

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`

2026/01/22, v3.5.2

<https://ctan.org/pkg/exframe>

<https://github.com/nbeisert/latex-pkg-nb>

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	3
2.1	Exercise Environments	3
2.2	Solution and Problem Display	4
2.3	Metadata	6
2.4	Points	9
2.5	Labels and Tags	11
2.6	Layout	12
2.7	Exercise Styles	13
2.8	Package Options	14
3	Customisations	16
3.1	Guidance	16
3.2	Examples	18
4	Information	20
4.1	Copyright	20
4.2	Files and Installation	20
4.3	Related Packages	20
4.4	Feature Suggestions	22
4.5	Revision History	22

A	Standalone Sample	24
B	Multipart Sample	31
B.1	Main File	31
B.2	Sheet File	36
B.3	Individual Problem Files	38
B.4	Make Scripts	39
C	Implementation	42
C.1	General Definitions	42
C.2	Package Setup	44
C.3	Configuration	46
C.4	Styles	55
C.5	Metadata	58
C.6	Counters	62
C.7	Buffers	63
C.8	Points	66
C.9	Tag Lists	70
C.10	Sheet Environment	71
C.11	Problem Environment	73
C.12	Problem Blocks	78
C.13	Subproblem Environment	79
C.14	Solution Environment	83
C.15	Solution Blocks	89
C.16	Interaction with metastr	90

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 \LaTeX 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 \LaTeX 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 `chiddoc` 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

```
\usepackage{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 (*env.*) 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 (*env.*) The **subproblem** environment describes a subproblem, part or an individual question of an exercise problem:

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

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

solution (*env.*) The **solution** environment describes the solution to a problem or a subproblem:

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

A **solution** environment should be at the end of a **subproblem** or **problem** environment (it is not mandatory to provide a **solution**). It can be contained within the corresponding environment or it can follow it (where needed, the option **forproblem** associates a

solution following a `subproblem` to the encompassing `problem` rather than the preceding `subproblem`). 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 environment; this may also have some technical advantages and produce a slightly better result in terms of layout.

`sheet` (*env.*) 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 and Problem Display

There are several options to control the output of solutions and of problems.

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

```
\exercisecsetup{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 `onlysolutions` solutions is performed automatically, the conditional would typically be used to change some details, e.g. for adjusting titles:

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

Alternatively, content to be processed only in solutions mode can be enclosed in an `onlysolutions` block:

```
\begin{onlysolutions}
...
\end{onlysolutions}
```

This structure can be useful to hide auxiliary text or material if all solution content is to be stripped from a source file, e.g. by an automated `sed` filter rule (doubling of backslashes required for `sed` as well as for shell script strings):

```
sed "/\\\\\\begin{solution}/,\\\\\\end{solution}/d;" \
"/\\\\\\begin{onlysolutions}/,\\\\\\end{onlysolutions}/d"
```

`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 environment. The alternate modes **problem*** and **subproblem*** positions the solution *after* the (sub)problem environment such that it does not inherit any layout, but also no definitions made in the parent environment. 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.

`\writesolutions` 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`; the extension `.sol` can be customised by the configuration `extsolutions`. 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}`.

`solutionbuf` The package offers similar functionality to control the display of problems. In order to have
`problembuf` any control over the content of **problem** and **subproblem** environments, the latter need to
`subproblembuf` be read into an internal buffer. Reading of solutions and problems to internal buffers is activated or deactivated by:

`\exercisecsetup{solutionbuf[=true|false]}`
`\exercisecsetup{problembuf[=true|false]}`
`\exercisecsetup{subproblembuf[=true|false]}`

By default, **solution** environments are read to an internal buffer, while the content of **[sub]problem** environments is processed directly by the T_EX engine. Therefore, the below options to control the display of **[sub]problem** environments require the statement `\exercisecsetup{[sub]problembuf}`.

`problemmanual` The immediate display of **problem** environments is controlled by:
`\insertproblems`

`\exercisestyle{problemmanual[=true|false]}`

In the default automatic mode, problems are displayed directly where they are declared. In manual mode, problems are collected to an internal buffer, and only displayed by issuing `\insertproblems`.

Note that `solution` environments should be declared within the corresponding `problem` environment in order to preserve their appropriate association. The `solution` environment is then processed at the place where the `problem` environment is displayed, and it may (or may not) be deferred further.

`\writeproblems` Problems can be written out to an external file for later usage. The functionality is analogous to `\readproblems` to solutions and uses the macros:

```
\writeproblems[filename]
\readproblems[filename]
```

The optional argument describes the filename as `filename.prb`; no argument defaults to the main tex filename as `\jobname.prb`; the extension `.prb` can be customised by the configuration `extproblems`.

`disable` The display of a particular `[sub]problem` can be suppressed altogether by an optional argument: `insertproblemsselect`
`sertsubproblemsselect`

```
\begin{[sub]problem}[disable]
```

This option can be exploited to automatically suppress certain classes of problems as follows: A hook function `insert[sub]problemselect` declared by:

```
\exerciseconfig{insert[sub]problemselect}[1]{code}
```

can call `\set[sub]problemdata{disable}` whenever a problem is to be suppressed. In order to decide, the optional argument of the `[sub]problem` environment is passed on to the hook function as the single argument. Note that the argument needs to be processed manually.

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:

```
\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 `LATEX` command `\author`; will be written to pdf documents.
- **title:** title of the exercise collection; also invokes the `LATEX` command `\title`; will be written to pdf documents.
- **date:** date of the exercise collection; also invokes the `LATEX` 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:

`\defexercisedata{key}`

`\getexercisedata` Global metadata should typically be specified somewhere at the top of the main document, `\exercisedataempty` 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:

`\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:

`\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 `problem` lems:

`\begin{sheet}[opts]`
`\begin{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**.

\defsheetdata Metadata for sheets can be used in the same way as the global metadata. The following directives are analogous to **\defexercisedata**, **\exercisedata**, **\getexercisedata** and **\getsheetdata** **\exercisedataempty**:

```

\sheetdataempty
\defproblemdata      \defsheetdata{key}
\setproblemdata      \setsheetdata{data}
\getproblemdata      \getsheetdata{key}
\problemdataempty    \sheetdataempty{key}{empty code}{filled code}
                     \defproblemdata{key}
                     \setproblemdata{data}
                     \getproblemdata{key}
                     \problemdataempty{key}{empty code}{filled code}

```

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

```
\exercisecsetup{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.

problem There is an additional mechanism to keep track of metadata for problems, subproblems and **subproblem** solutions which can be displayed in the opening line of these entities. Displayed metadata **solution** serve two purposes: they are used to describe the quality of a problem or they are intended 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** (display enabled by default): whether addressing the problem is mandatory or optional; by default the text will be displayed after the title in italic shape.

By default, only the **optional** items are displayed, all other types of items are hidden; controlling the display for each type of item is described below.

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
  \addprobleminfo      \exerciseconfig{insertprobleminfo}{code}
  \addprobleminfo      \exerciseconfig{insertsubprobleminfo}{code}
\addprobleminfo*      \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 T_EX 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.

Bonus points can be specified in the format `points=[regular][+bonus]`. By default, such points will be printed as `[regular][+bonus]` where 0 components are omitted.

`problem``points``at` The location where points of problems and subproblems shall be displayed can be adjusted individually by:

`subproblem``points``at`
`solution``points``at`

```
\exercisestyle{problempointsat=start|start*|margin|end|manual|off}
\exercisestyle{subproblempointsat=start|start*|margin|end|manual|off}
\exercisestyle{solutionpointsat=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 displayed manually. Within the corresponding `sheet` environment the points can be accessed by:

```
\getsheetdata{points}
```

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

`\getproblem``points`
`\getsubproblem``points`
`\getsolution``points`
`\extract``points`
`\switch``points`

```
\getsheetpoints{[tag]}
\getproblempoints{[tag]}
\getsubproblempoints{[tag]}
\getsolutionpoints{}
```

Here *tag* is the tag assigned to the corresponding sheet or problem, see section 2.5. An empty argument *tag* refers to the current sheet, (sub)problem or solution. If bonus points are used, the points will be returned in the format `[regular][+bonus]`; the components *regular* and *bonus* can be extracted from the returned expression by `\extractpoints` and `\extractpoints*`, respectively. A convenient case switch of the returned value can be performed by:

```
\switchpoints{reg}{bonus}{both}{none}{val}
```

Here *val* is the value returned from the points register, *reg* is displayed for purely regular points, *bonus* is displayed for purely bonus points, *both* is displayed for mixed points, *none* is displayed for no points. In each of the four expressions, #1 will be replaced by the regular points and #2 by the bonus points.

`\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**, `\sheettag` **problem** and **subproblem** can be assigned a unique tag *tag* by the environment option: `\problemtag` `\subproblemtag`

```
tag={tag}
```

This tag is used for reading point totals as described in section 2.4. Furthermore, the macro `\sheettag`, `\problemtag` or `\subproblemtag` is set to the tag *tag* within the current environment. If no tag is specified it defaults to the number of the current sheet or (sub)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]}
```

`\getsheetlist` Tags can also be used to process a list of sheets, problems and subproblems:
`\getproblemlist`
`\getsubproblemlist`

```
\getsheetlist{}
\getproblemlist{[sheet-tag]|*}
\getsubproblemlist{[problem-tag]}
```

These commands return a list of sheets, problems and subproblems tags, where each item is encapsulated in braces. The commands `\get[sub]problemlist` return the (sub)problems within the specified sheet or problem. If no argument is given, the current sheet or problem is assumed. `\getproblemlist*` returns the list of all problems.

`\exerciseloop` The package defines two commands to walk through such a list:

```

\exerciseloopstr
\exerciseloopret
exerciseloop
\exerciseloop{items}{cmd}
\exerciseloopstr[ret]{items}{str}

```

Here, *items* is a list of items in braces, and `\exerciseloop` evaluates *cmd* for each item of the list where ‘#1’ is replaced by the item. The command `\exerciseloopstr` concatenates the evaluations of *str* and returns the resulting string in the macro *ret* (which defaults to `\exerciseloopret`). Both commands maintain a counter of items `exerciseloop` which can be accessed within *cmd/str* or after `\exerciseloop[str]` to obtain the total number of items in the list.

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:

```
\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:

```

\exerciseconfigappend{key}{value}
\exerciseconfigprepend{key}{value}

```

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

```
\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:

```
\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 C.3. Some particular aspects are described in more detail in section 3.1. The configuration includes 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.

We will highlight a few examples in section 3.2.

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 environments; see section 2.2 for details.
- `problempointsat=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.
- `subproblempointsat=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.
- `solutionpointsat=pos` (can take values `start`, `start*`, `margin`, `end` and `manual`; initially set to `end`) – displays points in solutions 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*.
- `problembysheet` – 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.
- `subproblemdelitem` – shift subproblem number delimiter ‘)’ from subproblem label to subproblem item composition; needed if `\ref` to a subproblem `\label` should produce a bare identifier without a trailing ‘)’; assumes (and restores) numbering `\alph` and delimiter ‘)’.

`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 C.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

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

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

`\PassOptionsToPackage` must be used before `\usepackage`; `\exercisecsetup` 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 `\exercisecsetup`:

- `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.
- `metastr[=true|false]` (no value implies `true`, initially set to `false`) – load `metastr` package and use to handle pdf metadata and basic internationalisation, see section C.16 for details.
- `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`.
- `warntext[=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`.
- `solutionbuf[=true|false]` (no value implies `true`, initially set to `true`) – enable/disable buffering for `solution` environments in order to control their display; disabling buffering can be helpful in debugging faulty `solution` environments; it might also resolve some tokenisation issues in special circumstances; note that the display of solutions cannot be suppressed with `\exercisestup{solutions=false}` when buffering is disabled.
- `problembuf[=true|false]` (no value implies `true`, initially set to `false`) – enable/disable buffering for `problem` environments in order to control their display.

- `subproblembuf[=true|false]` (no value implies `true`, initially set to `false`) – enable/disable buffering for `subproblem` environments in order to control their display.
- `emptytestchar=char` (initially set to `'&'`) – character to use for testing whether an argument `#1` is empty via `\if&...\fi`; if `'&'` causes trouble within tables, could try `'@'` instead.

3 Customisations

This section attempts to shed some light on the rich array of options for layout customisations. It is not at all complete, but it provides an overview and illustrates selected aspects by means of guidance and examples.

3.1 Guidance

The following provides some guidance on customisation options which might otherwise be hard to follow. Suggestions for topics to be expanded are welcome.

Package Philosophy. 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 package itself provides a functional basic layout rather than an elaborate or particularly appealing one. However, it provides means to adjust the basic layout in many ways and to predefine custom layout schemes. The package is meant to collaborate with other packages which provide solutions for particular applications in order to produce an appealing layout for the given content.

Framework for Layout Customisation. The framework for defining the layout consists of an internal and a public component which allow for minor adjustments and major changes both with reasonable efforts.

The internal component implements some very coarse choices for the presentation of the content. It is hard-coded in the package (using internal macros starting with `exf@`) and its operation can be adjusted mainly through the available package options, see section 2.8, and through configuration switches for individual blocks.

The public framework uses a hierarchy of configuration macros defined by `\exerciseconfig`, see section 2.6. The internal component interfaces with the public framework through a set of higher-level interface macros. These include macros which produce the desired text in a well-prescribed situation as well as macros which perform some \LaTeX operations such as adjusting presentation styles or modifying some variables. In many cases the structural higher-level macros have been set up to branch out to further lower-level macros prescribing presentation details.

The hierarchy has the following structure: High-level macros compose certain abstract elements into readable text, e.g. an intro/outro for some block of text made from space, decorations and abstract text elements. Medium-level macros compose elementary objects into abstract text elements, e.g. a header for some block of text composed from a label and a title with some space and delimiters in between. Low-level macros define elementary objects, such as delimiter characters, spaces, widths and words (translations).

The public framework has been set up such that elements at high, medium and low levels can be adjusted independently to arrive at a specific effect. For example, the composition of

problem titles can be adjusted abstractly, but also the representation of a specific delimiter can be tweaked from ‘.’ to ‘:’; either change can be obtained by adjusting a single definition. This flexibility comes along with an unfortunate complexity to allow for adjustments to be implemented individually but also universally. Nevertheless, the implementation of the public configuration macros for a particular setup can be rather simple because the particular setup will neither use all of the available coarse layout choices nor all of the customisation options. For example, a higher-level macro for a problem title may directly produce the desired text without branching into further lower-level definitions. Also, only those higher-level macros which actually come to use in a particular setup will need to be implemented.

The following paragraphs provide guidance on aspects of the public framework. There is no complete manual for the framework, but a skilled L^AT_EX tinkerer can find further information and pieces of code ready for tweaking in the implementation notes in section C.3 and section C.4.

Labels for Problems and Sheets. The package provides many methods with various names to customise labels for problem and solution blocks. These names follow certain patterns (but are sometimes stuck with odd assignments due to backward compatibility). The following guide describes which methods come to use in which situations.

The labels can be grouped into two categories, items and titles, and the corresponding methods are typically called `...item...` and `...title...`. Items are enumerative identifiers for the [sub]problems such as ‘1.’, ‘b)’ or ‘3iv)’. Titles serve as headers for [sub]problem or solution blocks which usually take the form “Problem 1”, “2. Fermat’s Last” or simply “c)” for a subproblem. Titles may use the corresponding item, and display it together with the type, name or using a different punctuation depending on the situation.

`composetitleproblem` The composition of problem titles through `composetitleproblem` depends mainly on whether an individual name is provided (`title` option to `problem` environment). The `composenamedproblem` method passes on to `composenamedproblem` if a name is provided (second argument non-empty), otherwise to `composebareproblem`. It might also produce a plain name if no item is provided (first argument empty).

`composenamedproblem` Titles and items are separated by some space (`...sep`) and/or delimiting characters (`...delim`). The item within a named title is separated from the following name by `composenamedproblemsep` and `composenamedproblem`. In an unnamed title, the `composebareproblem` type is separated from the following item by `composebareproblemsep`, and the item is followed by `composebareproblem`. A plain item not used within a title is delimited by `composeitemproblem`. The presentation of all titles and items is followed by `composeitemproblemsep`.

`composetocproblem` Problem titles can also be made to appear in the table of contents, their display is governed by `composetocproblem` which distinguishes between named and unnamed problems. The method uses the above delimiter `composenamedproblem` but no other customisations because detailed layout will be less welcome in the table of contents.

`composetitlesheet` Sheet titles generated by `composetitlesheet` are similar to problem titles but simpler. `composetocsheet` They may or may not have a name, but there are no pre-defined delimiters. The two similar methods `composetocsheet` and `composemetasheet` govern the display of sheet titles in the table of contents and in pdf metadata, respectively.

`composeitemsubproblem` Subproblem items and titles are simple as well because subproblems have no names. They are generated by `composeitemsubproblem` and `composetitlesubproblem`, respectively. The `composeitemsubproblem` subproblem delimiter and separator are determined by `composeitemsubproblem` and `composeitemsubproblemsep`, respectively. Note that the delimiter ‘)’ within ‘b)’ may be declared as either `composeitemsubproblem` or as part of the subproblem counter defined in `countersubproblem`. This behaviour is governed by the style `subproblem`.

Labels for Solutions. The display of appropriate solution items and titles is very complex as it depends on four factors: (i) The first factor is whether solutions are displayed as an itemised list or as a plain list. (ii) The second factor is whether solutions are displayed individually, grouped by problem or assembled as a list for multiple problems. (iii) The third factor is whether the solution corresponds to a problem or subproblem. (iv) A fourth factor is whether a solution is the first one within a problem. In addition, the list of solutions may have a title (v) and sections (vi). Let us discuss the relevant combinations.

<code>composetitlesolutions</code>	The solution list title (“Solutions”) (v) generated by <code>composetitlesolutions</code> only applies
<code>solutionsproblemsingle</code>	to lists of (potentially) multiple problems (ii). The solution section title (vi) generated by
<code>solutionsproblemmulti</code>	<code>composetitlesolutionsproblem...</code> indicates the corresponding problem. It comes in two
	variants <code>...single</code> and <code>...multi</code> which distinguish a single problem (“Solution”) from a
	list of (potentially) multiple problems (“Problem 2”) (ii).
<code>skipsolutionitem</code>	An itemised list of solutions (i) is invoked by a non-zero width <code>skipsolutionitem[sub]</code>
<code>skipsolutionitemsub</code>	(iii) for horizontal alignment of the solution text. In this case, solution items are dis-
<code>composeitemsolution</code>	played to the left of the line of horizontal alignment. These are generated by the configu-
<code>composeitemsolutionsub</code>	ration <code>composeitemsolution[sub]</code> (iii) irrespectively of the display mode (ii). By default,
<code>oseitemsolutionfirst</code>	the current problem item is displayed as a section title (vi), therefore the solution items
<code>itemsolutionfirstsub</code>	will only display the corresponding subproblem item leaving blank the solution items for
	problems. However, in an adjusted scheme without section titles (vi), the very first so-
	lution within a problem (iv) might additionally indicate the corresponding problem item
	<code>composeitemsolutionfirst[sub]</code> (iii).
<code>etitlesolutionsingle</code>	An individual solution (ii) formally part of a plain list (i) starts with the title “Solution:”
<code>oseitemsolutiondelim</code>	which is generated by the method <code>composetitlesolutionsingle</code> . The delimiter ‘:’ is
	defined by <code>composeitemsolutiondelim</code> .
<code>setitlesolutionmulti</code>	The title for a grouped solution (ii) within a plain list (i) is generated by the method
<code>titlesolutionproblem</code>	<code>composetitlesolutionmulti</code> which forwards to <code>composetitlesolution[sub]problem</code> (iii).
<code>lesolutionsubproblem</code>	Again, by default, these methods only display the subproblem item because the problem item
	is provided as a section title (vi) instead.
<code>composetocsolution</code>	Items for the table of contents corresponding to solution lists and problem solutions are
<code>composetocsolutions</code>	generated by <code>composetocsolution[s]</code> .

3.2 Examples

This section contains some sample customisations of the default (plain) layout which can be realised with moderate effort using the various tools provided by the package. Additional applications are shown in the sample files `exfsamp.tex` and `exfserm.tex`. Suggestions for further applications are welcome.

`insertsheettitle` **Sheet Title.** An important setting is:

```
\exerciseconfig{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 C.4.

`composetitleproblem` **Problem Title.** Another noteworthy example is `composetitleproblem` to compose the `composenamedproblem` title for a problem. It takes two parameters, the number and the title. The (somewhat `composebareproblem` simplified) default declarations are:

```
\exerciseconfig{composetitleproblem}[2]{\exerciseifempty{#2}
  {\getexerciseconfig{composebareproblem}{#1}}
  {\getexerciseconfig{composenamedproblem}{#1}{#2}}}
\exerciseconfig{composebareproblem}[1]{\getexerciseconfig{termproblem}
  #1\getexerciseconfig{composebareproblem}delim}}
\exerciseconfig{composenamedproblem}[2]
  {#1\getexerciseconfig{composenamedproblem}delim} #2}
```

This checks whether the title is empty. If no title is given use the configuration `composebareproblem` to compose “Problem #”, where the term “Problem” is made abstract by the configuration `termproblem` (e.g. to support internationalisation). If a title is given use the configuration `composenamedproblem` to compose “#. *title*”.

`\exerciseifempty` **Conditionals.** Handy conditional commands to check whether an expression *expr* is `\exerciseifnotempty` 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; also using this within tables can be troublesome; the character ‘&’ can be reconfigured by the package option `emptytestchar`).

`insertsolutionsbefore` **Decorate Solutions with Boxes.** The following adjustment changes the presentation of `insertsolutionsafter` solutions to be surrounded by boxes with background colour (thanks to Till Bargheer for `posetitlesolution...` the example). This yields an alternative presentation that visually distinguishes problem from solution content.

The adjustment uses the package `tcolorbox` to supply such boxes. The coloured box environment `tcolorbox` is invoked by the customisations `insertsolutionsbefore` and `...after` which are called just before and after a block of solutions is:

```
\RequirePackage{tcolorbox}
\tcbuselibrary{breakable}
\exerciseconfig{insertsolutionsbefore}{%
  \begin{tcolorbox}[breakable,boxrule=0.25pt]}
\exerciseconfig{insertsolutionsafter}{\end{tcolorbox}}
\exerciseconfig{composetitlesolutionsingle}[2]{}
```

With the highlighting of solutions by boxes, the default introductory statement “Solution:” (when `solutionbelow` is `here` or `subproblem`) of a solution block becomes obsolete. This is achieved by removing the content of `composetitlesolutionsingle` (or similar composition definitions if `solutionbelow` is set otherwise).

`solutionpointsat` Note that displaying points in the margin would have to be adjusted because the plain `insertpointsmargin` L^AT_EX command `\marginpar` does not work inside a `tcolorbox` (due to nested floats). If this combination is desired, one should adjust the placement of margin paragraphs to the package `marginnote`:

```

\exercisestyle{solutionpointsat=margin}
\RequirePackage{marginnote}
\exerciseconfig{insertpointsmargin}[1]{\marginnote{\footnotesize#1}}

```

4 Information

4.1 Copyright

Copyright © 2011–2026 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 <https://www.latex-project.org/lppl.txt> and version 1.3c or later is part of all distributions of L^AT_EX version 2008 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`, `exfsamp.tex`, `exfser x .tex` ($x=m, e, f, 01, 02, 03, aa$), `exfsermk.sh`, `exfsermk.mak` and `exframe.pdf`.

4.2 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>exfsamp.tex</code>	sample file
<code>exfserm.tex</code>	multipart sample main file
<code>exfser0n.tex</code>	multipart sample sheet $n=1, 2, 3$
<code>exfseraa.tex</code>	multipart sample unused problems
<code>exfserpx.tex</code>	multipart sample problem $x=e, f$
<code>exfsermk.sh</code>	multipart sample compile script
<code>exfsermk.mak</code>	multipart sample makefile
<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 samples consisting of `exfsamp.tex`, `exfser x .tex` ($x=m, e, f, 01, 02, 03, aa$), `exfsermk.sh`, `exfsermk.mak`. Copy the file `exframe.sty` to an appropriate directory of your L^AT_EX distribution, e.g. `texmf-root/tex/latex/exframe`.

4.3 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 L^AT_EX source file in the generated metapost file. Compatibility with the `currfile` package has been tested with v0.7c (2015/04/23).
- This package can use the package `metastr` to write pdf metadata and to handle basic translations. Compatibility with the `metastr` package has been tested with v1.0 (2020/02/06).

There are several other L^AT_EX packages which offer a similar functionality varying largely in scope and sophistication:

- The package `exsheets` and its successor `xsim` provide a L^AT_EX 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 provides a functional basic layout rather than an elaborate or particularly appealing one. However, it provides means to adjust the basic layout 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 on customisation assistance and examples and section 4.4 for feature suggestions.

4.4 Feature Suggestions

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

- Define structures for multiple-choice questions.
- Add more guidance to section 3.1 on customisation settings.
- Add more examples of customisation settings to achieve specific goals to section 3.2. Please send suggestions.
- Option to hide problem text while maintaining access to embedded solutions (for a version containing only solutions): this is difficult to implement because the problem environment cannot simply be discarded, but would have to be scanned very carefully for the embedded solution; instead process problems to some document and save solutions to file, then read solutions from different document.

4.5 Revision History

v3.5.2: 2026/01/22

- distinguish between problem, subproblem and solutions for display of points in margin

v3.5.1: 2025/03/25

- maintenance and manual update

v3.5: 2024/10/18

- more convenient delimiter configuration `compose...delim` for various items
- composition of problem titles in `composetitleproblem` split up further into the configurations `composebareproblem` and `composenamedproblem`; to enable delimiter ‘.’ for problems without a name, use

`\exerciseconfig{composebareproblemdelim}{.}`

- style `subproblemdelimitem` added to shift delimiter ‘)’ from subproblem reference labels to subproblem item composition.
- `composetitlesolutionsproblemulti` improved (thanks to Daniel Wegmann for pointing out insufficient behaviour)
- option `forproblem` for `solution` to release association to previous subproblem
- option to `disable` particular subproblems and solutions including following `solution` block
- handling of first solutions within a problem block

v3.4: 2020/02/24

- lists of sheets, problems and subproblems; list iterators
- tags for subproblems, tag customisation
- interaction with package `metastr`
- minor fixes

v3.31: 2020/01/11

- `onlysolutions` environment for solution mode content
- sample multipart setup streamlined

v3.3: 2019/06/15

- control display of `problem` environments via package option `problembuf`: manual display, write to file, disable individual problems
- `solutionbelow` mode `here*` superseded by package option `solutionbuf`
- display total points within solution: `solutionpointssat` (thanks to Till Bargheer for suggestion)
- read points for current sheet, (sub)problem and solution (thanks to Johannes Hahn for suggestion)
- case switch for bonus points (thanks to Johannes Hahn for suggestion)
- option to `disable` particular problems, control by hook function (thanks to Manuel Benz for suggestion)
- provided interface `\showfracpoints` and `\exerciseconfig{frac}` for fractional points display
- filename extensions configurable

v3.2: 2019/05/01

- bonus points can be specified as `points=[regular][+bonus]`
- `solutionbelow` mode `here*` added for direct processing of the solution environment
- multipart sample added

v3.11: 2019/04/15

- fix interaction with package `calc` (thanks to Johannes Hahn for bug report)
- fix style `fracpoints` in combination with some `[sub]problempointssat` choices (thanks to Johannes Hahn for bug report)
- fix spacing for `[sub]problempointssat=margin` (thanks to Johannes Hahn for bug report)

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 Standalone Sample

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

This example file describes a single exercise sheet. The other sheet of the series would be declared analogously 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 `amstext`, `hyperref` packages:

```
8 \usepackage{amstext}
9 \usepackage{hyperref}
```

May include `metastr` package; below code adjusts to whether or not present:

```
10 \PassOptionsToPackage{loadlang=en|de}{metastr}
11 \PassOptionsToPackage{course=true}{metastr}
12 %%\usepackage{metastr}
13 %%\metasetlang{de}
```

Include `exframe` package:

```
14 \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:

```
15 \exercisecsetup{solutions=false}
16 %%\exercisecsetup{solutions=true}
```

Automatically put labels for (sub)problems:

```
17 \exercisecsetup{autolabelproblem=true}
```

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:

```
18 \exercisestyle{plainheader}
19 \exerciseconfig{composeheaderbelowright}{\getsheetdata{points}}%
```

Make the delimiter ‘)’ part of the subproblem item rather than the subproblem label:

```
20 \exercisestyle{subproblemdelitem}
```

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:

```
21 \exerciseconfig{countersheet}{\Roman{sheet}}
22 \exerciseconfig{countersubproblem}{\roman{subproblem}}
23 \exerciseconfig{countersubproblemmax}{vii}
```

Automatically display an asterisk for all subproblems with bonus points only; remove space to separate items:

```
24 \exerciseconfig{insertsubprobleminfo}{%
25  \switchpoints{}{\addprobleminfo*{%
26    \hspace{-\getexerciseconfig{skipsubprobleminfo}}*}}%
27  {}{}{\getsubproblempoints{}}}
```

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

```
28 \ifdefined\metaset
29 \metasetterm[en]{sheet}{Exercise Sheet}
30 \metasetterm[en]{sheets}{Exercise Sheets}
31 \metasetterm[de]{sheet}{\ "Ubungsblatt}
32 \metasetterm[de]{sheets}{\ "Ubungsbl\ "atter}
33 \else
34 \exerciseconfig{termsheet}{\ "Ubungsblatt}
35 \exerciseconfig{termsheets}{\ "Ubungsbl\ "atter}
36 \fi
```

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

```
37 \exercisestyle{problempointsat=margin}
38 \reversemarginpar
39 \exerciseconfig{composepointsmargin}[1]{#1p.}
40 \exerciseconfig{composepointspairmargin}[2]{%
41   \ifdim#2pt=0pt#1p.%
42   \else\ifdim#1pt=0pt+#2p.%
43   \else#1+#2p.%
44   \fi\fi}
```

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

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

Add a significant amount of space below problems:

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

Display half points as fractions:

```
47 \exercisestyle{fracpoints}
```

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

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

Indent solutions for subproblems:

```
49 \exerciseconfig{skipsolutionitemsub}{-1pt}
```

Separate solutions by horizontal lines (extended style):

```
50 \exercisestyle{solutionsep}
```

Write metadata for sheet:

```
51 \exercisecsetup{pdfdata=sheet}
```

Set title and author for pdf metadata; title is course plus material or sheet title; author is instructor or sheet author plus institution:

```
52 \ifdefined\metaset
53 \metaset[sep]{subtitle}{, }
```

```

54 \metaset{subtitle}{\ifsolutions\metatranslate[#1]{solutions} \fi%
55 \metaif[use]{sheettitle}
56   {\metapick[#1]{sheettitle}}
57   {\metapick[#1]{material}}}
58 \metaset{author}{\exerciseifempty{\getsheetdata{author}}%
59   {\metapick[#1]{instructor}}{\metapick[#1]{sheetauthor}},
60   \metapick[#1]{institution}}
61 \else
62 \exerciseconfig{composemetasheet}[2]{\getexercisedata{course},
63   \ifsolutions\getexerciseconfig{termsolutions} \fi%
64   \exerciseifempty{#2}{\getexerciseconfig{termsheet} #1}{#2}}
65 \exercisedata{title=%
66   {\getexercisedata{course},
67   \ifsolutions\getexercisedata{solutions} \fi%
68   \getexercisedata{material}}}
69 \exercisedata{author=%
70   {\getexercisedata{instructor}, \getexercisedata{institution}}}
71 \fi

```

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

```

72 \ifdefined\metaset
73 \metaset{institution}{Katharinen-Volksschule}
74 \metaset{de}{course}{Mathematik}
75 \metaset{en}{course}{Mathematics}
76 \metaset{instructor}{J.\ G.\ B\"uttner}
77 \metaset{period}{ca.\ 1786}
78 \metaset{de}{material}{\"Ubungsaufgaben}
79 \metaset{en}{material}{Exercise Problems}
80 \else
81 \exercisedata{institution={Katharinen-Volksschule}}
82 \exercisedata{course={Mathematik}}
83 \exercisedata{instructor={J.\ G.\ B\"uttner}}
84 \exercisedata{period={ca.\ 1786}}
85 \exercisedata{material={\"Ubungsaufgaben}}
86 \fi

```

Body.

```
87 \begin{document}
```

Start sheet number 5:

```
88 \begin{sheet}[number=5,label={sheet5}]
```

Start a problem with a title:

```
89 \begin{problem}[title={Sums},points=99+4]
```

Summary of points:

```

90 \exerciseloopstr{\getsubproblemlist{}}{c}%
91 \hfill\begin{tabular}{c|\exerciseloopret|c}
92 \exerciseloop{\getsubproblemlist{}}
93   {\&\ref{\getexerciseconfig{labelsubproblem}{#1}}}
94   {\&\ref{prob:\problemtag}}\hline
95 \getexerciseconfig{termpoints}
96 \exerciseloop{\getsubproblemlist{}}{\&\extractpoints{\getsubproblempoints{#1}}}
97 \&\extractpoints{\getproblempoints{}}
98 \

```

```

99 extra
100 \exerciseloop{\getsubproblemlist{}}{\&\extractpoints*\getsubproblempoints{#1}}
101 &\extractpoints*\getproblempoints{}}
102 \end{tabular}

```

Some introduction to the problem:

```

103 This problem deals with sums and series.

```

A subproblem with a local label:

```

104 \begin{subproblem}[points=2,difficulty=simple,label={\problemtag-simplesum}]
105 Compute the sum
106 \showpoints
107 \begin{equation}
108 1+2+3.
109 \end{equation}

```

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

```

110 \begin{solution}
111 The result is
112 \begin{equation}
113 1+2+3=6.
114 \end{equation}
115 \end{solution}

```

End subproblem:

```

116 \end{subproblem}

```

Another subproblem:

```

117 \begin{subproblem}[points=97+0.5,difficulty=lengthy]
118 Compute the sum
119 \begin{equation}
120 1+2+3+\ldots+98+99+100.
121 \end{equation}
122 Keep calm and calculate!
123 %%That ought to keep him occupied for a while
124 \end{subproblem}

```

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

```

125 \begin{solution}[author={C.\ F.\ Gau\ss}]
126 We use the result  $1+2+3=6$  from part \ref{\problemtag-simplesum})
127 to jumpstart the calculation. The remaining sums yield
128 \awardpoints*[1 for each remaining sum]{97}
129 \begin{equation}
130 6+4+5+\ldots+99+100=5050.
131 \end{equation}
132 Alternatively the summands can be grouped into pairs as follows:
133 \begin{align}
134 1+100&=101,\\
135 2+99&=101,\\
136 3+98&=101,\\
137 \ldots &\nonumber\\
138 50+51&=101.
139 \end{align}
140 These amount to 50 times the same number 101.
141 Therefore the sum equals

```

```

142 \begin{equation}
143 1+2+\ldots+99+100=50\cdot 101=5050.
144 \end{equation}
145 \textit{Ligget se!} \awardpoints{97+0.5}
146 \end{solution}

```

Some text between subproblems:

```

147 You may give the final part a try:

```

Final subproblem; this one is optional:

```

148 \begin{subproblem}[optional={optional},
149   difficulty={requires inspiration},points={+3.5}]
150 Compute the series
151 \showpoints
152 \begin{equation}
153 1+2+3+\ldots
154 \end{equation}

```

Provide a solution:

```

155 \begin{solution}
156 The series is divergent, so the result is  $\infty$  \awardpoints{+1}.
157 \par
158 However, after subtracting the divergent part,
159 the result clearly is
160 \begin{equation}
161 \zeta(-1)=-\frac{1}{12}\backslash,,
162 \end{equation}
163 where the zeta-function  $\zeta(s)$  is defined by
164 \begin{equation}
165 \zeta(s):=\sum_{k=1}^{\infty} \frac{1}{k^s}\backslash,.
166 \end{equation}
167 This definition holds only for  $s>1$  where the sum is convergent,
168 but one can continue the complex analytic function to  $s<0$ 
169 \awardpoints{+1.5}.
170 \par
171 Another way of understanding the result
172 is to use the indefinite summation formula
173 for arbitrary exponent  $s$  in the summand
174 (which also follows from the Euler--MacLaurin formula)
175 \begin{equation}
176 \sum_n n^s
177 = \frac{n^{s+1}}{s+1}
178 - \sum_{j=0}^s \frac{\zeta(j-s)\backslash,s!}{(s-j)!\backslash,j!}\backslash,n^j
179 = \ldots - \zeta(-s)\backslash,n^0.
180 \end{equation}
181 Curiously, the constant term with  $j=0$  is just the desired result
182 but with the wrong sign
183 (in fact, the constant term of an indefinite sum is ambiguous;
184 for the claim we merely set  $j=0$ 
185 in the expression which holds for others values of  $j$ )
186 \awardpoints{+0.5}.
187 In order to understand the sign,
188 we propose that the above formula describes the regularised result
189 for the sum with limits  $+\infty$  and  $n$ 
190 \begin{equation}
191 \sum_{k=+\infty}^n k^s
192 \simeq \frac{n^{s+1}}{s+1}
193 - \sum_{j=0}^s \frac{\zeta(j-s)\backslash,s!}{(s-j)!\backslash,j!}\backslash,n^j.

```

```

194 \end{equation}
195 Then we flip the summation limits of the desired sum
196 to bring it into the above form
197 \awardpoints{+0.5}
198 \begin{equation}
199 \sum_{k=1}^{\infty} k^s
200 = -\sum_{k=\infty}^0 k^s
201 \simeq \zeta(-s).
202 \end{equation}
203 \end{solution}

```

End subproblem:

```
204 \end{subproblem}
```

End problem:

```
205 \end{problem}
```

Another problem; this one is untitled:

```

206 \begin{problem}[points=1, difficulty=insane]
207 Show that the equation
208 \begin{equation}
209 a^3+b^3=c^3
210 \end{equation}
211 has no positive integer solutions.
212 \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):

```

213 \begin{solution}
214 \normalmarginpar
215 This is beyond the scope of this example.
216 \marginpar{\footnotesize\raggedright does not fit here.\par}
217 \end{solution}

```

Summary of points per problem and per subproblem for grading:

```

218 \ifsolutions\else
219 \textbf{Grading:}\par
220 \exerciseloopstr{\getproblemlist{}}{|c|}
221 \begin{tabular}{|c|\exerciseloopret||c|}\hline
222 \getexerciseconfig{termsheet} \ref{sheet5}
223 \exerciseloop{\getproblemlist{*}}
224 {&\ref{\getexerciseconfig{labelproblem}{#1}}}%
225 &total
226 \\ \hline
227 value
228 \exerciseloop{\getproblemlist{*}}
229 {&\extractpoints{\getproblempoints{#1}}}%
230 &\extractpoints{\getsheetpoints{}}
231 \\ \hline
232 \exerciseloop{\getproblemlist{*}}{&}
233 &\\ \hline
234 \end{tabular}\quad
235 \exerciseloop{\getproblemlist{*}}{
236 \exerciseloopstr{\getsubproblemlist{#1}}{|c|}
237 \ifnum\value{exerciseloop}>0\relax
238 \begin{tabular}{|c|\exerciseloopret||c|}\hline
239 \getexerciseconfig{termproblem} \ref{\getexerciseconfig{labelproblem}{#1}}

```

```

240 \exerciseloop{\getsubproblemlist{#1}}
241   {\&ref{\getexerciseconfig{labelsubproblem}{##1}}}}
242 &total
243 \\\hline
244 value
245 \exerciseloop{\getsubproblemlist{#1}}
246   {\&extractpoints{\getsubproblempoints{##1}}}%
247 &\extractpoints{\getproblempoints{#1}}
248 \\\hline
249 \exerciseloop{\getsubproblemlist{#1}}{\&}
250 &\\ \hline
251 \end{tabular}\quad
252 \fi
253 }
254 \fi

```

End sheet:

```
255 \end{sheet}
```

End of document body:

```
256 \end{document}
```

B Multipart Sample

The second example describes a series of exercise sheets which can be compiled as a collection or as individual sheets. This example describes a versatile setup with several convenient features; most of these features can be adjusted or removed easily as they mostly enhance the setup and do not interact with each other strongly.

B.1 Main File

The main source file is called `exfserm.tex`. It is referenced at several places within the setup, and when changing the name they need to be adjusted accordingly.

childdoc Mechanism. The setup uses the package `childdoc` to allow compilation of the series as a whole or in parts and with various sets of options:

```

257 \input{childdoc.def}
258 \childdocmain{}

```

Compilation Switches. Define compilation switches and declare their default settings. `\printsol` controls whether solutions should be printed or not; by default solutions are activated for compilation of a part, but not for the complete document. `\draftver` controls whether the final version of the document is to be compiled; concretely this affects the compilations of metapost figures, see below:

```

259 \ifchilddoc
260 \providecommand{\printsol}{y}
261 \else
262 \providecommand{\printsol}{n}
263 \fi
264 \providecommand{\draftver}{y}
265 \newif\ifdraft\if\draftver y\drafttrue\else\draftfalse\fi

```

Preamble. Standard document class:

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

267 % \ctanpkg{graphicx} package to display license logo:
268 \RequirePackage{graphicx}
```

hyperref Package. Use the hyperref package. Declare some options, e.g. use bookmarks only for complete document:

```
269 \PassOptionsToPackage{bookmarks=\ifchilddoc false\else true\fi}{hyperref}
270 \PassOptionsToPackage{bookmarksopen=true}{hyperref}
271 \RequirePackage{hyperref}
```

metastr Package. Use the metastr package to handle metadata and copyright information; disable usage of hyperxmp package if not installed; write auxiliary pdf metadata and rights information:

```
272 \ifdraft
273 \PassOptionsToPackage{draft}{metastr}
274 \fi
275 \IfFileExists{hyperxmp.sty}{\PassOptionsToPackage{hyperxmp=false}{metastr}}
276 \RequirePackage{course}{metastr}
277 \metaset{aux}{writepdf}{}
278 \metaset{rights}{writepdf}{}%
```

exframe Package. Invoke exframe with extended data and styles:

```
279 \RequirePackage[extdata,extstyle]{exframe}
```

Set solutions switch and declare two-sided layout only if no solutions are printed; do not use solution buffer in draft mode for easier identification of potential compile errors:

```
280 \if\printsol n
281 \exerciseshow{solutions=false}
282 \exerciseshow{twoside=true}
283 \else
284 \exerciseshow{solutions=true}
285 \exerciseshow{twoside=false}
286 \ifdraft
287 \exerciseshow{solutionbuf=false}
288 \fi
289 \fi%
```

Might want to display some metadata (only for partial compile):

```
290 %\if\printsol n\else\showproblem{author,source,recycle}\fi%
```

Set some options. Automatically assign labels to problems. Include sheets and problems in table of contents. Separate solutions by horizontal rules. Count problems, equations and pages by sheet (unless compiling single problem). Collect solutions below each problem. Write pdf metadata for sheet when compiling single sheet:

```
291 \exerciseshow{autolabelproblem}
292 \exerciseshow{contents,solutionsep}
293 \ifchilddocmanual\else
294 \exerciseshow{pagebysheet,problemby sheet,equationbysheet,sheetequation}
295 \fi
296 \exerciseshow{solutionbelow={problem}}
297 \ifchilddoc\ifchilddocmanual\else\exerciseshow{pdfdata=sheet}\fi\fi%
```


Language. Redefine some terms:

```
298 \metasetlang{en-GB}
299
300 \metasetterm[en]{sheet}{sheet}
301 \metasetterm[en]{sheets}{sample sheets}
302 \metasetterm[en]{solution}{Solution}
303 \metasetterm[en]{solutions}{solutions}
```

Layout Definitions. Set page dimensions and layout:

```
304 \RequirePackage[a4paper,margin=2.5cm]{geometry}
305 \pagestyle{plain}
```

Remove paragraph indentation:

```
306 \setlength\parindent{0pt}
307 \setlength\parskip{\smallskipamount}
```

Show overfull lines:

```
308 \setlength\overfullrule{5pt}
```

Define turn page over mark; hide when printing solutions:

```
309 \newcommand{\turnover}{\ifsolutions\else\vfill%
310 \hfill{\mathversion{bold}$\longrightarrow$\}\newpage\fi}
```

Sheet Banner. Use standard sheet banner; show sheet `editdate` below banner (if declared); when compiling single sheet, display sheet due date instead:

```
311 \exercisestyle{plainheader}
312 \exerciseconfig{composeheaderbelowright}
313 {\sheetdataempty{editdate}}{\version: \getsheetdata{editdate}}{ }
314 \ifchilddoc\ifsolutions\else
315 \exerciseconfig{composeheaderbelowright}
316 {\sheetdataempty{due}}{\due: \getsheetdata{due}}{ }
317 \fi\fi
```

Lorem. Define a macro `\lorem` to write out some paragraph of text for the example:

```
318 \def\lorem{Lorem ipsum dolor sit amet, consectetur adipisicing elit,
319 sed eiusmod tempor incididunt ut labore et dolore magna aliqua.
320 Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris
321 nisi ut aliquid ex ea commodi consequat.
322 Quis aute iure reprehenderit in voluptate velit esse
323 cillum dolore eu fugiat nulla pariatur.
324 Excepteur sint obcaecat cupiditat non proident,
325 sunt in culpa qui officia deserunt mollit anim id est laborum.\par}
```

metapost Setup. Use package `mpostinl` (if available) to include some metapost figures within the source of the problems:

```
326 \IfFileExists{mpostinl.sty}{\RequirePackage{mpostinl}}{ }
327 \ifdefined\mpostsetup
```

Setup `mpostinl`. Use checksums to invoke metapost only when figures change. Process all figures individually and immediately when in draft mode for child documents (avoids a

second compilation pass). Number figures within sheet to provide a stable numbering upon insertion/deletion of new figures or partial compilation:

```
328 \mpostsetup{checksum}
329 \ifchilddoc\ifdraft\mpostsetup{now,nowall}\fi\fi
330 \ifchilddocmanual\else\mpostsetup{numberwithin={sheet}}\fi
```

Global metapost definitions. Define some latex macro to demonstrate usage of latex for typesetting labels. Define some global metapost variables, `paths` is an array of path variables, `xu` serves as a length unit to scale individual figures (use `interim xu:=...`; to set scale for local figure only), `pensize` changes the size of the pen, `fillshape` fills and outlines a shape:

```
331 \mpostsetup{globaldef=true}
332 \begin{mposttex}
333 \def\figure{figure}
334 \end{mposttex}
335 \begin{mpostdef}
336 path paths[];
337 newinternal numeric xu;
338 xu:=1cm;
339 def pensize(expr s)=withpen pencircle scaled s enddef;
340 def fillshape(expr p,c)=
341   fill p withcolor c;
342   draw p
343 enddef;
344 \end{mpostdef}
345 \mpostsetup{globaldef=false}
```

Close optional mpostnl processing:

```
346 \fi
```

Document Data. Set document data for series of problem sheets:

```
347 \metaset{course}{exframe package samples}
348 \metaset{instructor}{N.\ Beisert}
349 \metaset{author}{Niklas Beisert, \metapick[#1]{institution}}
350 \metaset{institution}{exframe academy}
351 \metaset{period}{spring 2019}
352 \metaset{copyrightdate}{2019--2026}
```

For child documents declare part of:

```
353 \ifchilddoc
354 \metaset{partof}{\metatranslate[#1]{sheets} \metapick[#1]{course}}
355 \fi
```

Assemble some entries from given data:

```
356 \metaset{sep}{draft}{ -- }
357 \metaset{sep}{subtitle}{, }
358 \metaset{material}{\ifsolutions\metatranslate[#1]{solutions} \fi%
359   \metatranslate[#1]{sheets}}
360 \exerciseconfig{composetitlesheet}[2]{\exerciseifempty{#2}%
361   {\ifsolutions\metaterm{solutions}\else\metaterm{sheet}\fi \ #1}%
362   {\ifsolutions\metaterm{solutions} \fi \ #2}}
363 \metaset{sheettitle}{\ifsolutions\metatranslate[#1]{solutions} \fi%
364   \exerciseifempty{\getsheetdata{rawtitle}}%
365   {\metatranslate[#1]{sheet} \thesheet}%

```

```

366   {\getsheetdata{rawtitle}}
367 \metaset{subject}{Lecture Series,
368   \metapick[#1]{institution}, \metapick[#1]{period}}

```

License. It is good practice to specify a copyright line and a license for the document:

```

369 \metaset{copyrightowner}{\metapick[#1]{author}}
370 \ifchilddoc
371 \metacopyright{doc}
372 \else
373 \metacopyright{doc-parts}
374 \fi
375 \metaset{licenseprovider}{of \metapick[#1]{institution}}
376 \metalicense{consent}

```

Apply Creative Commons BY-SA license under certain conditions (no solutions, final version):

```

377 \ifsolutions\else\ifdraft\else
378 \metalicensecc{by-sa}
379 \fi\fi

```

Body. Start document body:

```

380 \begin{document}

```

Single Problem Display. The following code handles the compilation of individual problems from their own source file. Make sure to leave the conditional before issuing `\end{document}`:

```

381 \def\tmp{}
382 \ifchilddocmanual
383 \def\tmp{\end{document}}
384 \input{\childdocname}
385 \fi\tmp

```

Frontmatter. Do not print frontmatter for individual sheets. Define a plain page counter for frontmatter:

```

386 \setcounter{section}{-1}
387 \begingroup\ifchilddoc\else
388 \renewcommand{\thepage}{\arabic{page}}

```

Prepare a title page to display some relevant data:

```

389 \pdfbookmark[1]{\metaterm{title}}{title}
390 \thispagestyle{empty}
391 \vspace*{\fill}
392 \begin{center}
393 \metapick{course}{titletext}
394 \end{center}
395 \vspace*{\fill}\vspace*{\fill}
396 \newpage

```

Prepare a copyright and license page using data specified above:

```

397 \phantomsection\pdfbookmark[1]{\metaterm{copyright}}{copyright}
398 \thispagestyle{empty}

```

```

399 \vspace*{\fill}\vspace*{\fill}
400 \begin{center}\begin{minipage}{11cm}\raggedright
401 \metapick[print]{rightstext}
402 \end{minipage}\end{center}
403 \vspace*{\fill}\vspace*{\fill}\vspace*{\fill}
404 \newpage

```

Print table of contents:

```

405 \makeatletter\renewcommand\@pnumwidth{2.4em}\makeatother
406 \setcounter{tocdepth}{2}
407 \phantomsection\pdfbookmark[1]{\metaterm{contents}}{contents}
408 {\parskip0pt\tableofcontents}
409 \exercisecleardoublepage\setcounter{page}{1}

```

End of frontmatter:

```

410 \fi\endgroup

```

Include Sheets. Include problem sheets:

```

411 \include{exfser01}
412 \include{exfser02}
413 \include{exfser03}

```

Include sheet to collect unused problems (only process for individual sheets, i.e. itself):

```

414 \def\jobnameunused{exfseraa}
415 \ifx\childdocname\jobnameunused\include{\jobnameunused}\fi

```

End. End of document body:

```

416 \end{document}

```

B.2 Sheet File

Provide some source files `exfser01.tex`, `exfser02.tex`, `exfser03.tex`, `exfseraa.tex` for problem sheets.

childdoc Mechanism. Instruct the package `childdoc` to compile only the present sheet if source is compiled by latex:

```

417 %\providecommand{\printsol}{n}
418 \input{childdoc.def}
419 \childdocof{exfserm}

```

The parameter of `\childdocof` must match the main file name `exfserm`. Uncommenting the commented line suppressed printing of the solution.

Sheet Environment. Declare a sheet with intended due date:

```

420 \begin{sheet}[due={2019-04-29}]

```

Adjust due date for each sheet. For sheet containing unused problems `exfseraa.tex`, declare a sheet `title={unused problems}` instead of due date.

Problems. Start a problem:

```
421 \begin{problem}[title={Sample A}]
```

Let us declare a figure using `mpostinl` (if available). Denote it by the label `tag-fig`, where `tag` is the tag of the problem (in order to avoid potential conflicts with other problems; `tag` is assigned automatically or by specifying the option `tag` for the `problem` environment). Also declare a figure `tag-solfig` for use within the solution:

```
422 \ifdefined\mpostuse
423 \begin{mpostfig}[label={\problemtag-fig}]
424 interim xu:=1.5cm;
425 paths[1]:=fullcircle scaled 1xu;
426 fillshape(paths[1], 0.7white) pensize(1pt);
427 label(btex \figure etex, center(paths[1]));
428 \end{mpostfig}
429 \begin{onlvsolutions}
430 \begin{mpostfig}[label={\problemtag-solfig}]
431 interim xu:=1.5cm;
432 paths[1]:=fullcircle scaled 1xu;
433 paths[2]:=((subpath (-2,2) of paths[1])--cycle) shifted (+0.5xu,0);
434 paths[3]:=((subpath (2,6) of paths[1])--cycle) shifted (-0.5xu,0);
435 fillshape(paths[2], 0.7white) pensize(1pt);
436 fillshape(paths[3], 0.7white) pensize(1pt);
437 \end{mpostfig}
438 \end{onlvsolutions}
439 \fi
```

Write a problem body with figure, some subproblems and a solution:

```
440 \lorem
441
442 \begin{subproblem}
443 \lorem
444 \begin{center}
445 \ifdefined\mpostuse\mpostuse{\problemtag-fig}\else figure\fi
446 \end{center}
447 \lorem
448 \end{subproblem}
449
450 \begin{solution}
451 \lorem
452 \begin{center}
453 \ifdefined\mpostuse\mpostuse{\problemtag-solfig}\else figure\fi
454 \end{center}
455 \lorem
456 \end{solution}
457
458 \begin{subproblem}
459 \lorem
460 \end{subproblem}
461
462 \begin{solution}
463 \lorem
464 \end{solution}
465
466 \lorem
467
468 \begin{subproblem}
469 \lorem
```

```

470 \end{subproblem}
471
472 \begin{solution}
473 \lorem
474 \end{solution}

```

End the problem:

```
475 \end{problem}
```

Start new page:

```
476 \turnover
```

Write a second problem to accompany the first one:

```
477 \begin{problem}
```

Problem body without a figure; this time the `solution` environments are included in the `subproblem` environments:

```

478 \lorem
479
480 \begin{subproblem}
481 \lorem
482 \begin{solution}
483 \lorem
484 \end{solution}
485 \end{subproblem}
486
487 \begin{subproblem}
488 \lorem
489 \begin{solution}
490 \lorem
491 \end{solution}
492 \end{subproblem}

```

End the problem:

```
493 \end{problem}
```

End Sheet. End the sheet:

```
494 \end{sheet}
```

B.3 Individual Problem Files

It may be more convenient to define each problem in an individual file, so that a sheet can be composed by including the appropriate problem files. In such a setup, the `childdoc` mechanism allows to compile each problem individually.

To that end, prepare a file `exfserpnn.tex` containing the `problem` environment. This file should start with:

```

495 %%\providecommand{\printsol}{n}
496 \input{childdoc.def}
497 \childdocby{exfserm}

```

and end with `\endinput`. Then compose the problem sheet by including the appropriate set of problem files via `\input{exfserpnn}`.

In the example, a sheet file `exfser03.tex` includes two problem files `exfserpe.tex` and `exfserpf.tex`.

B.4 Make Scripts

The setup allows the compilation in various modes for editing purposes. In order to generate and update a complete set of documents for distribution (all individual sheets and a collection of sheets; with and without solutions), it makes sense to use the software development utility `make`. The compilation of individual components is simplified by a `bash` shell script.

Compile Script. The `bash` shell script `exfsermk.sh` compiles one part of the collection of problem sheets in a given mode.

Shebang for `bash` script:

```
498 #!/bin/bash
```

Configure and declare variables with default values. `srcmain` defines the name of the main source file; `srcsecnn` defines the name of the sheet source file; `trglist` defines the target file names; `trgsol` defines the target modes; `sheets` is a list of allowable sheet identifiers `nn`:

```
499 srcmain="exfserm"
500 srcsec="exfser"
501 trglist=(Problems Solutions)
502 trgsol=(n y)
503 secnum="01 02 03 aa"
```

Display usage:

```
504 if [ -z $1 ]
505 then
506     echo "Usage:
507     $0 number [version]
508         number: number of sheet, 0 for combined document
509         version: 0 for problems, 1 for solutions
510     $0 filename
511         filename: target file to be compiled"
512     exit 1
513 fi
```

Configure and declare variables with default values. `num` takes the sheet number; `ver` takes the compile mode;

```
514 num="$1"
515 ver="$2"
516 nl=$'\n'
517 secokay=""
518 make=".pdf"
```

Check if the parameter matches any of the acceptable output file names:

```
519 for v in "${trglist[@]}"
520 do
521     if [[ $num =~ ^$v ]]
522     then
523         ver=$v
524         num=${num#$v}
```

```

525     if [[ $num =~ ^.*\.tex$ ]]; then make=".tex"; fi
526     num=${num%.*}
527 fi
528 done

```

Ensure that num is a two-digit number, prepend '0' otherwise:

```

529 if [[ $num =~ ^[0-9]$ ]]; then num="0$num"; fi
530 if [[ $num == "00" ]]; then num=""; fi

```

Check whether num is acceptable:

```

531 if [[ -z $num ]]; then secokay="okay"; fi
532 for v in $secnum
533 do
534     if [[ "$num" == "$v" ]]; then secokay="okay"; fi
535 done

```

Otherwise display error message and exit:

```

536 if [[ -z $secokay ]]
537 then
538     echo "error: unknown sheet"
539     exit 1
540 fi

```

Disable newline character for command line tex code:

```

541 if [[ "$make" == ".pdf" ]]; then nl=""; fi

```

Function to compile a component. Set up childdoc mechanism according to desired component. Compile two passes, first in `-draftmode`. Suppress messages by `mpost`. Display warning messages in log file:

```

542 function docompile
543 {
544     if [[ -z $num ]]
545     then
546         job="$srcmain"
547         fwd="\childdocforward{$srcmain}"
548     else
549         job="$srcsec$num"
550         fwd="\childdocforward[$srcmain]{$srcsec$num}"
551     fi
552     body="\def\jobname{$job}$optdef\input{childdoc.def}$fwd"
553     for pass in first main extra
554     do
555         par="";
556         if [[ "$pass" == "first" ]]; then par="-draftmode"; fi
557         drop="This is|entering extended mode|\write18"
558         drop="$drop|Preloading the plain mem file|mpost\.mp|plain\.mp"
559         pdflatex -shell-escape -interaction=batchmode $par \
560             -jobname "$trg" "$body" | grep -vE "$drop"
561         if [[ "$pass" != "main" ]]; then continue; fi
562         if ! (grep -E -q "may have changed|rerunfilecheck Warning" "$trg.log")
563         then break; fi
564     done
565     grep -E "^! |Warning|Error|Undefined|Overfull|Underfull" "$trg.log"
566 }

```

Function to generate a childdoc compile file with specific options:


```

567 function writesource
568 {
569   if [[ -z $num ]]
570   then
571     fwd="\childdocforward{$srcmain}"
572   else
573     fwd="\childdocforwardprefix{$srcmain}{$target}{$srcsec}"
574   fi
575   body="$optdef\\input{childdoc.def}$nl$fwd"
576   echo "$body" > $trg.tex
577 }

```

Translate versions to parameter values, configure variables and select appropriate function for intended task:

```

578 for i in "${!trglist[@]}"
579 do
580   if [[ -z $ver || "$ver" == "${trglist[$i]}" || $ver = $i ]]
581   then
582     target="${trglist[$i]}"
583     sol="${trgsol[$i]}"
584     trg="$target$num"
585     optdef="\def\\draftver{n}$nl\\def\\printsol{$sol}$nl"
586     if [[ "$make" == ".pdf" ]]; then doccompile; else writesource; fi
587   fi
588 done

```

Finish with blank line:

```

589 echo

```

Makefile. Define a make file `exfsermk.mak` for the project. The compilation is then started by `make -f exfsermk.mak ...`. More conveniently, in a single-project setup within the directory, this file would be called `Makefile` in which case it suffices to just run `make ...`.

Configuration definitions:

```

590 SRCMAIN = exfserm
591 SRCSEC = exfser
592 SRCPRB = exfserp
593 SCRIPT = exfsermk.sh
594 MAKEFILE = exfsermk.mak
595 TRGLIST = Problems Solutions
596 SECNUM = 01 02 03 aa
597 PREREQS = $(SRCMAIN).tex
598
599 SRCSECFILES = $(SECNUM:%=$(SRCSEC)%.tex)
600 TRGMAINFILES = $(foreach trg,$(TRGLIST),$(trg).pdf)
601 TRGSECFILES = $(foreach trg,$(TRGLIST),$(trg).pdf $(SECNUM:%=$(trg)%.pdf))
602 GENFILES = $(foreach trg,$(TRGLIST),$(trg).tex $(SECNUM:%=$(trg)%.tex))
603 BAKFILES = $(PREREQS) $(SRCSECFILES) $(GENFILES)\
604             $(MAKEFILE) $(SCRIPT) $(SRCPRB)*

```

Define some abstract targets; `default` is the default target when no parameters are given:

```

605 default: sheets ;
606 main: $(TRGMAINFILES) ;
607 sheets: $(TRGSECFILES) ;
608 sheet%: $(foreach trg,$(TRGLIST),$(trg)%.pdf) ;

```

```

609 all: main sheets ;
610 sources: $(GENFILES) ;

```

Compile particular files via `exfsermk.sh` bash script. Note that command lines have to start with a tab character (represented by 8 spaces here):

```

611 $(TRGMAINFILES): $(SRCSECFILES) $(PREREQS)
612   bash ./$(SCRIPT) $@
613 $(word 1,$(TRGLIST)).pdf: $(SRCSEC).tex $(PREREQS)
614   bash ./$(SCRIPT) $@
615 $(word 2,$(TRGLIST)).pdf: $(SRCSEC).tex $(PREREQS)
616   bash ./$(SCRIPT) $@
617 $(GENFILES):
618   bash ./$(SCRIPT) $@

```

Touch main file for recompile:

```

619 touch:
620   touch $(SRCMAIN).tex

```

Define `clean` target to remove all intermediate compilation files:

```

621 clean:
622   rm -f $(foreach ext,.aux .log,$(SECNUM:%=$(SRCSEC)%$(ext)))
623   rm -f $(foreach trg,$(TRGLIST),$(SECNUM:%=$(trg).log) $(trg).log)
624   rm -f $(foreach ext,.aux .log .out .toc,$(SRCMAIN)$(ext))
625   rm -f $(foreach ext,.mp .mpx -*.mps,$(SRCMAIN)$(ext))
626   rm -f $(foreach ext,-tmp.log -tmp.mp -tmp.mpx,$(SRCMAIN)$(ext))
627   rm -f mpxerr.tex mpxerr.log mpxerr.dvi texput.log
628   rm -f $(patsubst %,$(SRCPRB)*%,.aux .log .mp .mpx -*.mps)
629   rm -f $(patsubst %,$(SRCPRB)*%,-tmp.log -tmp.mp -tmp.mpx)

```

Define `clean-bak` target to remove all backup files ending in ‘~’ or ‘.bak’:

```

630 clean-bak:
631   rm -f $(BAKFILES:%=%~) $(BAKFILES:%=%.bak)

```

Define `clean-all` target to remove all generated files for a clean source directory:

```

632 clean-all: clean
633   rm -f $(TRGSECFILES) $(TRGMAINFILES) $(GENFILES)
634   rm -f $(SECNUM:%=$(SRCSEC).pdf) $(SRCMAIN).pdf
635   rm -f $(SRCPRB)*.pdf

```

C Implementation

This section describes the package `exframe.sty`.

C.1 General Definitions

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

```

636 \RequirePackage{verbatim}
637 \RequirePackage{xkeyval}

```

General Definitions.

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

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

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

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

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

```
640 \long\def\exf@expswitch#1#2{#2{#1}}
641 \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`

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

`\exf@csor` A macro to return `\csname` or default if not exists:

```
644 \def\exf@csor#1#2{\ifcsname#1\endcsname\csname#1\endcsname\else#2\fi}
```

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

`\exf@prepend@def`

```
645 \long\def\exf@append@def#1#2{\exf@exptwo\def#1\expandafter{#1#2}}
646 \long\def\exf@prepend@switch#1#2#3{#2{#3#1}}
647 \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:

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

`\exf@isif` Execute #3 if content of macro #1 equals #2:

```
650 \newcommand{\exf@ifis}[3]{%
651   {\def\exf@tmp{#2}\ifx#1\exf@tmp#3\fi}}
```

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

```
652 \newcommand{\exf@href}[2]{%
653   \ifdefined#1\ifx#1\exf@empty#2\else%
654     \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

`\exf@ensuretext` case the latter package is loaded. `exf@ensuretext` makes sure the text is set in text mode or within an `\mbox` in math math:

```
655 \newcommand{\exf@text}[1]{\ifdefined\text\text{#1}\else#1\fi}
656 \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:

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

Auxiliary File Data Storage.

`\exf@notedata` Process stored data by handler:

```
659 \newcommand{\exf@notedata}[3]{\csname exf@notedata@#1\endcsname{#2}{#3}}
```

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

```
660 \AtBeginDocument{\immediate\write\@auxout{%  
661   \string\providecommand{\string\exf@notedata}[3]{}}}
```

`\exf@writedata` Write data to the .aux file:

```
662 \newcommand{\exf@writedata}[3]{%  
663   {\immediate\write\@auxout{\string\exf@notedata{#1}{#2}{#3}}}
```

C.2 Package Setup

Initialisation Options.

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

Configure names of main environments and counters:

```
664 \def\exf@problemname{problem}  
665 \def\exf@subproblemname{sub\exf@problemname}  
666 \def\exf@solutionname{solution}  
667 \def\exf@sheetname{sheet}  
668 \def\exf@problemcounter{problem}  
669 \def\exf@subproblemcounter{sub\exf@problemcounter}  
670 \def\exf@solutioncounter{solution}  
671 \def\exf@sheetcounter{sheet}  
672 \define@key{exframe.sty}{problemenv}{\def\exf@problemname{#1}}  
673 \define@key{exframe.sty}{subproblemenv}{\def\exf@subproblemname{#1}}  
674 \define@key{exframe.sty}{solutionenv}{\def\exf@solutionname{#1}}  
675 \define@key{exframe.sty}{sheetenv}{\def\exf@sheetname{#1}}  
676 \define@key{exframe.sty}{problemcounter}{\def\exf@problemcounter{#1}}  
677 \define@key{exframe.sty}{subproblemcounter}{\def\exf@subproblemcounter{#1}}  
678 \define@key{exframe.sty}{solutioncounter}{\def\exf@solutioncounter{#1}}  
679 \define@key{exframe.sty}{sheetcounter}{\def\exf@sheetcounter{#1}}
```

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

```
680 \define@boolkey{exframe.sty}[exf@]{extdata}[true]{}  
681 \define@boolkey{exframe.sty}[exf@]{extstyle}[true]{}
```

Whether to load package metastr (available while loading only):

```
682 \define@boolkey{exframe.sty}[exf@]{metastr}[true]{}
```

Setup Options.

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

Main switch for solutions:

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

Switch for writing pdf metadata:

```
684 \define@choicekey{exf@setup}{pdfdata}%
```

```

685 {auto,manual,sheet,off}[auto]{\def\exf@metadata{#1}}
686 \def\exf@metadata{auto}

```

Write line number indicators to output file:

```

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

```

Prepare two-sided sheets:

```

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

```

Generate hyperreferences from solutions to corresponding problems:

```

689 \define@boolkey{exf@setup}[exf@]{solutionhref}[true]{}
690 \exf@solutionhreftrue

```

Automatically generate labels for sheets and problems:

```

691 \define@boolkey{exf@setup}[exf@]{autolabelsheet}[true]{}
692 \define@boolkey{exf@setup}[exf@]{autolabelproblem}[true]{}

```

Write warning message to document for better detection of inconsistencies:

```

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

```

Activate buffering of solutions:

```

694 \define@boolkey{exf@setup}[exf@]{solutionbuf}[true]{}
695 \exf@solutionbuftrue

```

Activate buffering of problems:

```

696 \define@boolkey{exf@setup}[exf@]{problembuf}[true]{}

```

Activate buffering of subproblems:

```

697 \define@boolkey{exf@setup}[exf@]{subproblembuf}[true]{}

```

`\exf@emptytestchar` Redefine character for testing emptiness:

```

698 \def\exf@emptytestchar{&}
699 \define@key{exf@setup}{emptytestchar}{\def\exf@emptytestchar{#1}}

```

Processing. Process global options while loading package:

```

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

```

`\exercisestop` Configure package when package is already loaded:

```

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

```

Additional Packages. Load metastr package if desired:

```

702 \ifexf@metastr
703 \PassOptionsToPackage{course=true}{metastr}
704 \RequirePackage{metastr}
705 \fi

```

Solutions Only Processing.

`onlysolutions` Process block only in solutions mode:

```

706 \newenvironment{onlysolutions}%
707 {\ifsolutions\else%
708   \let\endonlysolutions\endcomment%
709   \expandafter\comment\fi}%
710 {}

```

C.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:

```

711 \newcommand{\exerciseconfig}[1]{%
712   \@ifnextchar[{\exf@configopt{#1}}{\exf@confignoopt{#1}}}%
713   \long\def\exf@configopt#1[#2]#3{%
714     \exf@csdo\def{exf@config@#1}{#3}%
715     \exf@csdo\renewcommand{exf@config@#1}[#2]{#3}}%
716   \long\def\exf@confignoopt#1#2{\exf@csdo\def{exf@config@#1}{#2}}

```

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

```

\exerciseconfigprepend
717 \newcommand{\exerciseconfigappend}[2]{%
718   \exf@csdo\exf@append@def{exf@config@#1}{#2}}
719 \newcommand{\exerciseconfigprepend}[2]{%
720   \exf@csdo\exf@prepend@def{exf@config@#1}{#2}}

```

`\getexerciseconfig` Get configuration macro:

```

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

```

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

```

722 \newcommand{\exerciseconfigempty}[3]{\exf@csdo\ifx{exf@config@#1}\exf@empty%
723   #2\else#3\fi}

```

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

```

\exerciseifnotempty
724 \long\def\exerciseifempty#1#2#3{%
725   \if\exf@emptytestchar#1\exf@emptytestchar#2\else#3\fi}
726 \long\def\exerciseifnotempty#1#2{%
727   \if\exf@emptytestchar#1\exf@emptytestchar\else#2\fi}

```

Terms.

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

```

728 \exerciseconfig{termsheet}{Sheet}
729 \exerciseconfig{termsheets}{Sheets}
730 \exerciseconfig{termproblem}{Problem}
731 \exerciseconfig{termproblems}{Problems}

```

```

732 \exerciseconfig{termsolution}{Solution}
733 \exerciseconfig{termsolutions}{Solutions}
734 \exerciseconfig{termpoint}{point}
735 \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:

```
736 \exerciseconfig{styletext}{\normalsize\normalfont}
```

Style for problems:

```
737 \exerciseconfig{styletextproblem}{{}}
```

Style for solutions:

```
738 \exerciseconfig{styletextsolution}{\footnotesize}
```

Basic style for titles:

```
739 \exerciseconfig{styletitle}{\bfseries}
```

Style for problem titles:

```
740 \exerciseconfig{styletitleproblem}{\large}
```

Style for subproblem titles:

```
741 \exerciseconfig{styletitlesubproblem}{{}}
```

Style for solution titles:

```
742 \exerciseconfig{styletitlesolution}{{}}
```

Style for problem section title in solution block:

```
743 \exerciseconfig{styletitlesolutionsproblem}{\small}
```

Style for solution block title:

```
744 \exerciseconfig{styletitlesolutions}{\normalsize}
```

Style for problem block title:

```
745 \exerciseconfig{styletitleproblems}{\Large}
```

Spacing.

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

Space above problem environment:

```
746 \exerciseconfig{skipproblemabove}{3.25ex plus 1ex minus 1.5ex}
```

Space below problem environment:

```
747 \exerciseconfig{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):

```
748 \exerciseconfig{skipproblemtitle}{3pt plus 1pt minus 1pt}
```

Horizontal space between items in the problem opening line:

```
749 \exerciseconfig{skipproblemitem}{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:

```
750 \exerciseconfig{skipproblemitem}{0pt}
```

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

```
751 \exerciseconfig{skipsubproblemabove}{1.5ex plus 0.5ex minus 1ex}
```

```
752 \exerciseconfig{skipsubproblembelow}{1.5ex plus 0.5ex minus 1ex}
```

```
753 \exerciseconfig{skipsubproblemtitle}{-1em}
```

```
754 \exerciseconfig{skipsubproblemitem}{0.25em}
```

```
755 \exerciseconfig{skipsubproblemitem}{-1pt}
```

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

```
756 \exerciseconfig{skipsolutionabove}{0ex}
```

```
757 \exerciseconfig{skipsolutionbelow}{1.5ex plus 0.5ex minus 1ex}
```

```
758 \exerciseconfig{skipsolutiontitle}{-0.5em}
```

```
759 \exerciseconfig{skipsolutioninfo}{0.25em}
```

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

```
760 \exerciseconfig{skipsolutionitem}{0pt}
```

```
761 \exerciseconfig{skipsolutionitemsub}{0pt}
```

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

```
762 \exerciseconfig{skipsolutionsabove}{1.5ex plus 0.5ex minus 1ex}
```

```
763 \exerciseconfig{skipsolutionsbelow}{1.5ex plus 0.5ex minus 1ex}
```

Space above problem titles in a solution block:

```
764 \exerciseconfig{skipsolutionsproblemabove}{1.0ex plus 0ex minus 0.5ex}
```

Space following problem titles in a solution block (with legacy definition):

```
765 \exerciseconfig{skipsolutionsproblemtitle}{1.0ex plus 0ex minus 0.5ex}
```

```
766 \exerciseconfig{skipsolutionsproblem}{\exf@config@skipsolutionsproblemtitle}
```

Space following title of a solution block:

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

Spaces related to problem blocks; space above and below a problem block:

```
768 \exerciseconfig{skipproblemsabove}{1.5ex plus 0.5ex minus 1ex}
```

```
769 \exerciseconfig{skipproblemsbelow}{1.5ex plus 0.5ex minus 1ex}
```

Space following title of a problem block:

```
770 \exerciseconfig{skipproblemstitle}{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:

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

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

```
772 \exerciseconfig{insertsheetclearpage}{\exercisecleardoublepage}
```

Code to insert before a sheet is displayed:

```
773 \exerciseconfig{insertsheetbefore}{}
```

Code to insert after a sheet is displayed:

```
774 \exerciseconfig{insertsheetafter}{}
```

Code to insert before a solution block is displayed:

```
775 \exerciseconfig{insertsolutionsbefore}{}
```

Code to insert after a solution block is displayed:

```
776 \exerciseconfig{insertsolutionsafter}{}
```

Code to insert before a problem block is displayed:

```
777 \exerciseconfig{insertproblemsbefore}{}
```

Code to insert after a problem block is displayed:

```
778 \exerciseconfig{insertproblemsafter}{}
```

Code to insert before a problem is displayed:

```
779 \exerciseconfig{insertproblembefore}{}
```

Code to insert after a problem is displayed:

```
780 \exerciseconfig{insertproblemafter}{}
```

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

```
781 \exerciseconfig{insertproblemsolution}{}
```

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

```
782 \exerciseconfig{insertprobleminfo}{}
```

Code to determine the display of problems:

```
783 \exerciseconfig{insertproblemsselect}[1]{}
```

Analogous definitions for subproblems:

```
784 \exerciseconfig{insertsubproblembefore}{}
```

```
785 \exerciseconfig{insertsubproblemafter}{}
```

```
786 \exerciseconfig{insertsubprobleminfo}{}
```

```
787 \exerciseconfig{insertsubproblemsolution}{}
```

```
788 \exerciseconfig{insertsubproblemsselect}[1]{}
```

Analogous definitions for solutions:

```
789 \exerciseconfig{insertsolutionbefore}{}
```

```
790 \exerciseconfig{insertsolutionafter}{}
```

```
791 \exerciseconfig{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:

```
792 \exerciseconfig{composeitemsep}{\ }
```

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

```
793 \exerciseconfig{composetitlesheet}[2]%  
794 {\exerciseifempty{#2}{\getexerciseconfig{termsheet} #1}{#2}}
```

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

```
795 \exerciseconfig{composemetasheet}[2]%  
796 {\getexerciseconfig{composetitlesheet}{#1}{#2}}
```

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

```
797 \exerciseconfig{composetocsheet}[2]%  
798 {\exerciseifempty{#2}{\getexerciseconfig{termsheet} #1}{#2}}
```

Problem item separator, delimiter:

```
799 \exerciseconfig{composeitemproblemsep}{\getexerciseconfig{composeitemsep}}  
800 \exerciseconfig{composeitemproblemdelim}{.}
```

Compose problem item; argument is problem number:

```
801 \exerciseconfig{composeitemproblem}[1]%  
802 {#1\getexerciseconfig{composeitemproblemdelim}}
```

Unnamed problem separator, delimiter:

```
803 \exerciseconfig{composebareproblemsep}{ }  
804 \exerciseconfig{composebareproblemdelim}{}
```

Compose unnamed problem title; arguments are problem number and name; includes exception for legacy behaviour:

```
805 \exerciseconfig{composebareproblem}[1]%  
806 {\getexerciseconfig{termproblem}\getexerciseconfig{composebareproblemsep}%  
807 \ifx\exf@config@composebareproblemdelim\exf@config@composeitemproblemdelim%  
808 \getexerciseconfig{composeitemproblem}{#1}\else%  
809 #1\getexerciseconfig{composebareproblemdelim}\fi}
```

Named problem separator, delimiter:

```
810 \exerciseconfig{composenamedproblemsep}{ }  
811 \exerciseconfig{composenamedproblemdelim}{.}
```

Compose named problem title; arguments are problem number and name; includes exception for legacy behaviour:

```
812 \exerciseconfig{composenamedproblem}[2]%  
813 {\ifx\exf@config@composenamedproblemdelim\exf@config@composeitemproblemdelim%  
814 \getexerciseconfig{composeitemproblem}{#1}\else%  
815 #1\getexerciseconfig{composenamedproblemdelim}\fi%  
816 \getexerciseconfig{composenamedproblemsep}#2}
```

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”:

```
817 \exercisecfg{composetitleproblem}[2]{\exerciseifempty{#1}%
818   {\exerciseifempty{#2}{#2}}}%
819   {\exerciseifempty{#2}{\getexercisecfg{composebareproblem}{#1}}}%
820   {\getexercisecfg{composenamedproblem}{#1}{#2}}}
```

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

```
821 \exercisecfg{composetocproblem}[2]%
822   {\exerciseifempty{#2}{\getexercisecfg{termproblem} #1}%
823   {#1\getexercisecfg{composenamedproblem} #2}}
```

Subproblem item separator, delimiter:

```
824 \exercisecfg{composeitemsubproblemsep}{\getexercisecfg{composeitemsep}}
825 \exercisecfg{composeitemsubproblem}{}%
```

Compose subproblem item; argument is subproblem number:

```
826 \exercisecfg{composeitemsubproblem}[1]%
827   {#1\getexercisecfg{composeitemsubproblem}}%
```

Compose subproblem title; argument is subproblem number:

```
828 \exercisecfg{composetitlesubproblem}[1]%
829   {\getexercisecfg{composeitemsubproblem}{#1}}%
```

Solution item separator, delimiter:

```
830 \exercisecfg{composeitemsolutionsep}{\getexercisecfg{composeitemsep}}
831 \exercisecfg{composeitemsolution}{}%
```

Compose solution item; arguments are problem and subproblem number:

```
832 \exercisecfg{composeitemsolution}[2]{%
833   \exercisecfg{composeitemsolutionsub}[2]%
834   {\getexercisecfg{composeitemsubproblem}{#2}}%
```

Compose solution item for first solution in a problem section; arguments are problem and subproblem number:

```
835 \exercisecfg{composeitemsolutionfirst}[2]%
836   {\getexercisecfg{composeitemsolution}{#1}{#2}}%
837 \exercisecfg{composeitemsolutionfirstsub}[2]%
838   {\getexercisecfg{composeitemsolutionsub}{#1}{#2}}%
```

Compose title for single solution (solution intro if `solutionbelow` is `here` or `subproblem`); arguments are corresponding problem and subproblem number:

```
839 \exercisecfg{composetitlesolutionsingle}[2]%
840   {\getexercisecfg{termsolution}%
841   \getexercisecfg{composeitemsolution}}%
```

Compose title for one out of several solutions (solution intro if `solutionbelow` is `problem`, `sheet` or `manual`); arguments are corresponding problem and subproblem number:

```
842 \exercisecfg{composetitlesolutionproblem}[1]{%
843   \exercisecfg{composetitlesolutionsubproblem}[2]%
844   {\getexercisecfg{composeitemsubproblem}{#2}}%
845 \exercisecfg{composetitlesolutionmulti}[2]{\exerciseifempty{#2}%
846   {\getexercisecfg{composetitlesolutionproblem}{#1}}%
847   {\getexercisecfg{composetitlesolutionsubproblem}{#1}{#2}}%
```

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

```
848 \exerciseconfig{composetocsolution}[2]%  
849 {\getexerciseconfig{composetocproblem}{#1}{#2}}
```

Compose title for solution block:

```
850 \exerciseconfig{composetitlesolutions}%  
851 {\getexerciseconfig{termsolutions}}
```

Compose title for problem block:

```
852 \exerciseconfig{composetitleproblems}%  
853 {\getexerciseconfig{termproblems}}
```

Compose table of contents line for solution block:

```
854 \exerciseconfig{composetocsolutions}%  
855 {\getexerciseconfig{composetitlesolutions}}
```

Compose table of contents line for problem block:

```
856 \exerciseconfig{composetocproblems}%  
857 {\getexerciseconfig{composetitleproblems}}
```

Compose sectional title for solution following a single problem (problem section intro if `solutionbelow` is `problem`); arguments are problem number and raw title:

```
858 \exerciseconfig{composetitlesolutionsproblemsingle}[2]%  
859 {\getexerciseconfig{termsolution}}
```

Compose sectional title for solution of one problem within a solution block (problem section intro if `solutionbelow` is `sheet` or `manual`); arguments are problem number and raw title:

```
860 \exerciseconfig{composetitlesolutionsproblemmulti}[2]%  
861 {\exerciseifempty{#2}{\getexerciseconfig{composebareproblem}{#1}}}%  
862 {\getexerciseconfig{composenamedproblem}{#1}{#2}}}
```

Compose label:

```
863 \exerciseconfig{composeitemsolutionlabel}[2]{#1#2}
```

Points. Compose number of points:

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

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

```
865 \exerciseconfig{composepoints}[1]{\getexerciseconfig{composepointsnum}{#1}}~%  
866 \ifdim #1pt=1pt\getexerciseconfig{termpoint}%  
867 \else\getexerciseconfig{termpoints}\fi}
```

Compose points declaration for use in opening line:

```
868 \exerciseconfig{composepointsstart}[1]{\getexerciseconfig{composepoints}{#1}}
```

Compose points declaration for use in margin:

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

Compose points declaration for use in text:

```
870 \exerciseconfig{composepointsbody}[1]{\getexerciseconfig{composepoints}{#1}}
```

Compose points declaration for use in sheet data:

```
871 \exerciseconfig{composepointssheet}[1]{%
872   \exerciseifnotempty{#1}{\getexerciseconfig{composepoints}{#1}}}
```

Compose points declaration for solution with comment:

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

Compose alternative points for solution declaration with comment:

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

Compose pairs of points; omit 0 components:

```
877 \exerciseconfig{composepointspair}[2]{%
878   \ifdim#2pt=0pt%
879     \getexerciseconfig{composepoints}{#1}%
880   \else\ifdim#1pt=0pt%
881     +\getexerciseconfig{composepoints}{#2}%
882   \else%
883     \getexerciseconfig{composepointssnum}{#1}+%
884     \getexerciseconfig{composepointssnum}{#2}~%
885     \getexerciseconfig{termpoints}%
886   \fi\fi}
```

Compose pairs of points for designated use; recycle plain definition if no bonus points given:

```
887 \exerciseconfig{composepointspairbody}[2]{%
888   \ifdim#2pt=0pt\getexerciseconfig{composepointssbody}{#1}\else%
889   (\getexerciseconfig{composepointspair}{#1}{#2})\fi}
890 \exerciseconfig{composepointspairstart}[2]{%
891   \ifdim#2pt=0pt\getexerciseconfig{composepointssstart}{#1}\else%
892   (\getexerciseconfig{composepointspair}{#1}{#2})\fi}
893 \exerciseconfig{composepointspairmargin}[2]{%
894   \ifdim#2pt=0pt\getexerciseconfig{composepointssmargin}{#1}\else%
895   \getexerciseconfig{composepointspair}{#1}{#2}\fi}
896 \exerciseconfig{composepointspairsheet}[2]{%
897   \ifdim#2pt=0pt\getexerciseconfig{composepointsssheet}{#1}\else%
898   \getexerciseconfig{composepointspair}{#1}{#2}\fi}
899 \exerciseconfig{composepointspairaward}[3]{%
900   \ifdim#2pt=0pt\getexerciseconfig{composepointsaward}{#1}{#3}\else%
901   (\getexerciseconfig{composepointspair}{#1}{#2}%
902     \exerciseifnotempty{#3}{; #3})\fi}
903 \exerciseconfig{composepointspairawardalt}[3]{%
904   \ifdim#2pt=0pt\getexerciseconfig{composepointsawardalt}{#1}{#3}\else%
905   (\getexerciseconfig{composepointspair}{#1}{#2}*%
906     \exerciseifnotempty{#3}{; #3})\fi}
```

Compose pairs of points for designated situations:

```
907 \exerciseconfig{composepointspairbodyproblem}[2]{%
908   \getexerciseconfig{composepointspairbody}{#1}{#2}}
909 \exerciseconfig{composepointspairbodysubproblem}[2]{%
910   \getexerciseconfig{composepointspairbody}{#1}{#2}}
911 \exerciseconfig{composepointspairbodysolution}[2]{%
912   \getexerciseconfig{composepointspairbody}{#1}{#2}}
913 \exerciseconfig{composepointspairstartproblem}[2]{%
914   \getexerciseconfig{composepointspairstart}{#1}{#2}}
```

```

915 \exerciseconfig{composepointspairstartsubproblem}[2]{%
916   \getexerciseconfig{composepointspairstart}{#1}{#2}}
917 \exerciseconfig{composepointspairstartsolution}[2]{%
918   \getexerciseconfig{composepointspairstart}{#1}{#2}}
919 \exerciseconfig{composepointspairmarginproblem}[2]{%
920   \getexerciseconfig{composepointspairmargin}{#1}{#2}}
921 \exerciseconfig{composepointspairmarginsubproblem}[2]{%
922   \getexerciseconfig{composepointspairmargin}{#1}{#2}}
923 \exerciseconfig{composepointspairmarginresolution}[2]{%
924   \getexerciseconfig{composepointspairmargin}{#1}{#2}}

```

Display points in the margin:

```

925 \exerciseconfig{insertpointsmargin}[1]{\marginpar{\footnotesize #1}}
926 \exerciseconfig{insertpointsmarginproblem}[1]{%
927   \getexerciseconfig{insertpointsmargin}{#1}}
928 \exerciseconfig{insertpointsmarginsubproblem}[1]{%
929   \getexerciseconfig{insertpointsmargin}{#1}}
930 \exerciseconfig{insertpointsmarginresolution}[1]{%
931   \getexerciseconfig{insertpointsmargin}{#1}}

```

Display warning about points mismatch:

```

932 \exerciseconfig{insertwarnpoints}[3]{%
933   {\textbf{points mismatch for #1 (#2 determined vs.\ #3 given)}}}

```

Display warning about points changed:

```

934 \exerciseconfig{insertwarnpointsrerun}[1]{%
935   {\textbf{points changed for #1 (please recompile)}}}

```

Counters. Define counter display via configuration interface:

```

936 \exerciseconfig{countersheet}{\arabic{\exf@sheetcounter}}
937 \exerciseconfig{counterproblem}{\arabic{\exf@problemcounter}}
938 \exerciseconfig{counterproblemmax}{10}
939 \exerciseconfig{countersubproblem}{\alph{\exf@subproblemcounter}}
940 \exerciseconfig{countersubproblemmax}{m}
941 \exerciseconfig{countersheetequation}{\arabic{equation}}
942 \exerciseconfig{counterproblemequation}{P\arabic{equation}}
943 \exerciseconfig{countersolutionequation}{S\arabic{equation}}

```

Further Definitions.

tagsheet	Templates for automatic generation of tags:
tagproblem	
tagsubproblem	<pre> 944 \exerciseconfig{tagsheet}{\arabic{\exf@sheetcounter}} 945 \exerciseconfig{tagproblem}{\csname the\exf@problemcounter\endcsname} 946 \exerciseconfig{tagsubproblem}{\problemtag-\arabic{\exf@subproblemcounter}} </pre>

labelsheet	Templates for automatic generation of labels from tags:
labelproblem	
labelsubproblem	<pre> 947 \exerciseconfig{labelsheet}[1]{sheet:#1} 948 \exerciseconfig{labelproblem}[1]{prob:#1} 949 \exerciseconfig{labelsubproblem}[1]{\getexerciseconfig{labelproblem}{#1}} </pre>

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

```

950 \exerciseconfig{toclevelsheet}{}
951 \exerciseconfig{toclevelproblem}{}
952 \exerciseconfig{toclevelproblems}{}
953 \exerciseconfig{toclevelsolution}{}
954 \exerciseconfig{toclevelsolutions}{}

```

extsolutions Filename extension for solution and problem blocks:

```

extproblems 955 \exerciseconfig{extsolutions}{.sol}
          956 \exerciseconfig{extproblems}{.prb}

```

C.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:

```

957 \newcommand{\defexercisestylearg}[3][{}]{%
958   \def\exf@tmp{#1}\ifx\exf@tmp\exf@empty%
959     \define@key{exf@style}{#2}{#3}\else%
960     \define@key{exf@style}{#2}[#1]{#3}\fi}

```

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

```

961 \newcommand{\defexercisestyle}[2]{%
962   \exf@csdotwo\long\def{exf@style@code@#1}{#2}%
963   \exf@exparg{\define@boolkey{exf@style}[exf@style@]{#1}[true]}%
964   {\csname ifexf@style@#1\endcsname\csname exf@style@code@#1\endcsname\fi}}

```

\exercisestyle Process styles:

```

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

```

Default Exercise Styles.

problemmanual Delay display of problems:

```

966 \define@boolkey{exf@style}[exf@]{problemmanual}[true]{}
967 \exf@problemmanualfalse

```

solutionbelow Choose location for solutions:

```

968 \def\exf@solutionbelow{subproblem}
969 \define@choicekey{exf@style}{solutionbelow}%
970   {here,subproblem,subproblem*,problem,problem*,sheet>manual}%
971   {\ifexf@solfile@open\else\gdef\exf@solutionbelow{#1}\fi}

```

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

```

problemequation 972 \defexercisestyle{sheetequation}{}
solutionequation 973 \defexercisestyle{problemequation}{}
          974 \defexercisestyle{solutionequation}{}
          975 \exf@style@solutionequationtrue

```

problempointsat Choose where points of (sub)problems and solutions are displayed:

```

subproblempointsat
solutionpointsat
976 \def\exf@pointsat{start}
977 \define@choicekey{exf@style}{problempointsat}%
978   {start,start*,margin,end,manual,off}{\def\exf@pointsat{#1}}
979 \define@choicekey{exf@style}{pointsat}%
980   {start,start*,margin,end,manual,off}{\def\exf@pointsat{#1}}
981 \def\exf@subpointsat{end}
982 \define@choicekey{exf@style}{subproblempointsat}%
983   {start,start*,margin,end,manual,off}{\def\exf@subpointsat{#1}}
984 \define@choicekey{exf@style}{subpointsat}%
985   {start,start*,margin,end,manual,off}{\def\exf@subpointsat{#1}}
986 \def\exf@solpointsat{off}
987 \define@choicekey{exf@style}{solutionpointsat}%
988   {start,start*,margin,end,manual,off}{\def\exf@solpointsat{#1}}
989 \define@choicekey{exf@style}{solpointsat}%
990   {start,start*,margin,end,manual,off}{\def\exf@solpointsat{#1}}

```

problemby Declare problems or equations as subcounter of other counter:

```

equationby
991 \defexercisestylearg{problemby}{\exf@numberproblemwithin{#1}}
992 \defexercisestylearg{equationby}{\exf@numberequationwithin{#1}}

```

pagebysheet Number pages, problems or equations by sheet:

```

problembysheet
equationbysheet
993 \defexercisestyle{pagebysheet}{%
994   \def\thepage{\csname the\exf@sheetcounter\endcsname.\arabic{page}}%
995   \def\theHpage{\csname theH\exf@sheetcounter\endcsname.\arabic{page}}%
996   \exerciseconfigappend{insertsheetbefore}{\setcounter{page}{1}}
997 \defexercisestyle{problembysheet}%
998   {\exf@numberproblemwithin{\exf@sheetcounter}}
999 \defexercisestyle{equationbysheet}%
1000   {\exf@numberequationwithin{\exf@sheetcounter}}

```

fracpoints Use vulgar fractions to display binary fractional points:

```

1001 \defexercisestyle{fracpoints}%
1002   {\exerciseconfig{composepointsnum}[1]{\protect\showfracpoints{##1}}}

```

twoside Use two-sided layout for sheets:

```

1003 \defexercisestylearg{true}{twoside}{\exercisecsetup{twoside={#1}}}

```

subproblemdelitem to append delimiter ‘)’ to subproblem reference labels Use two-sided layout for sheets:

```

1004 \defexercisestyle{subproblemdelitem}%
1005   {\exerciseconfig{composeitemsubproblemdelim}{}}%
1006   \exerciseconfig{countersubproblem}{\alph{\exf@subproblemcounter}}%
1007   \exerciseconfig{countersubproblemmax}{m}}

```

Extended Exercise Styles. Declare more specific styles:

```

1008 \ifexf@extstyle

```

contents Add sheets and problems to table of contents:

```

1009 \defexercisestyle{contents}{%
1010   \exerciseconfig{toclevelsheet}{section}%
1011   \exerciseconfig{toclevelproblem}{subsection}}

```


solutionsf Use sans serif font for solutions:

```
1012 \defexercisestyle{solutionsf}{%
1013   \exerciseconfigappend{styletextsolution}{\sffamily\let\itshape\slshape}}
```

solutiondimproblem Dim problem text if solutions are displayed:

```
1014 \defexercisestyle{solutiondimproblem}{%
1015   \RequirePackage{color}%
1016   \exerciseconfigappend{styletextsolution}{\color{gray}{0}}%
1017   \exerciseconfigappend{styletextproblem}{\color{gray}{0.2}}}
```

solutionsep Separate solutions by horizontal lines:

```
1018 \defexercisestyle{solutionsep}{%
1019   \exerciseconfig{insertsolutionsbefore}{\hrule\nopagebreak[3]\vspace{0.5ex}}%
1020   \exerciseconfig{insertsolutionsafter}{%
1021     {\removelastskip\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:

```
1022 \defexercisestyle{plainheader}{%
1023   \exerciseconfig{styleheadertitle}{\Large\bfseries}%
1024   \exerciseconfig{styleheadercourse}{\sffamily}%
1025   \exerciseconfig{styleheaderbelow}{\footnotesize}%
1026   \exerciseconfig{skipheaderbelow}{3ex}%
1027   \exerciseconfig{composeheaderbelowleft}{}%
1028   \exerciseconfig{composeheaderbelowright}{}%
1029   \exerciseconfig{composeheaderbelowcenter}{}%
1030   \exerciseconfig{insertsheettitle}{\noindent%
1031     \begin{minipage}{\textwidth}%
1032       {\getexerciseconfig{styleheadertitle}%
1033         \makebox[0pt][l]{\getexercisedata{course}}%
1034         \hfill\makebox[0pt][r]{\getsheetdata{title}}\par}%
1035       {\getexerciseconfig{styleheadercourse}%
1036         \makebox[0pt][l]{\getexercisedata{institution}%
1037           \exercisedataempty{period}{}, \getexercisedata{period}}}%
1038         \hfill\makebox[0pt][r]{\getexercisedata{instructor}}}%
1039       \vphantom{g}\par}%
1040   \hrule%
1041   {\def\tmp{}}%
1042   \exerciseconfigempty{composeheaderbelowleft}{}{\def\tmp{.}}%
1043   \exerciseconfigempty{composeheaderbelowcenter}{}{\def\tmp{.}}%
1044   \exerciseconfigempty{composeheaderbelowright}{}{\def\tmp{.}}%
1045   \exerciseifnotempty{\tmp}%
1046     {\getexerciseconfig{styleheaderbelow}\vphantom{\^A}%
1047       \makebox[0pt][l]{\getexerciseconfig{composeheaderbelowleft}}%
1048       \hfill\makebox[0pt][c]{\getexerciseconfig{composeheaderbelowcenter}}%
1049       \hfill\makebox[0pt][r]{\getexerciseconfig{composeheaderbelowright}}%
1050       \vspace*{-\baselineskip}\vspace*{-\parskip}\par}%
1051   \end{minipage}%
1052   \par\addvspace{\getexerciseconfig{skipheaderbelow}}}
```

Done with extended styles:

```
1053 \fi
```

C.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`:

```
1054 \newcommand{\defexercisedata}[1]{%
1055   \exf@csdo\def{exf@data@#1}{}%
1056   \define@key{exf@data}{#1}%
1057   {\exf@csdo\gdef{exf@data@#1}{##1}}}
```

`\exercisedata` Process key-value pairs:

```
1058 \newcommand{\exercisedata}[1]{\setkeys{exf@data}{#1}}
```

`\getexercisedata` Read global metadata:

```
1059 \newcommand{\getexercisedata}[1]{\csname exf@data@#1\endcsname}
```

`\exercisedataempty` Check whether the field is empty:

```
1060 \newcommand{\exercisedataempty}[3]{\exf@csdo\ifx{exf@data@#1}\exf@empty%
1061   #2\else#3\fi}
```

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

```
1062 \defexercisedata{author}
1063 \defexercisedata{title}
1064 \defexercisedata{subject}
1065 \defexercisedata{keyword}
```

Declare additional general purpose fields:

```
1066 \defexercisedata{date}
```

Declare metadata related to courses:

```
1067 \defexercisedata{instructor}
1068 \defexercisedata{course}
1069 \defexercisedata{institution}
1070 \defexercisedata{period}
1071 \defexercisedata{material}
```

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

```
1072 \define@key{exf@data}{author}{\gdef\exf@data@author{#1}\author{#1}}
1073 \define@key{exf@data}{title}{\gdef\exf@data@title{#1}\title{#1}}
1074 \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`:

```
1075 \newcommand{\defsheetdata}[1]{%
1076   \exf@csdo\def{exf@data@sheet@#1}{}%
1077   \define@key{exf@sheet}{#1}%
1078   {\exf@csdo\def{exf@data@sheet@#1}{##1}}}
```

`\setsheetdata` Set sheet metadata:

```
1079 \newcommand{\setsheetdata}[1]{\setkeys{exf@sheet}{#1}}
```

`\getsheetdata` Read sheet metadata:

```
1080 \newcommand{\getsheetdata}[1]{\csname exf@data@sheet@#1\endcsname}
```

`\sheetdataempty` Check whether the field is empty:

```
1081 \newcommand{\sheetdataempty}[3]{\exf@csdo@ifx{exf@data@sheet@#1}\exf@empty%
1082   #2\else#3\fi}
```

Declare general purpose fields:

```
1083 \defsheetdata{due}
1084 \defsheetdata{handout}
1085 \defsheetdata{editdate}
1086 \defsheetdata{author}
1087 \defsheetdata{editor}
```

Special title processing:

```
1088 \def\exf@data@sheet@rawtitle{}
1089 \define@key{exf@sheet}{title}{\def\exf@data@sheet@rawtitle{#1}}
1090 \def\exf@data@sheet@title{\exf@config@composetitlesheet%
1091   {\csname the\exf@sheetcounter\endcsname}{\exf@data@sheet@rawtitle}}%
```

Special points processing:

```
1092 \def\exf@data@sheet@points{\ifdefined\exf@sheet@points%
1093   \expandafter\exf@config@composepointspairsheet\exf@sheet@points\fi}%

```

Problem Metadata.

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

```
1094 \newcommand{\defproblemdata}[1]{%
1095   \exf@csdo\def{exf@data@problem@#1}{}%
1096   \define@key{exf@problem}{#1}%
1097   {\exf@csdo\def{exf@data@problem@#1}{##1}}}
```

`\setproblemdata` Set problem metadata:

```
1098 \newcommand{\setproblemdata}[1]{\setkeys{exf@problem,exf@scanproblem}{#1}}
```

`\setsubproblemdata` Set subproblem metadata:

```
1099 \newcommand{\setsubproblemdata}[1]{%
1100   \setkeys{exf@subproblem,exf@scansubproblem}{#1}}
```

`\getproblemdata` Read problem metadata:

```
1101 \newcommand{\getproblemdata}[1]{\csname exf@data@problem@#1\endcsname}
```

`\problemdataempty` Check whether the field is empty:

```
1102 \newcommand{\problemdataempty}[3]{\exf@csdo@ifx{exf@data@problem@#1}\exf@empty%
1103   #2\else#3\fi}
```

Special title processing:

```
1104 \def\exf@data@problem@rawtitle{}
1105 \define@key{exf@problem}{title}{\def\exf@data@problem@rawtitle{#1}}
1106 \def\exf@data@problem@title{\exf@config@composetitleproblem}%
1107 \csname the\exf@problemcounter\endcsname{\exf@data@problem@rawtitle}}%
```

Problem Environment Code.

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

```
1108 \newskip\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):

```
1109 \newcommand{\exf@section}[2]{%
1110   \ifdim\glueexpr#1\relax<0pt%
1111     \protected@edef\exf@tmp{#2}%
1112   \else%
1113     \def\exf@tmp{#2}%
1114   \fi%
1115   \exf@exparg{\@startsection{}{}{0pt}{0pt}{#1}{*}}{\exf@tmp}}
```

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

```
1116 \newcommand{\exf@init@block}[1]{%
1117   \def\exf@intro{}\def\exf@intro@skip{#1}%
1118   \exf@addmargin0pt\def\exf@introitem{}}
```

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

```
1119 \newcommand{\exf@append@intro}[1]{%
1120   {\exf@append@def\exf@intro{#1\hspace{\exf@intro@skip}}}}
```

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

```
1121 \newcommand{\exf@prepend@intro}[1]{%
1122   {\exf@prepend@def\exf@intro{#1\hspace{\exf@intro@skip}}}}
```

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

```
1123 \newcommand{\exf@open@block}[1]{%
1124   \advance\leftskip\exf@addmargin%
1125   \advance\linewidth-\exf@addmargin%
1126   \advance\@totalleftmargin\exf@addmargin%
1127   \ifx\exf@intro\exf@empty%
1128     \exf@section{0pt}{\exf@introitem}%
1129   \else%
1130     \exf@section{#1}{\exf@introitem\exf@intro\unskip}%
1131   \fi}%

```

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

```
1132 \newcommand{\exf@close@block}{%
1133   \advance\leftskip-\exf@addmargin%
1134   \advance\linewidth\exf@addmargin%
1135   \advance\@totalleftmargin-\exf@addmargin}%

```

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

```
1136 \newcommand{\addprobleminfo}{\@ifstar\exf@prepend@intro\exf@append@intro}
```

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

```
1137 \newcommand{\exf@addinfoswitch}[1]%  
1138 {\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:

```
1139 \newcommand{\defprobleminfo}[2]{%  
1140 \exf@addinfoswitch{#1}%  
1141 \exercisecconfig{compose@probleminfo@#1}[1]{#2}%  
1142 \exf@exparg{\define@key{exf@probleminfo}{#1}}%  
1143 {\csname ifexf@showdata@#1\endcsname\exf@append@intro{%  
1144 \csname exf@config@compose@probleminfo@#1\endcsname{##1}}\fi}}
```

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

```
1145 \newcommand{\showprobleminfo}[1]{\exf@expsetkeys{exf@infoswitch}{#1}}
```

Problem Info Declarations. Declare general purpose fields:

```
1146 \defprobleminfo{optional}{\emph{#1:}}  
1147 \showprobleminfo{optional}  
1148 \defprobleminfo{difficulty}{( #1 )}
```

Declare fields for internal information (mostly):

```
1149 \defprobleminfo{comment}{#1}  
1150 \defprobleminfo{author}{\langle$#1$\rangle$}  
1151 \defprobleminfo{editor}{\{ $#1 $\}  
1152 \defprobleminfo{source}{[ #1 ]}  
1153 \defprobleminfo{keyword}{\#( #1 )}
```

Declare more specific fields:

```
1154 \ifexf@extdata  
1155 \defprobleminfo{review}{#1}  
1156 \defprobleminfo{recycle}{[[ #1 ]]}  
1157 \defprobleminfo{timesolve}{\{ #1 \}}  
1158 \defprobleminfo{timepresent}{\{ \! \{ #1 \} \! \}}  
1159 \fi
```

Write Metadata to PDF Files.

`\exf@writemetadata` Write Metadata to PDF Files in case hyperref is available:

```
1160 \newcommand{\exf@writemetadata}{%  
1161 \ifdefined\hypersetup%
```

Write author, title, subject and keywords:

```
1162 \ifx\exf@data@author\exf@empty\else%  
1163 \hypersetup{pdfauthor={\exf@data@author}}\fi%  
1164 \ifx\exf@data@title\exf@empty\else%  
1165 \hypersetup{pdftitle={\exf@data@title}}\fi%  
1166 \ifx\exf@data@subject\exf@empty\else%
```

```

1167 \hypersetup{pdfsubject={\exf@data@subject}}\fi%
1168 \ifx\exf@data@keyword\exf@empty\else%
1169 \hypersetup{pdfkeywords={\exf@data@keyword}}\fi%
1170 \fi}

```

Automatic writing at `\begin{document}`:

```

1171 \AtBeginDocument{\exf@ifis\exf@metadata{auto}%
1172 {\exf@writemetadata\gdef\exf@metadata{off}}}

```

`\writeexercisedata` Write metadata manually:

```

1173 \newcommand{\writeexercisedata}{\exf@ifis\exf@metadata{manual}%
1174 {\exf@writemetadata\gdef\exf@metadata{off}}}

```

C.6 Counters

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

```

problem 1175 \newcounter{\exf@sheetcounter}
subproblem 1176 \newcounter{\exf@problemcounter}
solution 1177 \newcounter{\exf@subproblemcounter}[\exf@problemcounter]
1178 \newcounter{\exf@solutioncounter}[\exf@problemcounter]
1179 \newcount\exf@eqsav
1180 \newcounter{\exf@sheetequation}
1181 \newcounter{\exf@problemequation}
1182 \newcounter{\exf@solutionequation}

```

Implement counter display; take care of corresponding hyperref labels:

```

1183 \exf@csdo\def{the\exf@sheetcounter}{\exf@config@countersheet}
1184 \exf@csdo\def{the\exf@problemcounter}{\exf@config@counterproblem}
1185 \exf@csdo\def{the\exf@subproblemcounter}{\exf@config@countersubproblem}
1186 \def\theexf@sheetequation{\exf@config@countersheetequation}
1187 \def\theHexf@sheetequation{sheet.\arabic{equation}}
1188 \def\theexf@problemequation{\exf@config@counterproblemequation}
1189 \def\theHexf@problemequation{prob.\arabic{equation}}
1190 \def\theexf@solutionequation{\exf@config@countersolutionequation}
1191 \def\theHexf@solutionequation{sol.\arabic{equation}}

```

`@numberproblemwithin` Declare problem counter as subcounter of #1:

```

1192 \newcommand{\exf@numberproblemwithin}[1]{%
1193 \@addtoreset{\exf@problemcounter}{#1}%
1194 \exf@csdo\def{the\exf@problemcounter}{%
1195 {\csname the#1\endcsname.\exf@config@counterproblem}%
1196 \edef\exf@tmp{#1}%
1197 \ifx\exf@tmp\exf@sheetcounter%
1198 \exerciseconfig{tagproblem}{\ifdefined\sheettag\sheettag-\fi%
1199 \arabic{\exf@problemcounter}}}%
1200 \else%
1201 \exerciseconfig{tagproblem}{\csname the#1\endcsname-%
1202 \arabic{\exf@problemcounter}}}%
1203 \fi}

```

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

```

1204 \newcommand{\exf@numberequationwithin}[1]{%

```

```

1205 \addtoreset{exf@sheetequation}{#1}%
1206 \def\theexf@sheetequation%
1207   {\csname the#1\endcsname.\exf@config@countersheetequation}%
1208 \def\theHexf@sheetequation%
1209   {\csname theH#1\endcsname.sheet.\arabic{equation}}}%
1210 \addtoreset{exf@problemequation}{#1}%
1211 \def\theexf@problemequation%
1212   {\csname the#1\endcsname.\exf@config@counterproblemequation}%
1213 \def\theHexf@problemequation%
1214   {\csname theH#1\endcsname.prob.\arabic{equation}}}%
1215 \addtoreset{exf@solutionequation}{#1}%
1216 \def\theexf@solutionequation%
1217   {\csname the#1\endcsname.\exf@config@countersolutionequation}%
1218 \def\theHexf@solutionequation%
1219   {\csname theH#1\endcsname.sol.\arabic{equation}}}%

```

C.7 Buffers

File Output.

