NAME

file2c **-** convert file to c-source

SYNOPSIS

file2c [**-sx**] [**-n** *count*] [*prefix* [*suffix*]]

DESCRIPTION

The **file2c** utility reads a file from stdin and writes it to stdout, converting each byte to its decimal or hexadecimal representation on the fly. The byte values are separated by a comma. This also means that the last byte value is not followed by a comma. By default the byte values are printed in decimal, but when the **-x** option is given, the values will be printed in hexadecimal. When **-s** option is given, each line is printed with a leading tab and each comma is followed by a space except for the last one on the line.

If more than 70 characters are printed on the same line, that line is ended and the output continues on the next line. With the **-n** option this can be made to happen after the specified number of byte values have been printed. The length of the line will not be considered anymore. To have all the byte values printed on the same line, give the **-n** option a negative number.

A prefix and suffix strings can be printed before and after the byte values (resp.) If a suffix is to be printed, a prefix must also be specified. The first non-option word is the prefix, which may optionally be followed by a word that is to be used as the suffix.

This program is typically used to embed binary files into C source files. The prefix is used to define an array type and the suffix is used to end the C statement. The **-n**, **-s** and **-x** options are useful when the binary data represents a bitmap and the output needs to remain readable and/or editable. Fonts, for example, are a good example of this.

EXAMPLES

The command:

```
date | file2c ’const char date[] = {’ ’,0};’
```

will produce:

```
const char date[] = {
83,97,116,32,74,97,110,32,50,56,32,49,54,58,50,56,58,48,53,
32,80,83,84,32,49,57,57,53,10
,0};
```