86.15.11 ext/threads/t/libc

If this test fails, it indicates that your libc (C library) is not threadsafe. This particular test stress tests the localtime() call to find out whether it is threadsafe. See *perlthrtut* for more information.

86.15.12 Failure of Thread (5.005-style) tests

Note that support for 5.005-style threading is deprecated, experimental and practically unsupported. In 5.10, it is expected to be removed. You should migrate your code to ithreads.

The following tests are known to fail due to fundamental problems in the 5.005 threading implementation. These are not new failures–Perl 5.005_0x has the same bugs, but didn't have these tests.

```
../ext/B/t/xref.t                    255 65280    14   12  85.71%  3-14
../ext/List/Util/t/first.t           255 65280     7    4  57.14%  2 5-7
../lib/English.t                       2   512    54    2   3.70%  2-3
../lib/FileCache.t                                 5    1  20.00%  5
../lib/Filter/Simple/t/data.t                      6    3  50.00%  1-3
../lib/Filter/Simple/t/filter_only.                9    3  33.33%  1-2 5
../lib/Math/BigInt/t/bare_mbf.t                 1627    4   0.25%  8 11 1626-1627
../lib/Math/BigInt/t/bigfltpm.t                 1629    4   0.25%  10 13 1628-
                                                                   1629
../lib/Math/BigInt/t/sub_mbf.t                  1633    4   0.24%  8 11 1632-1633
../lib/Math/BigInt/t/with_sub.t                 1628    4   0.25%  9 12 1627-1628
../lib/Tie/File/t/31_autodefer.t     255 65280    65   32  49.23%  34-65
../lib/autouse.t                                  10    1  10.00%  4
op/flip.t                                         15    1   6.67%  15
```

These failures are unlikely to get fixed as 5.005-style threads are considered fundamentally broken. (Basically what happens is that competing threads can corrupt shared global state, one good example being regular expression engine's state.)

86.15.13 Timing problems

The following tests may fail intermittently because of timing problems, for example if the system is heavily loaded.

```
t/op/alarm.t
ext/Time/HiRes/HiRes.t
lib/Benchmark.t
lib/Memoize/t/expmod_t.t
lib/Memoize/t/speed.t
```

In case of failure please try running them manually, for example

```
./perl -Ilib ext/Time/HiRes/HiRes.t
```

86.15.14 Tied/Magical Array/Hash Elements Do Not Autovivify

For normal arrays `$foo = \$bar[1]` will assign `undef` to `$bar[1]` (assuming that it didn't exist before), but for tied/magical arrays and hashes such autovivification does not happen because there is currently no way to catch the reference creation. The same problem affects slicing over non-existent indices/keys of a tied/magical array/hash.