NAME

groff_char − groff glyph names

DESCRIPTION

This manual page lists the standard **groff** glyph names and the default input mapping, latin1. The glyphs in this document look different depending on which output device was chosen (with option **−T** for the **man**(1) program or the roff formatter). Glyphs not available for the device that is being used to print or view this manual page are marked with ‘(N/A)’; the device currently used is ‘ps’.

In the actual version, **groff** provides only 8-bit characters for direct input and named entities for further glyphs. On ASCII platforms, input character codes in the range 0 to 127 (decimal) represent the usual 7-bit ASCII characters, while codes between 127 and 255 are interpreted as the corresponding characters in the *latin1* (*ISO-8859-1*) code set by default. This mapping is contained in the file `latin1.tmac` and can be changed by loading a different input encoding. Note that some of the input characters are reserved by **groff**, either for internal use or for special input purposes. On EBCDIC platforms, only code page *cp1047* is supported (which contains the same characters as latin1; the input encoding file is called `cp1047.tmac`). Again, some input characters are reserved for internal and special purposes.

All roff systems provide the concept of named glyphs. In traditional roff systems, only names of length 2 were used, while groff also provides support for longer names. It is strongly suggested that only named glyphs are used for all character representations outside of the printable 7-bit ASCII range.

Some of the predefined groff escape sequences (with names of length 1) also produce single glyphs; these exist for historical reasons or are printable versions of syntactical characters. They include ‘`\\`’, ‘`\'`’, ‘`` \` ``’, ‘`\-`’, ‘`\.`’, and ‘`\e`’; see **groff**(7).

In groff, all of these different types of characters and glyphs can be tested positively with the ‘`.if c`’ conditional.

REFERENCE

In this section, the glyphs in groff are specified in tabular form. The meaning of the columns is as follows.

Output shows how the glyph is printed for the current device; although this can have quite a different shape on other devices, it always represents the same glyph.

Input specifies how the glyph is input either directly by a key on the keyboard, or by a groff escape sequence.

Code applies to glyphs which can be input with a single character, and gives the ISO latin1 decimal code of that input character. Note that this code is equivalent to the lowest 256 Unicode characters, including 7-bit ASCII in the range 0 to 127.

PostScript
gives the usual PostScript name of the glyph.

Unicode
is the glyph name used in composite glyph names. The names in the Unicode column look like **u0021** or **u0041_0300**. In groff, the corresponding Unicode characters can be constructed by adding a backslash and a pair of square brackets, for example **\[u0021]** or **\[u0041_0300]**.

7-bit Character Codes 32–126

These are the basic glyphs having 7-bit ASCII code values assigned. They are identical to the printable characters of the character standards ISO-8859-1 (latin1) and Unicode (range *Basic Latin*). The glyph names used in composite glyph names are ‘u0020’ up to ‘u007E’.

Note that input characters in the range 0−31 and character 127 are *not* printable characters. Most of them are invalid input characters for **groff** anyway, and the valid ones have special meaning. For EBCDIC, the printable characters are in the range 66−255.

48−57 Decimal digits 0 to 9 (print as themselves).

65−90 Upper case letters A−Z (print as themselves).

97−122 Lower case letters a–z (print as themselves).

Most of the remaining characters not in the just described ranges print as themselves; the only exceptions are the following characters:

**`** the ISO latin1 ‘Grave Accent’ (code 96) prints as ‘, a left single quotation mark (Unicode u2018). The same output glyph can be requested explicitly with ‘`\(oq`’. The original character can be obtained with ‘``\` ``’ (Unicode u0060).

' the ISO latin1 ‘Apostrophe’ (code 39) prints as ’, a right single quotation mark (Unicode u2019). The same output glyph is commonly used in typography to represent a punctation apostrophe, for example in contractions. It can be requested explicitly with ‘`\(cq`’. The original character can be obtained with ‘`\(aq`’ (Unicode u0027).

- the ISO latin1 ‘Hyphen, Minus Sign’ (code 45) prints as a hyphen (Unicode u2010). The same output glyph can be requested explicitly with ‘`\(hy`’. A minus sign can be obtained with ‘`\-`’ (Unicode u2212).

˜ the ISO latin1 ‘Tilde’ (code 126) is reduced in size to be usable as a diacritic (Unicode u02DC). A larger glyph can be obtained with ‘`\(ti`’ (Unicode u007E).

ˆ the ISO latin1 ‘Circumflex Accent’ (code 94) is reduced in size to be usable as a diacritic (Unicode u02C6); a larger glyph can be obtained with ‘`\(ha`’ (Unicode u005E).

