

The exframe Package

Niklas Beisert

Institut für Theoretische Physik
Eidgenössische Technische Hochschule Zürich
Wolfgang-Pauli-Strasse 27, 8093 Zürich, Switzerland

nbeisert@itp.phys.ethz.ch

2019/01/21, v3.1

Abstract

`exframe` is a $\text{\LaTeX} 2_{\epsilon}$ package which provides a general purpose framework to describe and typeset exercises and exam questions along with their solutions. The package features mechanisms to hide or postpone solutions, to assign and handle points, to collect problems on exercise sheets, to store and use metadata and to implement a consistent numbering. It also provides a very flexible interface for configuring and customising the formatting, layout and representation of the exercise content.

Contents

1	Introduction	2
2	Usage	2
2.1	Exercise Environments	3
2.2	Solution Display	3
2.3	Metadata	4
2.4	Points	8
2.5	Labels and Tags	9
2.6	Layout	9
2.7	Exercise Styles	11
2.8	Package Options	12
3	Information	14
3.1	Copyright	14
3.2	Files and Installation	14
3.3	Related Packages	14
3.4	Feature Suggestions	15
3.5	Revision History	16
A	Sample Files	17
A.1	Single Exercise Sheet	17
B	Implementation	21
B.1	General Definitions	21
B.2	Package Setup	22
B.3	Configuration	24

B.4	Styles	30
B.5	Metadata	32
B.6	Counters	37
B.7	Buffers	38
B.8	Points	40
B.9	Environments	42

1 Introduction

This package provides a framework to describe and typeset exercises (homework problems, classroom exercises, quizzes, exam questions, exercise questions in books and lecture notes, ...) and their solutions or answers. The aim of this package is to set up a few L^AT_EX environments into which questions and corresponding answers can be filled conveniently. The main task of the package is to manage the text and data that are provided in the source document, perform some common operations on them, and then output the content appropriately. The package has the following goals, tasks and features:

- The package is designed with generality in mind. It is meant to be usable in many different situations. The primary target is science and education, but it may well be useful in other areas.
- The package defines a basic functional layout for the output and provides many options to reshape the layout and formatting according to the author's needs and wishes.
- The package can handle two layers of exercises: main problems and subproblems. The use of subproblems is optional.
- The display of solutions can be configured: Solutions can be hidden for a hand-out version of exercise sheets. When displayed, they may appear immediately, collectively after the problem, at the end of each sheet or at some manually defined location.
- The package can handle exercise sheets which combine several exercise problems: A L^AT_EX document can consist of an individual sheet or of a collection of sheets (e.g. spanning a lecture course). In the latter case, the document files can be set up such that single sheets as well as a collection of all sheets can be compiled; the package `childdoc` may be of assistance.
- The package can handle points to be credited: Points will be displayed according to the layout. Overall points for a problem or a sheet can be added automatically. Points can also be stored and used elsewhere.
- The package provides an interface to specify exercise metadata (author, source, ...): Some basic types of metadata are predefined and more specific metadata categories can be added.
- The package can use alternative counters for equations within solutions (and problems). This is to ensure a consistent numbering independently of whether solutions are output or not.

2 Usage

To use the package `exframe` add the command

$$\backslash\text{usepackage}\{\text{exframe}\}$$

to the preamble of the L^AT_EX document.

2.1 Exercise Environments

The package provides four environments to describe the main entities of exercise problems. Additional information on the exercises can be provided in the optional arguments to these environments which will be discussed in the following sections. Furthermore, a limited set of commands is provided for control and extra features, see the sections below for details.

problem The `problem` environment describes an exercise problem:

```
\begin{problem}[opts]  
  problem text and subproblems  
\end{problem}
```

As one of the many available options *opts*, one can provide a title for the exercise by specifying `title={title}`. If no title is given, the problem number will be displayed instead. See [section 2.3](#) and [section 2.4](#) for a description of the available options.

subproblem The `subproblem` environment describes a subproblem, part or an individual question of an exercise problem:

```
\begin{subproblem}[opts]  
  subproblem text  
\end{subproblem}
```

A `subproblem` block must be contained within a `problem` block (however, a `problem` block need not contain `subproblem` blocks).

solution The `solution` environment describes the solution to a problem or a subproblem:

```
\begin{solution}[opts]  
  solution text  
\end{solution}
```

A `solution` block should be at the end of a `subproblem` or `problem` block (it is not mandatory to provide a `solution`). It can be contained within the corresponding block or it can follow it. Depending on the choice of solution display, see [section 2.2](#), the output may have a slightly different layout. In terms of logic, it is preferred to define a solution *within* the corresponding block; this may also have some technical advantages and produce a slightly better result in terms of layout.

sheet The `sheet` environment describes an exercise sheet:

```
\begin{sheet}[opts]  
  sheet text and problems  
\end{sheet}
```

A sheet typically contains one or several problems (it is not mandatory to group problems into a `sheet`). There may or may not be additional auxiliary text introducing the problems. A header will be added to the sheet according to the specified layout.

2.2 Solution Display

There are several options to control the output of solutions.

solutions Most importantly, the display of solutions can be disabled or enabled altogether:

```
\exercisestop{solutions[=true|false]}
```

Solutions are hidden by default, and their display needs to be activated explicitly (it suffices to specify the option `solutions` without the value `true`). It is also possible to control the display by an analogous package option `solutions`, see [section 2.8](#) for further information.

`\ifsolutions` The display of solutions is reflected by the conditional `\ifsolutions`. As the hiding of solutions is performed automatically, the conditional would typically be used to change some details, e.g. for adjusting titles:

```
\ifsolutions Solutions\else Exercises\fi
```

`solutionequation` As solutions can contain numbered equations while the display of solutions can be switched on and off, it is important to assign a different counter for equations within solutions in order for the equation numbers to be stable. A separate counter for equations within solutions is enabled by default. It can be disabled by:

```
\exercisestyle{solutionequation=false}
```

This option prepends the letter ‘S’ to equation numbers within solutions which are counted separately; the display can be configured differently, see [section 2.6](#).

`solutionbelow` The package allows to collect solutions and defer their display to particular locations:

`\insertsolutions`

```
\exercisestyle{solutionbelow=pos}
```

The available choices for *pos* are to display solutions where they are defined (`here`), defer them to the end of the current subproblem (`subproblem`), problem (`problem`) or sheet (`sheet`) or display them at a manually chosen location (`manual`). Note that typically solutions are defined at the end of a (sub)problem and therefore the choice `here` is similar to `(sub)problem`. The latter form, however, makes sure that a solution does not inherit the margin of the parent block. The alternate modes `problem*` and `subproblem*` positions the solution *after* the (sub)problem block such that it does not inherit any layout, but also the no definitions made in the parent block. In `manual` mode, all solutions are collected (with appropriate headers) until they are output by the directive `\insertsolutions`. If no solutions are stored in the buffer (or if the mode is not `manual`), `\insertsolutions` has no effect.

`\readsolutions` Another option to handle solutions is to write them to a file for later use. Writing to a file is initiated by:

```
\writesolutions[filename]
```

The optional argument describes the filename as *filename.sol*; no argument defaults to the main tex filename as `\jobname.sol`. This mode overrides the `solutionbelow` behaviour described above; all subsequent solutions are written to the file. The file is closed by `\closesolutions` and the display of solutions returns to manual mode. It is not necessary to close a file as it will be closed automatically by reading from a file, writing to another file or by the end of the document.

`\readsolutions` Solutions are read from a file by:

```
\readsolutions[filename]
```

This command outputs a sectional title and reads the file via `\input{filename.sol}`.

2.3 Metadata

In a collection of exercise problems it makes sense to keep track of metadata for the overall collection as well as for individual problems and potentially display some of them. The

framework defines a standard set of metadata fields and offers functionality to add more specialised metadata fields.

`\exercisedata` Global metadata is specified by the command:

$$\backslash\text{exercisedata}\{data\}$$

The argument *data* is a comma-separated list of metadata specifications in the form *key={value}*. The standard set of global metadata keys consists of:

- **author**: principal author(s) of the exercise collection; also invokes the L^AT_EX command `\author`; will be written to pdf documents.
- **title**: title of the exercise collection; also invokes the L^AT_EX command `\title`; will be written to pdf documents.
- **date**: date of the exercise collection; also invokes the L^AT_EX command `\date`; will be written to pdf documents.
- **subject**: subject area of the exercise collection; will be written to pdf documents.
- **keyword**: keyword(s) for the exercise collection; will be written to pdf documents.
- **course**: title of the course (class, lecture, module, ...) for the exercise collection.
- **institution**: institution (school, department, institute, university, ...) offering the course or exercise collection.
- **instructor**: instructor(s) for the course or exercise; this field refers to person(s) who organise the corresponding course or exercises whereas **author** refers to the principal creator of the material.
- **period**: period (year, season, date, term identifier, ...) of the corresponding course.
- **material**: type of material (exercises, homework assignments, exam, quizzes, solutions, ...).

`\defexercisedata` Additional custom fields for global metadata can be created with:

$$\backslash\text{defexercisedata}\{key\}$$

`\getexercisedata` Global metadata should typically be specified somewhere at the top of the main document, and it can be inserted wherever needed. There are two commands to read and process metadata. To insert the value of metadata field *key* use:

$$\backslash\text{getexercisedata}\{key\}$$

In some situations the output should depend on whether a metadata has been filled (e.g. to fill a default value or to display something else instead). This can be checked with the conditional:

$$\backslash\text{exercisedataempty}\{key\}\{empty\ code\}\{filled\ code\}$$

The *empty code* is executed if no value or an empty value has been specified; otherwise the *filled code* is executed.

`sheet` The package offers a similar mechanism to describe and use metadata for sheets and problems:
`problem`

$$\backslash\text{begin}\{\text{sheet}\}\{opts\}\ \backslash\text{begin}\{\text{problem}\}\{opts\}$$

The argument *opt* is a comma-separated list which can contain metadata specifications in the form *key={value}*. The standard set of metadata keys for sheets consists of:

- **due**: indication of the due date for the exercise sheet.
- **handout**: indication of the handout date for the exercise sheet.
- **title**: specifies a title for the sheet; when reading value (see below), returns composed title; untitled sheets will be displayed by their number; title will be written to pdf documents.
- **rawtitle** (for reading only): contains the raw title as specified by **title**.
- **author**: author(s) of the sheet; will be written to pdf documents.
- **editor**: editor(s) of the sheet; this field refers to a person who makes adjustments to the sheet whereas **author** refers to the creator of the sheet.
- **editdate**: indication of the date when the sheet was last edited.

The standard set of metadata keys for problems consists of:

- **title**: specifies a title for the problem; when reading value (see below), returns composed title; untitled problems will be displayed by their number.
- **rawtitle** (for reading only): contains the raw title as specified by **title**.

<code>\defsheetdata</code> <code>\getsheetdata</code> <code>\sheetdataempty</code> <code>\defproblemdata</code> <code>\getproblemdata</code> <code>\problemdataempty</code>	<p>Metadata for sheets can be used in the same way as the global metadata. The following directives are analogous to <code>\defexercisedata</code>, <code>\getproblemdata</code> and <code>\problemdataempty</code>:</p> <pre> \defsheetdata{key} \getsheetdata{key} \sheetdataempty{key}{empty code}{filled code} \defproblemdata{key} \getproblemdata{key} \problemdataempty{key}{empty code}{filled code} </pre>
--	---

<code>pdfdata</code> <code>\writeexercisedata</code>	<p>The most relevant metadata can be written to the metadata section of pdf files (using pdfL^AT_EX and the package <code>hyperref</code> whenever loaded). This feature is configured by:</p>
---	--

```
\exercisestop{pdfdata[=auto|manual|sheet|off]}
```

The option `auto` writes the global metadata `title`, `author`, `subject` and `keyword` to the corresponding fields in the pdf file. To make this work, these must be defined before the `\begin{document}` directive. The option `manual` allows to manually write these metadata by the command `\writeexercisedata`. It should be issued after the metadata have been set, but before any content is written to the pdf file. In other words, it can be anywhere in the document preamble directly after `\begin{document}`, or following a couple of content-free definitions at the beginning of the document body (in case the metadata should be set within the document body for some reason). The option `sheet` writes out the metadata at the beginning of the first `sheet` environment (which should follow `\begin{document}` without any content in between). This option is primarily for filling the `author` and `title` fields with metadata of a sheet rather than a collection of exercises. Note that if no `author` is defined for the sheet, the global metadata `author` is used. The option `off` disables all writing of metadata.

<code>problem</code> <code>subproblem</code> <code>solution</code>	<p>There is an additional mechanism to keep track of metadata for problems, subproblems and solutions which can be displayed in the opening line of these entities. Displayed metadata serve two purposes: they are used to describe the quality of a problem or they are intended</p>
--	--

for internal documentation purposes. Their output can be controlled individually, e.g. only in development versions of a document. Note that specifying a key more than once will display the content multiple times in the order in which they are encountered. Displayed metadata are specified at the top of the corresponding environment:

```
\begin{problem}[opts]  
\begin{subproblem}[opts]  
\begin{solution}[opts]
```

The standard set of displayed metadata keys consists of:

- **author**: author(s) of the problem (or subproblem, solution).
- **editor**: editor(s) of the problem; this field refers to a person who has made adjustments to the problem whereas **author** refers to the creator of the problem.
- **source**: source of the problem; in case the problem has been taken from elsewhere (conceptually or literally).
- **difficulty**: indication of the level of difficulty of the problem.
- **keyword**: keyword(s) for the problem;
- **comment**: some comment on the problem.
- **optional** (enabled by default): whether addressing the problem is mandatory or optional; by default the text will be displayed after the title in italic shape.

extdata Further displayed metadata keys are defined by the package option **extdata**, see [section 2.8](#):

- **review**: field to review the aspects of the problem (quality, length, appropriateness, difficulty, ...).
- **recycle**: indication of previous instances where this problem was used.
- **timesolve**: indication of the time needed to solve this problem (or subproblem).
- **timepresent**: indication of the time needed to present this problem (or subproblem, solution).

\showprobleminfo The display of the above metadata fields for a problem (or subproblem, solution) is controlled by:

```
\showprobleminfo{keys}
```

Here *keys* is a comma-separated list of keys to be activated (*key* or *key=true*) or deactivated (*key=false*).

\defprobleminfo Displayable metadata can be defined or adjusted by:

```
\defprobleminfo{key}{code}
```

Here *key* specifies the metadata field and *code* the code to display this type of metadata where the argument **#1** represents the data to be displayed.

insertprobleminfo Additional information can be injected into the opening line of problems and solutions by the definitions:

insertsubprobleminfo

insertsolutioninfo

info

\addprobleminfo

\addsubprobleminfo*

```
\exerciseconfig{insertprobleminfo}{code}
```

```
\exerciseconfig{insertsubprobleminfo}{code}
```

```
\exerciseconfig{insertsolutioninfo}{code}
```

The hook code *code* will be called after processing the environment arguments. Information can be added to the opening line by:

```
\addprobleminfo{info}
\addprobleminfo*{info}
```

The unstarred command adds information at the end of the opening line, the starred version at the beginning (but after the title or identifier).

2.4 Points

points Exercise problems or certain parts of them can be credited with points (credits, awards, ...). The package provides an interface to specify and manage such points. Points are declared by the option `points=points` for the environments `sheet`, `problem` and `subproblem`. These numbers will be printed to the opening line of problems and subproblems.

Note that the points should normally be integer numbers. Fractional points are permissible as well, but the internal storage by the $\text{T}_{\text{E}}\text{X}$ engine is somewhat limited, so that only fractions with powers of two as denominators (.5, multiples of .25, .125, .0625, ...) are reliable. More general fractional decimal numbers such as multiples of 0.2 will be subject to rounding errors and will not display nicely.

pointsat **subpointsat** The location where points of problems and subproblems shall be displayed can be adjusted individually by:

```
\problemstyle{points=start|start*|margin|end|manual|off}
\problemstyle{subpoints=start|start*|margin|end|manual|off}
```

The default values are `start` and `end` for problems and subproblems, respectively. The option `start` displays points at the very end of the opening line; the option `start*` displays them at the start of it. The option `end` displays points at the end of the problem or subproblem text. The option `margin` displays points in the margin. The option `manual` displays points at a manually chosen location specified by the directive `\showpoints`. Note that `\showpoints` can also be used for the option `end` to display the points prematurely (e.g. if the text ends with a displayed equation, it may make sense to display the points just before the equation). The option `off` disables the display of points.

\getsheetdata Points for sheets are only stored by the package; they must be displayed manually. Within the corresponding `sheet` environment the points can be accessed by:

```
\getsheetdata{points}
```

\getsheetpoints **\getproblempoints** The package allows to read the point totals for other sheets and problems:

```
\getsheetpoints{tag}
\getproblempoints{tag}
```

Here *tag* is the tag assigned to the corresponding sheet or problem, see [section 2.5](#).

