In-line math: \[\frac{x+1}{x-1}\], display math: \[
\frac{x+1}{x-1}
\] (indented)

\[
\frac{x+1}{x-1}
\] (centered)
In order to integrate the provided code, it is essential to ensure that the problem is structured correctly. The code below indicates the initial setup for a problem, including the inclusion of necessary packages and the declaration of variables and functions. The problem is divided into sections, each with a clear beginning and ending, ensuring readability and maintainability.

The problems are designed to test various mathematical concepts, such as

- Vector context: Using vector context to define and manipulate vectors.
- Context object: Utilizing the context object for dynamic problem creation.
- Current context: Setting the current context to modify the problem's behavior.

Each problem includes a header indicating it as the first or last executable line, ensuring the code's proper execution.

- Number twelve: Testing the number twelve with various mathematical representations.
- Formula: Evaluating the formula $1+x$.
- Twelve: Examining the modulus operation $2 \mod 10$.
- Twelve: Applying different comparison methods to the number twelve.

The problems are structured to test students' understanding of vectors, context objects, and context manipulation, ensuring a comprehensive assessment of these concepts.
\[ f(x) = \sqrt{x^2 - 1} \cap (1,2) \]

\[ \int x \, dx = \left[ \frac{x^2}{2} \right] + C \]

\[ \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \]
```perl
DOCUMENT(); # This should be the first executable line in the problem.

loadMacros(
  "PGstandard.pl",
  "PGML.pl",
  "MathObjects.pl",
  "parserMultiAnswer.pl",
  "PGcourse.pl",
);

TEXT(beginproblem);

$showPartialCorrectAnswers = 1;

######################################################################

$mp = MultiAnswer(12,6)->with(checker=>sub {1}, singleResult=>1);

TEXT(PGML::Format2(<<'END_PGML'));

Twelve and six: \[_____]{$mp} and \[_____]{$mp}

END_PGML

######################################################################

ENDDOCUMENT(); # This should be the last executable line in the problem.

DOCUMENT(); # This should be the first executable line in the problem.

loadMacros(
  "PGstandard.pl",
  "PGML.pl",
  "MathObjects.pl",
  "parserMultiAnswer.pl",
  "PGcourse.pl",
);

TEXT(beginproblem);

$showPartialCorrectAnswers = 1;

######################################################################

Context("Vector");

TEXT(PGML::Format2(<<'END_PGML'));

\[:<1,2,3>:\]* = \[__________\]

\[:<1,2,3>:\]* = \[__________\]{name=>"vec1"}

END_PGML

ANS(Vector(1,2,3)->cmp(showCoordinateHints=>0));

NAMED_ANS(vec1=>Vector(1,2,3)->cmp);

######################################################################

ENDDOCUMENT(); # This should be the last executable line in the problem.

DOCUMENT(); # This should be the first executable line in the problem.

loadMacros(
  "PGstandard.pl",
  "PGML.pl",
  "MathObjects.pl",
);
Here is a list:
1. This is the first list item continued on the next line.
2. Additional items are easy to add.
3. Continuation need not be indented, such as this line.

A paragraph break ends the list...
1. Unless you indent the paragraph...

...in which case it is part of the list item.
2. See?

A list with alphabetic markers:
a) You can use dots
b) or parens to indicate the items

A paragraph break ends the list.

A list with roman numeral markers:

Item 1
Item 2

Ending with three spaces also ends the list

A list with numeric markers:

1. Item 1
2. Item 2

A list with alphabetic markers:
a) You can use dots
b) or parens to indicate the items

A paragraph break ends the list.

A list with roman numeral markers:

Item 1
Item 2

Ending with three spaces also ends the list

A list with numeric markers:
A list can be with stars:
* Item 1
* Item 1

Or with plus or minus:
+ Item 1
+ Item 2

Paragraphs can be used between items:

1. A list
   - with a sub-list
   - of three items
   - (indent the sub list)
2. Back to the main list

\[
\frac{x+1}{x-1} = \text{Formula}()
\]
\[ a = \{a\}, \quad f = \{f\}. \]
In math: \[ f = \{f\} \] (TeX inserted automatically), parsed: \[ f = \{f\} \] (string inserted automatically).
### Example PGML Usage:

```perl

p18.pg: TEXT(PGML::Format2(<<'END_PGML'));
P18.pg:
  Force line break by
  ending a line with two spaces

p18.pg: TEXT(PGML::Format2(<<'END_PGML'));

p18.pg: $x = "has math: [:x+1:] and ${BBOLD}bold${EBOLD}";

p18.pg: # This should be the last executable line in the problem.

```

---

### Example PGML Usage:

```perl

p19.pg: TEXT(PGML::Format2(<<'END_PGML'));

p19.pg: \$x \= \"has math: [:x+1:] and \${BBOLD}bold\${EBOLD}\";

p19.pg: \"Escaped: \[${BBOLD}] not bold \[${EBOLD}\]";

p19.pg: \"Verbatim: \[${BBOLD}]* bold \[${EBOLD}]*\";

p19.pg: \"Two stars forces the contents to be processed further:\n\$x\,";

p19.pg: \"Escaped: \[${x}\]";

p19.pg: \"Verbatim: \[${x}\]*";

p19.pg: \"Processed: \[${x}\]**\";

```

---

### Example PGML Usage:

```perl

p20.pg: TEXT(PGML::Format2(<<'END_PGML'));

p20.pg: \$x \= \"has math: [:x+1:] and \${BBOLD}bold\${EBOLD}\";

p20.pg: \"Escaped: \[${BBOLD}] not bold \[${EBOLD}\]";

p20.pg: \"Verbatim: \[${BBOLD}]* bold \[${EBOLD}]*\";

p20.pg: \"Two stars forces the contents to be processed further:\n\$x\,";

p20.pg: \"Escaped: \[${x}\]";

p20.pg: \"Verbatim: \[${x}\]*";

p20.pg: \"Processed: \[${x}\]**\";

```

---

### Example PGML Usage:

```perl

p21.pg: TEXT(PGML::Format2(<<'END_PGML'));

p21.pg: \$x \= \"has math: [:x+1:] and \${BBOLD}bold\${EBOLD}\";

p21.pg: \"Escaped: \[${BBOLD}] not bold \[${EBOLD}\]";

p21.pg: \"Verbatim: \[${BBOLD}]* bold \[${EBOLD}]*\";

p21.pg: \"Two stars forces the contents to be processed further:\n\$x\,";

p21.pg: \"Escaped: \[${x}\]";

p21.pg: \"Verbatim: \[${x}\]*";

p21.pg: \"Processed: \[${x}\]**\";

```

---

### Example PGML Usage:

```perl

p22.pg: TEXT(PGML::Format2(<<'END_PGML'));

p22.pg: \$x \= \"has math: [:x+1:] and \${BBOLD}bold\${EBOLD}\";

p22.pg: \"Escaped: \[${BBOLD}] not bold \[${EBOLD}\]";

p22.pg: \"Verbatim: \[${BBOLD}]* bold \[${EBOLD}]*\";

p22.pg: \"Two stars forces the contents to be processed further:\n\$x\,";

p22.pg: \"Escaped: \[${x}\]";

p22.pg: \"Verbatim: \[${x}\]*";