Output	Input	Code	PostScript	Unicode	Notes
!	!	33	exclam	u0021	exclamation mark (bang)
"	"	34	quotedbl	u0022	double quote
#	#	35	numbersign	u0023	number sign
$	$	36	dollar	u0024	currency dollar sign
%	%	37	percent	u0025	percent
&	&	38	ampersand	u0026	ampersand
’	'	39	quoteright	u2019	right quote
'	\(aq		quotesingle	u0027	apostrophe quote
(	(	40	parenleft	u0028	parentheses left
)	)	41	parenright	u0029	parentheses right
*	*	42	asterisk	u002A	asterisk
+	+	43	plus	u002B	plus
,	,	44	comma	u002C	comma
-	-	45	hyphen	u2010	hyphen
−	\-		minus	u2212	minus sign
.	.	46	period	u002E	period, dot
/	/	47	slash	u002F	slash
:	:	58	colon	u003A	colon
;	;	59	semicolon	u003B	semicolon
<	<	60	less	u003C	less than
=	=	61	equal	u003D	equal
>	>	62	greater	u003E	greater than
?	?	63	question	u003F	question mark
@	@	64	at	u0040	at
[	[	91	bracketleft	u005B	square bracket left
\	\	92	backslash	u005C	backslash
]	]	93	bracketright	u005D	square bracket right
ˆ	^	94	circumflex	u02C6	modifier circumflex
^	\(ha		asciicircum	u005E	circumflex accent
_	_	95	underscore	u005F	underscore
‘	`	96	quoteleft	u2018	left quote
`	\(ga		grave	u0060	grave accent
{	{	123	braceleft	u007B	curly brace left

|	|	124	bar	u007C	bar
}	}	125	braceright	u007D	curly brace right
	~	126	tilde	u02DC	small tilde
~	\(ti		asciitilde	u007E	tilde

8-bit Character Codes 160 to 255

They are interpreted as printable characters according to the *latin1* (*ISO-8859-1*) code set, being identical to the Unicode range *Latin-1 Supplement*.

Input characters in range 128−159 (on non-EBCDIC hosts) are not printable characters.

160 the ISO latin1 *no-break space* is mapped to '\~', the stretchable space character.

173 the soft hyphen control character. **groff** never uses this character for output (thus it is omitted in the table below); the input character 173 is mapped onto '\%'.

The remaining ranges (161−172, 174−255) are printable characters that print as themselves. Although they can be specified directly with the keyboard on systems with a latin1 code page, it is better to use their glyph names; see the next section.

Output	Input	Code	PostScript	Unicode	Notes
¡	¡	161	exclamdown	u00A1	inverted exclamation mark
¢	¢	162	cent	u00A2	currency unit
£	£	163	sterling	u00A3	pound sterling
¤	¤	164	currency	u00A4	generic currency symbol
¥	¥	165	yen	u00A5	Japanese currency symbol
¦	¦	166	brokenbar	u00A6	broken bar
§	§	167	section	u00A7	section sign
¨	¨	168	dieresis	u00A8	dieresis (umlaut)
©	©	169	copyright	u00A9	copyright symbol
ª	ª	170	ordfeminine	u00AA	feminine ordinal (Spanish)
«	«	171	guillemotleft	u00AB	left guillemet [sic]
¬	¬	172	logicalnot	u00AC	logical not
®	®	174	registered	u00AE	registered mark symbol
¯	¯	175	macron	u00AF	overbar accent
°	°	176	degree	u00B0	degree sign
±	±	177	plusminus	u00B1	plus-minus sign
²	²	178	twosuperior	u00B2	superscript 2
³	³	179	threesuperior	u00B3	superscript 3
´	´	180	acute	u00B4	acute accent
µ	µ	181	mu	u00B5	micro sign
¶	¶	182	paragraph	u00B6	end of paragraphs marker
·	·	183	periodcentered	u00B7	centered period
¸	¸	184	cedilla	u00B8	cedilla accent
¹	¹	185	onesuperior	u00B9	superscript 1
º	º	186	ordmasculine	u00BA	masculine ordinal (Spanish)
»	»	187	guillemotright	u00BB	right guillemet [sic]
¼	¼	188	onequarter	u00BC	1/4 symbol
½	½	189	onehalf	u00BD	1/2 symbol
¾	¾	190	threequarters	u00BE	3/4 symbol
¿	¿	191	questiondown	u00BF	inverted question mark
À	À	192	Agrave	u0041_0300	A grave
Á	Á	193	Aacute	u0041_0301	A acute
Â	Â	194	Acircumflex	u0041_0302	A circumflex
Ã	Ã	195	Atilde	u0041_0303	A tilde
Ä	Ä	196	Adieresis	u0041_0308	A dieresis (umlaut)
Å	Å	197	Aring	u0041_030A	A ring

