

NAME

printing – producing formatted output

DESCRIPTION

The `print` command and the `printf()` and `fprintf()` functions are used to send output to files or to the screen. In `spec`, files and devices are turned *on* or *off* using special commands (see the *files* help file), and printed output generally goes to all devices that are currently *on*. The `fprintf()` function differs in that it turns off output to everything except the file or device specified in its first argument.

The `print` command prints each of its arguments separated by spaces, and then prints a newline. Strings are printed as is, numbers are printed using the "%g" format.

The functions `printf()`, `fprintf()` and `sprintf()` use format specifications just like those in C. A few of those format specifications are %s to print a string, %g to print a floating point number and %d to print an integer. An embedded \n prints a newline. See the description of *printf()* in a C manual for more details.

BUILT-IN COMMANDS AND FUNCTIONS

`print a, b ...` – Prints each argument, separated by spaces.

`printf(format, a, b ...)` – Prints a, b, etc. using *format*.

`fprintf(file_name, format [, a, ...])` – Does formatted printing on *file_name*.

All other devices (except log files) are turned off while the string is printed.

`sprintf(format, a, b ...)` – Returns a string holding the formatted print.

EXAMPLES

```
1.SPEC> print sqrt(2), PI, PRINTER
1.41412 3.14159 /dev/lp
```

```
2.SPEC> printf("Today is %s. PI = %.3g\n", \
date(), PI)
Today is Sat Jan 23 00:54:23 1988. PI = 3.14
```

```
3.SPEC> FILENAME = sprintf("%s/data/run.%d", \
HOME, run);
```

```
4.SPEC> print FILENAME
/usr/gerry/data/run.12
```

SEE ALSO

files

printf() in any C-language programming manual.