```
#!/usr/bin/perl
# $Id: pfmt.perl,v 1.4 2012-01-05 19:14:21-08 - - $
use strict;
use warnings;

$0 =~ s|^.*/||;
my $exit_status = 0;
END {exit $exit_status}
sub note(@) {print STDERR "@_"};
$SIG{'__WARN__'} = sub {note @_; $exit_status = 1};
$SIG{'__DIE__'} = sub {warn @_; exit};

my $linelen = 65;
if (@ARGV and $ARGV[0] =~ m/^-(.+)/) {
   $linelen = $1;
   die "Usage: $0 [-width] [filename...]\n" if $linelen =~ m/\D/;
   shift @ARGV
}

sub print_paragraph (@) {
   my (@words) = @_;
   print "\n";
   my $char_count = 0;
   for my $word (@words) {
      if ($char_count == 0) {
         print $word;
         $char_count = length $word;
      }else {
         $char_count += 1 + length $word;
         if ($char_count > $linelen) {
            print "\n", $word;
            $char_count = length $word;
         }else {
            print " ", $word;
         }
      }
   }
   print "\n" if $char_count > 0;
}

push @ARGV, "-" unless @ARGV;
for my $filename (@ARGV) {
   open my $file, "<$filename" or warn "$0: $filename: $!\n" and next;
   my @output_words;
   for (;;) {
      my $input_line = <$file>;
      last unless defined $input_line;
      my @input_words = split " ", $input_line;
      if (@input_words) {
         push @output_words, @input_words;
      }else {
         print_paragraph @output_words if @output_words;
         @output_words = ();
      }
   }
   print_paragraph @output_words;
   close $file;
}

```

```
#!/bin/sh
# $Id: mkp,v 1.1 2012-01-05 19:16:54-08 - - $
./pfmt.perl ../.score/*.dat >pfmt.output1
./pfmt.perl -40 *.java >pfmt.output2
mkpspdf pfmt.listing.ps pfmt.perl $0 pfmt.output*
```

```

This is test file #1. This is test file #1. This is test file #1.
This is test file #1. This is test file #1. This is test file #1.
This is test file #1. This is test file #1.

It is very regular and is used to check to see if word wrap
works. It is very regular and is used to check to see if word
wrap works. It is very regular and is used to check to see if
word wrap works. It is very regular and is used to check to see
if word wrap works. It is very regular and is used to check to
see if word wrap works. It is very regular and is used to check
to see if word wrap works. It is very regular and is used to
check to see if word wrap works. It is very regular and is used
to check to see if word wrap works.

Does it work with a one line paragraph?

$Id: input1.dat,v 1.1 2010-12-13 17:24:57-08 - - $


This is another file of test data for test number two. Some lines
are short. Other lines are very long lines, exceeding even the
line length that checksource.perl likes to see and will complain
about.

Are multiple input blank lines squeezed to a single output blank
line?

What happens if there is only one word per line.

$Id: input2.dat,v 1.1 2010-12-13 17:24:57-08 - - $

This paragraph is indented by a tab. Are tabs deleted at the
front of the line?

What about spaces? Do they work like pfmt.perl?

a long word should be on a line by itself
sometimesthereisaverylongwordwhichpokesoutsidethenormalmarginsometimesthereisaverylongwordwhichpokesoutsidethenormalmargin
if the word exceeds the margin

This paragraph has lots of tabs on input. Tabs should be replaced
by spaces on output.

This paragraph has lots of leading spaces and trailing tabs on
input.

$Id: input3.dat,v 1.1 2010-12-13 17:24:57-08 - - $
```

```

// $Id: jarname.java,v 1.5 2012-01-05
18:42:34-08 - - $ // // // NAME //
jarname - Print out the name of the
current jar file. // // DESCRIPTION //
Makes use of the fact that the
java.class.path, when Java // is run
from a jar, is the name of the jar. //

import static java.lang.System.*;

class jarname { public static void main
(String[] args) { String jarpath =
getProperty ("java.class.path");
out.printf ("jarpath = \"%s\"%n",
jarpath); int lastslash =
jarpath.lastIndexOf ('/'); String
jarbase = lastslash < 0 ? jarpath :
jarpath.substring (lastslash + 1);
out.printf ("jarbase = \"%s\"%n",
jarbase); } }

//TEST// ./jarname >jartest.out //TEST//
mkpspdf jarlist.ps jarname.java
jartest*.out


// $Id: parseint.java,v 1.3 2012-01-05
15:13:14-08 - - $

// // Illustrate try-catch convert args
to integers. // Iterate over each
element of args and attempt to convert
it to // an integer. If it succeeds,
print the integer. If not, catch // the
error and print an error message. //

import static java.lang.System.*;

class parseint { public static void main
(String[] args) { for (int argi = 0;
argi < args.length; ++argi) { try { int
value = Integer.parseInt (args[argi]);
out.printf ("%s = %d%n", args[argi],
value); }catch (NumberFormatException
error) { out.printf ("%s: exn %s%n",
args[argi], error.getMessage()); } } } }

//TEST// ./parseint 214748 hello -33 ''
987 >parsetest.out //TEST// mkpspdf
parsetest.ps parseint.java parsetest.out

```