Æ	Æ	198	AE	u00C6	A+E combined
Ç	Ç	199	Ccedilla	u0043_0327	C cedilla
È	È	200	Egrave	u0045_0300	E grave
É	É	201	Eacute	u0045_0301	E acute
Ê	Ê	202	Ecircumflex	u0045_0302	E circumflex
Ë	Ë	203	Edieresis	u0045_0308	E dieresis (umlaut)
Ì	Ì	204	Igrave	u0049_0300	I grave
Í	Í	205	Iacute	u0049_0301	I acute
Î	Î	206	Icircumflex	u0049_0302	I circumflex
Ï	Ï	207	Idieresis	u0049_0308	I dieresis
Ð	Ð	208	Eth	u00D0	E th
Ñ	Ñ	209	Ntilde	u004E_0303	N tilde
Ò	Ò	210	Ograve	u004F_0300	O grave
Ó	Ó	211	Oacute	u004F_0301	O acute
Ô	Ô	212	Ocircumflex	u004F_0302	O circumflex
Õ	Õ	213	Otilde	u004F_0303	O tilde
Ö	Ö	214	Odieresis	u004F_0308	O dieresis (umlaut)
×	×	215	multiply	u00D7	multiply
Ø	Ø	216	Oslash	u00D8	O slash
Ù	Ù	217	Ugrave	u0055_0300	U grave
Ú	Ú	218	Uacute	u0055_0301	U acute
Û	Û	219	Ucircumflex	u0055_0302	U circumflex
Ü	Ü	220	Udieresis	u0055_0308	U dieresis (umlaut)
Ý	Ý	221	Yacute	u0059_0301	Y acute
Þ	Þ	222	Thorn	u00DE	Thorn
ß	ß	223	germandbls	u00DF	German double s (sharp s)
à	à	224	agrave	u0061_0300	a grave
á	á	225	aacute	u0061_0301	a acute
â	â	226	acircumflex	u0061_0302	a circumflex
ã	ã	227	atilde	u0061_0303	a tilde
ä	ä	228	adieresis	u0061_0308	a dieresis (umlaut)
å	å	229	aring	u0061_030A	a ring
æ	æ	230	ae	u00E6	a+e combined
ç	ç	231	ccedilla	u0063_0327	c cedilla
è	è	232	egrave	u0065_0300	e grave
é	é	233	eacute	u0065_0301	e acute
ê	ê	234	ecircumflex	u0065_0302	e circumflex
ë	ë	235	edieresis	u0065_0308	e dieresis (umlaut)
ì	ì	236	igrave	u0069_0300	i grave
í	í	237	iacute	u0069_0301	i acute
î	î	238	icircumflex	u0069_0302	i circumflex
ï	ï	239	idieresis	u0069_0308	i dieresis (umlaut)
ð	ð	240	eth	u00F0	e th
ñ	ñ	241	ntilde	u006E_0303	n tilde
ò	ò	242	ograve	u006F_0300	o grave
ó	ó	243	oacute	u006F_0301	o acute
ô	ô	244	ocircumflex	u006F_0302	o circumflex
õ	õ	245	otilde	u006F_0303	o tilde
ö	ö	246	odieresis	u006F_0308	o dieresis (umlaut)
÷	÷	247	divide	u00F7	divide
ø	ø	248	oslash	u00F8	o slash
ù	ù	249	ugrave	u0075_0300	u grave
ú	ú	250	uacute	u0075_0301	u acute

û	û	251	ucircumflex	u0075_0302	u circumflex
ü	ü	252	udieresis	u0075_0308	u dieresis (umlaut)
ý	ý	253	yacute	u0079_0301	y acute
þ	þ	254	thorn	u00FE	thorn
ÿ	ÿ	255	ydieresis	u0079_0308	y dieresis (umlaut)

Named Glyphs

Glyph names can be embedded into the document text by using escape sequences. **groff**(7) describes how these escape sequences look. Glyph names can consist of quite arbitrary characters from the ASCII or latin1 code set, not only alphanumeric characters. Here some examples:

`\(`*ch* A glyph having the 2-character name *ch*.

`\[`*char_name*`]`
A glyph having the name *char_name* (having length 1, 2, 3, ...). Note that '*c*' is not the same as '`\[`*c*`]`' (*c* a single character): The latter is internally mapped to glyph name '\c'. By default, groff defines a single glyph name starting with a backslash, namely '\\-', which can be either accessed as '`\-`' or '`\[-]`'.

`\[`*base_glyph composite_1 composite_2 ...*`]`
A composite glyph; see below for a more detailed description.

In groff, each 8-bit input character can also referred to by the construct '`\[char`*n*`]`' where *n* is the decimal code of the character, a number between 0 and 255 without leading zeros (those entities are *not* glyph names). They are normally mapped onto glyphs using the `.trin` request.

Another special convention is the handling of glyphs with names directly derived from a Unicode code point; this is shown in the 'Unicode' column of the table below. In general, all glyphs not having a name as listed in this manual page can be accessed with the '`\[u`*XXXX*`]`' construct. Refer to section "Using Symbols" in *Groff: The GNU Implementation of troff*, the *groff* Texinfo manual, which describes how *groff* glyph names are constructed.

Moreover, new glyph names can be created by the `.char` request; see **groff**(7).

In the following, a plus sign '+' in the 'Notes' column indicates that this particular glyph name appears in the PS version of the original troff documentation, CSTR 54.

Entries marked with '***' denote glyphs for mathematical purposes (mainly used for DVI output). Normally, such glyphs have metrics which make them unusable in normal text.

Output	Input	PostScript	Unicode	Notes
Ð	\[-D]	Eth	u00D0	uppercase eth
ð	\[Sd]	eth	u00F0	lowercase eth
Þ	\[TP]	Thorn	u00DE	uppercase thorn
þ	\[Tp]	thorn	u00FE	lowercase thorn
ß	\[ss]	germandbls	u00DF	German double s (sharp s)

Ligatures and Other Latin Glyphs

Output	Input	PostScript	Unicode	Notes
ff	\[ff]	ff	u0066_0066	ff ligature +
fi	\[fi]	fi	u0066_0069	fi ligature +
fl	\[fl]	fl	u0066_006C	fl ligature +
ffi	\[Fi]	ffi	u0066_0066_0069	ffi ligature +
ffl	\[Fl]	ffl	u0066_0066_006C	ffl ligature +
Ł	\[/L]	Lslash	u0141	L slash (Polish)
ł	\[/l]	lslash	u0142	l slash (Polish)
Ø	\[/O]	Oslash	u00D8	O slash (Scandinavian)
ø	\[/o]	oslash	u00F8	o slash (Scandinavian)
Æ	\[AE]	AE	u00C6	A+E combined