p22.pg: \"Processed: \[${x}\]**\";

```

---
Indent a section by using four spaces or a tab

    This is indented,

and continues on a second line.

Another four spaces indents again.

Go back to four to end the inner indenting.

Note, however, that you only need to indent
the first line of a paragraph to have all of it
be indented. (That may need to be changed.)

End the paragraph to go back to no indenting
or use three spaces to end the line
and that will end the indenting


Use angle brackets to center a phrase:

    >> This is centered <<

You can center several lines as a paragraph:

    >> These lines will <<

be combined <<

Or you can force line breaks with two spaces at the end:

    >> These lines will <<

be centered separately <<

A whole paragraph can be centered:

    >> This is a paragraph

that will be centered <<


Use right angle brackets to force a line or paragraph to be right-justified:

- Several lines combined
- Or a whole paragraph that is pushed to the right
- Or two lines
- justified separately.

### Two separate lines ###
### are combined ###
### A whole paragraph can be a heading ###
### End with two spaces ###
### for two lines separately ###
### The trailing hashes are optional. ###

# Heading size 1 #
## Heading size 2 ##
### Heading size 3 ###
#### Heading size 4 ####
##### Heading size 5 ######
###### Heading size 6 ########

## centered heading ##
## right-justified ##

# This should be the last executable line in the problem.

# This should be the first executable line in the problem.
Three or more dashes or equals on a line by itself forms a rule

You can specify the width and size if you want:

You can center and right-justify rules:

These words are in *bold* or _italic_.

Stars can be used in*side* a word,

but underlines_don't_work_that_way.

These should be the last executable line in the
problem.

These should be the first executable line in the
problem.

These words are in *bold* or _italic_.
Quotes are "smart" ("even here"), and don't forget about 'other' quotes.

You can quote a quote: "dumb quotes".

Preformatted text starts with a colon and three spaces:

::   This is preformatted,
::       and can include any text, e.g., <, >, $, etc.,
::   but [@ "commands" @] and other *mark up* are performed normally.
::   Use [verbatim model] if you want to include commands literally,
::use a slash to escape them: \[$x\].

The formatting can be indented, too:

Here is some indenting
: with preformatting
: on several lines.
: Now back to normal, but indented.

Text that includes commands can be enclosed to prevent interpretation:

This is not math ["x+1"] in here.

You can use more vertical bars to make verbatim verbatims:

This is [verbatimmodel].

Use backslashes to escape command characters if you need to:

This occurred in the year 1.

(Prevent accidental list).
Don’t do comment: \% you will see this \%.

Other characters quote themselves on their own: <, >, &, %, $, ^, etc.

######################################################################
# This should be the last executable line in the problem.
END_PGML

DOCUMENT();  # This should be the first executable line in the problem.

loadMacros(
  "PGstandard.pl",
  "PGML.pl",
  "MathObjects.pl",
  "PGcourse.pl",
);

TEXT(beginproblem);

$showPartialCorrectAnswers = 1;

######################################################################
Context("Interval");

$a = random(1,8,1);
$b = random(8,15,1);
$min = $a-$b; $max = $a+$b;

TEXT(PGML::Format2(<<'END_PGML'));

Solve the following inequality and enter your answer using interval notation:

\[
|x-[$a]| > [$b]
\]

Answer: \([x]\) must be in \([____________________________}\){(-inf,$min)U($max,inf)}

END_PGML

######################################################################

DOCUMENT();  # This should be the last executable line in the problem.

loadMacros(
  "PGstandard.pl",
  "PGML.pl",
  "PGgraphmacros.pl",
  "MathObjects.pl",
  "PGcourse.pl",
);

TEXT(beginproblem);

$showPartialCorrectAnswers = 1;

######################################################################
Context("Numeric");

$b=non_zero_random(-3,1,1)+1; # b=1 makes answers equal
$f = Formula("x+$b"); $g = Formula("(x-2)^2");

$F = "$f for x in \([-1,5]\) using color:blue and weight:2";
$G = "$g for x in \([0,4]\) using color:red and weight:2"

$graph = init_graph(-2,-4,6,8,axes=>[0,0],grid=>[8,12],size=>[200,200]);
Let $f$ be the linear function (in blue) and let $g$ be the parabolic function (in red) below.

1. $((f \circ g)(2)) = \underline{b}$
2. $((g \circ f)(2)) = \underline{b^2}$
3. $((f \circ f)(2)) = \underline{2+2b}$
4. $((g \circ g)(2)) = \underline{4}$
Use the laws of logarithms to rewrite the expression

\[ \ln(\sqrt{a} \text{ of } xy) = A \ln x + B \ln y \]

with constants

\[ A = \frac{1}{a} \] and

\[ B = \frac{1}{b} \].

A fence is to be built to enclose a rectangular area of $a$ square feet. The fence along three sides is to be made of material that costs $b$ dollars per foot, and the material for the fourth side costs $c$ dollars per foot. Find the dimensions of the enclosure that is most economical to construct.

Dimensions: \[ \text{[ ]} \times \text{[ ]} \] feet
p33.pg:                                                                                                                                                                                                                       
p33.pg:                                                                                                           
p33.pg:ENDDOCUMENT();        # This should be the last executable line in the problem.