`\ifexf@solfile@open` Conditional whether output files are presently in use:

```

1220 \newif\ifexf@solfile@open\exf@solfile@openfalse
1221 \newif\ifexf@probfile@open\exf@probfile@openfalse

```

`\exf@solfile` Reserve file handles:

```

1222 \newwrite\exf@solfile
1223 \newwrite\exf@probfile

```

`\exf@writeline` Write a line to the file:

```

1224 \newcommand{\exf@writeline}[2]{\immediate\write#1{#2}}

```

`\exf@linesep` Return a separator line:

```

1225 \newcommand{\exf@linesep}%
1226   {\@percentchar-----}

```

`\exf@lineno` Return current position in source file; display line number and source file name (if available via package `currfile`):

```

1227 \newcommand{\exf@lineno}{\@percentchar%
1228   \ifdefined\currfilename\currfilename\space\fi%
1229   1.\the\inputlineno}

```

`\exf@start@solfile` Open a new solution file #1.sol (do nothing if already open); indicate source, switch to manual solution display mode:

```

1230 \newcommand{\exf@start@solfile}[1]{%
1231   \ifexf@solfile@open\else%
1232     \exercisestyle{solutionbelow>manual}%
1233     \global\exf@solfile@opentrue%
1234     \edef\exf@tmp{#1}%
1235     \immediate\openout\exf@solfile\exf@tmp\exf@config@extsolutions\relax%
1236     \exf@writeline\exf@solfile{\@percentchar%
1237       generated from file ‘\jobname’ by exframe.sty}%
1238     \ifexf@lineno\exf@writeline\exf@solfile{\exf@lineno}\fi%

```

```

1239 \exf@writeline\exf@solfile{%
1240 \fi}

```

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

```

1241 \newcommand{\exf@close@solfile}{%
1242 \ifexf@solfile@open%
1243 \ifexf@lineno\exf@writeline\exf@solfile{\exf@linesep}%
1244 \exf@writeline\exf@solfile{\exf@lineno}\fi%
1245 \exf@writeline\exf@solfile{\@backslashchar endinput}%
1246 \immediate\closeout\exf@solfile%
1247 \global\exf@solfile@openfalse%
1248 \fi}

```

\exf@start@probfile Open a new problem file #1.prb (do nothing if already open); indicate source, switch to manual problem display mode:

```

1249 \newcommand{\exf@start@probfile}[1]{%
1250 \ifexf@probfile@open\else%
1251 \global\exf@probfile@opentruetrue%
1252 \edef\exf@tmp{#1}%
1253 \immediate\openout\exf@probfile\exf@tmp\exf@config\extproblems\relax%
1254 \exf@writeline\exf@probfile{\@percentchar%
1255 generated from file '\jobname' by exframe.sty}%
1256 \ifexf@lineno\exf@writeline\exf@probfile{\exf@lineno}\fi%
1257 \exf@writeline\exf@probfile{%
1258 \fi}

```

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

```

1259 \newcommand{\exf@close@probfile}{%
1260 \ifexf@probfile@open%
1261 \ifexf@lineno\exf@writeline\exf@probfile{\exf@linesep}%
1262 \exf@writeline\exf@probfile{\exf@lineno}\fi%
1263 \exf@writeline\exf@probfile{\@backslashchar endinput}%
1264 \immediate\closeout\exf@probfile%
1265 \global\exf@probfile@openfalse%
1266 \fi}

```

Make sure to properly close files at the end:

```

1267 \AtEndDocument{\exf@close@solfile\exf@close@probfile}

```

Buffers.

\exf@solbuf Declare token buffers for storing problems and solutions and conditionals indicating whether the buffers have been used:

```

\exf@subprobbuf
\ifexf@solbuf@clean 1268 \newtoks\exf@solbuf
\ifexf@probbuf@clean 1269 \newtoks\exf@probbuf
\ifexf@subprobbuf@clean 1270 \newtoks\exf@subprobbuf
1271 \newif\ifexf@solbuf@clean\exf@solbuf@cleantrue
1272 \newif\ifexf@probbuf@clean\exf@probbuf@cleantrue

```

\exf@clear@solbuf Clear a buffer and mark clean:

```

\exf@clear@probbuf
\exf@clear@subprobbuf 1273 \def\exf@clear@solbuf{\global\exf@solbuf@cleantrue\global\exf@solbuf={}}
1274 \def\exf@clear@probbuf{\global\exf@probbuf@cleantrue\global\exf@probbuf={}}
1275 \def\exf@clear@subprobbuf{\global\exf@subprobbuf={}}

```


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

```
1276 \def\exf@append@buf#1#2{\global#1=\expandafter{\the#1#2}}
```

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

```
1277 \def\exf@addline#1#2{\protected@edef\exf@tmp{#2}%
1278 \exf@exparg{\exf@append@buf#1}{\exf@tmp^^J}}
```

`\exf@source@buf` Source a buffer into the document:

```
1279 \def\exf@source@buf#1{\exf@exptwo\scantokens{\the#1}}
```

`\exf@write@buf` Write the buffer into the solution file:

```
1280 \def\exf@write@buf#1#2{\exf@writeline#1{\the#2}}
```

Verbatim Processing.

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

```
1281 \newcommand{\exf@verbatim}{%
1282 \begingroup%
1283 \@bsphack%
1284 \let\do\@makeother\dospecials%
1285 \catcode'\^^M\active%
1286 \def\verbatim@processline{\exf@exptwo\exf@verbatim@process%
1287 {\the\verbatim@line^^J}}%
1288 \verbatim@start}
```

`\exf@endverbatim` Stop reading the buffer:

```
1289 \newcommand{\exf@endverbatim}{\@esphack\endgroup}
```

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

```
1290 \def\exf@scanblock#1{%
```

Check for empty first line:

```
1291 \@ifnextchar\par{\exf@scanblock@par{#1}}{\exf@scanblock@sel{#1}}
```

Handle empty first line, implies no optional argument:

```
1292 \long\def\exf@scanblock@par#1\par{\exf@scanblock@sel{#1}[]}
```

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

```
1293 \def\exf@scanblock@sel#1{\@ifnextchar[{\exf@scanblock@opt{#1}}%
1294 {\@ifnextchar\end{\exf@scanblock@end{#1}}{\exf@scanblock@noopt{#1}}}
```

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

```
1295 \def\exf@scanblock@end#1\end#2{%
1296 \def\exf@tmp{#2}\ifx\exf@tmp\@currentenv%
1297 \def\exf@verbatim{}\def\exf@endverbatim{%
1298 \fi%
1299 #1}{\scantokens{\end{#2}}}
```

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

```
1300 \def\exf@scanblock@noopt#1#2#{#1}{\scantokens#2}}
1301 \def\exf@scanblock@opt#1[#2]{#1{#2}{}}
```

C.8 Points

Points Arithmetic.

\exf@addtopoints Add points #2+#3 to macro #1 using metric register:

```
1302 \newcommand{\exf@addtopoints}[3]{%
1303   \ifdefined#1\else\def#1{{0}{0}}\fi%
1304   \dimen@ \dimexpr\expandafter\@firstoftwo#1pt\relax%
1305   \advance\dimen@ \dimexpr#2pt\relax%
1306   \edef\exf@tmp{\strip@pt\dimen@}%
1307   \dimen@ \dimexpr\expandafter\@secondoftwo#1pt\relax%
1308   \advance\dimen@ \dimexpr#3pt%
1309   \xdef#1{{\exf@tmp}{\strip@pt\dimen@}}}
```

\exf@pointsmismatch Execute #3 if points disagree:

```
1310 \newcommand{\exf@pointsmismatch}[3]{%
1311   \let\exf@tmp\@undefined%
1312   \ifdim\expandafter\@firstoftwo#1pt=\expandafter\@firstoftwo#2pt\else%
1313     \def\exf@tmp{}\fi%
1314   \ifdim\expandafter\@secondoftwo#1pt=\expandafter\@secondoftwo#2pt\else%
1315     \def\exf@tmp{}\fi%
1316   \ifdefined\exf@tmp#3\fi}
```

Points Expansion.

\exf@outpoints If points macro #3 is set, expand #3, pass on as #1{\protect#2}#3 and clear #3 globally:

```
1317 \def\exf@outpoints#1#2#3{\ifdefined#3%
1318   \exf@exptwo\exf@outpoints@switch{#3}{#1}{#2}%
1319   \global\let#3\@undefined\fi}
1320 \def\exf@outpoints@switch#1#2#3{#2{\protect#3#1}}
```

\exf@scanpoints Call as \exf@scanpoints#1[*regular*][+*bonus*]+@ to write {*regular*}{*bonus*} into #1; fill with 0 if empty:

```
1321 \def\exf@scanpoints#1#2+#3+#4{%
1322   \edef#1{\if @#2@0\else#2\fi}%
1323   \edef#1{{#1}{\if @#3@0\else#3\fi}}}
```

\exf@formatpoints Format as [#1][+#2]; remove 0 components:

```
1324 \def\exf@formatpoints#1#2{\ifdim#2pt=0pt#1\else%
1325   \ifdim#1pt=0pt+#2\else#1+#2\fi\fi}
```

\extractpoints Extract main (plain) or bonus (starred) part from saved points register:

```
1326 \newcommand{\extractpoints}{\@ifstar{\exf@extractpoints\@secondoftwo}%
1327   {\exf@extractpoints\@firstoftwo}}
1328 \newcommand{\exf@extractpoints}[2]{\edef\exf@tmp{#2}%
1329   \exf@exptwo\exf@scanpoints\exf@tmp\exf@tmp++@%
1330   \expandafter#1\exf@tmp}
```

`\switchpoints` Extract main (plain) and bonus (starred) part from points, and execute one of three:

```

1331 \newcommand{\switchpoints}[5]{\edef\exf@tmp{#5}%
1332   \exf@exptwo\exf@scanpoints\exf@tmp\exf@tmp++0%
1333   \expandafter\exf@switchpoints\exf@tmp{#1}{#2}{#3}{#4}}
1334 \newcommand{\exf@switchpoints}[6]{%
1335   \ifdim#2pt=0pt\ifdim#1pt=0pt\def\exf@tmp##1##2{#6}%
1336   \else\def\exf@tmp##1##2{#3}\fi%
1337   \else\ifdim#1pt=0pt\def\exf@tmp##1##2{#4}%
1338   \else\def\exf@tmp##1##2{#5}\fi\fi\exf@tmp{#1}{#2}}

```

Tools.

`\pointsmarginproblem` Combination to typeset points in margin:

`\intsmarginsubproblem`
`\pointsmarginsolution`

```

1339 \newcommand{\exf@makepointsmarginproblem}[2]{%
1340   \exf@config@insertpointsmarginproblem{%
1341     \exf@config@composepointspairmarginproblem{#1}{#2}}}
1342 \newcommand{\exf@makepointsmarginsubproblem}[2]{%
1343   \exf@config@insertpointsmarginsubproblem{%
1344     \exf@config@composepointspairmarginsubproblem{#1}{#2}}}
1345 \newcommand{\exf@makepointsmarginsolution}[2]{%
1346   \exf@config@insertpointsmarginsolution{%
1347     \exf@config@composepointspairmarginsolution{#1}{#2}}}

```

`\exf@warnmismatch` If points #3 and #4 are defined and disagree, issue a warning message:

```

1348 \newcommand{\exf@warnmismatch}[4]{%
1349   \ifdefined#4\ifdefined#3\exf@pointsmismatch#3#4{%
1350     \let\exf@tmp\PackageWarning%
1351     \ifx#1\exf@solutionname\let\exf@tmp\PackageWarningNoLine\fi%
1352     \exf@tmp{exframe}{points mismatch %
1353       (\expandafter\exf@formatpoints#3 determined %
1354       vs. \expandafter\exf@formatpoints#4 given) %
1355       for #1 \csname the#2\endcsname}%
1356     \ifexf@warntext\edef\exf@tmp{%
1357       {\expandafter\exf@formatpoints#3}{\expandafter\exf@formatpoints#4}}%
1358     \exf@exptwo\exf@config@insertwarnpoints#1\exf@tmp\fi}%
1359     \fi\fi}

```

`\exf@warnrerun` If points #3 and #4 are defined and disagree, issue advice to recompile:

```

1360 \newcommand{\exf@warnrerun}[4]{%
1361   \ifdefined#4\ifdefined#3\exf@pointsmismatch#3#4{%
1362     \PackageWarning{exframe}{points changed %
1363       for #1 \csname the#2\endcsname; rerun to fix}%
1364     \ifexf@warntext\exf@config@insertwarnpointsrerun#1\fi}%
1365     \fi\fi}

```

Binary Rational Numbers.

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

`\exf@splitdecimal`

```

1366 \def\exf@splitsign#1-#2-#3&{\def\exf@splitnum{#1#2}\def\exf@splitminus{#3}}
1367 \def\exf@splitdecimal#1.#2.#3&{\def\exf@splitint{#1}\def\exf@splitdec{#2}}

```

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

```

1368 \newcommand{\showfracpoints}[1]{%
1369   \edef\exf@tmp{#1}%
1370   \expandafter\exf@splitsign\exf@tmp--&%
1371   \expandafter\exf@splitdecimal\exf@splitnum..&%
1372   \if @\exf@splitint @\def\exf@splitint{0}\fi%
1373   \if @\exf@splitdec @\def\exf@splitdec{0}\fi%
1374   \def\exf@tmp{\exf@splitint.\exf@splitdec}%
1375   \ifnum\exf@splitdec=0\def\exf@tmp{\exf@splitint}\fi%
1376   \ifnum\exf@splitdec=5\def\exf@tmp{\exf@config@frac{\exf@splitint}{1}{2}}\fi%
1377   \ifnum\exf@splitdec=25\def\exf@tmp{\exf@config@frac{\exf@splitint}{1}{4}}\fi%
1378   \ifnum\exf@splitdec=75\def\exf@tmp{\exf@config@frac{\exf@splitint}{3}{4}}\fi%
1379   \ifnum\exf@splitdec=125\def\exf@tmp{\exf@config@frac{\exf@splitint}{1}{8}}\fi%
1380   \ifnum\exf@splitdec=375\def\exf@tmp{\exf@config@frac{\exf@splitint}{3}{8}}\fi%
1381   \ifnum\exf@splitdec=625\def\exf@tmp{\exf@config@frac{\exf@splitint}{5}{8}}\fi%
1382   \ifnum\exf@splitdec=875\def\exf@tmp{\exf@config@frac{\exf@splitint}{7}{8}}\fi%
1383   \ifx\exf@splitminus\exf@empty\else$\exf@splitminus$\fi\exf@tmp}

```

`\exf@config@frac` Display a vulgar fraction such as $12\frac{3}{4}$:

```

1384 \newcommand{\exf@config@frac}[3]{%
1385   \ifnum#1=0\relax\else#1\fi%
1386   \ifnum#2=0\relax\else$%
1387     ^{\exf@text{#2}}}%
1388   \mskip-4mu/\mskip-2mu%
1389   _{\exf@text{#3}}$\fi}

```

Sheet Points Code.

`notedata@sheetpoints` Store a sheet point number in a macro:

```

1390 \newcommand{\exf@notedata@sheetpoints}[2]{%
1391   \exf@csdo\gdef{\exf@sheetpoints@#1}{#2}}

```

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

```

1392 \newcommand{\exf@writesheetpoints}[2]{%
1393   {\exf@writedata{sheetpoints}{\sheettag}{\exf@formatpoints{#1}{#2}}}}

```

`\getsheetpoints` Read points for current sheet or from .aux file:

```

1394 \newcommand{\getsheetpoints}[1]{\if @#1%
1395   \ifdefined\exf@sheet@points%
1396     \expandafter\exf@formatpoints\exf@sheet@points\else 0\fi%
1397   \else\exf@csor{\exf@sheetpoints@#1}{0}\fi}

```

Problem Points Code.

`tedata@problempoints` Store a problem point number in a macro:

```

1398 \newcommand{\exf@notedata@problempoints}[2]{%
1399   \exf@csdo\gdef{\exf@problempoints@#1}{#2}}

```

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

```

1400 \newcommand{\exf@writeproblempoints}[2]{%
1401   {\exf@writedata{problempoints}{\problemtag}{\exf@formatpoints{#1}{#2}}}}

```

`\getproblempoints` Read points for current problem or from .aux file:

```
1402 \newcommand{\getproblempoints}[1]{\if @#1@%
1403   \ifdefined\exf@problem@points%
1404     \expandafter\exf@formatpoints\exf@problem@points\else 0\fi%
1405   \else\exf@csor{exf@problempoints@#1}{0}\fi}
```

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

```
1406 \newcommand{\showpoints}{%
1407   \ifdefined\exf@in@solution\exf@outpoints{\exf@ensuretext}%
1408   {\exf@config@composepointspairbodysolution}{\exf@solution@points@show}%
1409   \else\ifdefined\exf@in@subproblem\exf@outpoints{\exf@ensuretext}%
1410   {\exf@config@composepointspairbodysubproblem}{\exf@subproblem@points@show}%
1411   \else\ifdefined\exf@in@problem\exf@outpoints{\exf@ensuretext}%
1412   {\exf@config@composepointspairbodyproblem}{\exf@problem@points@show}%
1413   \fi\fi\fi}
```

Subproblem Points Code.

`\ata@subproblempoints` Store a subproblem point number in a macro:

```
1414 \newcommand{\exf@notedata@subproblempoints}[2]{%
1415   \exf@csdo\gdef{exf@subproblempoints@#1}{#2}}
```

`\ritesubproblempoints` Write subproblem points to the .aux file:

```
1416 \newcommand{\exf@writesubproblempoints}[2]{%
1417   {\exf@writedata{subproblempoints}%
1418     {\subproblemtag}{\exf@formatpoints{#1}{#2}}}}
```

`\getsubproblempoints` Read points for current subproblem:

```
1419 \newcommand{\getsubproblempoints}[1]{\if @#1@%
1420   \ifdefined\exf@subproblem@points%
1421     \expandafter\exf@formatpoints\exf@subproblem@points\else 0\fi%
1422   \else\exf@csor{exf@subproblempoints@#1}{0}\fi}
```

Solution Points Code.

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

```
1423 \newcommand{\exf@awardpointsalt}[2] [] {\exf@scanpoints\exf@tmp#2++@%
1424   \exf@exptwo\exf@ensuretext{%
1425     \expandafter\exf@config@composepointspairawardalt\exf@tmp{#1}}}
```

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

```
1426 \newcommand{\exf@awardpointsreg}[2] [] {\exf@scanpoints\exf@tmp#2++@%
1427   \exf@exptwo\exf@addtopoints\exf@solution@points@total\exf@tmp%
1428   \exf@scanpoints\exf@tmp#2++@%
1429   \exf@exptwo\exf@ensuretext{%
1430     \expandafter\exf@config@composepointspairaward\exf@tmp{#1}}}
```

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

```
1431 \newcommand{\awardpoints}{\@ifstar\exf@awardpointsalt\exf@awardpointsreg}
```

`\getsolutionpoints` Read points for current solution:

```
1432 \newcommand{\getsolutionpoints}[1]{\if @#1@%
1433   \ifdefined\exf@solution@points%
1434     \expandafter\exf@formatpoints\exf@solution@points\else 0\fi%
1435   \else 0\fi}
```

C.9 Tag Lists

Loop Lists.

`exerciseloop` Counters for item number within list and depth of loop nesting:

```
exf@loopdepth
1436 \newcounter{exerciseloop}
1437 \newcounter{exf@loopdepth}
```

`exf@listwalk` Step through a list; call the callback function #1 with current item #2:

```
1438 \def\exf@listwalk#1#2{\if @#2@\def\exf@tmp{}\else%
1439   \def\exf@tmp{#1{#2}\exf@listwalk#1}\fi\exf@tmp}
```

`\exerciseloop` Loop through a list of items in braces #1 which is expanded first; store code #2 to be executed in a callback function (need to use different callback function for each loop depth, callback function must be global to work within table cells); initialise item counter:

```
1440 \newcommand{\exerciseloop}[2]{\addtocounter{exf@loopdepth}{1}%
1441   \setcounter{exerciseloop}{0}%
1442   \exf@csdo\gdef{exf@listcallback@\roman{exf@loopdepth}}##1%
1443     {\stepcounter{exerciseloop}#2}%
1444   \edef\exf@tmp{#1}%
1445   \exf@csdotwo\exf@exptwo\exf@listwalk%
1446     {exf@listcallback@\roman{exf@loopdepth}}\exf@tmp}%
1447   \addtocounter{exf@loopdepth}{-1}}
```

`\exerciseloopstr` Loop through list and save result to `\exerciseloopret` (or optional argument #1):

```
1448 \newcommand{\exerciseloopstr}[3][\exerciseloopret]{%
1449   \def#1{\exerciseloop{#2}{\protected@edef#1{#1#3}}}
```

Lists Management.

`\exf@notedata@sheet` Store a sheet tag:

```
1450 \def\exf@sheetlist{}
1451 \newcommand{\exf@notedata@sheet}[2]{%
1452   \xdef\exf@sheetlist{\exf@sheetlist{#1}}}
```

`exf@notedata@problem` Store a problem tag:

```
1453 \def\exf@problemlist{}
1454 \newcommand{\exf@notedata@problem}[2]{%
1455   \xdef\exf@problemlist{\exf@problemlist{#1}}%
1456   \if @#2@\else%
1457     \ifcsname exf@problemlist@#2\endcsname\else%
1458       \exf@csdo\gdef{exf@problemlist@#2}{}\fi%
1459     \exf@csdo\xdef{exf@problemlist@#2}%
1460     {\csname exf@problemlist@#2\endcsname{#1}}%
1461   \fi}
```

`@notedata@subproblem` Store a subproblem tag:

```

1462 \newcommand{\exf@notedata@subproblem}[2]{%
1463   \ifcsname exf@subproblemlist@#2\endcsname\else%
1464     \exf@csdo\gdef{exf@subproblemlist@#2}{\fi%
1465     \exf@csdo\xdef{exf@subproblemlist@#2}%
1466     {\csname exf@subproblemlist@#2\endcsname{#1}}}
```

`\getsheetlist` Get list of sheet tags, problem tags (all, within current or particular sheet), subproblem

`\getproblemlist` tags (within current or particular problem):

`\getsubproblemlist`

```

1467 \newcommand{\getsheetlist}[1]{\exf@sheetlist}
1468 \newcommand{\getproblemlist}[1]{\if @#1%
1469   \ifdefined\sheettag\exf@csor{exf@problemlist@\sheettag}{}%
1470   \else\exf@problemlist\fi%
1471   \else%
1472   \if *#1\exf@problemlist\else\exf@csor{exf@problemlist@#1}{\fi%
1473   \fi}
1474 \newcommand{\getsubproblemlist}[1]{\if @#1%
1475   \exf@csor{exf@subproblemlist@problemtag}{\else%
1476   \exf@csor{exf@subproblemlist@#1}{\fi}
```

C.10 Sheet Environment

`exf@sheet` Define options for sheet environment:

```

1477 \define@key{exf@sheet}{points}{\exf@scanpoints\exf@sheet@points#1++@}
1478 \define@key{exf@sheet}{number}{\setcounter{exf@sheetcounter}{#1}%
1479   \addtocounter{exf@sheetcounter}{-1}\refstepcounter{exf@sheetcounter}}
1480 \define@key{exf@sheet}{label}{\def\exf@label{#1}}
1481 \define@key{exf@sheet}{tag}{\def\sheettag{#1}}
```

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

```

1482 \newenvironment{\exf@sheetname}[1][\fi%
```

Insert hook code to clear page, step counter:

```

1483   \exf@config@insertsheetclearpage%
1484   \refstepcounter{exf@sheetcounter}%
```

Use equation counter for sheets:

```

1485   \ifexf@style@sheetequation%
1486     \exf@eqsav\value{equation}\relax%
1487     \setcounter{equation}{\value{exf@sheetequation}}%
1488     \let\theequation\theexf@sheetequation%
1489     \let\theHequation\theHexf@sheetequation%
1490   \fi%
```

Reset optional arguments, process arguments:

```

1491   \let\exf@sheet@points\@undefined%
1492   \def\sheettag{\getexerciseconfig{tagsheet}}%
1493   \let\exf@sheet@points@total\@undefined%
1494   \let\exf@label\@undefined%
1495   \setkeys{exf@sheet}{#1}%
```

Process automatic and manual labels:

```

1496   \ifexf@autolabelsheet\label{\exf@config@labelsheet{\sheettag}}\fi%
```

```

1497 \ifdefined\exf@label\label{\exf@label}\fi%
1498 \exf@writedata{sheet}{\sheettag}{}%

```

Set points from explicit input or from .aux storage:

```

1499 \ifdefined\exf@sheet@points%
1500 \let\exf@sheet@points@given\exf@empty%
1501 \else%
1502 \let\exf@sheet@points@given\@undefined%
1503 \ifcsname exf@sheetpoints@\sheettag\endcsname%
1504 \exf@csdotwo\let\exf@tmp{\exf@sheetpoints@\sheettag}%
1505 \exf@exptwo\exf@scanpoints\exf@sheet@points\exf@tmp++@%
1506 \fi\fi%

```

Process metadata:

```

1507 \exf@ifis\exf@metadata{sheet}{{%
1508 \ifx\exf@data@sheet@author\exf@empty\else%
1509 \let\exf@data@author\exf@data@sheet@author\fi%
1510 \def\exf@data@title{\exf@config@composemetasheet%
1511 {\csname the\exf@sheetcounter\endcsname}{\exf@data@sheet@rawtitle}}%
1512 \exf@writemetadata}\gdef\exf@metadata{off}}%

```

Insert hook code:

```

1513 \exf@config@insertsheetbefore%

```

Add table of contents line:

```

1514 \ifx\exf@config@toclevelsheet\exf@empty\else%
1515 \ifdefined\phantomsection\phantomsection\fi\fi%
1516 \exf@addcontentsline{\exf@config@toclevelsheet}%
1517 {\exf@config@composetocsheet{\csname the\exf@sheetcounter\endcsname}%
1518 {\exf@data@sheet@rawtitle}}%

```

Write sheet title:

```

1519 \exf@config@insertsheettitle}%

```

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

```

1520 {\ifdefined\exf@sheet@points@given%
1521 \exf@warnmismatch{\exf@sheetname}{\exf@sheetcounter}%
1522 {\exf@sheet@points@total}{\exf@sheet@points}%

```

Test whether points have changed since last compile:

```

1523 \else%
1524 \exf@warnrerun{\exf@sheetname}{\exf@sheetcounter}%
1525 {\exf@sheet@points@total}{\exf@sheet@points}%

```

Store points:

```

1526 \let\exf@sheet@points\exf@sheet@points@total%
1527 \fi%

```

Write sheet points to .aux file:

```

1528 \ifdefined\exf@sheet@points%
1529 \expandafter\exf@writesheetpoints\exf@sheet@points%
1530 \fi%

```

Insert solutions:

```

1531 \exf@ifis\exf@solutionbelow{sheet}{\insertsolutions}%

```


Insert hook code:

```
1532 \exf@config@insertsheetafter%
1533 \exf@config@insertsheetclearpage%
```

Restore original equation counter:

```
1534 \ifexf@style@sheetequation%
1535 \setcounter{exf@sheetequation}{\value{equation}}%
1536 \setcounter{equation}{\exf@eqsav}%
1537 \fi%
```

Done:

```
1538 \ignorespacesafterend}
```

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

```
1539 \newcommand{\exercisecleardoublepage}{%
1540 \clearpage\ifexf@twoside\ifodd\value{page}\else%
1541 \thispagestyle{empty}\hbox{}\newpage\fi\fi}
```

C.11 Problem Environment

Print Problems.

`exf@problem` Define options for problem environment:

```
1542 \define@key{exf@problem}{points}{\exf@scanpoints\exf@problem@points#1++@}
1543 \define@key{exf@problem}{label}{\def\exf@label{#1}}
1544 \define@key{exf@problem}{tag}{\def\problemtag{#1}}
1545 \define@key{exf@problem}{sollabel}{\xdef\exf@sollabel{#1}}
```

`printproblem` Define `printproblem` environment:

```
1546 \newenvironment{printproblem}[1]{%
```

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

```
1547 \par\exf@config@styletext\addvspace{\exf@config@skipproblemabove}%
```

Step problem counter:

```
1548 \refstepcounter{\exf@problemcounter}%
```

Insert hook code:

```
1549 \exf@config@insertproblembefore%
```

Begin inner group, mark in problem:

```
1550 \begingroup%
1551 \def\exf@in@problem{}%
```

Use equation counter for problems:

```
1552 \ifexf@style@problemequation%
1553 \exf@eqsav\value{equation}\relax%
1554 \setcounter{equation}{\value{exf@problemequation}}%
1555 \let\theequation\theexf@problemequation%
1556 \let\theHequation\theHexf@problemequation%
1557 \fi%
```

Initialise variables, process arguments:

```

1558 \exf@init@block{\exf@config@skipprobleminfo}%
1559 \def\problemtag{\getexerciseconfig{tagproblem}}%
1560 \let\exf@problem@points\@undefined%
1561 \let\exf@label\@undefined%
1562 \global\let\exf@sollabel\@undefined%
1563 \let\exf@problem@points@total\@undefined%
1564 \setkeys{exf@problem,exf@probleminfo,exf@scanproblem}{#1}%

```

Process automatic and manual labels:

```

1565 \ifexf@autolabelproblem\label{\exf@config@labelproblem{\problemtag}}\fi%
1566 \ifdefined\exf@label\label{\exf@label}\fi%
1567 \exf@writedata{problem}{\problemtag}{\ifdefined\sheettag\sheettag\fi}%

```

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

```

1568 \global\let\exf@problem@solfirst\exf@empty%
1569 \xdef\exf@prevprob{\csname the\exf@problemcounter\endcsname}%
1570 \ifcsname the\exf@problemcounter\endcsname%
1571 \xdef\exf@prevprobhref{\exf@problemcounter.%
1572 \csname the\exf@problemcounter\endcsname}%
1573 \fi%
1574 \ifx\exf@data@problem@rawtitle\exf@empty%
1575 \global\let\exf@prevprobttitle\@undefined%
1576 \else%
1577 \protected\xdef\exf@prevprobttitle{\exf@data@problem@rawtitle}%
1578 \fi%
1579 \exf@clear@prevsubprob%

```

Set points from explicit input or from .aux storage:

```

1580 \ifdefined\exf@problem@points%
1581 \let\exf@problem@points@given\exf@empty%
1582 \else%
1583 \let\exf@problem@points@given\@undefined%
1584 \ifcsname exf@problempoints@\problemtag\endcsname%
1585 \exf@csdotwo\let\exf@tmp{\exf@problempoints@\problemtag}%
1586 \exf@exptwo\exf@scanpoints\exf@problem@points\exf@tmp++@%
1587 \fi\fi%
1588 \global\let\exf@prevpoints\exf@problem@points%
1589 \let\exf@problem@points@show\@undefined%
1590 \ifdefined\exf@problem@points%
1591 \let\exf@problem@points@show\exf@problem@points%
1592 \fi%

```

Disable points display if desired:

```

1593 \exf@ifis\exf@pointsat{off}{\let\exf@problem@points@show\@undefined}%

```

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

```

1594 \exf@ifis\exf@pointsat{start}{\exf@outpoints{\exf@append@intro}%
1595 {\exf@config@composepointspairstartproblem}{\exf@problem@points@show}}%
1596 \exf@ifis\exf@pointsat{start*}{\exf@outpoints{\exf@prepend@intro}%
1597 {\exf@config@composepointspairstartproblem}{\exf@problem@points@show}}%

```

Insert hook code, set problem body style:

```

1598 \exf@config@insertprobleminfo%
1599 \exf@config@styletextproblem%

```

Write title without item:

```

1600 \exf@addmargin\glueexpr\exf@config@skipproblemitem\relax%
1601 \ifdim\exf@addmargin=0pt%
1602 \exf@prepend@intro{%
1603 \exf@config@styletitle\exf@config@styletitleproblem%
1604 \exf@config@composetitleproblem{\csname the\exf@problemcounter\endcsname}%
1605 {\exf@data@problem@rawtitle}}}%

```

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

```

1606 \else%
1607 \ifdim\exf@addmargin<0pt%
1608 \settowidth\exf@addmargin{%
1609 \exf@config@styletitle\exf@config@styletitleproblem%
1610 \exf@config@composeitemproblem{\exf@config@counterproblemmax}%
1611 \exf@config@composeitemproblemsep}%
1612 \fi%

```

Define item label:

```

1613 \def\exf@introitem{\makebox[0cm][r]{%
1614 \exf@config@styletitle\exf@config@styletitleproblem%
1615 \exf@config@composeitemproblem{\csname the\exf@problemcounter\endcsname}%
1616 \exf@config@composeitemproblemsep}}%

```

Compose title:

```

1617 \ifx\exf@data@problem@rawtitle\exf@empty\else%
1618 \exf@prepend@intro{%
1619 \exf@config@styletitle\exf@config@styletitleproblem%
1620 \exf@config@composetitleproblem{\exf@empty}{\exf@data@problem@rawtitle}}}%
1621 \fi%
1622 \fi%

```

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

```

1623 \exf@ifis\exf@pointsat{margin}{%
1624 \exf@outpoints{\exf@prepend@def\exf@introitem}%
1625 {\exf@makepointsmarginproblem}{\exf@problem@points@show}}%

```

Write out opening line:

```

1626 \exf@open@block{\exf@config@skipproblemtitle}%

```

Add table of contents line:

```

1627 \exf@addcontentsline{\exf@config@toclevelproblem}%
1628 {\exf@config@composetocproblem{\csname the\exf@problemcounter\endcsname}%
1629 {\exf@data@problem@rawtitle}}}%

```

Done:

```

1630 \@afterindentfalse}%

```

End environment, show points if desired:

```

1631 {\exf@ifis\exf@pointsat{end}{\showpoints}}%

```

Perform sanity checks on total points if given explicitly:

```

1632 \ifdefined\exf@problem@points@given%
1633 \exf@warnmismatch{\exf@problemname}{\exf@problemcounter}%
1634 {\exf@problem@points@total}{\exf@problem@points}%

```

Warn if calculated total points have changed:

```
1635 \else%
1636 \exf@warnrerun{\exf@problemname}{\exf@problemcounter}%
1637 {\exf@problem@points@total}{\exf@problem@points}%
```

Read computed total points:

```
1638 \let\exf@problem@points\exf@problem@points@total%
1639 \fi%
```

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

```
1640 \ifdefined\exf@problem@points%
1641 \expandafter\exf@writeproblempoints\exf@problem@points%
1642 \exf@exptwo\exf@addtopoints\exf@sheet@points@total\exf@problem@points%
```

Warn if no points given for present problem but previously:

```
1643 \else\ifdefined\exf@sheet@points@total%
1644 \PackageWarning{exframe}{no points defined for \exf@problemname}%
1645 \fi\fi%
```

Solutions to subproblems must be declared within problem environment:

```
1646 \exf@clear@prevsubprob%
```

End paragraph and environment:

```
1647 \par\exf@close@block%
```

Display solution if desired:

```
1648 \exf@ifis\exf@solutionbelow{problem}{%
1649 \exf@config@insertproblemsolution%
1650 \exf@showsolutions{\exf@config@composetitlesolutionmulti}{}}%
```

Restore original equation counter:

```
1651 \ifexf@style@problemequation%
1652 \setcounter{exf@problemequation}{\value{equation}}%
1653 \setcounter{equation}{\exf@eqsav}%
1654 \fi%
```

End inner group:

```
1655 \endgroup%
```

Insert hook code, vertical skip:

```
1656 \exf@config@insertproblemafter%
1657 \addvspace{\exf@config@skipproblembelow}%
```

Display solution if desired:

```
1658 \exf@ifis\exf@solutionbelow{problem*}{%
1659 \exf@showsolutions{\exf@config@composetitlesolutionmulti}{}}%
```

Done:

```
1660 \ignorespacesafterend}
```

Read Problem Environment.

exf@scanproblem Define options for scanning problem environment:

```
1661 \define@boolkey{exf@scanproblem}[exf@scanproblem@]{disable}[true]{}%
```

exf@problem@direct Define direct output version of problem environment; pass on to printproblem environment:

```
1662 \newenvironment{exf@problem@direct}[1][]{%
1663   {\printproblem{#1}}{\endprintproblem\ignorespacesafterend}}
```

exf@problem@scan Define scan version of problem environment; use \exf@scanblock to properly parse optional argument and pass on to exf@scanproblem:

```
1664 \newenvironment{exf@problem@scan}%
1665   {\exf@scanblock{\exf@scanproblem}}{\endexf@scanproblem}%
1666 \newenvironment{exf@scanproblem}[2]{%
```

Determine problem display:

```
1667   \global\exf@scanproblem@disablefalse%
1668   \setkeys*{exf@scanproblem}{#1}%
1669   \exf@config@insertproblemselect{#1}%
1670   \ifexf@scanproblem@disable\global\exf@scanproblem@disabletrue\fi%
```

Circumvent problem display:

```
1671   \ifexf@scanproblem@disable%
1672     \def\exf@verbatim@process{\@gobble}%
1673   \else%
```

Write separator and printproblem environment to buffer:

```
1674     \ifexf@lineno\exf@addline\exf@probbuf{\exf@linesep}%
1675     \exf@addline\exf@probbuf{\exf@lineno}\fi%
1676     \exf@addline\exf@probbuf%
1677     {\@backslashchar begin{printproblem}{#1}}%
```

Start verbatim processing:

```
1678     \def\exf@verbatim@process{\exf@append@buf\exf@probbuf}%
1679     \fi%
1680     \exf@verbatim#2}%%
```

End verbatim processing; close printproblem environment:

```
1681   {\exf@endverbatim%
1682   \ifexf@scanproblem@disable\else%
1683     \exf@addline\exf@probbuf{\@backslashchar end{printproblem}}}%
1684   \global\exf@probbuf@cleanfalse%
1685   \fi%
```

Write buffer to file if output file open:

```
1686   \ifexf@probfile@open%
1687     \exf@write@buf\exf@probfile\exf@probbuf%
1688     \exf@clear@probbuf%
1689   \fi%
```

Output buffer immediately:

```
1690   \ifexf@problemmanual\else%
1691     \exf@source@buf\exf@probbuf%
1692     \exf@clear@probbuf%
1693   \fi%
```

Done:

```
1694 \ignorespacesafterend}
```

problem Define problem environment (potentially using custom name) to choose between direct and buffered version:

```
1695 \newenvironment{\exf@problemname}%  
1696 {\ifexf@problembuf\let\exf@tmp\exf@problem@scan%  
1697 \else\let\exf@tmp\exf@problem@direct\fi%  
1698 \exf@tmp}%  
1699 {\ifexf@problembuf\let\exf@tmp\endexf@problem@scan%  
1700 \else\let\exf@tmp\endexf@problem@direct\fi%  
1701 \exf@tmp}
```

C.12 Problem Blocks

Problem Block Handling.

\exf@problemstitle Compose the title for a problem block section:

```
1702 \newcommand{\exf@problemstitle}{%
```

Check whether title is empty:

```
1703 \protected@edef\exf@problemstitleexp{\exf@config@composetitleproblems}%  
1704 \ifx\exf@problemstitleexp\exf@empty\else%
```

Output section line:

```
1705 \exf@section{\exf@config@skipproblemstitle}%  
1706 {\exf@config@styletitle\exf@config@styletitleproblems}%  
1707 \exf@problemstitleexp}%  
1708 \exf@addcontentsline{\exf@config@toclevelproblems}%  
1709 {\exf@config@composetocproblems}%  
1710 \fi}
```

\exf@showproblemsin Output problem block intro:

```
1711 \newcommand{\exf@showproblemsin}{%
```

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

```
1712 \par\exf@config@styletext\addvspace{\exf@config@skipproblemsabove}%  
1713 \exf@config@insertproblemsbefore}
```

\exf@showproblemsout Output problem block outro:

```
1714 \newcommand{\exf@showproblemsout}{%
```

Insert hook code; close paragraph; add vertical space:

```
1715 \exf@config@insertproblemsafter%  
1716 \par\exf@config@styletext\addvspace{\exf@config@skipproblemsbelow}}
```

\exf@showproblems Output problem block in buffer:

```
1717 \newcommand{\exf@showproblems}{%
```

Do nothing if buffer is empty (avoid titles):

```
1718 \ifexf@probbuf@clean\else\begingroup%
```

Execute output problem block intro:

```
1719 \exf@showproblemsin%
1720 \exf@problemstitle%
```

Source and clear buffer:

```
1721 \exf@source@buf\exf@probbuf%
1722 \exf@clear@probbuf%
```

Execute output problem block outro:

```
1723 \exf@showproblemsout%
1724 \endgroup\fi}
```

Problems Buffer Interface.

`\writeproblems` Open a file #1.prb for writing problems; default is present main file name:

```
1725 \newcommand{\writeproblems}[1][\jobname]{%
1726 \exf@close@probfile\exf@start@probfile{#1}}
```

`\closeproblems` Close problems output file (if open):

```
1727 \newcommand{\closeproblems}{\exf@close@probfile}
```

`\readproblems` Read problems from file #1.prb; default is present main file name; switch layout and add heading:

```
1728 \newcommand{\readproblems}[1][\jobname]{\exf@close@probfile%
1729 \beginngroup%
1730 \exf@config@styletext\exf@config@styletextproblem%
1731 \exf@problemstitle%
1732 \input{#1\exf@config@extproblems}%
1733 \endgroup}
```

`\insertproblems` Show problems buffer:

```
1734 \newcommand{\insertproblems}{\exf@showproblems}
```

C.13 Subproblem Environment

Print Subproblems.