æ	\[ae]	ae	u00E6	a+e combined
Œ	\[OE]	OE	u0152	O+E combined
œ	\[oe]	oe	u0153	o+e combined
IJ	\[IJ]	IJ	u0132	I+J combined (Dutch)
ij	\[ij]	ij	u0133	i+j combined(Dutch)
ı	\[.i]	dotlessi	u0131	i without a dot (Turkish)
	\[.j]	dotlessj	u0237	j without a dot

Accented Characters

Output	Input	PostScript	Unicode	Notes
Á	\[’A]	Aacute	u0041_0301	A acute
Ć	\[’C]	Cacute	u0043_0301	C acute
É	\[’E]	Eacute	u0045_0301	E acute
Í	\[’I]	Iacute	u0049_0301	I acute
Ó	\[’O]	Oacute	u004F_0301	O acute
Ú	\[’U]	Uacute	u0055_0301	U acute
Ý	\[’Y]	Yacute	u0059_0301	Y acute
á	\[’a]	aacute	u0061_0301	a acute
ć	\[’c]	cacute	u0063_0301	c acute
é	\[’e]	eacute	u0065_0301	e acute
í	\[’i]	iacute	u0069_0301	i acute
ó	\[’o]	oacute	u006F_0301	o acute
ú	\[’u]	uacute	u0075_0301	u acute
ý	\[’y]	yacute	u0079_0301	y acute
Ä	\[:A]	Adieresis	u0041_0308	A dieresis (umlaut)
Ë	\[:E]	Edieresis	u0045_0308	E dieresis (umlaut)
Ï	\[:I]	Idieresis	u0049_0308	I dieresis (umlaut)
Ö	\[:O]	Odieresis	u004F_0308	O dieresis (umlaut)
Ü	\[:U]	Udieresis	u0055_0308	U dieresis (umlaut)
Ÿ	\[:Y]	Ydieresis	u0059_0308	Y dieresis (umlaut)
ä	\[:a]	adieresis	u0061_0308	a dieresis (umlaut)
ë	\[:e]	edieresis	u0065_0308	e dieresis (umlaut)
ï	\[:i]	idieresis	u0069_0308	i dieresis (umlaut)
ö	\[:o]	odieresis	u006F_0308	o dieresis (umlaut)
ü	\[:u]	udieresis	u0075_0308	u dieresis (umlaut)
ÿ	\[:y]	ydieresis	u0079_0308	y dieresis (umlaut)
Â	\[ˆA]	Acircumflex	u0041_0302	A circumflex
Ê	\[ˆE]	Ecircumflex	u0045_0302	E circumflex
Î	\[ˆI]	Icircumflex	u0049_0302	I circumflex
Ô	\[ˆO]	Ocircumflex	u004F_0302	O circumflex
Û	\[ˆU]	Ucircumflex	u0055_0302	U circumflex
â	\[ˆa]	acircumflex	u0061_0302	a circumflex
ê	\[ˆe]	ecircumflex	u0065_0302	e circumflex
î	\[ˆi]	icircumflex	u0069_0302	i circumflex
ô	\[ˆo]	ocircumflex	u006F_0302	o circumflex
û	\[ˆu]	ucircumflex	u0075_0302	u circumflex
À	\[‘A]	Agrave	u0041_0300	A grave
È	\[‘E]	Egrave	u0045_0300	E grave
Ì	\[‘I]	Igrave	u0049_0300	I grave
Ò	\[‘O]	Ograve	u004F_0300	O grave
Ù	\[‘U]	Ugrave	u0055_0300	U grave
à	\[‘a]	agrave	u0061_0300	a grave
è	\[‘e]	egrave	u0065_0300	e grave
ì	\[‘i]	igrave	u0069_0300	i grave

ò	\[‘o]	ograve	u006F_0300	o grave
ù	\[‘u]	ugrave	u0075_0300	u grave
Ã	\[˜A]	Atilde	u0041_0303	A tilde
Ñ	\[˜N]	Ntilde	u004E_0303	N tilde
Õ	\[˜O]	Otilde	u004F_0303	O tilde
ã	\[˜a]	atilde	u0061_0303	a tilde
ñ	\[˜n]	ntilde	u006E_0303	n tilde
õ	\[˜o]	otilde	u006F_0303	o tilde
Š	\[vS]	Scaron	u0053_030C	S caron
š	\[vs]	scaron	u0073_030C	s caron
Ž	\[vZ]	Zcaron	u005A_030C	Z caron
ž	\[vz]	zcaron	u007A_030C	z caron
Ç	\[,C]	Ccedilla	u0043_0327	C cedilla
ç	\[,c]	ccedilla	u0063_0327	c cedilla
Å	\[oA]	Aring	u0041_030A	A ring
å	\[oa]	aring	u0061_030A	a ring

Accents

The **composite** request is used to map most of the accents to non-spacing glyph names; the values given in parentheses are the original (spacing) ones.

