keys (or the values); in particular for Data::Dumper to use the `Sortkeys` option. If some particular order is really important, use tied hashes: for example the Tie::IxHash module which by default preserves the order in which the hash elements were added.

More subtle problem is reliance on the order of "global destruction". That is what happens at the end of execution: Perl destroys all data structures, including user data. If your destructors (the DESTROY subroutines) have assumed any particular ordering to the global destruction, there might be problems ahead. For example, in a destructor of one object you cannot assume that objects of any other class are still available, unless you hold a reference to them. If the environment variable PERL_DESTRUCT_LEVEL is set to a non-zero value, or if Perl is exiting a spawned thread, it will also destruct the ordinary references and the symbol tables that are no longer in use. You can't call a class method or an ordinary function on a class that has been collected that way.

The hash randomisation is certain to reveal hidden assumptions about some particular ordering of hash elements, and outright bugs: it revealed a few bugs in the Perl core and core modules.

To disable the hash randomisation in runtime, set the environment variable PERL_HASH_SEED to 0 (zero) before running Perl (for more information see PERL_HASH_SEED in *perlrun*), or to disable the feature completely in compile time, compile with `-DNO_HASH_SEED` (see *INSTALL*).

See Algorithmic Complexity Attacks in *perlsec* for the original rationale behind this change.

85.2.2 UTF-8 On Filehandles No Longer Activated By Locale

In Perl 5.8.0 all filehandles, including the standard filehandles, were implicitly set to be in Unicode UTF-8 if the locale settings indicated the use of UTF-8. This feature caused too many problems, so the feature was turned off and redesigned: see §85.3.

85.2.3 Single-number v-strings are no longer v-strings before "=>"

The version strings or v-strings (see Version Strings in *perldata*) feature introduced in Perl 5.6.0 has been a source of some confusion– especially when the user did not want to use it, but Perl thought it knew better. Especially troublesome has been the feature that before a "=>" a version string (a "v" followed by digits) has been interpreted as a v-string instead of a string literal. In other words:

```
%h = ( v65 => 42 );
```

has meant since Perl 5.6.0

```
%h = ( 'A' => 42 );
```

(at least in platforms of ASCII progeny) Perl 5.8.1 restores the more natural interpretation

```
%h = ( 'v65' => 42 );
```

The multi-number v-strings like v65.66 and 65.66.67 still continue to be v-strings in Perl 5.8.

85.2.4 (Win32) The -C Switch Has Been Repurposed

The -C switch has changed in an incompatible way. The old semantics of this switch only made sense in Win32 and only in the "use utf8" universe in 5.6.x releases, and do not make sense for the Unicode implementation in 5.8.0. Since this switch could not have been used by anyone, it has been repurposed. The behavior that this switch enabled in 5.6.x releases may be supported in a transparent, data-dependent fashion in a future release.

For the new life of this switch, see §85.3.1, and -C in *perlrun*.