41.1.4 CHARACTER CLASSES

```
[amy]    Match 'a', 'm' or 'y'
[f-j]    Dash specifies "range"
[f-j-]   Dash escaped or at start or end means 'dash'
[^f-j]   Caret indicates "match any character _except_ these"
```

The following sequences work within or without a character class. The first six are locale aware, all are Unicode aware. The default character class equivalent are given. See *perllocale* and *perlunicode* for details.

```
\d      A digit                   [0-9]
\D      A nondigit                [^0-9]
\w      A word character          [a-zA-Z0-9_]
\W      A non-word character      [^a-zA-Z0-9_]
\s      A whitespace character    [ \t\n\r\f]
\S      A non-whitespace character  [^ \t\n\r\f]

\C      Match a byte (with Unicode, '.' matches a character)
\pP     Match P-named (Unicode) property
\p{...} Match Unicode property with long name
\PP     Match non-P
\P{...} Match lack of Unicode property with long name
\X      Match extended unicode sequence
```

POSIX character classes and their Unicode and Perl equivalents:

```
alnum   IsAlnum              Alphanumeric
alpha   IsAlpha              Alphabetic
ascii   IsASCII              Any ASCII char
blank   IsSpace  [ \t]       Horizontal whitespace (GNU extension)
cntrl   IsCntrl              Control characters
digit   IsDigit  \d          Digits
graph   IsGraph              Alphanumeric and punctuation
lower   IsLower              Lowercase chars (locale and Unicode aware)
print   IsPrint              Alphanumeric, punct, and space
punct   IsPunct              Punctuation
space   IsSpace  [\s\ck]     Whitespace
        IsSpacePerl   \s     Perl's whitespace definition
upper   IsUpper              Uppercase chars (locale and Unicode aware)
word    IsWord   \w          Alphanumeric plus _ (Perl extension)
xdigit  IsXDigit [0-9A-Fa-f] Hexadecimal digit
```

Within a character class:

```
 POSIX       traditional   Unicode
 [:digit:]       \d        \p{IsDigit}
 [:^digit:]      \D        \P{IsDigit}
```

41.1.5 ANCHORS

All are zero-width assertions.

```
^  Match string start (or line, if /m is used)
$  Match string end (or line, if /m is used) or before newline
\b Match word boundary (between \w and \W)
\B Match except at word boundary (between \w and \w or \W and \W)
\A Match string start (regardless of /m)
\Z Match string end (before optional newline)
\z Match absolute string end
\G Match where previous m//g left off
```