Offset outside string

(F) You tried to do a read/write/send/recv operation with an offset pointing outside the buffer. This is difficult to imagine. The sole exception to this is that `sysread()`ing past the buffer will extend the buffer and zero pad the new area.

%s() on unopened %s

(W unopened) An I/O operation was attempted on a filehandle that was never initialized. You need to do an open(), a sysopen(), or a socket() call, or call a constructor from the FileHandle package.

-%s on unopened filehandle %s

(W unopened) You tried to invoke a file test operator on a filehandle that isn't open. Check your control flow. See also -X in *perlfunc*.

oops: oopsAV

(S internal) An internal warning that the grammar is screwed up.

oops: oopsHV

(S internal) An internal warning that the grammar is screwed up.

Operation '%s': no method found, %s

(F) An attempt was made to perform an overloaded operation for which no handler was defined. While some handlers can be autogenerated in terms of other handlers, there is no default handler for any operation, unless `fallback` overloading key is specified to be true. See *overload*.

Operator or semicolon missing before %s

(S ambiguous) You used a variable or subroutine call where the parser was expecting an operator. The parser has assumed you really meant to use an operator, but this is highly likely to be incorrect. For example, if you say "*foo *foo" it will be interpreted as if you said "*foo * 'foo'".

"our" variable %s redeclared

(W misc) You seem to have already declared the same global once before in the current lexical scope.

Out of memory!

(X) The malloc() function returned 0, indicating there was insufficient remaining memory (or virtual memory) to satisfy the request. Perl has no option but to exit immediately.

At least in Unix you may be able to get past this by increasing your process datasize limits: in csh/tcsh use `limit` and `limit datasize n` (where `n` is the number of kilobytes) to check the current limits and change them, and in ksh/bash/zsh use `ulimit -a` and `ulimit -d n`, respectively.

Out of memory during "large" request for %s

(F) The malloc() function returned 0, indicating there was insufficient remaining memory (or virtual memory) to satisfy the request. However, the request was judged large enough (compile-time default is 64K), so a possibility to shut down by trapping this error is granted.

Out of memory during %s extend

(X) An attempt was made to extend an array, a list, or a string beyond the largest possible memory allocation.

Out of memory during request for %s

(X|F) The malloc() function returned 0, indicating there was insufficient remaining memory (or virtual memory) to satisfy the request.

The request was judged to be small, so the possibility to trap it depends on the way perl was compiled. By default it is not trappable. However, if compiled for this, Perl may use the contents of `$^M` as an emergency pool after die()ing with this message. In this case the error is trappable *once*, and the error message will include the line and file where the failed request happened.