\awardpoints Grading instructions with points to be awarded can be specified in the solution text by:

```
\awardpoints[details]{points}
\awardpoints*[details]{points}
```

Here *details* is an optional text with further details, e.g. to explain under which conditions these points are to be awarded. The starred form is used to specify optional points or

alternative paths with alternative grading instructions. These points will be marked and not be used for the computation of a total.

warn_{text} The package attempts to add up the points of subproblems to the problem total and likewise the points of problems to the sheet total. The package also performs some sanity checks on the provided numbers: If points are specified for both subproblems and problems or for both problems and sheets, they will be compared. Also the points within solutions (excluding optional or alternative points) are added up and compared to the corresponding problem or subproblem. Furthermore the package checks whether points are defined for all subproblems within a problem or all problems within a sheet. Mismatches are reported as package warnings. As point mismatches can be rather severe, there is an option to write such warnings directly into the output document (to be removed before distribution):

```
\exercisecsetup{warntext[=true|false]}
```

frac_{points} The package offers pretty display of fractional points with denominators 2, 4 and 8 by writing the decimal part as a fraction, e.g. $1.75 \rightarrow 1\frac{3}{4}$. This feature is enabled by:

```
\exercisecstyle{fracpoints}
```

2.5 Labels and Tags

label L^AT_EX provides labels to make references to remote parts of the text. Labels can be set as usual by `\label{label}` within the `problem`, `subproblems` and `sheet` environments. Alternatively, they can be specified as the environment option:

```
label={label}
```

tag The package provides an additional mechanism to tag sheets and problems. Each `sheet` and each `problem` can be assigned a unique tag `tag` by the environment option:
`\sheettag`
`\problemtag`

```
tag={tag}
```

This tag is used for reading point totals as described in [section 2.4](#). Furthermore, the macro `\sheettag` or `\problemtag` is set to the tag `tag` within the current environment. If no tag is specified it matches the number of the sheet or problem; note that this number can change by reordering sheets and problems and therefore it should not be used to identify the entity from other parts of the document.

A useful application for tags is to encapsulate labels within individual sheets and problems which are part of a collection of exercises. Labels which are composed as `\sheettag-label` or `\problemtag-label` can be considered local and will not clash with labels defined within a different environment. Within the same sheet or problem, local labels can be accessed by the same construction. They can also be accessed from remote parts of the document by fully expanding `\sheettag` or `\problemtag` for the desired target environment.

autolabel_{sheet} If unique tags are specified, the package can automatically create labels for sheets
autolabel_{problem} (`sheet:tag`) and problems (`prob:tag`) by:

```
\exercisecsetup{autolabelsheet[=true|false]}
\exercisecsetup{autolabelproblem[=true|false]}
```

2.6 Layout

The package provides a large number of parameters to adjust the display of exercises to a desired layout.

`\exerciseconfig` Configuration settings are declared and modified by the command:

$$\backslash\text{exerciseconfig}\{key\}[narg]\{value\}$$

Here *key* is a key and *value* is its assigned value. Configuration options can also be macros with arguments in which case *narg* is the number of arguments and *value* is the macro definition using arguments *#n*. The command `\exerciseconfig` therefore is analogous to `\(re)newcommand` except that the definitions are encapsulated by the package and any previous definition is overwritten without checking.

`\exerciseconfigappend` In some cases it may be useful to be able to append or prepend to a (parameterless) definition
`\exerciseconfigprepend` by:

$$\backslash\text{exerciseconfigappend}\{key\}\{value\}$$
$$\backslash\text{exerciseconfigprepend}\{key\}\{value\}$$

`\getexerciseconfig` Configuration definitions can be read by:
`\exerciseconfigempty`

$$\backslash\text{getexerciseconfig}\{key\}[arguments]$$

The number of arguments after *{key}* must match the optional argument *nargs* of the definition. Furthermore, it can be checked whether a configuration definition is empty:

$$\backslash\text{exerciseconfigempty}\{key\}\{empty\ code\}\{filled\ code\}$$

The *empty code* is executed if no value or an empty value has been specified. Otherwise the *filled code* is executed.

The package defines numerous layout configuration options. They are listed along with their original definition and a brief description in [section B.3](#). They include options to:

- adjust the language for the principal entities of this package like ‘sheet(s)’, ‘problem(s)’, ‘solution(s)’, ‘points(s)’;
- adjust the fonts styles of various parts of the text;
- adjust the spacing above, below, between various elements;
- define code to process data and insert text at various locations;
- compose text to be used in various situations;
- adjust the appearance of counters;
- adjust some other behaviour of the package.

The following will highlight only few examples.

`insertsheettitle` An important setting is:

$$\backslash\text{exerciseconfig}\{\text{insertsheettitle}\}\{code\}$$

The code *code* is meant to print the title or header of an exercise sheet. The minimalistic default code `\centerline{\getsheetdata{title}}` merely prints the sheet title “Sheet #” at the centre of a line. Commonly, one would replace this by a more elaborate header (potentially with some more information, appealing layout, logos, ...). In order to design a header template, it makes sense to retrieve data via `\getexercisedata` and `\getsheetdata` described in [section 2.3](#). Likewise `\exercisedataempty` and `\sheetdataempty` can be used to display default values or alternative data if some particular data is not provided. An example is given by the `plainheader` extended style option defined in [section B.4](#).

`composetitleproblem` Another noteworthy example is `composetitleproblem` to compose the title for a problem. It takes two parameters, the number and the title. The (somewhat simplified) default declaration is:

```
\exerciseconfig{composetitleproblem}[2]{\exerciseifempty{#2}
  {\getexerciseconfig{termpproblem}
   \getexerciseconfig{composeitemproblem}{#1}}
  {\getexerciseconfig{composeitemproblem}{#1} #2}}
```

This checks whether the title is empty. If no title is given use “Problem #.”, otherwise use “#. *title*”. Here the term “Problem” is made abstract by the configuration `termpproblem` (e.g. to support internationalisation) and the problem number is further composed obtained by the configuration `composeitemproblem` which takes the bare number as argument and returns it followed by a dot.

`\exerciseifempty`
`\exerciseifnotempty` Handy conditionals command to check whether an expression *expr* is empty are:

```
\exerciseifempty{expr}{empty code}{filled code}
\exerciseifnotempty{expr}{filled code}
```

Their main purpose is to test whether some provided expression *expr* is empty. They expand to the common T_EX constructs `\if\else#3\fi` and `\if&\else#2\fi` which work assuming that *expr* is not too exotic (e.g. it should not start with the character ‘&’ and other special T_EX characters or macros are potentially dangerous).

2.7 Exercise Styles

The package provides a mechanism to define exercise styles which customise the display of exercises in some coordinated fashion.

`\exercisestyle` Style(s) are activated by the command:

```
\exercisestyle{styles}
```

Here *styles* is a comma-separated list of styles, where each style is given by a pair *style*[=*argument*]. The package defines a couple of standard styles:

- `solutionbelow=pos` (can take values `here`, `subproblem`, `subproblem*`, `problem`, `problem*`, `sheet` and `manual`; initially set to `subproblem`) – positions the solutions below the indicated blocks; see [section 2.2](#) for details.
- `pointsat=pos` (can take values `start`, `start*`, `margin`, `end` and `manual`; initially set to `start`) – displays points in problems at the indicated location; see [section 2.4](#) for details.
- `subpointsat=pos` (can take values `start`, `start*`, `margin`, `end` and `manual`; initially set to `end`) – displays points in subproblems at the indicated location; see [section 2.4](#) for details.
- `problemby={counter}` – number problems with the prefix *counter*, i.e. reset the problem counter whenever *counter* increases and use a composite label *counter.problem* to identify problems.
- `equationby={counter}` – number the dedicated equation counters for sheets, problems and solutions with the prefix *counter*.
- `problembsheet` – number problems by sheet.

- `equationbysheet` – number dedicated equations for sheets, problems and solutions by sheet; note that the main equation counter is unaffected by this setting, it therefore makes sense to also activate the style `sheetequation` or use `\counterwithin{equation}{sheet}`.
- `pagebysheet` – number pages by sheet and denote pages by `sheet.page`; this style is useful to generate stable page numbers for a collection of sheets.
- `sheetequation[=true|false]` (no value implies `true`, initially set to `false`) – use a dedicated equation counter within sheets.
- `problemequation[=true|false]` (no value implies `true`, initially set to `false`) – use a dedicated equation counter within problems.
- `solutionequation[=true|false]` (no value implies `true`, initially set to `true`) – use a dedicated equation counter within solutions.
- `fracpoints[=true|false]` (no value implies `true`, initially set to `false`) – display fractional points for denominators 2, 4, 8; see [section 2.4](#) for details.
- `twoside[=true|false]` (no value implies `true`, initially set to `false`) – enable/disable two-sided layout; in two-sided layout, sheets will start on odd pages and empty pages are added at the end of sheets to produce an even number of pages.

`extstyle` Further exercise styles are defined by the package option `extstyle`, see [section 2.8](#):

- `plainheader` – define a plain sheet header to display some essential exercise and sheet data: `course`, `institution`, `instructor`, `period` (optional), sheet `title`, see [section 2.3](#); the line below the header, font styles and spaces can be adjusted, see the definition in [section B.4](#).
- `contents` – display sheets and problems in the table of contents (as sections and subsections).
- `solutionsf` – display solutions in sans serif font family.
- `solutiondimproblem` – dim the problem text whenever solutions are displayed.
- `solutionsep` – separate the solutions from the remaining text by horizontal lines.

`\defexercisestyle` Custom styles can be defined by:

```
\defexercisestyle{style}{init}
\defexercisestylearg[default]{style}{init}
```

This feature can be used to predefine certain aspects of the exercises layout. For example, different default page layouts could be declared in this way. The first version declares a style which is initialised by the code `item` upon activation by `\exercisestyle{style[=true]}`. Note that `\exercisestyle{style=false}` does nothing. The second version declares a style which is activated by `\exercisestyle{style[={arg}]}` and which calls `item` with the argument #1 referring to `arg` (or `default` if no argument is given).

2.8 Package Options

`\exercissetup` Features and options of general nature can be selected by the commands:

```
\usepackage[opts]{exframe}
or \PassOptionsToPackage{opts}{exframe}
or \exercissetup{opts}
```

`\PassOptionsToPackage` must be used before `\usepackage`; `\exercisestop` must be used afterwards. *opts* is a comma-separated list of options.

The following options are available only when loading the package, i.e. they will not work within `\exercisestop`:

- `extdata[=true|false]` (no value implies `true`, initially set to `false`) – define some more advanced metadata entries.
- `extstyle[=true|false]` (no value implies `true`, initially set to `false`) – define some more advanced styles.
- `problemenv=name` – redefine environment name `problem`. This and the following alike options may be useful in quickly adjusting existing sources to the `exframe` framework if the original framework works similarly and no special features are used. Otherwise, it is highly advisable to leave the names of environments and counters defined by the package untouched.
- `subproblemenv=name` – redefine environment name `subproblem`.
- `solutionenv=name` – redefine environment name `solution`.
- `sheetenv=name` – redefine environment name `sheet`.
- `problemcounter=name` – redefine counter name `problem`.
- `subproblemcounter=name` – redefine counter name `subproblem`.
- `solutioncounter=name` – redefine counter name `solution`.
- `sheetcounter=name` – redefine counter name `sheet`.

The following options can be specified by all three methods described above:

- `solutions[=true|false]` (no value implies `true`, initially set to `false`) – Enable/disable display of solutions. Sets the conditional `\ifsolutions` accordingly.
- `pdfdata[=auto|manual|sheet|off]` (no value implies `auto`, initially set to `auto`) – control writing most relevant metadata to pdf files; has no effect without package `hyperref`.
- `lineno[=true|false]` (no value implies `true`, initially set to `false`) – enable/disable writing of line numbers as comments into solution files.
- `twoside[=true|false]` (no value implies `true`, initially set to `false`) – enable/disable two-sided layout; see [section 2.7](#) for details.
- `solutionhref[=true|true]` (no value implies `true`, initially set to `false`) – enable/disable use of hyper-references from solutions to the corresponding problems; has no effect without package `hyperref`.
- `warnstext[=true|false]` (no value implies `true`, initially set to `false`) – enable/disable writing of relevant warning messages (points mismatch, point sums require update) into the document output for easier detection.
- `autolabelsheet[=true|false]` (no value implies `true`, initially set to `false`) – enable/disable automatically assigning labels (`sheet:\sheettag`; can be adjusted) to sheets according to their tag `\sheettag`.
- `autolabelproblem[=true|false]` (no value implies `true`, initially set to `false`) – enable/disable automatically assigning labels (`prob:\problemtag`; can be adjusted) to problems according to their tag `\problemtag`.

3 Information

3.1 Copyright

Copyright © 2011–2019 Niklas Beisert

This work may be distributed and/or modified under the conditions of the L^AT_EX Project Public License, either version 1.3 of this license or (at your option) any later version. The latest version of this license is in <http://www.latex-project.org/lppl.txt> and version 1.3 or later is part of all distributions of L^AT_EX version 2005/12/01 or later.

This work has the LPPL maintenance status ‘maintained’.

The Current Maintainer of this work is Niklas Beisert.

This work consists of the files `README.txt`, `exframe.ins` and `exframe.dtx` as well as the derived files `exframe.sty`, `exfsmp1.tex` and `exframe.pdf`.

3.2 Files and Installation

The package consists of the files:

<code>README.txt</code>	readme file
<code>exframe.ins</code>	installation file
<code>exframe.dtx</code>	source file
<code>exframe.sty</code>	package file
<code>exfsmp1.tex</code>	sample file
<code>exframe.pdf</code>	manual

The distribution consists of the files `README.txt`, `exframe.ins` and `exframe.dtx`.

- Run (pdf)L^AT_EX on `exframe.dtx` to compile the manual `exframe.pdf` (this file).
- Run L^AT_EX on `exframe.ins` to create the package `exframe.sty` and the sample `exfsmp1.tex`. Copy the file `exframe.sty` to an appropriate directory of your L^AT_EX distribution, e.g. `texmf-root/tex/latex/exframe`.

3.3 Related Packages

The package makes use of other packages available at CTAN:

- This package relies on some functionality of the package `verbatim` to read verbatim code from the L^AT_EX source without expansion of macros. Compatibility with the `verbatim` package has been tested with v1.5q (2014/10/28).
- This package uses the package `xkeyval` to process the options for the package, environments and macros. Compatibility with the `xkeyval` package has been tested with v2.7a (2014/12/03).
- This package can use the package `hyperref` to include hyperlinks between problems and solutions. Compatibility with the `hyperref` package has been tested with v6.88e (2018/11/30).
- This package can use the package `amstext` (which is automatically loaded by `amsmath`) to display text within equations. Compatibility with the `amstext` package has been tested with v2.01 (2000/06/29).

- This package uses the command `\currfilename` provided by the package `currfile` (if available and loaded) to indicate the \LaTeX source file in the generated metapost file. Compatibility with the `currfile` package has been tested with v0.7c (2015/04/23).
- This package cooperates well with the package `childdoc` to produce individual sheets, single problems as well as a collection of all exercises.

There are several other \LaTeX packages which offer a similar functionality varying largely in scope and sophistication:

- The package `exsheets` and its successor `xsim` provide a \LaTeX 3 style for typesetting exercises with solutions. They offer options to hide or delay solutions, print only specific problems, deal with points specify metadata, handle exercise collections, as well as some more specific options. They allow to adjust the layout and choose among predefined ones.
- The package `exercise` provides a style for typesetting exercises with solutions. It offers many options to hide or delay solutions, print only specific problems, specify some metadata as well as some more specific options. It allows to customise the layout.
- The package `exercises` provides a style for typesetting exercises with solutions. It offers options to hide solutions and deal with points. It allows basic customisation of the layout.
- The package `exam` provides a document class for typesetting exams conveniently. It offers many options to hide solutions, deal with points and deal with other exam-specific tasks. It allows to adjust the layout and choose among predefined ones.
- The package `probsoln` provides a style for typesetting exercises with solutions which are stored in a collection. It offers options to hide solutions and to assemble problems from an external collection.
- The packages `uebungsblatt`, `uassign`, `mathexam`, `exsol`, `homework`, `jhw hw` provide basic functionality for somewhat more particular situations.

See CTAN categories `exercise` and `exam` for further up-to-date packages.