Output	Input	PostScript	Unicode	Notes
˝	\[a"]	hungarumlaut	u030B (u02DD)	Hungarian umlaut
¯	\[a-]	macron	u0304 (u00AF)	overbar accent
˙	\[a.]	dotaccent	u0307 (u02D9)	dot accent
ˆ	\[aˆ]	circumflex	u0302 (u005E)	circumflex accent
´	\[aa]	acute	u0301 (u00B4)	acute accent +
`	\[ga]	grave	u0300 (u0060)	grave accent +
˘	\[ab]	breve	u0306 (u02D8)	breve accent
¸	\[ac]	cedilla	u0327 (u00B8)	cedilla accent
¨	\[ad]	dieresis	u0308 (u00A8)	umlaut accent
ˇ	\[ah]	caron	u030C (u02C7)	caron accent
˚	\[ao]	ring	u030A (u02DA)	small circle, ring accent
˜	\[a˜]	tilde	u0303 (u007E)	tilde accent
˛	\[ho]	ogonek	u0328 (u02DB)	hook accent
^	\[ha]	asciicircum	u005E	high circumflex, ASCII character, in mathematics the power sign
~	\[ti]	asciitilde	u007E	tilde in vertical middle, ASCII, in Unix-like the home directory

Quotes

Output	Input	PostScript	Unicode	Notes
„	\[Bq]	quotedblbase	u201E	low double comma quote
‚	\[bq]	quotesinglbase	u201A	low single comma quote
“	\[lq]	quotedblleft	u201C	left double quote
”	\[rq]	quotedblright	u201D	right double quote
‘	\[oq]	quoteleft	u2018	single open (left) quote
’	\[cq]	quoteright	u2019	single closing (right) quote
'	\[aq]	quotesingle	u0027	apostrophe quote (ASCII 39)
"	\[dq]	quotedbl	u0022	double quote (ASCII 34)
«	\[Fo]	guillemotleft	u00AB	left guillemet [sic]
»	\[Fc]	guillemotright	u00BB	right guillemet [sic]
‹	\[fo]	guilsinglleft	u2039	single left-pointing angle quotation mark
›	\[fc]	guilsinglright	u203A	single right-pointing angle quotation mark

Punctuation

Output	Input	PostScript	Unicode	Notes
¡	\[r!]	exclamdown	u00A1	inverted exclamation mark
¿	\[r?]	questiondown	u00BF	inverted question mark
—	\[em]	emdash	u2014	em-dash symbol +
–	\[en]	endash	u2013	en-dash symbol
‐	\[hy]	hyphen	u2010	hyphen symbol +

Brackets

The extensible bracket pieces are font-invariant glyphs. In classical troff only one glyph was available to vertically extend brackets, braces, and parentheses: ‘bv’. We map it rather arbitrarily to u23AA.

Note that not all devices contain extensible bracket pieces which can be piled up with ‘`\b`’ due to the restrictions of the escape’s piling algorithm. A general solution to build brackets out of pieces is the following macro:

```
.\" Make a pile centered vertically 0.5em
.\" above the baseline.
.\" The first argument is placed at the top.
.\" The pile is returned in string 'pile'
.eo
.de pile-make
.  nr pile-wd 0
.  nr pile-ht 0
.  ds pile-args
.
.  nr pile-# \n[.$]
.  while \n[pile-#] \{\
.    nr pile-wd (\n[pile-wd] >? \w'\$[\n[pile-#]]')
.    nr pile-ht +(\n[rst] - \n[rsb])
.    as pile-args \v'\n[rsb]u'\"
.    as pile-args \Z'\$[\n[pile-#]]'\"
.    as pile-args \v'-\n[rst]u'\"
.    nr pile-# -1
.  \}
.
.  ds pile \v'(-0.5m + (\n[pile-ht]u / 2u))'\"
.  as pile \*[pile-args]\"
.  as pile \v'((\n[pile-ht]u / 2u) + 0.5m)'\"
.  as pile \h'\n[pile-wd]u'\"
..
.ec
```

Another complication is the fact that some glyphs which represent bracket pieces in original troff can be used for other mathematical symbols also, for example ‘lf’ and ‘rf’ which provide the ‘floor’ operator. Other devices (most notably for DVI output) don’t unify such glyphs. For this reason, the four glyphs ‘lf’, ‘rf’, ‘lc’, and ‘rc’ are not unified with similarly looking bracket pieces. In **groff**, only glyphs with long names are guaranteed to pile up correctly for all devices (provided those glyphs exist).

Output	Input	PostScript	Unicode	Notes
[	\[lB]	bracketleft	u005B	left square bracket
]	\[rB]	bracketright	u005D	right square bracket
{	\[lC]	braceleft	u007B	left curly brace
}	\[rC]	braceright	u007D	right curly brace
⟨	\[la]	angleleft	u27E8	left angle bracket
⟩	\[ra]	angleright	u27E9	right angle bracket

