The p and P formats should be used with care. Since Perl has no way of checking whether the value passed to unpack() corresponds to a valid memory location, passing a pointer value that's not known to be valid is likely to have disastrous consequences.

If there are more pack codes or if the repeat count of a field or a group is larger than what the remainder of the input string allows, the result is not well defined: in some cases, the repeat count is decreased, or unpack() will produce null strings or zeroes, or terminate with an error. If the input string is longer than one described by the TEMPLATE, the rest is ignored.

See pack for more examples and notes.

untie VARIABLE

Breaks the binding between a variable and a package. (See tie.) Has no effect if the variable is not tied.

unshift ARRAY,LIST

Does the opposite of a shift. Or the opposite of a push, depending on how you look at it. Prepends list to the front of the array, and returns the new number of elements in the array.

```
unshift(@ARGV, '-e') unless $ARGV[0] =~ /^-/;
```

Note the LIST is prepended whole, not one element at a time, so the prepended elements stay in the same order. Use reverse to do the reverse.

use Module VERSION LIST

use Module VERSION

use Module LIST

use Module

use VERSION

Imports some semantics into the current package from the named module, generally by aliasing certain subroutine or variable names into your package. It is exactly equivalent to

```
BEGIN { require Module; import Module LIST; }
```

except that Module *must* be a bareword.

VERSION may be either a numeric argument such as 5.006, which will be compared to $], or a literal of the form v5.6.1, which will be compared to $^V (aka $PERL_VERSION. A fatal error is produced if VERSION is greater than the version of the current Perl interpreter; Perl will not attempt to parse the rest of the file. Compare with require, which can do a similar check at run time.

Specifying VERSION as a literal of the form v5.6.1 should generally be avoided, because it leads to misleading error messages under earlier versions of Perl which do not support this syntax. The equivalent numeric version should be used instead.

```
use v5.6.1;        # compile time version check
use 5.6.1;         # ditto
use 5.006_001;     # ditto; preferred for backwards compatibility
```

This is often useful if you need to check the current Perl version before useing library modules that have changed in incompatible ways from older versions of Perl. (We try not to do this more than we have to.)

The BEGIN forces the require and import to happen at compile time. The require makes sure the module is loaded into memory if it hasn't been yet. The import is not a builtin–it's just an ordinary static method call into the Module package to tell the module to import the list of features back into the current package. The module can implement its import method any way it likes, though most modules just choose to derive their import method via inheritance from the Exporter class that is defined in the Exporter module. See *Exporter*. If no import method can be found then the call is skipped.

If you do not want to call the package's import method (for instance, to stop your namespace from being altered), explicitly supply the empty list: