rindex STR,SUBSTR

Works just like index() except that it returns the position of the LAST occurrence of SUBSTR in STR. If POSITION is specified, returns the last occurrence at or before that position.

rmdir FILENAME

rmdir

Deletes the directory specified by FILENAME if that directory is empty. If it succeeds it returns true, otherwise it returns false and sets `$!` (errno). If FILENAME is omitted, uses `$_`.

s///

The substitution operator. See *perlop*.

scalar EXPR

Forces EXPR to be interpreted in scalar context and returns the value of EXPR.

```
@counts = ( scalar @a, scalar @b, scalar @c );
```

There is no equivalent operator to force an expression to be interpolated in list context because in practice, this is never needed. If you really wanted to do so, however, you could use the construction `@{[ (some expression) ]}`, but usually a simple `(some expression)` suffices.

Because `scalar` is unary operator, if you accidentally use for EXPR a parenthesized list, this behaves as a scalar comma expression, evaluating all but the last element in void context and returning the final element evaluated in scalar context. This is seldom what you want.

The following single statement:

```
print uc(scalar(&foo,$bar)),$baz;
```

is the moral equivalent of these two:

```
&foo;
print(uc($bar),$baz);
```

See *perlop* for more details on unary operators and the comma operator.

seek FILEHANDLE,POSITION,WHENCE

Sets FILEHANDLE's position, just like the `fseek` call of `stdio`. FILEHANDLE may be an expression whose value gives the name of the filehandle. The values for WHENCE are `0` to set the new position *in bytes* to POSITION, `1` to set it to the current position plus POSITION, and `2` to set it to EOF plus POSITION (typically negative). For WHENCE you may use the constants `SEEK_SET`, `SEEK_CUR`, and `SEEK_END` (start of the file, current position, end of the file) from the Fcntl module. Returns `1` upon success, `0` otherwise.

Note the *in bytes*: even if the filehandle has been set to operate on characters (for example by using the `:utf8` open layer), tell() will return byte offsets, not character offsets (because implementing that would render seek() and tell() rather slow).

If you want to position file for `sysread` or `syswrite`, don't use `seek`–buffering makes its effect on the file's system position unpredictable and non-portable. Use `sysseek` instead.

Due to the rules and rigors of ANSI C, on some systems you have to do a seek whenever you switch between reading and writing. Amongst other things, this may have the effect of calling stdio's clearerr(3). A WHENCE of `1` (`SEEK_CUR`) is useful for not moving the file position:

```
seek(TEST,0,1);
```