⎪	\[bv]	braceex	u23AA	curly brace vertical extension *** +
⎪	\[braceex]	braceex	u23AA	curly brace vertical extension
⎡	\[bracketlefttp]	bracketlefttp	u23A1	left square bracket top
⎣	\[bracketleftbt]	bracketleftbt	u23A3	left square bracket bottom
⎢	\[bracketleftex]	bracketleftex	u23A2	left square bracket extension
⎤	\[bracketrighttp]	bracketrighttp	u23A4	right square bracket top
⎦	\[bracketrightbt]	bracketrightbt	u23A6	right square bracket bottom
⎥	\[bracketrightex]	bracketrightex	u23A5	right square bracket extension
⎧	\[lt]	bracelefttp	u23A7	left curly brace top +
⎧	\[bracelefttp]	bracelefttp	u23A7	left curly brace top
⎨	\[lk]	braceleftmid	u23A8	left curly brace middle +
⎨	\[braceleftmid]	braceleftmid	u23A8	left curly brace middle
⎩	\[lb]	braceleftbt	u23A9	left curly brace bottom +
⎩	\[braceleftbt]	braceleftbt	u23A9	left curly brace bottom
⎪	\[braceleftex]	braceleftex	u23AA	left curly brace extension
⎫	\[rt]	bracerighttp	u23AB	right curly brace top +
⎫	\[bracerighttp]	bracerighttp	u23AB	right curly brace top
⎬	\[rk]	bracerightmid	u23AC	right curly brace middle +
⎬	\[bracerightmid]	bracerightmid	u23AC	right curly brace middle
⎭	\[rb]	bracerightbt	u23AD	right curly brace bottom +
⎭	\[bracerightbt]	bracerightbt	u23AD	right curly brace bottom
⎪	\[bracerightex]	bracerightex	u23AA	right curly brace extension
⎛	\[parenlefttp]	parenlefttp	u239B	left parenthesis top
⎝	\[parenleftbt]	parenleftbt	u239D	left parenthesis bottom
⎜	\[parenleftex]	parenleftex	u239C	left parenthesis extension
⎞	\[parenrighttp]	parenrighttp	u239E	right parenthesis top
⎠	\[parenrightbt]	parenrightbt	u23A0	right parenthesis bottoom
⎟	\[parenrightex]	parenrightex	u239F	right parenthesis extension

Arrows

Output	Input	PostScript	Unicode	Notes
←	\[<-]	arrowleft	u2190	horizontal arrow left +
→	\[->]	arrowright	u2192	horizontal arrow right +
↔	\[<>]	arrowboth	u2194	horizontal arrow in both directions
↓	\[da]	arrowdown	u2193	vertical arrow down +
↑	\[ua]	arrowup	u2191	vertical arrow up +
↕	\[va]	arrowupdn	u2195	vertical arrow in both directions
⇐	\[lA]	arrowdblleft	u21D0	horizontal double arrow left
⇒	\[rA]	arrowdblright	u21D2	horizontal double arrow right
⇔	\[hA]	arrowdblboth	u21D4	horizontal double arrow in both directions
⇓	\[dA]	arrowdbldown	u21D3	vertical double arrow down
⇑	\[uA]	arrowdblup	u21D1	vertical double arrow up
	\[vA]	uni21D5	u21D5	vertical double arrow in both directions
⎯	\[an]	arrowhorizex	u23AF	horizontal arrow extension

Lines

The font-invariant glyphs ‘br’, ‘ul’, and ‘rn’ form corners; they can be used to build boxes. Note that both the PostScript and the Unicode-derived names of these three glyphs are just rough approximations.

‘rn’ also serves in classical troff as the horizontal extension of the square root sign.

‘ru’ is a font-invariant glyph, namely a rule of length 0.5m.

Output	Input	PostScript	Unicode	Notes

|	\[ba]	bar	u007C	
|	\[br]	SF110000	u2502	box rule +
_	\[ul]	underscore	u005F	+
‾	\[rn]	overline	u203E	+
_	\[ru]	---	---	baseline rule +
¦	\[bb]	brokenbar	u00A6	
/	\[sl]	slash	u002F	+
\	\[rs]	backslash	u005C	reverse solidus

Use ‘`\[radicalex]`’, not ‘`\[overline]`’, for continuation of square root.

Text markers

Output	Input	PostScript	Unicode	Notes
○	\[ci]	circle	u25CB	+
•	\[bu]	bullet	u2022	+
‡	\[dd]	daggerdbl	u2021	double dagger sign +
†	\[dg]	dagger	u2020	dagger +
◊	\[lz]	lozenge	u25CA	lozenge, diamond, pound key
□	\[sq]	uni25A1	u25A1	white square +
¶	\[ps]	paragraph	u00B6	end of paragraph marker
§	\[sc]	section	u00A7	section sign +
☜	\[lh]	uni261C	u261C	hand pointing left +
☞	\[rh]	a14	u261E	hand pointing right +
@	\[at]	at	u0040	at
#	\[sh]	numbersign	u0023	number sign
↵	\[CR]	carriagereturn	u21B5	carriage return
✓	\[OK]	a19	u2713	check mark, tick

Legal Symbols

Output	Input	PostScript	Unicode	Notes
©	\[co]	copyright	u00A9	+
®	\[rg]	registered	u00AE	+
™	\[tm]	trademark	u2122	
	\[bs]	---	---	AT&T Bell Labs logo +

The Bell Labs logo is not supported in groff.

Currency symbols

Output	Input	PostScript	Unicode	Notes
$	\[Do]	dollar	u0024	dollar
¢	\[ct]	cent	u00A2	cent +
€	\[eu]	---	u20AC	official Euro symbol
€	\[Eu]	Euro	u20AC	font-specific Euro glyph variant
¥	\[Ye]	yen	u00A5	Japanese Yen
£	\[Po]	sterling	u00A3	pound sterling (British)
¤	\[Cs]	currency	u00A4	Scandinavian currency sign
ƒ	\[Fn]	florin	u0192	Dutch currency sign

Units

Output	Input	PostScript	Unicode	Notes
°	\[de]	degree	u00B0	degree +
‰	\[%0]	perthousand	u2030	per thousand, per mille sign
′	\[fm]	minute	u2032	arc minute sign +
″	\[sd]	second	u2033	acr second sign
µ	\[mc]	mu	u00B5	mu, micro sign

ª	\[Of]	ordfeminine	u00AA	feminine ordinal (Spanish)
º	\[Om]	ordmasculine	u00BA	masculine ordinal (Spanish)

Logical Symbols

Output	Input	PostScript	Unicode	Notes
∧	\[AN]	logicaland	u2227	logical and
∨	\[OR]	logicalor	u2228	logical or
¬	\[no]	logicalnot	u00AC	logical not + ***
¬	\[tno]	logicalnot	u00AC	text variant of ‘no’
∃	\[te]	existential	u2203	there exists
∀	\[fa]	universal	u2200	for all
∍	\[st]	suchthat	u220B	sucht that
∴	\[3d]	therefore	u2234	therefore
∴	\[tf]	therefore	u2234	therefore
|	\[or]	bar	u007C	bitwise OR operator (as used in C) +

Mathematical Symbols

Output	Input	PostScript	Unicode	Notes
½	\[12]	onehalf	u00BD	1/2 symbol +
¼	\[14]	onequarter	u00BC	1/4 symbol +
¾	\[34]	threequarters	u00BE	3/4 symbol +
⅛	\[18]	oneeighth	u215B	1/8 symbol
⅜	\[38]	threeeighths	u215C	3/8 symbol
⅝	\[58]	fiveeighths	u215D	5/8 symbol
⅞	\[78]	seveneighths	u215E	7/8 symbol
¹	\[S1]	onesuperior	u00B9	superscript 1
²	\[S2]	twosuperior	u00B2	superscript 2
³	\[S3]	threesuperior	u00B3	superscript 3
+	\[pl]	plus	u002B	plus in special font +
−	\[mi]	minus	u2212	minus in special font +
	\[-+]	uni2213	u2213	minus-plus
±	\[+-]	plusminus	u00B1	plus-minus + ***
±	\[t+-]	plusminus	u00B1	text variant of \[+-]
·	\[pc]	periodcentered	u00B7	period centered
⋅	\[md]	dotmath	u22C5	multiplication dot
×	\[mu]	multiply	u00D7	multiply sign + ***
×	\[tmu]	multiply	u00D7	text variant of \[mu]
⊗	\[c*]	circlemultiply	u2297	multiply sign in circle
⊕	\[c+]	circleplus	u2295	plus sign in circle
÷	\[di]	divide	u00F7	division sign + ***
÷	\[tdi]	divide	u00F7	text variant of \[di]
⁄	\[f/]	fraction	u2044	bar for fractions
∗	\[**]	asteriskmath	u2217	mathematical asterisk +
≤	\[<=]	lessequal	u2264	less or equal +
≥	\[>=]	greaterequal	u2265	greater or equal +
≪	\[<<]	uni226A	u226A	much less
≫	\[>>]	uni226B	u226B	much greater
=	\[eq]	equal	u003D	equals in special font +
≠	\[!=]	notequal	u003D_0338	not equal +
≡	\[==]	equivalence	u2261	equivalent +
≢	\[ne]	uni2262	u2261_0338	not equivalent
≅	\[=~]	congruent	u2245	congruent, approx. equal

≃	\[|=]	uni2243	u2243	asymptot. equal to +
~	\[ap]	similar	u223C	similar +
≈	\[~~]	approxequal	u2248	almost equal to
≈	\[~=]	approxequal	u2248	almost equal to
∝	\[pt]	proportional	u221D	proportional +
∅	\[es]	emptyset	u2205	empty set +
∈	\[mo]	element	u2208	element of a set +
∉	\[nm]	notelement	u2208_0338	not element of set
⊂	\[sb]	propersubset	u2282	proper subset +
⊄	\[nb]	notsubset	u2282_0338	not supset
⊃	\[sp]	propersuperset	u2283	proper superset +
⊅	\[nc]	uni2285	u2283_0338	not superset
⊆	\[ib]	reflexsubset	u2286	subset or equal +
⊇	\[ip]	reflexsuperset	u2287	superset or equal +
∩	\[ca]	intersection	u2229	intersection, cap +
∪	\[cu]	union	u222A	union, cup +
∠	\[/_]	angle	u2220	angle
⊥	\[pp]	perpendicular	u22A5	perpendicular
∫	\[is]	integral	u222B	integral +
∫	\[integral]	integral	u222B	integral ***
Σ	\[sum]	summation	u2211	summation ***
∏	\[product]	product	u220F	product ***
	\[coproduct]	uni2210	u2210	coproduct ***
∇	\[gr]	gradient	u2207	gradient +
√	\[sr]	radical	u221A	square root +
√	\[sqrt]	radical	u221A	square root
‾	\[radicalex]	radicalex	---	square root continuation ***
‾	\[sqrtex]	radicalex	---	square root continuation ***
⌈	\[lc]	uni2308	u2308	left ceiling +
⌉	\[rc]	uni2309	u2309	right ceiling +
⌊	\[lf]	uni230A	u230A	left floor +
⌋	\[rf]	uni230B	u230B	right floor +
∞	\[if]	infinity	u221E	infinity +
ℵ	\[Ah]	aleph	u2135	aleph
ℑ	\[Im]	Ifraktur	u2111	Gothic I, imaginary
ℜ	\[Re]	Rfraktur	u211C	Gothic R, real
℘	\[wp]	weierstrass	u2118	Weierstrass p
∂	\[pd]	partialdiff	u2202	partial differentiation +
ℏ	\[-h]	uni210F	u210F	Planck constant / 2pi (h-bar)
ℏ	\[hbar]	uni210F	u210F	Planck constant / 2pi (h-bar)

