Since the focus of the subroutines in this library is to produce technical drawings, many of the "artistic" features of PostScript have not been addressed or included here.

The name PostScript is a registered trademark of Adobe Systems Incorporated. All instances of the name PostScript in this manual are references to the PostScript language as defined by Adobe Systems Incorporated.

Send questions, comments, suggestions to *kevin@ocean.nova.edu*.