```
@lines = <INPUT>;
```

you should think long and hard about why you need everything loaded at once. It's just not a scalable solution. You might also find it more fun to use the standard Tie::File module, or the DB_File module's $DB_RECNO bindings, which allow you to tie an array to a file so that accessing an element the array actually accesses the corresponding line in the file.

You can read the entire filehandle contents into a scalar.

```
{
    local(*INPUT, $/);
    open (INPUT, $file)     || die "can't open $file: $!";
    $var = <INPUT>;
}
```

That temporarily undefs your record separator, and will automatically close the file at block exit. If the file is already open, just use this:

```
$var = do { local $/; <INPUT> };
```

For ordinary files you can also use the read function.

```
    read( INPUT, $var, -s INPUT );
```

The third argument tests the byte size of the data on the INPUT filehandle and reads that many bytes into the buffer $var.

21.1.27 How can I read in a file by paragraphs?

Use the `$/` variable (see *perlvar* for details). You can either set it to `""` to eliminate empty paragraphs (`"abc\n\n\n\ndef"`, for instance, gets treated as two paragraphs and not three), or `"\n\n"` to accept empty paragraphs.

Note that a blank line must have no blanks in it. Thus `"fred\n \nstuff\n\n"` is one paragraph, but `"fred\n\nstuff\n\n"` is two.

21.1.28 How can I read a single character from a file? From the keyboard?

You can use the builtin `getc()` function for most filehandles, but it won't (easily) work on a terminal device. For STDIN, either use the Term::ReadKey module from CPAN or use the sample code in getc in *perlfunc*.

If your system supports the portable operating system programming interface (POSIX), you can use the following code, which you'll note turns off echo processing as well.

```
#!/usr/bin/perl -w
use strict;
$| = 1;
for (1..4) {
    my $got;
    print "gimme: ";
    $got = getone();
    print "--> $got\n";
}
exit;

BEGIN {
    use POSIX qw(:termios_h);

    my ($term, $oterm, $echo, $noecho, $fd_stdin);
```