The philosophy of the present package is to define a low-level framework to describe exercises with solutions to be used in various situations. The aim is to provide the means to describe the content (problems, solutions, sheets) in a simple fashion and separate it from the various layout definitions and choices which will define the appearance of the content. The interface was designed to reduce potential conflict with other packages and definitions. The package itself does not define an elaborate layout, but it provides means to adjust it in many ways and to predefine custom layout schemes. The package offers most of the functionality of the above packages, but (presently) misses out on some more advanced features, see [section 3.4](#).

3.4 Feature Suggestions

The following is a list of features which may be useful for future versions of this package:

- Option to hide problem text while maintaining access to embedded solutions (for a version containing only solutions) or to postpone problems (for collecting them in an exercises section): this is difficult to implement because the problem block cannot simply be discarded, but must be scanned very carefully for the embedded solution.
- Provide an example with multiple sheets in the same document such that the sheet can be compiled individually or as a collection.
- Define structures for multiple-choice questions.

3.5 Revision History

v3.1: 2019/01/21

- alternate placement modes for solutions
- fixed expansion of problem title
- reset font size for problem text

v3.0: 2019/01/16

- renamed to `exframe.sty`
- first version published on CTAN
- overhaul and streamline interface
- solution processing remodelled
- changed metadata handling
- changed and generalised points handling
- generalised sectioning layout
- changed layout specification model
- insert hyperlinks using `hyperref`
- manual, example and installation package added

v2.0 – v2.6: 2014/10/03 – 2018/11/05

- changed metadata interface
- broadened scope
- added more layout options
- added more metadata
- added sheet and problem tags
- add and remember points

v1.1 – v1.6: 2014/08/07 – 2014/09/14

- renamed to `nbprob.sty`
- added metadata
- added points
- added layout configuration
- removed specific macros

v1.0 – v1.02: 2011/09/23 – 2013/03/17

- first version as `problems.cls`
- dedicated layout and macros for author's exercise sheets

A Sample Files

This section provides examples of how to use some of the `exframe` features. The resulting layout will be somewhat messy due to a random selection of features...

A.1 Single Exercise Sheet

The first (and presently only) example describes a single exercise sheet. The other sheet of the series would be declared in independent documents.

Preamble. Standard document class:

```
1 \documentclass[12pt]{article}
```

Use package `geometry` to set the page layout; adjust the paragraph shape:

```
2 \usepackage{geometry}
3 \geometry{layout=a4paper}
4 \geometry{paper=a4paper}
5 \geometry{margin=2.5cm}
6 \parindent0pt
7 \parskip0.5ex
```

Include `amsmath`, `hyperref` and the `exframe` package:

```
8 \usepackage{amsmath}
9 \usepackage{hyperref}
10 \usepackage[extstyle]{exframe}
```

Solutions Switch. It will be useful to have the switch to turn on/off the display of solutions near the top of the source file, potentially with the opposite setting commented out:

```
11 \exercisecsetup{solutions=true}
12 %%\exercisecsetup{solutions=false}
```

Layout Declarations. The following layout declarations adjust the general layout of exercise sheets. They may as well be moved into an include file.

Declare a header for exercise sheets to display several relevant pieces of data; display points total:

```
13 \exercisecstyle{plainheader}
14 \exercisecconfig{composeheaderbelowright}{\getsheetdata{points}}%
```

Redefine the appearance of some counters; sheets should be labelled by capital roman numerals, subproblems by lowercase roman numerals; declare the widest subproblem item to be expected:

```
15 \exercisecconfig{countersheet}{\Roman{sheet}}
16 \exercisecconfig{countersubproblem}{\roman{subproblem}}
17 \exercisecconfig{countersubproblemmax}{vii}
```

Redefine the terms to be used for sheet(s); here, a German version:

```
18 \exercisecconfig{termsheet}{\ "Ubungsblatt}
19 \exercisecconfig{termsheets}{\ "Ubungsbl"atter}
```

Display points for problems in the margin; change margin display to use the left margin; use the abbreviated form ‘np.’:

```
20 \exercisestyle{pointsat=margin}
21 \reversemarginpar
22 \exerciseconfig{composepointsmargin}[1]{#1p.}
```

Change the basic font style for all titles to be bold sans-serif:

```
23 \exerciseconfig{styletitle}{\sffamily\bfseries}
```

Add a significant amount of space below problems:

```
24 \exerciseconfig{skipproblembelow}{1.5cm}
```

Display half points as fractions:

```
25 \exercisestyle{fracpoints}
```

Show solutions below each problem (may try alternatives `subproblem` or `sheet`):

```
26 \exercisestyle{solutionbelow=problem}
```

Separate solutions by horizontal lines (extended style):

```
27 \exercisestyle{solutionsep}
```

Add course name to sheet title metadata:

```
28 \exerciseconfig{composemetasheet}[2]{\getexercisedata{course},
29 \exerciseifempty{#2}{\getexerciseconfig{termsheet} #1}{#2}}
```

Set title and author for pdf metadata:

```
30 \exercisestop{pdfdata=sheet}
31 \exercisedata{title=%
32 {\getexercisedata{course}, \getexercisedata{material}}}
33 \exercisedata{author=%
34 {\getexercisedata{instructor}, \getexercisedata{institution}}}
```

Exercise Series Data. Set some data on the current series:

```
35 \exercisedata{institution={Katharinen-Volksschule}}
36 \exercisedata{course={Mathematik}}
37 \exercisedata{instructor={J.\ G.\ B\ "uttner}}
38 \exercisedata{period={ca.\ 1786}}
39 \exercisedata{material={"\Ubungsarbeiten}}
```

Body.

```
40 \begin{document}
```

Start sheet number 5:

```
41 \begin{sheet}[number=5]
```

Start a problem with a title:

```
42 \begin{problem}[title={Sums}]
```

Some introduction to the problem:

```
43 This problem deals with sums and series.
```

A subproblem with a local label:

```
44 \begin{subproblem}[points=2,difficulty=simple,label={\problemtag-simplesum}]
45 Compute the sum
46 \showpoints
47 \begin{equation}
48 1+2+3.
49 \end{equation}
```

Provide a solution for the subproblem (within the subproblem block):

```
50 \begin{solution}
51 The result is
52 \begin{equation}
53 1+2+3=6.
54 \end{equation}
55 \end{solution}
```

End subproblem:

```
56 \end{subproblem}
```

Another subproblem:

```
57 \begin{subproblem}[points=97,difficulty=lengthy]
58 Compute the sum
59 \begin{equation}
60 1+2+3+\ldots+98+99+100.
61 \end{equation}
62 Keep calm and calculate!
63 %%That ought to keep him occupied for a while
64 \end{subproblem}
```

Provide a solution for the previous subproblem (layout may differ slightly from declaration within); declare author:

```
65 \begin{solution}[author={C.\ F.\ Gau\ss}]
66 We use the result  $1+2+3=6$  from part \ref{\problemtag-simplesum}
67 to jumpstart the calculation. The remaining sums yield
68 \awardpoints[1 for each remaining sum]{97}
69 \begin{equation}
70 6+4+5+\ldots+99+100=5050.
71 \end{equation}
72 Alternatively the summands can be grouped into pairs as follows:
73 \begin{align}
74 1+100&=101, \\
75 2+99&=101, \\
76 3+98&=101, \\
77 \dots &\nonumber \\
78 50+51&=101.
79 \end{align}
80 These amount to 50 times the same number 101.
81 Therefore the sum equals
82 \begin{equation}
83 1+2+\ldots+99+100=50\cdot 101=5050.
84 \end{equation}
85 \textit{Ligget se!} \awardpoints*{97}
86 \end{solution}
```

Some text between subproblems:

```
87 You may give the final part a try:
```

Final subproblem; this one is optional:

```

88 \begin{subproblem}[optional={optional}, difficulty={requires inspiration}]
89 Compute the series
90 \begin{equation}
91 1+2+3+\ldots
92 \end{equation}

```

Provide a solution:

```

93 \begin{solution}
94 The series is divergent, so the result is  $\infty$  \awardpoints*{1}.
95 \par
96 However, after subtracting the divergent part,
97 the result clearly is
98 \begin{equation}
99 \zeta(-1)=-\frac{1}{12}\,,
100 \end{equation}
101 where the zeta-function  $\zeta(s)$  is defined by
102 \begin{equation}
103 \zeta(s):=\sum_{k=1}^{\infty} \frac{1}{k^s}\,,
104 \end{equation}
105 This definition holds only for  $s>1$  where the sum is convergent,
106 but one can continue the complex analytic function to  $s<0$ 
107 \awardpoints*{1.5}.
108 \par
109 Another way of understanding the result
110 is to use the indefinite summation formula
111 for arbitrary exponent  $s$  in the summand
112 (which also follows from the Euler--MacLaurin formula)
113 \begin{equation}
114 \sum_n n^s
115 = \frac{n^{s+1}}{s+1}
116 -\sum_{j=0}^s \frac{\zeta(j-s)\,s!}{(s-j)!\,j!}\,n^j
117 = \ldots - \zeta(-s)\,n^0.
118 \end{equation}
119 Curiously, the constant term with  $j=0$  is just the desired result
120 but with the wrong sign
121 (in fact, the constant term of an indefinite sum is ambiguous;
122 for the claim we merely set  $j=0$ 
123 in the expression which holds for others values of  $j$ )
124 \awardpoints*{0.5}.
125 In order to understand the sign,
126 we propose that the above formula describes the regularised result
127 for the sum with limits  $+\infty$  and  $n$ 
128 \begin{equation}
129 \sum_{k=+\infty}^n k^s
130 \simeq \frac{n^{s+1}}{s+1}
131 -\sum_{j=0}^s \frac{\zeta(j-s)\,s!}{(s-j)!\,j!}\,n^j.
132 \end{equation}
133 Then we flip the summation limits of the desired sum
134 to bring it into the above form
135 \awardpoints*{0.5}
136 \begin{equation}
137 \sum_{k=1}^{\infty} k^s
138 = -\sum_{k=\infty}^0 k^s
139 \simeq \zeta(-s).
140 \end{equation}
141 \end{solution}

```

End subproblem:

```
142 \end{subproblem}
```

End problem:

```
143 \end{problem}
```

Another problem; this one is untitled:

```
144 \begin{problem}[points=1, difficulty=insane]
145 Show that the equation
146 \begin{equation}
147 a^3+b^3=c^3
148 \end{equation}
149 has no positive integer solutions.
150 \end{problem}
```

A solution can also follow a problem (but the layout may be slightly different, e.g. here the space below the problem will appear before the solution):

```
151 \begin{solution}
152 \normalmarginpar
153 This is beyond the scope of this example.
154 \marginpar{\footnotesize\raggedright does not fit here.\par}
155 \end{solution}
```

End sheet:

```
156 \end{sheet}
```

End of document body:

```
157 \end{document}
```

B Implementation

This section describes the package `exframe.sty`.

B.1 General Definitions

Required Packages. The package loads the package `verbatim` and `xkeyval` if not yet present. `verbatim` is used for solution block reading and `xkeyval` is used for extended options processing:

```
158 \RequirePackage{verbatim}
159 \RequirePackage{xkeyval}
```

General Definitions.

`\exf@empty` Define an empty macro for comparison by `\ifx`:

```
160 \def\exf@empty{}
```

`\exf@tmpdim` Define a length for temporary storage:

```
161 \newlength\exf@tmpdim
```

`\exf@exptwo` A macro to conveniently expand the third token in line:

```
162 \def\exf@exptwo#1{\expandafter#1\expandafter}
```

`\exf@exparg` A macro to conveniently expand the first token of an argument following arbitrary code:

```
163 \long\def\exf@expswitch#1#2{#2{#1}}
164 \long\def\exf@exparg#1#2{\exf@exptwo\exf@expswitch{#2}{#1}}
```

`\exf@csdo` Some macros to conveniently expand `\csname` arguments before expanding the macro:

```
\exf@csdotwo
165 \def\exf@csdo#1#2{\expandafter#1\csname#2\endcsname}
166 \def\exf@csdotwo#1#2#3{\exf@exptwo#1#2\csname#3\endcsname}
```

`\exf@append@def` Add definitions to macros (after or before original content):

```
\exf@prepend@def
167 \long\def\exf@append@def#1#2{\exf@exptwo\def#1\expandafter{#1#2}}
168 \long\def\exf@prepend@switch#1#2#3{#2{#3#1}}
169 \long\def\exf@prepend@def#1#2{\exf@exptwo\exf@prepend@switch{#1}{\def#1}{#2}}
```

`\exf@expsetkeys` A version of `\setkeys` from `xkeyval` which expands first:

```
170 \newcommand{\exf@expsetkeys}[2]{\edef\exf@tmp{#2}%
171 \exf@exparg{\setkeys{#1}}{\exf@tmp}}
```

`\exf@href` Display text with hyperreference passed by macro #1 (in case `hyperref` is loaded and the reference is defined and not empty):

```
172 \newcommand{\exf@href}[2]{%
173 \ifdefined#1\ifx#1\exf@empty#2\else%
174 \ifdefined\hyperlink\protect\hyperlink{#1}{#2}\else#2\fi\fi\else#2\fi}
```

`\exf@text` Two macros to display text in math mode. `exf@text` is a wrapper for `\text` of `amstext` in case the latter package is loaded. `exf@ensuretext` makes sure the text is set in text mode or within an `\mbox` in math math:

```
175 \newcommand{\exf@text}[1]{\ifdefined\text\text{#1}\else#1\fi}
176 \newcommand{\exf@ensuretext}[1]{\ifmmode\mbox{#1}\else#1\fi}
```

`exf@addcontentsline` Add a line to the table of contents unless macro in argument #1 is empty:

```
177 \newcommand{\exf@addcontentsline}[2]{%
178 \ifx#1\exf@empty\else\addcontentsline{toc}{#1}{#2}\fi}
```

B.2 Package Setup

Initialisation Options.

`exframe.sty` Some setup options are available while loading the package only.

Configure names of main environments and counters:

```
179 \def\exf@problemname{problem}
180 \def\exf@subproblemname{sub\exf@problemname}
181 \def\exf@solutionname{solution}
182 \def\exf@sheetname{sheet}
183 \def\exf@problemcounter{problem}
184 \def\exf@subproblemcounter{sub\exf@problemcounter}
185 \def\exf@solutioncounter{solution}
```

```

186 \def\exf@sheetcounter{sheet}
187 \define@key{exframe.sty}{problemenv}{\def\exf@problemname{#1}}
188 \define@key{exframe.sty}{subproblemenv}{\def\exf@subproblemname{#1}}
189 \define@key{exframe.sty}{solutionenv}{\def\exf@solutionname{#1}}
190 \define@key{exframe.sty}{sheetenv}{\def\exf@sheetname{#1}}
191 \define@key{exframe.sty}{problemcounter}{\def\exf@problemcounter{#1}}
192 \define@key{exframe.sty}{subproblemcounter}{\def\exf@subproblemcounter{#1}}
193 \define@key{exframe.sty}{solutioncounter}{\def\exf@solutioncounter{#1}}
194 \define@key{exframe.sty}{sheetcounter}{\def\exf@sheetcounter{#1}}

```

Whether to provide some extended configuration options (available while loading only):

```

195 \define@boolkey{exframe.sty}[exf@]{extdata}[true]{}
196 \define@boolkey{exframe.sty}[exf@]{extstyle}[true]{}

```

Setup Options.

exf@setup All remaining setup options are available also when the package is already loaded.

Main switch for solutions:

```

197 \define@boolkey{exf@setup}[]{solutions}[true]{}

```

Switch for writing pdf metadata:

```

198 \define@choickey{exf@setup}{pdfdata}%
199 {auto,manual,sheet,off}[auto]{\def\exf@metadata{#1}}
200 \def\exf@metadata{auto}

```

Write line number indicators to output file:

```

201 \define@boolkey{exf@setup}[exf@]{lineno}[true]{}

```

Prepare two-sided sheets:

```

202 \define@boolkey{exf@setup}[exf@]{twoside}[true]{}

```

Generate hyperreferences from solutions to corresponding problems:

```

203 \define@boolkey{exf@setup}[exf@]{solutionhref}[true]{}
204 \exf@solutionhreftrue

```

Automatically generate labels for sheets and problems:

```

205 \define@boolkey{exf@setup}[exf@]{autolabelsheet}[true]{}
206 \define@boolkey{exf@setup}[exf@]{autolabelproblem}[true]{}

```

Write warning message to document for better detection of inconsistencies:

```

207 \define@boolkey{exf@setup}[exf@]{warntext}[true]{}

```

Processing. Process global options while loading package:

```

208 \ProcessOptionsX<exframe.sty,exf@setup>

```

\exercisestop Configure package when package is already loaded:

```

209 \newcommand{\exercisestop}[1]{\exf@expsetkeys{exf@setup}{#1}}

```

B.3 Configuration

This section defines and describes the various configuration options provided by the package. It also serves as a manual, and most code can be recycled and adjusted for individual configurations:

Definitions.

`\exerciseconfig` Set a configuration macro; store definition in `exf@config@#2`; use `\newcommand` for macros with arguments, but (non-long) `\def` for plain definitions:

```
210 \newcommand{\exerciseconfig}[1]{%
211   \@ifnextchar[{\exf@configopt{#1}}{\exf@confignoopt{#1}}
212   \long\def\exf@configopt#1[#2]#3{%
213     \exf@csdo\def{exf@config@#1}{}%
214     \exf@csdo\renewcommand{exf@config@#1}[#2]{#3}%
215   \long\def\exf@confignoopt#1#2{\exf@csdo\def{exf@config@#1}{#2}}
```

`\exerciseconfigappend` Append to a (parameterless) configuration macro:

`\exerciseconfigprepend`

```
216 \newcommand{\exerciseconfigappend}[2]{%
217   \exf@csdo\exf@append@def{exf@config@#1}{#2}}
218 \newcommand{\exerciseconfigprepend}[2]{%
219   \exf@csdo\exf@prepend@def{exf@config@#1}{#2}}
```

`\getexerciseconfig` Get configuration macro:

```
220 \newcommand{\getexerciseconfig}[1]{\csname exf@config@#1\endcsname}
```

`\exerciseconfigempty` Test whether configuration macro #1 is empty; execute #2 if empty, otherwise execute #3:

```
221 \newcommand{\exerciseconfigempty}[3]{\exf@csdo\ifx{exf@config@#1}\exf@empty%
222   #2\else#3\fi}
```

`\exerciseifempty` Code to test whether #1 (expanded) is empty; execute #2 if empty, otherwise execute #3:

`\exerciseifnotempty`

```
223 \long\def\exerciseifempty#1#2#3{\if&#1&#2\else#3\fi}
224 \long\def\exerciseifnotempty#1#2{\if&#1&\else#2\fi}
```

Terms.

`term...` Terms for sheet, problem, solution and points (for adjustment or internationalisation):

```
225 \exerciseconfig{termsheet}{Sheet}
226 \exerciseconfig{termsheets}{Sheets}
227 \exerciseconfig{termproblem}{Problem}
228 \exerciseconfig{termproblems}{Problems}
229 \exerciseconfig{termsolution}{Solution}
230 \exerciseconfig{termsolutions}{Solutions}
231 \exerciseconfig{termpoint}{point}
232 \exerciseconfig{termpoints}{points}
```

Formatting Styles.

`style...` Formatting styles to be applied for various parts of text. Different styles will be applied in sequence from more general to more specific.

Basic style for all exercise text:

```
233 \exercisecconfig{styletext}{\normalsize\normalfont}
```

Style for problems:

```
234 \exercisecconfig{styletextproblem}{}
```

Style for solutions:

```
235 \exercisecconfig{styletextsolution}{\footnotesize}
```

Basic style for titles:

```
236 \exercisecconfig{styletitle}{\bfseries}
```

Style for problem titles:

```
237 \exercisecconfig{styletitleproblem}{\large}
```

Style for subproblem titles:

```
238 \exercisecconfig{styletitlesubproblem}{}
```

Style for solution titles:

```
239 \exercisecconfig{styletitlesolution}{}
```

Style for problem section title in solution block:

```
240 \exercisecconfig{styletitlesolutionsproblem}{\small}
```

Style for solution block title:

```
241 \exercisecconfig{styletitlesolutions}{\normalsize}
```

Spacing.

`skip...` Spaces related to various elements. Vertical space is typically combined with space declared elsewhere using `\addvspace`.

Space above problem environment:

```
242 \exercisecconfig{skipproblemabove}{3.25ex plus 1ex minus 1.5ex}
```

Space below problem environment:

```
243 \exercisecconfig{skipproblembelow}{3pt plus 1pt minus 1pt}
```

Space below or after problem title; positive numbers generate vertical space (problem body is started in new paragraph), negative numbers generate horizontal space (problem body continues on opening line):

```
244 \exercisecconfig{skipproblemtitle}{3pt plus 1pt minus 1pt}
```

Horizontal space between items in the problem opening line:

```
245 \exercisecconfig{skipprobleminfo}{0.5em}
```

Space for problem item and indentation; `0pt` means no indentation and direct display of title; positive numbers define an absolute amount; `-1pt` (or any negative number) computes the amount of indentation from the width of (standard) item plus separator:

```
246 \exercisecconfig{skipproblemitem}{0pt}
```

Spaces related to subproblem environment; analogous to spaces related to problem environment, see above:

```
247 \exerciseconfig{skipsubproblemabove}{1.5ex plus 0.5ex minus 1ex}
248 \exerciseconfig{skipsubproblembelow}{1.5ex plus 0.5ex minus 1ex}
249 \exerciseconfig{skipsubproblemtitle}{-1em}
250 \exerciseconfig{skipsubprobleminfo}{0.25em}
251 \exerciseconfig{skipsubproblemitem}{-1pt}
```

Spaces related to solution environment; analogous to spaces related to problem environment,

```
252 \exerciseconfig{skipsolutionabove}{0ex}
253 \exerciseconfig{skipsolutionbelow}{1.5ex plus 0.5ex minus 1ex}
254 \exerciseconfig{skipsolutiontitle}{-0.5em}
255 \exerciseconfig{skipsolutioninfo}{0.25em}
```

`skipsolutionitem` and `skipsolutionitemsub` are analogous to `skipproblemitem` described above; they apply to solutions corresponding to problems and subproblems, respectively:

```
256 \exerciseconfig{skipsolutionitem}{0pt}
257 \exerciseconfig{skipsolutionitemsub}{0pt}
```

Spaces related to blocks of solutions; space above and below a solution block:

```
258 \exerciseconfig{skipsolutionsabove}{1.5ex plus 0.5ex minus 1ex}
259 \exerciseconfig{skipsolutionsbelow}{1.5ex plus 0.5ex minus 1ex}
```

Space following problem titles in a solution block:

```
260 \exerciseconfig{skipsolutionsproblem}{1.0ex plus 0ex minus 0.5ex}
```

Space following title of a solution block:

```
261 \exerciseconfig{skipsolutionstitle}{1.0ex plus 0ex minus 0.5ex}
```

Hook Code.

`insert...` Code to process data and to insert text at various points of processing.

Code to generate the title for a sheet; minimalistic default to display the sheet title:

```
262 \exerciseconfig{insertsheettitle}{\centerline{\getsheetdata{title}}}
```

Code to clear the page at the start and at the end of a new sheet:

```
263 \exerciseconfig{insertsheetclearpage}{\exercisecleardoublepage}
```

Code to insert before a sheet is displayed:

```
264 \exerciseconfig{insertsheetbefore}{}
```

Code to insert after a sheet is displayed:

```
265 \exerciseconfig{insertsheetafter}{}
```

Code to insert before a solution block is displayed:

```
266 \exerciseconfig{insertsolutionsbefore}{}
```

Code to insert after a solution block is displayed:

```
267 \exerciseconfig{insertsolutionsafter}{}
```

Code to insert before a problem is displayed:

```
268 \exercisecfg{insertproblembefore}{}
```

Code to insert after a problem is displayed:

```
269 \exercisecfg{insertproblemafter}{}
```

Code to insert between a problem and its solution (if style `solutionbelow` is problem):

```
270 \exercisecfg{insertproblemsolution}{}
```

Code to insert text into the problem opening line by means of `\addprobleminfo`:

```
271 \exercisecfg{insertprobleminfo}{}
```

Analogous definitions for subproblems:

```
272 \exercisecfg{insertsubproblembefore}{}
```

```
273 \exercisecfg{insertsubproblemafter}{}
```

```
274 \exercisecfg{insertsubprobleminfo}{}
```

```
275 \exercisecfg{insertsubproblemsolution}{}
```

Analogous definitions for solutions:

```
276 \exercisecfg{insertsolutionbefore}{}
```

```
277 \exercisecfg{insertsolutionafter}{}
```

```
278 \exercisecfg{insertsolutioninfo}{}
```

Text Composition for Environments.

`compose...` Macros to generate text for various situations. Preferably the output is plain text without formatting, but in some situations it may be required to address formatting in these macros.

Default separator for items:

```
279 \exercisecfg{composeitemsep}{\ } 
```

Compose sheet title; arguments are sheet number and raw title (empty if not specified); default is “Sheet #1” or given title “#2”:

```
280 \exercisecfg{composetitlesheet}[2]%
```

```
281 {\exercisecfg{empty}{#2}{\getexercisecfg{termsheet} #1}{#2}}
```

Compose sheet title for pdf metadata; arguments are sheet number and raw title:

```
282 \exercisecfg{composemetasheet}[2]%
```

```
283 {\getexercisecfg{composetitlesheet}{#1}{#2}}
```

Compose sheet title for table of contents; arguments are sheet number and raw title:

```
284 \exercisecfg{composetocsheet}[2]%
```

```
285 {\exercisecfg{empty}{#2}{\getexercisecfg{termsheet} #1}{#1. #2}}
```

Compose problem item; argument is problem number:

```
286 \exercisecfg{composeitemproblem}[1]{#1.}
```

Problem item separator:

```
287 \exercisecfg{composeitemproblemsep}%
```

```
288 {\getexercisecfg{composeitemsep}}
```

Compose problem title; arguments are problem number (empty if item is split off) and raw title (empty if not specified); default is “Problem #1.” or “#1. #2”:

```

289 \exerciseconfig{composetitleproblem}[2]{\exerciseifempty{#1}%
290   {\exerciseifempty{#2}{#2}}}%
291   {\exerciseifempty{#2}{\getexerciseconfig{termproblem}\ %
292     \getexerciseconfig{composeitemproblem}{#1}}}%
293   {\getexerciseconfig{composeitemproblem}{#1} #2}}

```

Compose problem title for table of contents; arguments are problem number and raw title:

```

294 \exerciseconfig{composetocproblem}[2]%
295   {\exerciseifempty{#2}{\getexerciseconfig{termproblem} #1}{#1. #2}}

```

Compose subproblem item; argument is subproblem number:

```

296 \exerciseconfig{composeitemsubproblem}[1]{#1}

```

Subproblem item separator:

```

297 \exerciseconfig{composeitemsubproblemsep}%
298   {\getexerciseconfig{composeitemsep}}

```

Compose subproblem title; argument is subproblem number:

```

299 \exerciseconfig{composetitlesubproblem}[1]{#1}

```

Compose solution item; argument is problem or subproblem number:

```

300 \exerciseconfig{composeitemsolution}[2]{#1.}
301 \exerciseconfig{composeitemsolutionsub}[2]{#2}

```

Solution item separator:

```

302 \exerciseconfig{composeitemsolutionsep}%
303   {\getexerciseconfig{composeitemsep}}

```

Compose title for single solution; arguments are corresponding problem and subproblem number:

```

304 \exerciseconfig{composetitlesolutionsingle}[2]%
305   {\getexerciseconfig{termsolution}:}

```

Compose title for one out of several solutions; arguments are corresponding problem and subproblem number:

```

306 \exerciseconfig{composetitlesolutionmulti}[2]{#2}

```

Compose table of contents line for solution; arguments are problem number and raw title:

```

307 \exerciseconfig{composetocsolution}[2]%
308   {\getexerciseconfig{composetocproblem}{#1}{#2}}

```

Compose title for solution block:

```

309 \exerciseconfig{composetitlesolutions}%
310   {\getexerciseconfig{termsolutions}}

```

Compose table of contents line for solution block:

```

311 \exerciseconfig{composetocsolutions}%
312   {\getexerciseconfig{composetitlesolutions}}

```

Compose sectional title for solution following a single problem; arguments are problem number and raw title:

```

313 \exerciseconfig{composetitlesolutionsproblemsingle}[2]%
314   {\getexerciseconfig{termsolution}}

```

Compose sectional title for solution of one problem within a block; arguments are problem number and raw title:

```
315 \exerciseconfig{composetitlesolutionsproblemmulti}[2]%
316 {\exerciseifempty{#2}{\getexerciseconfig{termproblem} #1}{#1. #2}}
```

Compose label:

```
317 \exerciseconfig{composeitemsolution}[2]{#1#2}
```

Points. Compose number of points:

```
318 \exerciseconfig{composepointsnum}[1]{#1}
```

Compose number of points followed by ‘points’; use singular ‘point’ for 1:

```
319 \exerciseconfig{composepoints}[1]{\getexerciseconfig{composepointsnum}{#1}~%
320 \ifdim #1pt=1pt\getexerciseconfig{termpoint}%
321 \else\getexerciseconfig{termpoints}\fi}
```

Compose points declaration for use in opening line:

```
322 \exerciseconfig{composepointsstart}[1]{(\getexerciseconfig{composepoints}{#1})}
```

Compose points declaration for use in margin:

```
323 \exerciseconfig{composepointsmargin}[1]{\getexerciseconfig{composepoints}{#1}}
```

Compose points declaration for use in text:

```
324 \exerciseconfig{composepointsbody}[1]{(\getexerciseconfig{composepoints}{#1})}
```

Compose points declaration for use in sheet data:

```
325 \exerciseconfig{composepointssheet}[1]{%
326 \exerciseifnotempty{#1}{\getexerciseconfig{composepoints}{#1}}}
```

Compose points declaration for solution with comment:

```
327 \exerciseconfig{composepointsaward}[2]%
328 {(\getexerciseconfig{composepoints}{#1}\exerciseifnotempty{#2}{; #2})}
```

Compose alternative points for solution declaration with comment:

```
329 \exerciseconfig{composepointsawardalt}[2]%
330 {(\getexerciseconfig{composepoints}{#1}*\exerciseifnotempty{#2}{; #2})}
```

Display points in the margin:

```
331 \exerciseconfig{insertpointsmargin}[1]{\marginpar{\footnotesize #1}}
```

Display warning about points mismatch:

```
332 \exerciseconfig{insertwarnpoints}[3]
333 {\textbf{points mismatch for #1 (#2 determined vs.\ #3 given)}}
```

Counters. Define counter display via configuration interface:

```
334 \exerciseconfig{countersheet}{\arabic{\exf@sheetcounter}}
335 \exerciseconfig{counterproblem}{\arabic{\exf@problemcounter}}
336 \exerciseconfig{counterproblemmax}{10}
337 \exerciseconfig{countersubproblem}{\alph{\exf@subproblemcounter}}
338 \exerciseconfig{countersubproblemmax}{m}
339 \exerciseconfig{countersheetequation}{\arabic{equation}}
340 \exerciseconfig{counterproblemequation}{P\arabic{equation}}
341 \exerciseconfig{countersolutionequation}{S\arabic{equation}}
```

Further Definitions.

`label...` Templates for automatic generation of labels from tags:

```
342 \exerciseconfig{labelsheet}[1]{sheet:#1}
343 \exerciseconfig{labelproblem}[1]{prob:#1}
```

`toclevel...` Table of contents levels for sheets, problems, solutions of problems and solution blocks; empty means no writing to table of contents:

```
344 \exerciseconfig{toclevelsheet}{}
345 \exerciseconfig{toclevelproblem}{}
346 \exerciseconfig{toclevelsolution}{}
347 \exerciseconfig{toclevelsolutions}{}
```

B.4 Styles

Styles are meant as a way to adjust several configuration options at the same time to achieve a consistent layout in some regard. Useful examples can be found among the extended exercise styles. They can serve a starting point for further custom styles.

Exercise Styles Code.

`defexercisestylearg` Define a style with an argument:

```
348 \newcommand{\defexercisestylearg}[3][ ]{%
349   \def\exf@tmp{#1}\ifx\exf@tmp\exf@empty%
350     \define@key{exf@style}{#2}{#3}\else%
351     \define@key{exf@style}{#2}[#1]{#3}\fi}
```

`\defexercisestyle` Define a style with a boolean argument; execute code only if true:

```
352 \newcommand{\defexercisestyle}[2]{%
353   \exf@csdotwo\long\def{exf@style@code@#1}{#2}%
354   \exf@exparg{\define@boolkey{exf@style}[exf@style@]{#1}[true]}%
355   {\csname ifexf@style@#1\endcsname\csname exf@style@code@#1\endcsname\fi}}
```

`\exercisestyle` Process styles:

```
356 \newcommand{\exercisestyle}[1]{\exf@expsetkeys{exf@style}{#1}}
```

Default Exercise Styles.

`solutionbelow` Choose location for solutions:

```
357 \def\exf@solutionbelow{subproblem}
358 \define@choicekey{exf@style}{solutionbelow}%
359   {here,subproblem,subproblem*,problem,problem*,sheet>manual}%
360   {\def\exf@solutionbelow{#1}}
```

Conditionals to compare solution locations:

```
361 \newcommand{\exf@solbelowis}[2]%
362   {\def\exf@tmp{#1}\ifx\exf@solutionbelow\exf@tmp#2\fi}
```

`sheetequation` Use separate equation counters for sheets, problems and solutions:
`problemequation`
`solutionequation`

```

363 \defexercisestyle{sheetequation}{}
364 \defexercisestyle{problemequation}{}
365 \defexercisestyle{solutionequation}{}
366 \exf@style@solutionequationtrue

```

pointsat Choose where points of problems and subproblems are displayed:
subpointsat

```

367 \def\exf@pointsat{start}
368 \define@choicekey{exf@style}{pointsat}%
369 {start,start*,margin,end>manual,off}{\def\exf@pointsat{#1}}
370 \def\exf@subpointsat{end}
371 \define@choicekey{exf@style}{subpointsat}%
372 {start,start*,margin,end>manual,off}{\def\exf@subpointsat{#1}}

```

Conditionals to compare point locations:

```

373 \newcommand{\exf@pointsatis}[2]%
374 {\def\exf@tmp{#1}\ifx\exf@pointsat\exf@tmp#2\fi}
375 \newcommand{\exf@subpointsatis}[2]%
376 {\def\exf@tmp{#1}\ifx\exf@subpointsat\exf@tmp#2\fi}

```

problemby Declare problems or equations as subcounter of other counter:
equationby

```

377 \defexercisestylearg{problemby}{\exf@numberproblemwithin{#1}}
378 \defexercisestylearg{equationby}{\exf@numberequationwithin{#1}}

```

pagebysheet Number pages, problems or equations by sheet:
problembysheet
equationbysheet

```

379 \defexercisestyle{pagebysheet}{}
380 \def\thepage{\csname the\exf@sheetcounter\endcsname.\arabic{page}}%
381 \def\theHpage{\csname theH\exf@sheetcounter\endcsname.\arabic{page}}%
382 \exerciseconfigappend{insertsheetbefore}{\setcounter{page}{1}}
383 \defexercisestyle{problembysheet}%
384 {\exf@numberproblemwithin{\exf@sheetcounter}}
385 \defexercisestyle{equationbysheet}%
386 {\exf@numberequationwithin{\exf@sheetcounter}}

```

fracpoints Use vulgar fractions to display binary fractional points:

```

387 \defexercisestyle{fracpoints}%
388 {\exerciseconfig{composepointsnum}[1]{\exf@nicefrac{##1}}}

```

twoside Use two-sided layout for sheets:

```

389 \defexercisestylearg[true]{twoside}{\exercisestop{twoside={#1}}}

```

Extended Exercise Styles. Declare more specific styles:

```

390 \ifexf@extstyle

```

contents Add sheets and problems to table of contents:

```

391 \defexercisestyle{contents}{}
392 \exerciseconfig{toclevelsheet}{section}%
393 \exerciseconfig{toclevelproblem}{subsection}}

```

solutionsf Use sans serif font for solutions:

```

394 \defexercisestyle{solutionsf}{}
395 \exerciseconfigappend{styletextsolution}{\sffamily\let\itshape\slshape}}

```

`solutiondimproblem` Dim problem text if solutions are displayed:

```
396 \defexercisestyle{solutiondimproblem}{%
397   \RequirePackage{color}%
398   \exerciseconfigappend{styletextsolution}{\color[gray]{0}}%
399   \exerciseconfigappend{styletextproblem}{\color[gray]{0.2}}}
```

`solutionsep` Separate solutions by horizontal lines:

```
400 \defexercisestyle{solutionsep}{%
401   \exerciseconfig{insertsolutionsbefore}{\hrule\nopagebreak[3]\vspace{0.5ex}}%
402   \exerciseconfig{insertsolutionsafter}{\nopagebreak[3]\vspace{1.0ex}\hrule}}
```

`plainheader` Declare a simple sheet header with some configurable options; the configuration options `styleheader...` define font styles, `skipheaderbelow` the space below the header and `composeheaderbelow...` some auxiliary text to be displayed on the line below the header:

```
403 \defexercisestyle{plainheader}{%
404   \exerciseconfig{styleheadertitle}{\Large\bfseries}%
405   \exerciseconfig{styleheadercourse}{\sffamily}%
406   \exerciseconfig{styleheaderbelow}{\footnotesize}%
407   \exerciseconfig{skipheaderbelow}{3ex}%
408   \exerciseconfig{composeheaderbelowleft}{}%
409   \exerciseconfig{composeheaderbelowright}{}%
410   \exerciseconfig{composeheaderbelowcenter}{}%
411   \exerciseconfig{insertsheettitle}{\noindent%
412     \begin{minipage}{\textwidth}%
413       {\getexercisestyle{styleheadertitle}}%
414       \makebox[0pt][l]{\getexercisedata{course}}%
415       \hfill\makebox[0pt][r]{\getsheetdata{title}}\par}%
416     {\getexercisestyle{styleheadercourse}}%
417     \makebox[0pt][l]{\getexercisedata{institution}}%
418     \exercisedataempty{period}{}{\getexercisedata{period}}}%
419     \hfill\makebox[0pt][r]{\getexercisedata{instructor}}%
420     \vphantom{g}\par}%
421   \hrule%
422   {\def\tmp{}}%
423   \exerciseconfigempty{composeheaderbelowleft}{}{\def\tmp{.}}%
424   \exerciseconfigempty{composeheaderbelowcenter}{}{\def\tmp{.}}%
425   \exerciseconfigempty{composeheaderbelowright}{}{\def\tmp{.}}%
426   \exerciseifnotempty{\tmp}%
427   {\getexercisestyle{styleheaderbelow}\vphantom{^A}%
428     \makebox[0pt][l]{\getexercisestyle{composeheaderbelowleft}}%
429     \hfill\makebox[0pt][c]{\getexercisestyle{composeheaderbelowcenter}}%
430     \hfill\makebox[0pt][r]{\getexercisestyle{composeheaderbelowright}}%
431     \vspace*{-\baselineskip}\vspace*{-\parskip}\par}%
432   \end{minipage}%
433   \par\addvspace{\getexercisestyle{skipheaderbelow}}}
```

Done with extended styles:

```
434 \fi
```

B.5 Metadata

Global Metadata Code.

`\defexercisedata` Declare global metadata field by defining a key *key* in category `exf@data` that stores the

chosen value in `\exf@data@key`:

```
435 \newcommand{\defexercisedata}[1]{%
436   \exf@csdo\def{exf@data@#1}{}%
437   \define@key{exf@data}{#1}%
438   {\exf@csdo\gdef{exf@data@#1}{##1}}}
```

`\exercisedata` Process key-value pairs:

```
439 \newcommand{\exercisedata}[1]{\setkeys{exf@data}{#1}}
```

`\getexercisedata` Read global metadata:

```
440 \newcommand{\getexercisedata}[1]{\csname exf@data@#1\endcsname}
```

`\exercisedataempty` Check whether the field is empty:

```
441 \newcommand{\exercisedataempty}[3]{\exf@csdo\ifx{exf@data@#1}\exf@empty%
442   #2\else#3\fi}
```

Global Metadata Declarations. Declare fields corresponding to standard pdf metadata:

```
443 \defexercisedata{author}
444 \defexercisedata{title}
445 \defexercisedata{subject}
446 \defexercisedata{keyword}
```

Declare additional general purpose fields:

```
447 \defexercisedata{date}
```

Declare metadata related to courses:

```
448 \defexercisedata{instructor}
449 \defexercisedata{course}
450 \defexercisedata{institution}
451 \defexercisedata{period}
452 \defexercisedata{material}
```

Overwrite standard definitions for `author`, `title`, `date` to also fill ordinary L^AT_EX structures:

```
453 \define@key{exf@data}{author}{\gdef\exf@data@author{#1}\author{#1}}
454 \define@key{exf@data}{title}{\gdef\exf@data@title{#1}\title{#1}}
455 \define@key{exf@data}{date}{\gdef\exf@data@date{#1}\date{#1}}
```

Sheet Metadata.

`\defsheetdata` Declare sheet metadata field by defining a key *key* in category `exf@sheet` that stores the chosen value in `\exf@data@sheet@key`:

```
456 \newcommand{\defsheetdata}[1]{%
457   \exf@csdo\def{exf@data@sheet@#1}{}%
458   \define@key{exf@sheet}{#1}%
459   {\exf@csdo\def{exf@data@sheet@#1}{##1}}}
```

`\getsheetdata` Read global metadata:

```
460 \newcommand{\getsheetdata}[1]{\csname exf@data@sheet@#1\endcsname}
```

`\sheetdataempty` Check whether the field is empty:

```
461 \newcommand{\sheetdataempty}[3]{\ifx\csdo\ifx{exf@data@sheet@#1}\exf@empty%
462 #2\else#3\fi}
```

Declare general purpose fields:

```
463 \defsheetdata{due}
464 \defsheetdata{handout}
465 \defsheetdata{editdate}
466 \defsheetdata{author}
467 \defsheetdata{editor}
```

Special title processing:

```
468 \def\exf@data@sheet@rawtitle{}
469 \define@key{exf@sheet}{title}{\def\exf@data@sheet@rawtitle{#1}}
470 \def\exf@data@sheet@title{\exf@config@composetitlesheet%
471 {\csname the\exf@sheetcounter\endcsname}{\exf@data@sheet@rawtitle}}%
```

Special points processing:

```
472 \def\exf@data@sheet@rawpoints{}
473 \def\exf@data@sheet@points{\exf@config@composepointssheet%
474 {\exf@data@sheet@rawpoints}}%
```

Problem Metadata.

`\defproblemdata` Declare sheet metadata field by defining a key *key* in category `exf@problem` that stores the chosen value in `\exf@data@problem@key`:

```
475 \newcommand{\defproblemdata}[1]{%
476 \exf@csdo\def{exf@data@problem@#1}{}}%
477 \define@key{exf@problem}{#1}%
478 {\exf@csdo\def{exf@data@problem@#1}{##1}}}
```

`\getproblemdata` Read global metadata:

```
479 \newcommand{\getproblemdata}[1]{\csname exf@data@problem@#1\endcsname}
```

`\problemdataempty` Check whether the field is empty:

```
480 \newcommand{\problemdataempty}[3]{\ifx\csdo\ifx{exf@data@problem@#1}\exf@empty%
481 #2\else#3\fi}
```

Special title processing:

```
482 \def\exf@data@problem@rawtitle{}
483 \define@key{exf@problem}{title}{\def\exf@data@problem@rawtitle{#1}}
484 \def\exf@data@problem@title{\exf@config@composetitleproblem%
485 \csname the\exf@problemcounter\endcsname}{\exf@data@problem@rawtitle}}%
```

Problem Block Code.

`\exf@addmargin` Define a length for block margin:

```
486 \newlength\exf@addmargin
```

`\exf@section` Write out problem opening line followed by some amount of skip (positive dimensions add vertical space, negative dimensions add horizontal space); protected expand argument if

in horizontal mode (because it will be held until text is output and some definitions may become invalid):

```

487 \newcommand{\exf@section}[2]{\setlength\exf@tmpdim{#1}%
488 \ifdim\exf@tmpdim<0pt%
489 \protected@edef\exf@tmp{#2}%
490 \else%
491 \def\exf@tmp{#2}%
492 \fi%
493 \exf@exparg{\@startsection{}{}{0pt}{0pt}{#1}{*}{\exf@tmp}}

```

`\exf@init@block` Clean info buffer, define amount of skip between items:

```

494 \newcommand{\exf@init@block}[1]{%
495 \def\exf@intro{}\def\exf@intro@skip{#1}%
496 \exf@addmarginOpt\def\exf@introitem{}}

```

`\exf@append@intro` Append to info buffer:

```

497 \newcommand{\exf@append@intro}[1]{%
498 {\exf@append@def\exf@intro{#1\hspace{\exf@intro@skip}}}

```

`\exf@prepend@intro` Prepend to info buffer:

```

499 \newcommand{\exf@prepend@intro}[1]{%
500 {\exf@prepend@def\exf@intro{#1\hspace{\exf@intro@skip}}}

```

`\exf@open@block` Open block, set margin, compose opening line:

```

501 \newcommand{\exf@open@block}[1]{%
502 \advance\leftskip\exf@addmargin%
503 \advance\linewidth-\exf@addmargin%
504 \advance\@totalleftmargin\exf@addmargin%
505 \ifx\exf@intro\exf@empty%
506 \exf@section{Opt}{\exf@introitem}%
507 \else%
508 \exf@section{#1}{\exf@introitem\exf@intro\unskip}%
509 \fi}%

```

`\exf@close@block` Close block, undo margin:

```

510 \newcommand{\exf@close@block}{%
511 \advance\leftskip-\exf@addmargin%
512 \advance\linewidth\exf@addmargin%
513 \advance\@totalleftmargin-\exf@addmargin}%

```

`\addprobleminfo` Interface to append or prepend to info buffer:

```

514 \newcommand{\addprobleminfo}{\@ifstar\exf@prepend@intro\exf@append@intro}

```

`\exf@addinfoswitch` Add a switch for displaying problem info:

```

515 \newcommand{\exf@addinfoswitch}[1]{%
516 {\define@boolkey{exf@infoswitch}[exf@showdata@]{#1}[true]{} }

```

`\defprobleminfo` Declare a problem info field, add corresponding info switch, process key-value pair if switch activated:

```

517 \newcommand{\defprobleminfo}[2]{%

```

```

518 \exf@addinfoswitch{#1}%
519 \exerciseconfig{compose@probleminfo@#1}[1]{#2}%
520 \exf@exparg{\define@key{exf@probleminfo}{#1}}%
521 {\csname ifexf@showdata@#1\endcsname\exf@append@intro{%
522 \csname exf@config@compose@probleminfo@#1\endcsname{##1}}\fi}}

```

`\showprobleminfo` Process info switches, expand argument first:

```

523 \newcommand{\showprobleminfo}[1]{\exf@expsetkeys{exf@infoswitch}{#1}}

```

Problem Info Declarations. Declare general purpose fields:

```

524 \defprobleminfo{optional}{\emph{#1:}}
525 \showprobleminfo{optional}
526 \defprobleminfo{difficulty}{( #1 )}

```

Declare fields for internal information (mostly):

```

527 \defprobleminfo{comment}{#1}
528 \defprobleminfo{author}{$\langle$#1$\rangle$}
529 \defprobleminfo{editor}{$\{#1\}$}
530 \defprobleminfo{source}{[#1]}
531 \defprobleminfo{keyword}{\#( #1 )}

```

Declare more specific fields:

```

532 \ifexf@extdata
533 \defprobleminfo{review}{#1}
534 \defprobleminfo{recycle}{[[#1]]}
535 \defprobleminfo{timesolve}{\{#1\}}
536 \defprobleminfo{timepresent}{\{!\{#1\}!\}}
537 \fi

```

Write Metadata to PDF Files.