Greek glyphs

These glyphs are intended for technical use, not for real Greek; normally, the uppercase letters have upright shape, and the lowercase ones are slanted. There is a problem with the mapping of letter phi to Unicode. Prior to Unicode version 3.0, the difference between U+03C6, GREEK SMALL LETTER PHI, and U+03D5, GREEK PHI SYMBOL, was not clearly described; only the glyph shapes in the Unicode book could be used as a reference. Starting with Unicode 3.0, the reference glyphs have been exchanged and described verbally also: In mathematical context, U+03D5 is the stroked variant and U+03C6 the curly glyph. Unfortunately, most font vendors didn't update their fonts to this (incompatible) change in Unicode. At the time of this writing (January 2006), it is not clear yet whether the Adobe Glyph Names 'phi' and 'phi1' also change its meaning if used for mathematics, thus compatibility problems are likely to happen – being conservative, groff currently assumes that ‘phi’ in a PostScript symbol font is the stroked version.

In groff, symbol ‘`\[*f]`’ always denotes the stroked version of phi, and ‘`\[+f]`’ the curly variant.

Output	Input	PostScript	Unicode	Notes
Α	\[*A]	Alpha	u0391	+
Β	\[*B]	Beta	u0392	+
Γ	\[*G]	Gamma	u0393	+
Δ	\[*D]	Delta	u0394	+
Ε	\[*E]	Epsilon	u0395	+
Ζ	\[*Z]	Zeta	u0396	+
Η	\[*Y]	Eta	u0397	+
Θ	\[*H]	Theta	u0398	+
Ι	\[*I]	Iota	u0399	+
Κ	\[*K]	Kappa	u039A	+
Λ	\[*L]	Lambda	u039B	+
Μ	\[*M]	Mu	u039C	+
Ν	\[*N]	Nu	u039D	+
Ξ	\[*C]	Xi	u039E	+
Ο	\[*O]	Omicron	u039F	+
Π	\[*P]	Pi	u03A0	+
Ρ	\[*R]	Rho	u03A1	+
Σ	\[*S]	Sigma	u03A3	+
Τ	\[*T]	Tau	u03A4	+
ϒ	\[*U]	Upsilon	u03A5	+
Φ	\[*F]	Phi	u03A6	+
Χ	\[*X]	Chi	u03A7	+
Ψ	\[*Q]	Psi	u03A8	+
Ω	\[*W]	Omega	u03A9	+
α	\[*a]	alpha	u03B1	+
β	\[*b]	beta	u03B2	+
γ	\[*g]	gamma	u03B3	+
δ	\[*d]	delta	u03B4	+
ε	\[*e]	epsilon	u03B5	+
ζ	\[*z]	zeta	u03B6	+
η	\[*y]	eta	u03B7	+
θ	\[*h]	theta	u03B8	+
ι	\[*i]	iota	u03B9	+
κ	\[*k]	kappa	u03BA	+
λ	\[*l]	lambda	u03BB	+
μ	\[*m]	mu	u03BC	+
ν	\[*n]	nu	u03BD	+
ξ	\[*c]	xi	u03BE	+
ο	\[*o]	omicron	u03BF	+
π	\[*p]	pi	u03C0	+
ρ	\[*r]	rho	u03C1	+
ς	\[ts]	sigma1	u03C2	terminal sigma +
σ	\[*s]	sigma	u03C3	+
τ	\[*t]	tau	u03C4	+
υ	\[*u]	upsilon	u03C5	+
ϕ	\[*f]	phi	u03D5	(stroked glyph) +
χ	\[*x]	chi	u03C7	+
ψ	\[*q]	psi	u03C8	+
ω	\[*w]	omega	u03C9	+
ϑ	\[+h]	theta1	u03D1	variant theta

φ	\[+f]	phi1	u03C6	variant phi (curly shape)
ϖ	\[+p]	omega1	u03D6	variant pi, looking like omega
	\[+e]	uni03F5	u03F5	variant epsilon

Card symbols

Output	Input	PostScript	Unicode	Notes
♣	\[CL]	club	u2663	black club suit
♠	\[SP]	spade	u2660	black spade suit
♥	\[HE]	heart	u2665	black heart suit
	\[u2661]	uni2661	u2661	white heart suit
♦	\[DI]	diamond	u2666	black diamond suit
	\[u2662]	uni2662	u2662	white diamond suit

AUTHORS

This document was written by James Clark ⟨jjc@jclark.com⟩, with additions by Werner Lemberg ⟨wl@gnu.org⟩ and Bernd Warken ⟨groff−bernd.warken−72@web.de⟩, and revised to use real tables by Eric S. Raymond ⟨esr@thyrsus.com⟩.

SEE ALSO

Groff: The GNU Implementation of troff, by Trent A. Fisher and Werner Lemberg, is the primary *groff* manual. Section “Using Symbols” may be of particular note. You can browse it interactively with “info '(groff)Using Symbols'”.

groff(1)
the GNU roff formatter

groff(7)
a short reference of the groff formatting language

An extension to the troff character set for Europe, E.G. Keizer, K.J. Simonsen, J. Akkerhuis; EUUG Newsletter, Volume 9, No. 2, Summer 1989

The Unicode Standard ⟨http://www.unicode.org⟩