```
3|
4| use File::Temp;
5| use Semi::Literate;
```

Weave a semi-literate program into Text, Markdown, etc. format

```
sub MAIN($input-file,
         Bool :l(:$line-numbers)  = True;
         Str :f(:$format) is copy = 'markdown';
         Str :o(:$output-file);
         Bool :v(:$verbose) = True;
    ) {
    my Str  @options;
    my Bool $no-output-file = False;

    note "Input Format =>  $format" if $verbose;
    $format .= trim;
    given $format {
        when  /:i ^ markdown | md $ / {
            $format    = 'MarkDown2';
        };
        when  /:i ^ [[plain][\-|_]?]? t[e]?xt $ / {
            $format    = 'Text';
        }
        when  /:i ^ [s]?htm[l]? $/ {
            $format    = 'HTML2';
        }
        when /:i ^ pdf $ / {
            $format         = 'PDF';
            @options        = "--save-as=$output-file" if $output-file;
            $no-output-file = True;
        }
        when /:i ^ pdf[\-|_]?lite  $ / {
            $format         = 'PDF::Lite';
            @options        = "--save-as=$output-file" if $output-file;
            $no-output-file = True;
        }
        when /:i ^ pod 6? $/ {
            $format    = 'Pod6';
        }
        when /:i ^ [la]? tex $/ {
            $format    = 'Latex';
        }
        when /:i man [page]? $/ {
            print "\n\e[33mPod::To::Man may not support pod comment blocks...\e[0m";
            $format    = 'Man';
        }

        default {
            ;
        }

    }
    my Str $woven = weave($input-file, :$line-numbers);

    my $output-file-handle = $output-file              ??
                                open(:w, $output-file) !!
                                $*OUT
                            unless $no-output-file;
```

```
59|
60|     if $format eq 'Pod6' {
61|         $output-file-handle.spurt: $woven;
62|         return;
63|     }
64|
65|     note "Weave Format =>  $format" if $verbose;
66|     my Str $f = "Pod::To::$format";
67|     try require ::($f);
68|     if ::($f) ~~ Failure {
69|         die "$format is not a supported output format"
70|     }
71|
72|     my ($pod-file, $fh) = tempfile(suffix =>  '.rakudoc', :!unlink);
73|     note "Temp file: $pod-file" if $verbose;
74|
75|     $pod-file.IO.spurt: $woven;
76|
77|     run $*EXECUTABLE,
78|         "--doc=$format",
79|         $pod-file,
80|         @options,
81|         :out($output-file-handle);
82|
83| }
```