`\exf@writemetadata` Write Metadata to PDF Files in case hyperref is available, choose between global and sheet version, pass through unfilled values:

```

538 \newcommand{\exf@writemetadata}[1]{%
539 \ifdefined\hypersetup%
540 \def\exf@tmp{#1}\ifx\exf@tmp\exf@empty%

```

Write global author and title:

```

541 \ifx\exf@data@author\exf@empty\else%
542 \hypersetup{pdfauthor={\exf@data@author}}\fi%
543 \ifx\exf@data@title\exf@empty\else%
544 \hypersetup{pdftitle={\exf@data@title}}\fi%
545 \else%

```

Write sheet author (if available) and compose sheet title:

```

546 \ifx\exf@data@sheet@author\exf@empty%
547 \ifx\exf@data@author\exf@empty\else%
548 \hypersetup{pdfauthor={\exf@data@author}}\fi%
549 \else\hypersetup{pdfauthor={\exf@data@sheet@author}}\fi%
550 \hypersetup{pdftitle={\exf@config@composemetasheet%
551 {\csname the\exf@sheetcounter\endcsname}{\exf@data@sheet@rawtitle}}}%
552 \fi%

```

Write subject and keywords:

```
553 \ifx\exf@data@subject\exf@empty\else%
554 \hypersetup{pdfsubject={\exf@data@subject}}\fi%
555 \ifx\exf@data@keyword\exf@empty\else%
556 \hypersetup{pdfkeywords={\exf@data@keyword}}\fi%
557 \fi%
```

Do not write again:

```
558 \gdef\exf@metadata{off}}
```

Automatic writing at `\begin{document}`:

```
559 \AtBeginDocument{\def\exf@tmp{auto}\ifx\exf@metadata\exf@tmp%
560 \exf@writemetadata{}}\fi}
```

`\writeexercisedata`

Write metadata manually:

```
561 \newcommand{\writeexercisedata}{\def\exf@tmp{manual}\ifx\exf@metadata\exf@tmp%
562 \exf@writemetadata{}}\fi}
```

B.6 Counters

sheet
problem
subproblem
solution

Define main counters (with customised names if necessary) and equation counters:

```
563 \newcounter{\exf@sheetcounter}
564 \newcounter{\exf@problemcounter}
565 \newcounter{\exf@subproblemcounter}[\exf@problemcounter]
566 \newcounter{\exf@solutioncounter}[\exf@problemcounter]
567 \newcount\exf@eqsav
568 \newcounter{exf@sheetequation}
569 \newcounter{exf@problemequation}
570 \newcounter{exf@solutionequation}
```

Implement counter display; take care of corresponding hyperref labels:

```
571 \exf@csdo\def{the\exf@sheetcounter}{\exf@config@countersheet}
572 \exf@csdo\def{the\exf@problemcounter}{\exf@config@counterproblem}
573 \exf@csdo\def{the\exf@subproblemcounter}{\exf@config@countersubproblem}
574 \def\theexf@sheetequation{\exf@config@countersheetequation}
575 \def\theHexf@sheetequation{sheet.\arabic{equation}}
576 \def\theexf@problemequation{\exf@config@counterproblemequation}
577 \def\theHexf@problemequation{prob.\arabic{equation}}
578 \def\theexf@solutionequation{\exf@config@countersolutionequation}
579 \def\theHexf@solutionequation{sol.\arabic{equation}}
```

`numberproblemwithin`

Declare problem counter as subcounter of #1:

```
580 \newcommand{\exf@numberproblemwithin}[1]{%
581 \@addtoreset{\exf@problemcounter}{#1}%
582 \exf@csdo\def{the\exf@problemcounter}{%
583 {\csname the#1\endcsname.\exf@config@counterproblem}}}
```

`numberequationwithin`

Declare various new equation counters as subcounter of #1; take care of corresponding hyperref labels:

```
584 \newcommand{\exf@numberequationwithin}[1]{%
585 \@addtoreset{exf@sheetequation}{#1}%
586 \def\theexf@sheetequation%
```

```

587  {\csname the#1\endcsname.\exf@config@countersheetequation}%
588  \def\theHexf@sheetequation%
589  {\csname theH#1\endcsname.sheet.\arabic{equation}}}%
590  \@addtoreset{exf@problemequation}{#1}%
591  \def\theexf@problemequation%
592  {\csname the#1\endcsname.\exf@config@counterproblemequation}%
593  \def\theHexf@problemequation%
594  {\csname theH#1\endcsname.prob.\arabic{equation}}}%
595  \@addtoreset{exf@solutionequation}{#1}%
596  \def\theexf@solutionequation%
597  {\csname the#1\endcsname.\exf@config@countersolutionequation}%
598  \def\theHexf@solutionequation%
599  {\csname theH#1\endcsname.sol.\arabic{equation}}}}

```

B.7 Buffers

File Output.

```

\ifexf@infile Conditional whether an output file is presently in use:
    600 \newif\ifexf@infile\exf@infilefalse

\exf@out Reserve file handle:
    601 \newwrite\exf@out

\exf@writeline Write a line to the file:
    602 \newcommand{\exf@writeline}[1]{\immediate\write\exf@out{#1}}

\exf@linesep Return a separator line:
    603 \newcommand{\exf@linesep}%
    604  {\@percentchar-----}

\exf@lineno Return current position in source file; display line number and source file name (if available
via package currfile):
    605 \newcommand{\exf@lineno}{\@percentchar%
    606  \ifdefined\currfilename\currfilename\space\fi%
    607  1.\the\inputlineno}

\exf@startfile Open a new file #1.sol (do nothing if already open); indicate source, switch to manual
solution display mode:
    608 \newcommand{\exf@startfile}[1]{%
    609  \ifexf@infile\else%
    610  \global\exf@infiletrue%
    611  \gdef\exf@solutionbelow{manual}%
    612  \edef\exf@tmp{#1}%
    613  \immediate\openout\exf@out\exf@tmp.sol%
    614  \exf@writeline{\@percentchar%
    615  generated from file '\jobname' by exframe.sty}%
    616  \ifexf@lineno\exf@writeline{\exf@lineno}\fi%
    617  \exf@writeline{}%
    618  \fi}

```

`\exf@closefile` Close file (if open); indicate position, close and reset variables:

```
619 \newcommand{\exf@closefile}{%
620   \ifexf@infile%
621   \ifexf@lineno%
622     \exf@writeline{\exf@linesep}\exf@writeline{\exf@lineno}\fi%
623   \exf@writeline{@backslashchar endinput}%
624   \immediate\closeout\exf@out%
625   \global\exf@infilefalse%
626   \fi}
```

Make sure to properly close file at the end:

```
627 \AtEndDocument{\exf@closefile}
```

Solution Buffer.

`\exf@buf` Declare a token buffer for storing the solutions and a conditional indicating whether the
`\ifexf@bufclean` buffer has been used:

```
628 \newtoks\exf@buf
629 \newif\ifexf@bufclean\exf@bufcleantrue
```

`\exf@clearbuf` Clear solution buffer and mark clean:

```
630 \def\exf@clearbuf{\global\exf@bufcleantrue\global\exf@buf={}}
```

`\exf@append@buf` Append tokens to buffer:

```
631 \def\exf@append@buf#1{\global\exf@buf=\expandafter{\the\exf@buf#1}}
```

`\exf@addbufline` Add a protected expanded line to the buffer:

```
632 \def\exf@addbufline#1{{\protected@edef\exf@tmp{#1}%
633   \exf@exptwo\exf@append@buf{\exf@tmp^^J}}}
```

`\exf@sourcebuf` Source the solution buffer into the document:

```
634 \def\exf@sourcebuf{\exf@exptwo\scantokens{\the\exf@buf}}
```

`\exf@writebuf` Write the buffer into the solution file:

```
635 \def\exf@writebuf{\exf@writeline{\the\exf@buf}}
```

`\exf@verbatim` Start reading the buffer from the environment body:

```
636 \newcommand{\exf@verbatim}{%
637   \begingroup%
638   \@bsphack%
639   \let\do\@makeother\dospecials%
640   \catcode'\^^M\active%
641   \def\verbatim@processline{\exf@exptwo\exf@append@buf{\the\verbatim@line^^J}}%
642   \verbatim@start}
```

`\exf@endverbatim` Stop reading the buffer:

```
643 \newcommand{\exf@endverbatim}{\@esphack\endgroup}
```

`\exf@scanblock` Scan an optional argument from a verbatim block; allow for an empty block and an empty first line; argument #1 is macro to be called eventually:

```
644 \def\exf@scanblock#1{%
```

Check for empty first line:

```
645 \@ifnextchar\par{\exf@scanblock@par{#1}}{\exf@scanblock@sel{#1}}
```

Handle empty first line, implies no optional argument:

```
646 \long\def\exf@scanblock@par#1\par{\exf@scanblock@sel{#1} []}
```

Check for optional argument ([]) and for block ending (\end):

```
647 \def\exf@scanblock@sel#1{\ifnextchar[{\exf@scanblock@opt{#1}}%
648 {\ifnextchar\end{\exf@scanblock@end{#1}}{\exf@scanblock@noopt{#1}}}}
```

Handle empty block, hopefully environment matches (otherwise?!):

```
649 \def\exf@scanblock@end#1\end#2{%
650 \def\exf@tmp{#2}\ifx\exf@tmp\currenenvir%
651 \def\exf@verbatim{\def\exf@endverbatim}%
652 \fi%
653 #1{\scantokens{\end{#2}}}}
```

Pass on without and with optional argument; pass on optional argument and any token scanned prematurely:

```
654 \def\exf@scanblock@noopt#1#2{#1{\scantokens#2}}
655 \def\exf@scanblock@opt#1[#2]{#1{#2}}
```

B.8 Points

Binary Rational Numbers.

`\exf@splitsign` Split a decimal float number into sign, integer and fractional part:
`\exf@splitdecimal`

```
656 \def\exf@splitsign#1-#2-#3&{\def\exf@splitnum{#1#2}\def\exf@splitminus{#3}}
657 \def\exf@splitdecimal#1.#2.#3&{\def\exf@splitint{#1}\def\exf@splitdec{#2}}
```

`\exf@nicefrac` Display a float number as a fraction with denominators 2, 4 or 8 when possible; first split number, complete missing zeros and handle cases:

```
658 \newcommand{\exf@nicefrac}[1]{%
659 \edef\exf@tmp{#1}%
660 \expandafter\exf@splitsign\exf@tmp--&%
661 \expandafter\exf@splitdecimal\exf@splitnum. .&%
662 \if&\exf@splitint&\def\exf@splitint{0}\fi%
663 \if&\exf@splitdec&\def\exf@splitdec{0}\fi%
664 \def\exf@tmp{\exf@splitint.\exf@splitdec}%
665 \ifnum\exf@splitdec=0\def\exf@tmp{\exf@splitint}\fi%
666 \ifnum\exf@splitdec=5\def\exf@tmp{\exf@intfrac{\exf@splitint}{1}{2}}\fi%
667 \ifnum\exf@splitdec=25\def\exf@tmp{\exf@intfrac{\exf@splitint}{1}{4}}\fi%
668 \ifnum\exf@splitdec=75\def\exf@tmp{\exf@intfrac{\exf@splitint}{3}{4}}\fi%
669 \ifnum\exf@splitdec=125\def\exf@tmp{\exf@intfrac{\exf@splitint}{1}{8}}\fi%
670 \ifnum\exf@splitdec=375\def\exf@tmp{\exf@intfrac{\exf@splitint}{3}{8}}\fi%
671 \ifnum\exf@splitdec=625\def\exf@tmp{\exf@intfrac{\exf@splitint}{5}{8}}\fi%
672 \ifnum\exf@splitdec=875\def\exf@tmp{\exf@intfrac{\exf@splitint}{7}{8}}\fi%
673 \ifx\exf@splitminus\exf@empty\else$\exf@splitminus$\fi\exf@tmp%
674 }
```


`\exf@intfrac` Display a vulgar fraction such as $12^3/4$:

```
675 \newcommand{\exf@intfrac}[3]{%
676   \ifnum#1=0\else#1\fi%
677   \ifnum#2=0\else$%
678   ^{\exf@text{#2}}}%
679   \mskip-4mu\mskip-2mu%
680   _{\exf@text{#3}}$\fi}
```

Sheet Points Code.

`\exf@sheet@points@dim` Declare a metric register to sum overall points on a sheet; equate unit pt to a point:

```
681 \newlength{\exf@sheet@points@dim}
```

`\exf@notesheetpoints` Store a sheet point number in a macro:

```
682 \newcommand{\exf@notesheetpoints}[2]{%
683   \exf@csdo\gdef{\exf@sheetpoints@#1}{#2}}
```

Make sure the macros in code written to the .aux file exist:

```
684 \AtBeginDocument{\immediate\write\@auxout{%
685   \string\providecommand{\string\exf@notesheetpoints}[2]{}}}
```

`\exf@writsheetpoints` Write sheet points to the .aux file:

```
686 \newcommand{\exf@writsheetpoints}[1]{%
687   {\immediate\write\@auxout{\string\exf@notesheetpoints{\sheettag}{#1}}}}
```

`\getsheetpoints` Read sheet points stored in .aux file:

```
688 \newcommand{\getsheetpoints}[1]{\ifcsname exf@sheetpoints@#1\endcsname%
689   \csname exf@sheetpoints@#1\endcsname\else 0\fi}
```

Problem Points Code.

`\exf@problem@points@dim` Declare a metric register to sum overall points in a problem; equate unit pt to a point:

```
690 \newlength{\exf@problem@points@dim}
```

`\exf@noteproblempoints` Store a problem point number in a macro:

```
691 \newcommand{\exf@noteproblempoints}[2]{%
692   \exf@csdo\gdef{\exf@problempoints@#1}{#2}}
```

Make sure the macros written to the .aux file exist:

```
693 \AtBeginDocument{\immediate\write\@auxout{%
694   \string\providecommand{\string\exf@noteproblempoints}[2]{}}}
```

`\exf@writeproblempoints` Write problem points to the .aux file:

```
695 \newcommand{\exf@writeproblempoints}[1]{%
696   {\immediate\write\@auxout{\string\exf@noteproblempoints{\problemtag}{#1}}}}
```

`\getproblempoints` Read problem points stored in .aux file:

```
697 \newcommand{\getproblempoints}[1]{\ifcsname exf@problempoints@#1\endcsname%
698   \csname exf@problempoints@#1\endcsname\else 0\fi}
```

`\showpoints` Show points within a problem or subproblem:

```
699 \newcommand{\showpoints}{%
700   \ifdefined\exfin@subproblem%
701     \ifdefined\exf@subproblem@points%
702       \exf@ensuretext{\exf@config@composepointsbody{\exf@subproblem@points}}%
703       \global\let\exf@subproblem@points\@undefined%
704     \fi%
705   \else\ifdefined\exfin@problem%
706     \ifdefined\exf@problem@pointsshow%
707       \exf@ensuretext{\exf@config@composepointsbody{\exf@problem@pointsshow}}%
708       \global\let\exf@problem@pointsshow\@undefined%
709     \fi%
710   \fi\fi}
```

Solution Points Code.

`\solution@points@dim` Declare a metric register to sum overall (mandatory) points in a solution; equate unit `pt` to a point:

```
711 \newlength{\exf@solution@points@dim}
```

`\exf@awardpointsalt` Award points for alternative or optional solution; does not count towards solution total:

```
712 \newcommand{\exf@awardpointsalt}[2] [] {%
713   \exf@ensuretext{\exf@config@composepointsawardalt{#2}{#1}}}
```

`\exf@awardpointsreg` Award points for regular solution; counts towards solution total:

```
714 \newcommand{\exf@awardpointsreg}[2] [] {%
715   \global\addtolength{\exf@solution@points@dim}{#2 pt}%
716   \exf@ensuretext{\exf@config@composepointsaward{#2}{#1}}}
```

`\awardpoints` Award points within solution with optional starred form:

```
717 \newcommand{\awardpoints}{\@ifstar\exf@awardpointsalt\exf@awardpointsreg}
```

B.9 Environments

Sheets.

`exf@sheet` Define options for `sheet` environment:

```
718 \define@key{exf@sheet}{points}{\def\exf@points@given{#1}}
719 \define@key{exf@sheet}{number}{\setcounter{\exf@sheetcounter}{#1}}
720 \define@key{exf@sheet}{label}{\def\exf@label{#1}}
721 \define@key{exf@sheet}{tag}{\def\sheettag{#1}}
```

`sheet` Define `sheet` environment (potentially using custom name):

```
722 \newenvironment{\exf@sheetname}[1] [] {%
```

Insert hook code to clear page, step counter:

```
723   \exf@config@insertsheetclearpage%
724   \refstepcounter{\exf@sheetcounter}%
```

Use equation counter for sheets:

```
725 \ifexf@style@sheetequation%
726 \exf@eqsav\value{equation}%
727 \setcounter{equation}{\value{exf@sheetequation}}%
728 \let\theequation\theexf@sheetequation%
729 \let\theHequation\theHexf@sheetequation%
730 \fi%
```

Reset optional arguments, process arguments:

```
731 \let\exf@points@given@undefined%
732 \def\sheettag{\csname the\exf@sheetcounter\endcsname}%
733 \setlength{\exf@sheet@points@dim}{0pt}%
734 \let\exf@label@undefined%
735 \setkeys{exf@sheet}{#1}%
```

Process automatic and manual labels:

```
736 \ifexf@autolabelsheet\label{\exf@config@labelsheet{\sheettag}}\fi%
737 \ifdefined\exf@label\label{\exf@label}\fi%
```

Set points from explicit input or from .aux storage:

```
738 \ifdefined\exf@points@given%
739 \let\exf@data@sheet@rawpoints\exf@points@given%
740 \else\ifcsname exf@sheetpoints@\sheettag\endcsname%
741 \exf@csdotwo\let\exf@data@sheet@rawpoints{exf@sheetpoints@\sheettag}%
742 \fi\fi%
```

Process metadata:

```
743 \def\exf@tmp{sheet}\ifx\exf@metadata\exf@tmp%
744 \exf@writemetadata{sheet}\fi%
```

Insert hook code:

```
745 \exf@config@insertsheetbefore%
```

Add table of contents line:

```
746 \ifx\exf@config@toclevelsheet\exf@empty\else%
747 \ifdefined\phantomsection\phantomsection\fi\fi%
748 \exf@addcontentsline{\exf@config@toclevelsheet}%
749 {\exf@config@composetocsheet{\csname the\exf@sheetcounter\endcsname}%
750 {\exf@data@sheet@rawtitle}}%
```

Write sheet title:

```
751 \exf@config@insertsheettitle%
```

End of environment; perform sanity check on total points if given explicitly:

```
752 {\ifdefined\exf@points@given%
753 \ifdim\exf@sheet@points@dim=0pt\else%
754 \ifdim\exf@sheet@points@dim=\exf@data@sheet@rawpoints pt\else%
755 \PackageWarning{exframe}{points mismatch for %
756 \exf@sheetname\space\csname the\exf@sheetcounter\endcsname}%
757 \ifexf@warntext\exf@config@insertwarnpoints{\exf@sheetname}%
758 {\strip@pt\exf@sheet@points@dim}{\exf@data@sheet@rawpoints}}\fi%
759 \fi\fi%
```

Test whether points have changed since last compile:

```

760 \else%
761 \ifx\exf@data@sheet@rawpoints\exf@empty\else%
762 \ifdim\exf@sheet@points@dim=\exf@data@sheet@rawpoints pt\else%
763 \PackageWarning{exframe}{points changed for %
764 \exf@sheetname\space\csname the\exf@sheetcounter\endcsname;
765 rerun to fix}%
766 \ifexf@warntext\exf@config@insertwarnpoints{\exf@sheetname}%
767 {\strip@pt\exf@sheet@points@dim}{\exf@data@sheet@rawpoints}\fi%
768 \fi\fi%

```

Store points if not given explicitly:

```

769 \ifdim\exf@sheet@points@dim=0pt%
770 \def\exf@data@sheet@rawpoints{}%
771 \else%
772 \edef\exf@data@sheet@rawpoints{\strip@pt\exf@sheet@points@dim}%
773 \fi%
774 \fi%

```

Write sheet points to .aux file:

```

775 \ifx\exf@data@sheet@rawpoints\exf@empty\else%
776 \exf@writeshheetpoints{\exf@data@sheet@rawpoints}%
777 \fi%

```

Insert solutions:

```

778 \exf@solbelowis{sheet}{\insertsolutions}%

```

Insert hook code:

```

779 \exf@config@insertsheetafter%
780 \exf@config@insertsheetclearpage%

```

Restore original equation counter:

```

781 \ifexf@style@sheetequation%
782 \setcounter{exf@sheetequation}{\value{equation}}%
783 \setcounter{equation}{\exf@eqsav}%
784 \fi%

```

Done:

```

785 \ignorespacesafterend}

```

`exercisecleardoublepage` Clear the current page, clear even page with a totally empty page:

```

786 \newcommand{\exercisecleardoublepage}{%
787 \clearpage\ifexf@twoside\ifodd\value{page}\else%
788 \thispagestyle{empty}\hbox{} \newpage\fi\fi}

```

Problems.

`exf@problem` Define options for problem environment:

```

789 \define@key{exf@problem}{points}{\def\exf@points@given{#1}}
790 \define@key{exf@problem}{label}{\def\exf@label{#1}}
791 \define@key{exf@problem}{tag}{\def\problemtag{#1}}
792 \define@key{exf@problem}{sollabel}{\xdef\exf@sollabel{#1}}

```

`problem` Define problem environment (potentially using custom name):

```

793 \newenvironment{\exf@problemname}[1][\fi]{%

```

Start with new paragraph, set text style, add vspace:

```
794 \par\exf@config@styletext\addvspace{\exf@config@skipproblemabove}%
```

Step problem counter:

```
795 \refstepcounter{\exf@problemcounter}%
```

Insert hook code:

```
796 \exf@config@insertproblembefore%
```

Begin inner group, mark in problem:

```
797 \begingroup%
798 \def\exf@in@problem{}
```

Use equation counter for problems:

```
799 \ifexf@style@problemequation%
800 \exf@eqsav\value{equation}%
801 \setcounter{equation}{\value{exf@problemequation}}%
802 \let\theequation\theexf@problemequation%
803 \let\theHequation\theHexf@problemequation%
804 \fi%
```

Initialise variables, process arguments:

```
805 \exf@init@block{\exf@config@skipprobleminfo}%
806 \def\problemtag{\csname the\exf@problemcounter\endcsname}%
807 \let\exf@points@given\undefined%
808 \let\exf@label\undefined%
809 \global\let\exf@sollabel\undefined%
810 \setlength{\exf@problem@points@dim}{0pt}%
811 \setkeys{exf@problem,exf@probleminfo}{#1}%
```

Process automatic and manual labels:

```
812 \ifexf@autolabelproblem\label{\exf@config@labelproblem{\problemtag}}\fi%
813 \ifdefined\exf@label\label{\exf@label}\fi%
```

Mark for new solution section; remember problem counter, title:

```
814 \gdef\exf@problem@solnewsec{}%
815 \xdef\exf@prevprob{\csname the\exf@problemcounter\endcsname}%
816 \ifcsname theH\exf@problemcounter\endcsname%
817 \xdef\exf@prevprobhref{\exf@problemcounter.%
818 \csname theH\exf@problemcounter\endcsname}%
819 \fi%
820 \ifx\exf@data@problem@rawtitle\exf@empty%
821 \global\let\exf@prevprobttitle\undefined%
822 \else%
823 \protected@xdef\exf@prevprobttitle{\exf@data@problem@rawtitle}%
824 \fi%
825 \global\let\exf@prevsubprob\undefined%
826 \global\let\exf@prevsubprobhref\undefined%
```

Set points from explicit input or from .aux storage:

```
827 \let\exf@problem@points\undefined%
828 \ifdefined\exf@points@given%
829 \let\exf@problem@points\exf@points@given%
830 \else%
831 \ifcsname exf@problempoints@\problemtag\endcsname%
```

```

832 \ex@csdotwo\let\ex@problem@points{ex@problempoints@\problemtag}%
833 \fi\fi%
834 \global\let\ex@prevpoints\ex@problem@points%
835 \let\ex@problem@pointsshow\@undefined%
836 \ifdefined\ex@problem@points\ifdim\ex@problem@points pt=0pt\else%
837 \let\ex@problem@pointsshow\ex@problem@points%
838 \fi\fi%

```

Disable points display if desired:

```

839 \ex@pointsatis{off}{\let\ex@problem@pointsshow\@undefined}%

```

Display points in opening line if desired; expand points into argument and remove points:

```

840 \ex@pointsatis{start}{\ifdefined\ex@problem@pointsshow%
841 \ex@exptwo\ex@append@intro{\expandafter%
842 \ex@config@composepointsstart\expandafter{\ex@problem@pointsshow}}}%
843 \let\ex@problem@pointsshow\@undefined\fi}%
844 \ex@pointsatis{start*}{\ifdefined\ex@problem@pointsshow%
845 \ex@exptwo\ex@prepend@intro{\expandafter%
846 \ex@config@composepointsstart\expandafter{\ex@problem@pointsshow}}}%
847 \let\ex@problem@pointsshow\@undefined\fi}%

```

Insert hook code, set problem body style:

```

848 \ex@config@insertprobleminfo%
849 \ex@config@styletextproblem%

```

Write title without item:

```

850 \ifdim\ex@config@skipproblemitem=0pt%
851 \ex@prepend@intro{%
852 \ex@config@styletitle\ex@config@styletitleproblem%
853 \ex@config@composetitleproblem{\csname the\ex@problemcounter\endcsname}%
854 {\ex@data@problem@rawtitle}}}%

```

Write item with fixed total width or item width plus space:

```

855 \else%
856 \ifdim\ex@config@skipproblemitem>0pt%
857 \setlength\ex@addmargin{\ex@config@skipproblemitem}%
858 \else%
859 \settowidth\ex@addmargin{%
860 \ex@config@styletitle\ex@config@styletitleproblem%
861 \ex@config@composeitemproblem{\ex@config@counterproblemmax}%
862 \ex@config@composeitemproblemsep}%
863 \fi%

```

Define item label:

```

864 \def\ex@introitem{\makebox[0cm][r]{%
865 \ex@config@styletitle\ex@config@styletitleproblem%
866 \ex@config@composeitemproblem{\csname the\ex@problemcounter\endcsname}%
867 \ex@config@composeitemproblemsep}}}%

```

Compose title:

```

868 \ifx\ex@data@problem@rawtitle\ex@empty\else%
869 \ex@prepend@intro{%
870 \ex@config@styletitle\ex@config@styletitleproblem%
871 \ex@config@composetitleproblem{\ex@empty}{\ex@data@problem@rawtitle}}}%
872 \fi%
873 \fi%

```

Write points into margin if desired; expand points into argument and remove points:

```
874 \exf@pointsatis{margin}{\ifdefined\exf@problem@pointsshow%
875 \expandafter\exf@prepend@def\expandafter\exf@intro\expandafter%
876 {\expandafter\protect\expandafter%
877 \exf@config@insertpointsmargin\expandafter{\expandafter%
878 \exf@config@composepointsmargin\expandafter{\exf@problem@pointsshow}}}%
879 \let\exf@problem@pointsshow\@undefined\fi}%
```

Write out opening line:

```
880 \exf@open@block{\exf@config@skipproblemtitle}%
```

Add table of contents line:

```
881 \exf@addcontentsline{\exf@config@toclevelproblem}%
882 {\exf@config@composetocproblem{\csname the\exf@problemcounter\endcsname}%
883 {\exf@data@problem@rawtitle}}%
```

Done:

```
884 \@afterindentfalse}%
```

End environment, show points if desired:

```
885 {\exf@pointsatis{end}{\showpoints}%
```

Perform sanity checks on total points if given explicitly:

```
886 \ifdefined\exf@points@given%
887 \ifdim\exf@problem@points@dim=0pt\else%
888 \ifdim\exf@problem@points@dim=\exf@problem@points pt\else%
889 \PackageWarning{exframe}{points mismatch for %
890 \exf@problemname\space\csname the\exf@problemcounter\endcsname}%
891 \ifexf@warntext\exf@config@insertwarnpoints{\exf@problemname}%
892 {\strip@pt\exf@problem@points@dim}{\exf@problem@points}\fi%
893 \fi\fi%
```

Warn if calculated total points have changed:

```
894 \else%
895 \ifdefined\exf@problem@points%
896 \ifdim\exf@problem@points@dim=\exf@problem@points pt\else%
897 \PackageWarning{exframe}{points changed for %
898 \exf@problemname\space\csname the\exf@problemcounter\endcsname;
899 rerun to fix}%
900 \ifexf@warntext\exf@config@insertwarnpoints{\exf@problemname}%
901 {\strip@pt\exf@problem@points@dim}{\exf@problem@points}\fi%
902 \fi\fi%
```

Read computed total points, check whether zero:

```
903 \ifdim\exf@problem@points@dim=0pt%
904 \let\exf@problem@points\@undefined%
905 \else%
906 \edef\exf@problem@points{\strip@pt\exf@problem@points@dim}%
907 \fi%
908 \fi%
```

Write points to .aux file; add to sheet total:

```
909 \ifdefined\exf@problem@points%
910 \exf@writeproblempoints{\exf@problem@points}%
911 \global\addtolength{\exf@sheet@points@dim}{\exf@problem@points pt}%
912 \fi%
```

Warn if some but not all problems on sheet declare points:

```
913 \ifdefined\exf@problem@points\else\ifdim\exf@sheet@points@dim=0pt\else%
914 \PackageWarning{exframe}{no points defined for \exf@problemname}%
915 \fi\fi%
```

Solutions to subproblems must be declared within problem block:

```
916 \global\let\exf@prevsubprob\@undefined%
917 \global\let\exf@prevsubprohref\@undefined%
```

End paragraph and block:

```
918 \par\exf@close@block%
```

Display solution if desired:

```
919 \exf@solbelowis{problem}{%
920 \exf@config@insertproblemsolution%
921 \exf@showsolutions{\exf@config@composetitlesolutionmulti}{}}%
```

Restore original equation counter:

```
922 \ifexf@style@problemequation%
923 \setcounter{exf@problemequation}{\value{equation}}%
924 \setcounter{equation}{\exf@eqsav}%
925 \fi%
```

End inner group:

```
926 \endgroup%
```

Insert hook code, vertical skip:

```
927 \exf@config@insertproblemafter%
928 \addvspace{\exf@config@skipproblembelow}%
```

Display solution if desired:

```
929 \exf@solbelowis{problem*}{%
930 \exf@showsolutions{\exf@config@composetitlesolutionmulti}{}}%
```

Done:

```
931 \ignorespacesafterend}
```

Subproblems.

exf@subproblem Define options for subproblem environment:

```
932 \define@key{exf@subproblem}{points}{\def\exf@subproblem@points{#1}}
933 \define@key{exf@subproblem}{label}{\def\exf@label{#1}}
```

subproblem Define subproblem environment (potentially using custom name):

```
934 \newenvironment{\exf@subproblemname}[1] [] {%
```

Start with new paragraph, set text style, add vspace and step counter:

```
935 \par{\exf@config@styletext\addvspace{\exf@config@skipsubproblemabove}}%
936 \refstepcounter{\exf@subproblemcounter}%
```

Insert hook code:

```
937 \exf@config@insertsubproblembefore%
```


Start inner group, mark in subproblem:

```
938 \begingroup%
939 \def\exf@in@subproblem{}
```

Initialise variables, process arguments:

```
940 \exf@init@block{\exf@config@skips@subproblem@info}%
941 \let\exf@subproblem@points\@undefined%
942 \let\exf@label\@undefined%
943 \setkeys{exf@subproblem,exf@problem@info}{#1}%
```

Process manual label:

```
944 \ifdefined\exf@label\label{\exf@label}\fi%
```

Remember subproblem counter for solution:

```
945 \xdef\exf@prevsubprob{\csname the\exf@subproblemcounter\endcsname}%
946 \ifcsname theH\exf@subproblemcounter\endcsname%
947 \xdef\exf@prevsubprobhref{\exf@subproblemcounter.%
948 \csname theH\exf@subproblemcounter\endcsname}%
949 \fi%
```

Add given points to problem total:

```
950 \ifdefined\exf@subproblem@points%
951 \global\let\exf@prevpoints\exf@subproblem@points%
952 \global\addtolength{\exf@problem@points@dim}{\exf@subproblem@points pt}%
```

Warn if no points given for present subproblem but previously:

```
953 \else%
954 \ifdim\exf@problem@points@dim=0pt\else%
955 \PackageWarning{exframe}{no points defined for \exf@subproblemname}%
956 \fi%
957 \fi%
```

Disable points display if desired:

```
958 \exf@subpoints@satis{off}{\let\exf@subproblem@points\@undefined}%
```

Display points in opening line if desired; expand points into argument and remove points:

```
959 \exf@subpoints@satis{start}{\ifdefined\exf@subproblem@points%
960 \exf@exptwo\exf@append@intro{\expandafter%
961 \exf@config@composepointsstart\expandafter{\exf@subproblem@points}}%
962 \let\exf@subproblem@points\@undefined\fi}%
963 \exf@subpoints@satis{start*}{\ifdefined\exf@subproblem@points%
964 \exf@exptwo\exf@prepend@intro{\expandafter%
965 \exf@config@composepointsstart\expandafter{\exf@subproblem@points}}%
966 \let\exf@subproblem@points\@undefined\fi}%
```

Insert hook code:

```
967 \exf@config@insert@subproblem@info%
```

Write opening line without item:

```
968 \ifdim\exf@config@skips@subproblem@item=0pt%
969 \exf@prepend@intro{{%
970 \exf@config@style@title\exf@config@style@titlesubproblem%
971 \exf@config@compose@titlesubproblem{%
972 \csname the\exf@subproblemcounter\endcsname}}}%
```

Write item with fixed total width or item width plus space:

```
973 \else%
974 \ifdim\exf@config@skipsubproblemitem>0pt%
975 \setlength\exf@addmargin{\exf@config@skipsubproblemitem}%
976 \else%
977 \settowidth\exf@addmargin{%
978 \exf@config@styletitle\exf@config@styletitlesubproblem%
979 \exf@config@composeitemsubproblem{\exf@config@countersubproblemmax}%
980 \exf@config@composeitemsubproblemsep}%
981 \fi%
```

Define item label:

```
982 \def\exf@introitem{\makebox[0cm][r]{%
983 \exf@config@styletitle\exf@config@styletitlesubproblem%
984 \exf@config@composeitemsubproblem%
985 {\csname the\exf@subproblemcounter\endcsname}%
986 \exf@config@composeitemsubproblemsep}}%
987 \fi%
```

Write points into margin if desired; expand points into argument and remove points:

```
988 \exf@subpointsatisf{margin}{\ifdefined\exf@subproblem@points%
989 \expandafter\exf@prepend@def\expandafter\exf@intro\expandafter%
990 {\expandafter\protect\expandafter%
991 \exf@config@insertpointsmargin\expandafter{\expandafter%
992 \exf@config@composepointsmargin\expandafter{\exf@subproblem@points}}}%
993 \let\exf@subproblem@points\undefined\fi%
```

Write out opening line:

```
994 \exf@open@block{\exf@config@skipsubproblemtitle}%
```

Done:

```
995 \@afterindentfalse}%
```

End environment, show points if desired:

```
996 {\exf@subpointsatisf{end}{\showpoints}}%
```

End paragraph and block:

```
997 \par\exf@close@block%
```

Display solution if desired:

```
998 \exf@solbelowis{subproblem*}{%
999 \exf@config@insertsubproblemsolution%
1000 \exf@showsolutions{\exf@config@composetitlesolutionsingle}{}}%
```

End inner group:

```
1001 \endgroup%
```

Insert hook code, vertical skip:

```
1002 \exf@config@insertsubproblemafter%
1003 {\exf@config@styletext\addvspace{\exf@config@skipsubproblembelow}}%
```

Display solution if desired:

```
1004 \exf@solbelowis{subproblem}{%
1005 \exf@showsolutions{\exf@config@composetitlesolutionsingle}{}}%
```

Done:

```
1006 \ignorespacesafterend}
```

Print Solutions.

`exf@solution` Define options for solution environment:

```
1007 \define@key{exf@solution}{prob}{\def\exf@solprob{#1}}
1008 \define@key{exf@solution}{subprob}{\def\exf@solsubprob{#1}}
1009 \define@key{exf@solution}{problemtag}{\def\problemtag{#1}}
1010 \define@key{exf@solution}{sheettag}{\def\sheettag{#1}}
1011 \define@key{exf@solution}{href}{\def\exf@solhref{#1}}
1012 \define@key{exf@solution}{label}{\def\exf@label{#1}}
1013 \define@key{exf@solution}{points}{\def\exf@solution@points{#1}}
1014 \define@key{exf@solution}{probttitle}{\def\exf@solprobttitle{#1}}
```

`printsolution` Define `printsolution` environment to display a previously read solution block; this works analogously to `problem` and `subproblem`:

```
1015 \newenvironment{printsolution}[1]{%
```

Start new paragraph, add vertical space:

```
1016 \par{\exf@config@styletext\addvspace{\exf@config@skipsolutionabove}}%
```

Insert hook code:

```
1017 \exf@config@insertsolutionbefore%
```

Use equation counter for solutions:

```
1018 \ifexf@style@solutionequation%
1019 \exf@eqsav\value{equation}%
1020 \setcounter{equation}{\value{exf@solutionequation}}%
1021 \let\theequation\theexf@solutionequation%
1022 \let\theHequation\theHexf@solutionequation%
1023 \fi%
```

Start a block, initialise variables, process arguments:

```
1024 \begingroup%
1025 \def\exf@solprob{}%
1026 \def\exf@solsubprob{}%
1027 \let\exf@label\@undefined%
1028 \setlength{\exf@solution@points@dim}{0pt}%
1029 \def\exf@solhref{}%
1030 \exf@init@block{\exf@config@skipsolutioninfo}%
1031 \setkeys{exf@solution,exf@probleminfo}{#1}%
```

Set solution counter to reflect associated problem:

```
1032 \exf@csdo\def{the\exf@solutioncounter}%
1033 {\exf@config@composeitemsolution{\exf@solprob}{\exf@solsubprob}}%
1034 \refstepcounter{\exf@solutioncounter}%
```

Set label:

```
1035 \ifdefined\exf@label\label{\exf@label}\fi%
```

Insert hook code, set solution body style:

```
1036 \exf@config@insertsolutioninfo%
1037 \exf@config@styletext\exf@config@styletextsolution%
```

Determine solution for problem or subproblem:

```
1038 \ifx\exf@solsubprob\exf@empty%
1039 \let\exf@tmp\exf@config@skipsolutionitem%
1040 \else%
1041 \let\exf@tmp\exf@config@skipsolutionitemsub%
1042 \fi%
```

Write title without item:

```
1043 \ifdim\exf@tmp=0pt%
1044 \protected@edef\exf@solution@title{%
1045 \exf@composetitle{\exf@solprob}{\exf@solsubprob}}%
1046 \ifx\exf@solution@title\exf@empty\else%
1047 \exf@prepend@intro{%
1048 \exf@config@styletitle\exf@config@styletitlesolution%
1049 \ifexf@solutionhref\exf@href{\exf@solhref}%
1050 {\exf@solution@title}\else\exf@solution@title\fi}%
1051 \fi%
```

Write item with fixed total width or item width plus space:

```
1052 \else%
1053 \ifdim\exf@tmp>0pt%
1054 \setlength\exf@addmargin{\exf@tmp}%
1055 \else%
1056 \settowidth\exf@addmargin{%
1057 \exf@config@styletitle\exf@config@styletitlesolution%
1058 \ifx\exf@solsubprob\exf@empty%
1059 \exf@config@composeitemsolution{\exf@config@counterproblemmax}%
1060 {\exf@config@countersubproblemmax}%
1061 \else%
1062 \exf@config@composeitemsolutionsub{\exf@config@counterproblemmax}%
1063 {\exf@config@countersubproblemmax}%
1064 \fi\exf@config@composeitemsolutionsep}%
1065 \fi%
```

Set item label up depending on problem or subproblem:

```
1066 \ifx\exf@solsubprob\exf@empty%
1067 \protected@edef\exf@solution@item%
1068 {\exf@config@composeitemsolution{\exf@solprob}{\exf@empty}}%
1069 \else%
1070 \protected@edef\exf@solution@item%
1071 {\exf@config@composeitemsolutionsub{\exf@solprob}{\exf@solsubprob}}%
1072 \fi%
```

Define item label:

```
1073 \def\exf@introitem{\makebox[0cm][r]{%
1074 \exf@config@styletitle\exf@config@styletitlesubproblem%
1075 \ifexf@solutionhref\exf@href{\exf@solhref}{\exf@solution@item}%
1076 \else\exf@solution@item\fi%
1077 \exf@config@composeitemproblemsep}}%
1078 \fi%
```

Write out opening line:

```
1079 \exf@open@block{\exf@config@skipsolutiontitle}%
```

Done:

```
1080 \@afterindentfalse%
```

End environment, show points if desired, perform sanity check:

```
1081 {\ifdefined\exf@solution@points\ifdim\exf@solution@points@dim=0pt\else%
1082   \ifdim\exf@solution@points@dim=\exf@solution@points pt\else%
1083     \PackageWarning{exframe}{points mismatch in \exf@solutionname}%
1084     \ifexf@warntext\exf@config@insertwarnpoints{\exf@solutionname}%
1085       {\strip@pt\exf@solution@points@dim}{\exf@solution@points}\fi%
1086   \fi\fi\fi%
```

End paragraph and block:

```
1087 \par\exf@close@block%
```

Restore original equation counter:

```
1088 \ifexf@style@solutionequation%
1089   \setcounter{exf@solutionequation}{\value{equation}}%
1090   \setcounter{equation}{\exf@eqsav}%
1091 \fi%
```

End inner group:

```
1092 \endgroup%
```

Vertical skip, insert hook code:

```
1093 {\exf@config@styletext\addvspace{\exf@config@skipsolutionbelow}}%
1094 \exf@config@insertsolutionafter%
```

Done:

```
1095 \ignorespacesafterend}
```

`\solutionssection` Define a section for a problem within a block with multiple solutions:

```
1096 \newcommand{\solutionssection}[1]{\begingroup%
```

Initialise variables, process arguments:

```
1097 \def\exf@solprob{}%
1098 \def\exf@solsubprob{}%
1099 \def\exf@solprobttitle{}%
1100 \let\exf@label\@undefined%
1101 \let\exf@solhref\@undefined%
1102 \setkeys{exf@solution}{#1}%
```

Select title (and table of contents entry) corresponding to multiple problems vs. single problem:

```
1103 \let\exf@composetitle\exf@config@composetitlesolutionsproblemmulti%
1104 \def\exf@solutionstoc{\exf@addcontentsline{\exf@config@toclevelsolution}%
1105   {\exf@config@composetocsolution{\exf@solprob}{\exf@solprobttitle}}}%
1106 \exf@solbelowis{problem}{\let\exf@composetitle%
1107   \exf@config@composetitlesolutionsproblemsingle}%
1108 \exf@solbelowis{problem*}{\let\exf@composetitle%
1109   \exf@config@composetitlesolutionsproblemsingle}%
```

Write section line:

```
1110 \exf@solutionssection{\exf@config@styletitlesolutionsproblem}%
1111   {\exf@composetitle{\exf@solprob}{\exf@solprobttitle}}%
1112   {\exf@config@skipsolutionsproblem}%
1113   {\exf@solutionstoc}{\exf@label}{\exf@solhref}%
1114 \endgroup}
```

Read Solution Block to Buffer.

`solution` Define solution environment (potentially using custom name); pass on to `exf@scansolution` environment via `\exf@scanblock` mechanism:

```
1115 \newenvironment{\exf@solutionname}%  
1116 {\exf@scanblock{\exf@scansolution}}{\endexf@scansolution}%
```

`exf@scansolution` Define `exf@scansolution` to read solution to buffer and process:

```
1117 \newenvironment{exf@scansolution}[2]{%
```

If solution is to be displayed immediately, make sure to display points first:

```
1118 \exf@solbelowis{here}{\showpoints}%  
1119 \global\exf@bufcleanfalse%
```

If this is the first solution within a new section, display section heading:

```
1120 \ifdefined\exf@problem@solnewsec%  
1121 \def\exf@probarg{\ifdefined\exf@prevprob prob={\exf@prevprob}\fi%  
1122 \ifdefined\exf@prevprobttitle,probttitle={\exf@prevprobttitle}\fi%  
1123 \ifdefined\exf@prevprobhhref,href={\exf@prevprobhhref}\fi%  
1124 \ifdefined\exf@sollabel,label={\exf@sollabel}\fi}%  
1125 \exf@solbelowis{here}{\let\exf@probarg\@undefined}%  
1126 \exf@solbelowis{subproblem}{\let\exf@probarg\@undefined}%  
1127 \exf@solbelowis{subproblem*}{\let\exf@probarg\@undefined}%  
1128 \ifdefined\exf@probarg%  
1129 \ifexf@lineno\exf@addbuflinesep{\exf@linesep}\exf@addbuflinesep{\exf@lineno}\fi%  
1130 \exf@addbuflinesep{\@backslashchar solutionssection{\exf@probarg}}%  
1131 \exf@addbuflinesep{}%  
1132 \fi%  
1133 \global\let\exf@problem@solnewsec\@undefined%  
1134 \fi%
```

Write separator to buffer:

```
1135 \ifexf@lineno\exf@addbuflinesep{\exf@linesep}\exf@addbuflinesep{\exf@lineno}\fi%
```

Declare additional arguments to `printsolution` to describe corresponding problem and tags:

```
1136 \def\exf@subprobarg{%  
1137 \ifdefined\exf@prevprob prob={\exf@prevprob},\fi%  
1138 \ifdefined\exf@prevsubprob%  
1139 subprob={\exf@prevsubprob},%  
1140 \ifdefined\exf@prevsubprobhhref href={\exf@prevsubprobhhref},\fi%  
1141 \else%  
1142 \ifdefined\exf@prevprobhhref href={\exf@prevprobhhref},\fi%  
1143 \fi%  
1144 \ifdefined\exf@prevpoints points={\exf@prevpoints},\fi%  
1145 \ifdefined\sheettag sheettag={\sheettag},\fi%  
1146 \ifdefined\problemtag problemtag={\problemtag},\fi}%
```

Write `printsolution` opening to buffer:

```
1147 \exf@addbuflinesep{\@backslashchar begin{printsolution}{\exf@subprobarg#1}}%
```

Clean up:

```
1148 \global\let\exf@prevsubprob\@undefined%  
1149 \global\let\exf@prevsubprobhhref\@undefined%  
1150 \global\let\exf@prevpoints\@undefined%
```

Verbatim read block to buffer:

```
1151 \exf@verbatim#2}%
```

End environment; stop reading buffer:

```
1152 {\exf@endverbatim%
```

Write `printsolution` closing to buffer:

```
1153 \exf@addbuflines{\@backslashchar end{printsolution}}%
```

Write buffer to file if output file open:

```
1154 \ifexf@infile%  
1155 \exf@writebuf%  
1156 \exf@clearbuf%  
1157 \fi%
```

Drop buffer if solutions not to be displayed:

```
1158 \ifsolutions\else\exf@clearbuf\fi%
```

Display solution immediately in various cases:

```
1159 \exf@solbelowis{here}{\exf@showsolutions%  
1160 {\exf@config@composetitlesolutionsingle}{}}%  
1161 \ifdefined\exf@in@subproblem\else%  
1162 \exf@solbelowis{subproblem}{%  
1163 \exf@showsolutions{\exf@config@composetitlesolutionsingle}{}}%  
1164 \exf@solbelowis{subproblem*}{%  
1165 \exf@showsolutions{\exf@config@composetitlesolutionsingle}{}}\fi%  
1166 \ifdefined\exf@in@problem\else%  
1167 \exf@solbelowis{problem}{%  
1168 \exf@showsolutions{\exf@config@composetitlesolutionmulti}{}}%  
1169 \exf@solbelowis{problem*}{%  
1170 \exf@showsolutions{\exf@config@composetitlesolutionmulti}{}}\fi%
```

Done:

```
1171 \ignorespacesafterend}
```

Solution Block Handling.

`\exf@solutionssection` Output solutions section:

```
1172 \newcommand{\exf@solutionssection}[6]{%
```

Check whether title is empty:

```
1173 \protected@edef\exf@solutionstitleexp{#2}%  
1174 \ifx\exf@solutionstitleexp\exf@empty\else%
```

Define a label:

```
1175 \ifdefined#5%  
1176 \exf@csdo\def{the\exf@solutioncounter}%  
1177 {\exf@config@composeitemsolution{\exf@solprob}{\exf@solsubprob}}%  
1178 \refstepcounter{\exf@solutioncounter}\label{#5}%  
1179 \fi%
```

Output section line:

```
1180 \exf@section{#3}{\exf@config@styletitle\exf@config@styletitlesolution#1%
```

```

1181 \ifexf@solutionhref\exf@href{#6}{\exf@solutionstitleexp}%
1182 \else\exf@solutionstitleexp\fi#4%
1183 \fi}

```

`\exf@solutionstitle` Compose the title for a solution section:

```

1184 \newcommand{\exf@solutionstitle}{\exf@solutionssection%
1185 {\exf@config@styletitlesolutions}{%
1186 \exf@config@composetitlesolutions}{\exf@config@skipsolutionstitle}%
1187 {\exf@addcontentsline{\exf@config@toclevelsolutions}%
1188 {\exf@config@composetoccsolutions}}{\@undefined}{\@undefined}}

```

`\exf@showsolutions` Output solutions in buffer:

```

1189 \newcommand{\exf@showsolutions}[2]{%

```

Do nothing if buffer is empty (avoid titles):

```

1190 \ifexf@bufclean\else\beginngroup%

```

Set solution body style; add vertical space; insert hook code:

```

1191 \par\exf@config@styletext\advspace{\exf@config@skipsolutionsabove}%
1192 \exf@config@styletextsolution%
1193 \exf@config@insertsolutionsbefore%
1194 \let\exf@composetitle#1%
1195 #2%

```

Source and clear buffer:

```

1196 \exf@sourcebuf%
1197 \exf@clearbuf%

```

Ignore most recent skip; insert hook code; close paragraph; add vertical space:

```

1198 \removelastskip%
1199 \exf@config@insertsolutionsafter%
1200 \par\exf@config@styletext\advspace{\exf@config@skipsolutionbelow}%
1201 \endgroup\fi}

```

Solutions Buffer Interface.

`\writesolutions` Open a file #1.sol for writing solutions; default is present main file name:

```

1202 \newcommand{\writesolutions}[1][\jobname]{\exf@closefile\exf@startfile{#1}}

```

`\closesolutions` Close output file (if open):

```

1203 \newcommand{\closesolutions}{\exf@closefile}

```

`\readsolutions` Read solutions from file #1.sol; default is present main file name; switch layout and add heading:

```

1204 \newcommand{\readsolutions}[1][\jobname]{\exf@closefile%
1205 \ifsolutions\beginngroup%
1206 \exf@config@styletext\exf@config@styletextsolution%
1207 \let\exf@config@composetitlesolution\exf@config@composetitlesolutionmulti%
1208 \exf@solutionstitle%
1209 \input{#1.sol}%
1210 \endgroup\fi}

```


`\insertsolutions` Show solution buffer:

```
1211 \newcommand{\insertsolutions}{\exf@showsolutions%  
1212  {\exf@config@composetitlesolutionmulti}{\exf@solutionstitle}}
```