```
false 0 ; true -1 ;
off ( a -- ) true swap ! ; on ( a -- ) false swap ! ;
pad ( -- a ) temporary space Vhere @ 256 + ;
The number conversion routines A and B address registers.
hex 16 Base ! ; decimal 10 Base ! ; octal 8 Base ! ;
8 Infd 8 Outfd
type ( a n -- ) Outfd @ -rot ( fd a n ) syswrite drop ;
OREAD 0 ; OWRITE 1 ; ORDWR 2 ; OTRUNC 16 ; DMREAD 4 ; DMWRITE 2 ;
close ( fd -- ) sysclose drop ;
exit Infd @ close Outfd @ close bye ;
8 Type 8 Len 264 Str 8 Char
ctype ( c -- ) write a character from the stack to Outfd
    Char c! Char 1 type ;
swrite ( counted_string -- ) Outfd @ swap count syswrite drop ;
read ( a n -- ) Infd @ -rot ( fd a n ) sysread test if drop ; then eof exit ;
exit showpage  swrite Infd @ close Outfd @ close bye ;
space writes a space to outfd   swrite ;
newline Newline  swrite ;
tab Tab  swrite ;
none do nothing exit ;
>name read to Len and Str from Infd Len 1 read Str Len c@ read ;
slash \ swrite ;
slash? ( c -- ) writes a slash if the character is (, ), or \
    dup 92 ascii code of \ - ( c c-92 )
    if drop 40 ( 41 ) belongs -if slash then ; then
    ( c c-92=0 ) drop drop slash ;
slashed ( a n -- ) write n characters at a to Outfd.
    If the character is ) or (, write a slash before them
    swap A! begin c@+ dup slash? ctype 1 - until ;
name> ( counted_string -- ) writes the counted string, Len and Str to Outfd
    ( swrite Str Len c@ slashed )  swrite ;
name ( counted_string -- ) read the name and write the counted_string and name
    >name name> swrite ;
>base Base 1 read ;
num read and write the number
    >base >name name> decimal swrite ;
number Number  num ; literalcompile Literalcompile  num ;
textcomment Textcomment  name ; execute Execute  name ;
compile Compile  name ; define Define  name ;
variable Variable  name ; address Address  name ;
macrocompile Macrocompile  name ; string String  name ;
perm DMREAD DMWRITE or ; perms perm dup 3 shl over 6 shl or or ;
opens 1 arg OREAD sysopen Infd ! 2 arg OWRITE perms syscreate Outfd ! ;
>type Type 1 read Type c@ ;
interpret jump
    none ; textcomment ; execute ; number ; compile ; literalcompile ;
    macrocompile ; address ; string ; tab ; newline ;
    define ; variable ; none ;
loop >type interpret loop ;
header
```

```
    /maxx 612 def /maxy 792 def /size 13 def /margin 72 def  swrite
    /newpage { newpath 0 0 moveto 0 maxy lineto maxx maxy lineto maxx 0 lineto
swrite
    closepath 0 setgray fill startx starty moveto } def  swrite
    /Times-Roman findfont size scalefont setfont  swrite
    /startx {margin} def /stopx {maxx margin sub} def  swrite
    /starty {maxy margin sub} def /stopy {margin} def  swrite
    /y {starty} def /nextpage {showpage newpage /y {starty} def} def  swrite
    /Newline {y size sub /y exch def y stopy le {nextpage} if startx y moveto} def  swrite
    /showword { %stack: string  swrite
    dup stringwidth pop % calculate length of string  swrite
    currentpoint pop % get horizontal position  swrite
    add stopx gt % sum > right margin  swrite
    {Newline} {( ) show} ifelse % if so, next line  swrite
    show} def  swrite
    /Tab {(    ) showword} def  swrite
    /Textcomment {1 1 1 setrgbcolor showword} def  swrite
    /Execute {1 1 0 setrgbcolor showword} def  swrite
    /Number {1 0.7 0 setrgbcolor showword} def  swrite
    /Compile {0 1 0 setrgbcolor showword} def  swrite
    /Literalcompile {0 0.5 0 setrgbcolor showword} def  swrite
    /Macrocompile {0.6 0.6 0.31 setrgbcolor showword} def  swrite
    /Define {1 0 0 setrgbcolor showword} def  swrite
    /Variable {1 0 1 setrgbcolor showword} def  swrite
    /Address {0 0.38 0.73 setrgbcolor showword} def  swrite
    /String {0.7 0.3 0 setrgbcolor showword} def  swrite
    newpage  swrite ;
 main opens header loop ;
```