`exf@subproblem` Define options for subproblem environment:

```
1735 \define@key{exf@subproblem}{points}{\exf@scanpoints\exf@subproblem@points#1++@}
1736 \define@key{exf@subproblem}{label}{\def\exf@label{#1}}
1737 \define@key{exf@subproblem}{tag}{\def\subproblemtag{#1}}
```

`printsubproblem` Define printsubproblem environment:

```
1738 \newenvironment{printsubproblem}[1]{%
```

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

```
1739 \par{\exf@config@styletext\addvspace{\exf@config@skipsubproblemabove}}%
1740 \refstepcounter{\exf@subproblemcounter}%
```

Insert hook code:

```
1741 \exf@config@insertsubproblembefore%
```

Start inner group, mark in subproblem:

```
1742 \begingroup%
1743 \def\exf@in@subproblem{}%
```

Initialise variables, process arguments:

```
1744 \exf@init@block{\exf@config@skipsubprobleminfo}%
1745 \def\subproblemtag{\getexerciseconfig{tagsubproblem}}%
1746 \let\exf@subproblem@points\@undefined%
1747 \let\exf@label\@undefined%
1748 \setkeys{exf@subproblem,exf@probleminfo,exf@scansubproblem}{#1}%
```

Process automatic and manual labels:

```
1749 \ifexf@autolabelproblem\label{\exf@config@labelsubproblem%
1750   {\subproblemtag}}\fi%
1751 \ifdefined\exf@label\label{\exf@label}\fi%
1752 \exf@writedata{subproblem}{\subproblemtag}{\problemtag}%
```

Remember subproblem counter for solution:

```
1753 \xdef\exf@prevsubprob{\csname the\exf@subproblemcounter\endcsname}%
1754 \ifcsname theH\exf@subproblemcounter\endcsname%
1755   \xdef\exf@prevsubprobhref{\exf@subproblemcounter.%
1756     \csname theH\exf@subproblemcounter\endcsname}%
1757 \fi%
```

Remember points for display; disable points display if desired:

```
1758 \let\exf@subproblem@points@show\@undefined%
1759 \ifdefined\exf@subproblem@points%
1760   \let\exf@subproblem@points@show\exf@subproblem@points\fi%
1761 \exf@ifis\exf@subpointsat{off}{\let\exf@subproblem@points@show\@undefined}%
```

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

```
1762 \ifdefined\exf@subproblem@points%
1763   \global\let\exf@prevpoints\exf@subproblem@points%
1764   \expandafter\exf@writesubproblem@points\exf@subproblem@points%
1765   \exf@exptwo\exf@addtopoints\exf@problem@points@total\exf@subproblem@points%
```

Warn if no points given for present subproblem but previously:

```
1766 \else\ifdefined\exf@problem@points@total%
1767   \PackageWarning{exframe}{no points defined for \exf@subproblemname}%
1768 \fi\fi%
```

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

```
1769 \exf@ifis\exf@subpointsat{start}{\exf@outpoints{\exf@append@intro}%
1770   {\exf@config@composepointspairstartsubproblem}{\exf@subproblem@points@show}}%
1771 \exf@ifis\exf@subpointsat{start*}{\exf@outpoints{\exf@prepend@intro}%
1772   {\exf@config@composepointspairstartsubproblem}{\exf@subproblem@points@show}}%
```

Insert hook code:

```
1773 \exf@config@insertsubprobleminfo%
```

Write opening line without item:


```

1774 \exf@addmargin\glueexpr\exf@config@skipsubproblemitem\relax%
1775 \ifdim\exf@addmargin=0pt%
1776 \exf@prepend@intro{%
1777 \exf@config@styletitle\exf@config@styletitlesubproblem%
1778 \exf@config@composetitlesubproblem{%
1779 \csname the\exf@subproblemcounter\endcsname}}}%

```

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

```

1780 \else%
1781 \ifdim\exf@addmargin<0pt%
1782 \settowidth\exf@addmargin{%
1783 \exf@config@styletitle\exf@config@styletitlesubproblem%
1784 \exf@config@composeitemsubproblem{\exf@config@countersubproblemmax}%
1785 \exf@config@composeitemsubproblemsep}%
1786 \fi%

```

Define item label:

```

1787 \def\exf@introitem{\makebox[0cm][r]{%
1788 \exf@config@styletitle\exf@config@styletitlesubproblem%
1789 \exf@config@composeitemsubproblem%
1790 {\csname the\exf@subproblemcounter\endcsname}%
1791 \exf@config@composeitemsubproblemsep}}%
1792 \fi%

```

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

```

1793 \exf@ifis\exf@subpointsat{margin}{%
1794 \exf@outpoints{\exf@prepend@def\exf@introitem}%
1795 {\exf@makepointsmarginsubproblem}{\exf@subproblem@points@show}}%

```

Write out opening line:

```

1796 \exf@open@block{\exf@config@skipsubproblemtitle}%

```

Done:

```

1797 \@afterindentfalse}%

```

End environment, show points if desired:

```

1798 {\exf@ifis\exf@subpointsat{end}{\showpoints}%

```

End paragraph and environment:

```

1799 \par\exf@close@block%

```

Display solution if desired:

```

1800 \exf@ifis\exf@solutionbelow{subproblem*}{%
1801 \exf@config@insertsubproblemsolution%
1802 \exf@showsolutions{\exf@config@composetitlesolutionsingle}{}}%

```

End inner group:

```

1803 \endgroup%

```

Insert hook code, vertical skip:

```

1804 \exf@config@insertsubproblemhook%
1805 {\exf@config@styletext\addvspace{\exf@config@skipsubproblemhook}}%

```

Display solution if desired:

```

1806 \exf@ifis\exf@solutionbelow{subproblem}{%
1807 \exf@showolutions{\exf@config@composetitlesolutionsingle}{}}%

```

Done:

```

1808 \ignorespacesafterend}

```

Read Subproblem Environment.

exf@scansubproblem Define options for scanning subproblem environment:

```

1809 \define@boolkey{exf@scansubproblem}[exf@scansubproblem@]{disable}[true]{}

```

exf@subproblem@direct Define direct output version of subproblem environment; pass on to `printsubproblem` environment:

```

1810 \newenvironment{exf@subproblem@direct}[1][]{%
1811 {\printsubproblem{#1}}{\endprintsubproblem\ignorespacesafterend}

```

exf@subproblem@scan Define scan version of subproblem environment; use `\exf@scanblock` to properly parse optional argument and pass on to `exf@scansubproblem`:

```

1812 \newenvironment{exf@subproblem@scan}%
1813 {\exf@scanblock{\exf@scansubproblem}}{\endexf@scansubproblem}%
1814 \newenvironment{exf@scansubproblem}[2]{%

```

Determine subproblem display:

```

1815 \global\exf@scansubproblem@disablefalse%
1816 \setkeys*{exf@scansubproblem}{#1}%
1817 \exf@config@insertsubproblemsselect{#1}%
1818 \ifexf@scansubproblem@disable\global\exf@scansubproblem@disabletrue\fi%

```

Circumvent subproblem display:

```

1819 \exf@clear@subprobbuf%
1820 \ifexf@scansubproblem@disable%
1821 \global\let\exf@prevsubprob\exf@empty%
1822 \def\exf@verbatim@process{\@gobble}%
1823 \else%

```

Write `printsubproblem` environment to buffer:

```

1824 \exf@addline\exf@subprobbuf%
1825 {\@backslashchar begin{printsubproblem}{#1}}%

```

Start verbatim processing:

```

1826 \def\exf@verbatim@process{\exf@append@buf\exf@subprobbuf}%
1827 \fi%
1828 \exf@verbatim#2}%

```

End verbatim processing; close `printsubproblem` environment:

```

1829 {\exf@endverbatim%
1830 \ifexf@scansubproblem@disable\else%
1831 \exf@addline\exf@subprobbuf{\@backslashchar end{printsubproblem}}%
1832 \fi%

```

Output buffer immediately:

```

1833 \exf@source@buf\exf@subprobbuf%

```

Done:

```
1834 \ignorespacesafterend}
```

subproblem Define `subproblem` environment (potentially using custom name) to choose between direct and buffered version:

```
1835 \newenvironment{\exf@subproblemname}%
1836 {\ifexf@subproblembuf\let\exf@tmp\exf@subproblem@scan%
1837 \else\let\exf@tmp\exf@subproblem@direct\fi%
1838 \exf@tmp}%
1839 {\ifexf@subproblembuf\let\exf@tmp\endexf@subproblem@scan%
1840 \else\let\exf@tmp\endexf@subproblem@direct\fi%
1841 \exf@tmp}
```

C.14 Solution Environment

Print Solutions.

exf@solution Define options for solution environment:

```
1842 \define@key{exf@solution}{prob}{\def\exf@solprob{#1}}
1843 \define@key{exf@solution}{subprob}{\def\exf@solsubprob{#1}}
1844 \define@key{exf@solution}{problemtag}{\def\problemtag{#1}}
1845 \define@key{exf@solution}{sheettag}{\def\sheettag{#1}}
1846 \define@key{exf@solution}{href}{\def\exf@solhref{#1}}
1847 \define@key{exf@solution}{label}{\def\exf@label{#1}}
1848 \define@key{exf@solution}{points}{\exf@scanpoints\exf@solution@points#1++@}
1849 \define@key{exf@solution}{probttitle}{\def\exf@solprobttitle{#1}}
1850 \define@key{exf@solution}{firstsol}[]{\let\exf@solfirst\exf@empty}
```

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

```
1851 \newenvironment{printsolution}[1]{%
```

Start new paragraph, add vertical space:

```
1852 \par{\exf@config@styletext\addvspace{\exf@config@skipsolutionabove}}%
```

Insert hook code:

```
1853 \exf@config@insertsolutionbefore%
```

Use equation counter for solutions:

```
1854 \ifexf@style@solutionequation%
1855 \exf@eqsav\value{equation}\relax%
1856 \setcounter{equation}{\value{exf@solutionequation}}%
1857 \let\theequation\theexf@solutionequation%
1858 \let\theHequation\theHexf@solutionequation%
1859 \fi%
```

Start a group, initialise variables, process arguments:

```
1860 \begingroup%
1861 \def\exf@in@solution{}%
1862 \def\exf@solprob{}%
1863 \def\exf@solsubprob{}%
1864 \let\exf@solfirst\@undefined%
1865 \let\exf@label\@undefined%
```

```

1866 \let\exf@solution@points\@undefined%
1867 \let\exf@solution@points@total\@undefined%
1868 \def\exf@solhref{}\fi%
1869 \exf@init@block{\exf@config@skipsolutioninfo}%
1870 \setkeys{exf@solution,exf@probleminfo,exf@scansolution}{#1}%

```

Set solution counter to reflect associated problem:

```

1871 \exf@csdo\def{the\exf@solutioncounter}%
1872 {\exf@config@composeitemsolutionlabel{\exf@solprob}{\exf@solsubprob}}%
1873 \refstepcounter{\exf@solutioncounter}%

```

Set label:

```

1874 \ifdefined\exf@label\label{\exf@label}\fi%

```

Remember points for display; disable points display if desired:

```

1875 \let\exf@solution@points@show\@undefined%
1876 \ifdefined\exf@solution@points%
1877 \let\exf@solution@points@show\exf@solution@points\fi%
1878 \exf@ifis\exf@solpointsat{off}{\let\exf@solution@points@show\@undefined}%

```

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

```

1879 \exf@ifis\exf@solpointsat{start}{\exf@outpoints{\exf@append@intro}%
1880 {\exf@config@composepointspairstartsolution}{\exf@solution@points@show}}%
1881 \exf@ifis\exf@solpointsat{start*}{\exf@outpoints{\exf@prepend@intro}%
1882 {\exf@config@composepointspairstartsolution}{\exf@solution@points@show}}%

```

Insert hook code, set solution body style:

```

1883 \exf@config@insertsolutioninfo%
1884 \exf@config@styletext\exf@config@styletextsolution%

```

Determine solution for problem or subproblem:

```

1885 \exf@addmargin\glueexpr\ifx\exf@solsubprob\exf@empty%
1886 \exf@config@skipsolutionitem\else\exf@config@skipsolutionitemsub\fi\relax%

```

Write title without item:

```

1887 \ifdim\exf@addmargin=0pt%
1888 \protected@edef\exf@solution@title{%
1889 \exf@composetitle{\exf@solprob}{\exf@solsubprob}}%
1890 \ifx\exf@solsubprob\exf@empty\ifdefined\exf@solfirst\else%
1891 \let\exf@solution@title\exf@empty\fi\fi%
1892 \ifx\exf@solution@title\exf@empty\else%
1893 \exf@prepend@intro{%
1894 \exf@config@styletitle\exf@config@styletitlesolution%
1895 \ifexf@solutionhref\exf@href{\exf@solhref}%
1896 {\exf@solution@title}\else\exf@solution@title\fi}%
1897 \fi%

```

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

```

1898 \else%
1899 \ifdim\exf@addmargin<0pt%
1900 \settowidth\exf@addmargin{%
1901 \exf@config@styletitle\exf@config@styletitlesolution%
1902 \ifx\exf@solsubprob\exf@empty%
1903 \let\exf@tmp\exf@config@composeitemsolutionfirst\else%
1904 \let\exf@tmp\exf@config@composeitemsolutionfirstsub\fi%

```

```

1905 \protected@edef\exf@solution@item{\exf@tmp%
1906   {\exf@config@counterproblemmax}{\exf@config@countersubproblemmax}}%
1907 \ifx\exf@solution@item\exf@empty\else%
1908   \exf@solution@item\exf@config@composeitemsolutionsep\fi}%
1909 \fi%

```

Set item label depending on problem or subproblem:

```

1910 \ifx\exf@solsubprob\exf@empty\ifdefined\exf@solfirst%
1911   \let\exf@tmp\exf@config@composeitemsolutionfirst\else%
1912   \let\exf@tmp\exf@config@composeitemsolution\fi%
1913 \else\ifdefined\exf@solfirst%
1914   \let\exf@tmp\exf@config@composeitemsolutionfirstsub\else%
1915   \let\exf@tmp\exf@config@composeitemsolutionsub\fi\fi%
1916 \protected@edef\exf@solution@item{\exf@tmp{\exf@solprob}{\exf@solsubprob}}%

```

Define item label:

```

1917 \def\exf@introitem{\ifx\exf@solution@item\exf@empty\else\makebox[0cm][r]{%
1918   \exf@config@styletitle\exf@config@styletitlesubproblem%
1919   \ifx\exf@solutionhref\exf@href{\exf@solhref}{\exf@solution@item}%
1920   \else\exf@solution@item\fi%
1921   \exf@config@composeitemproblemsep}\fi}%
1922 \fi%

```

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

```

1923 \exf@ifis\exf@solpointsat{margin}{%
1924   \exf@outpoints{\exf@prepend@def\exf@introitem}%
1925   {\exf@makepointsmarginsolution}{\exf@solution@points@show}}%

```

Write out opening line:

```

1926 \exf@open@block{\exf@config@skipsolutiontitle}%

```

Done:

```

1927 \@afterindentfalse}%

```

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

```

1928 {\exf@ifis\exf@solpointsat{end}{\showpoints}%
1929   \exf@warnmismatch{\exf@solutionname}{\exf@solutioncounter}%
1930   {\exf@solution@points@total}{\exf@solution@points}%

```

End paragraph and environment:

```

1931 \par\exf@close@block%

```

Restore original equation counter:

```

1932 \ifx\exf@style@solutionequation%
1933   \setcounter{\exf@solutionequation}{\value{equation}}%
1934   \setcounter{equation}{\exf@eqsav}%
1935 \fi%

```

End inner group:

```

1936 \endgroup%

```

Vertical skip, insert hook code:

```

1937 {\exf@config@styletext\addvspace{\exf@config@skipsolutionbelow}}%
1938 \exf@config@insertsolutionafter%

```

Done:

```
1939 \ignorespacesafterend}
```

`\solutionssection` Define a section for a problem within a solution block:

```
1940 \newcommand{\solutionssection}[1]{\begingroup%
```

Initialise variables, process arguments:

```
1941 \def\exf@solprob{}%
1942 \def\exf@solsubprob{}%
1943 \def\exf@solprobttitle{}%
1944 \let\exf@label\@undefined%
1945 \let\exf@solhref\@undefined%
1946 \setkeys{exf@solution,exf@scansolution}{#1}%
```

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

```
1947 \let\exf@composesectitle\exf@config@composetitlesolutionsproblemmulti%
1948 \exf@ifis\exf@solutionbelow{problem}{\let\exf@composesectitle%
1949 \exf@config@composetitlesolutionsproblemsingle}%
1950 \exf@ifis\exf@solutionbelow{problem*}{\let\exf@composesectitle%
1951 \exf@config@composetitlesolutionsproblemsingle}%
1952 \def\exf@solutionstoc{\exf@addcontentsline{\exf@config@toclevelsolution}%
1953 {\exf@config@composetocsolution{\exf@solprob}{\exf@solprobttitle}}}%
```

Write section line:

```
1954 \addvspace{\exf@config@skipsolutionsproblemabove}%
1955 \exf@solutionsection{\exf@config@styletitlesolutionsproblem}%
1956 {\exf@composesectitle{\exf@solprob}{\exf@solprobttitle}}%
1957 {\exf@config@skipsolutionsproblemttitle}%
1958 {\exf@solutionstoc}{\exf@label}{\exf@solhref}%
1959 \endgroup}
```

Read Solution Environment.

`exf@scansolution` Define options for scanning solution environment:

```
1960 \define@key{exf@scansolution}{forproblem}[]{\exf@clear@prevsubprob}
1961 \define@boolkey{exf@scansolution}[exf@scansolution@]{disable}[true]{}
```

`exf@clear@prevsubprob` Clear previous subproblem register:

```
1962 \newcommand{\exf@clear@prevsubprob}{%
1963 \global\let\exf@prevsubprob\@undefined%
1964 \global\let\exf@prevsubprohref\@undefined%
1965 \global\let\exf@prevpoints\@undefined%
1966 \global\exf@scansubproblem@disablefalse}
```

`\exf@process@solsec` If this is the first solution within a problem section, write section heading to buffer:

```
1967 \newcommand{\exf@process@solsec}{%
1968 \def\exf@probarg{\ifdefined\exf@prevprob prob={\exf@prevprob}\fi%
1969 \ifdefined\exf@prevprobttitle,probttitle={\exf@prevprobttitle}\fi%
1970 \ifdefined\exf@prevprohref,href={\exf@prevprohref}\fi%
1971 \ifdefined\exf@sollabel,label={\exf@sollabel}\fi}%
1972 \exf@ifis\exf@solutionbelow{here}{\let\exf@probarg\@undefined}%
```

```

1973 \exf@ifis\exf@solutionbelow{subproblem}{\let\exf@probarg\@undefined}%
1974 \exf@ifis\exf@solutionbelow{subproblem*}{\let\exf@probarg\@undefined}%
1975 \ifdefined\exf@probarg%
1976 \ifexf@lineno\exf@addline\exf@solbuf{\exf@linesep}%
1977 \exf@addline\exf@solbuf{\exf@lineno}\fi%
1978 \exf@addline\exf@solbuf{\@backslashchar solutionssection{\exf@probarg}}%
1979 \exf@addline\exf@solbuf{}%
1980 \fi}

```

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

```

1981 \newcommand{\exf@generate@solprobarg}{%
1982 \edef\exf@solprobarg{%
1983 \ifdefined\exf@problem@solfirst firstsol,\fi%
1984 \ifdefined\exf@prevprob prob={\exf@prevprob},\fi%
1985 \ifdefined\exf@prevsubprob subprob={\exf@prevsubprob},%
1986 \ifdefined\exf@prevsubprobhref href={\exf@prevsubprobhref},\fi%
1987 \else%
1988 \ifdefined\exf@prevprobhref href={\exf@prevprobhref},\fi%
1989 \fi%
1990 \ifdefined\exf@prevpoints points=%
1991 {\expandafter\exf@formatpoints\exf@prevpoints},\fi%
1992 \ifdefined\sheettag sheettag={\sheettag},\fi%
1993 \ifdefined\problemtag problemtag={\problemtag},\fi}}

```

exf@solution@direct Define direct output version of solution environment; pass on to `printsolution` environment:

```

1994 \newenvironment{exf@solution@direct}[1][]%
1995 {\showpoints%
1996 \exf@generate@solprobarg%
1997 \global\let\exf@problem@solfirst\@undefined%
1998 \exf@clear@prevsubprob%
1999 \exf@showsolutionsin%
2000 \let\exf@composetitle\exf@config@composetitlesolutionsingle%
2001 \exf@exptwo\printsolution{\exf@solprobarg#1}}%
2002 {\endprintsolution%
2003 \exf@showsolutionsout%
2004 \ignorespacesafterend}

```

exf@solution@scan Define scan version of solution environment; use `\exf@scanblock` to properly parse optional argument and pass on to `exf@scansolution`:

```

2005 \newenvironment{exf@solution@scan}%
2006 {\exf@scanblock{\exf@scansolution}}{\endexf@scansolution}%
2007 \newenvironment{exf@scansolution}[2]{%

```

Determine solution association and display:

```

2008 \exf@scansolution@disablefalse%
2009 \setkeys*{exf@scansolution}{#1}%
2010 \ifdefined\exf@prevsubprob%
2011 \ifexf@scansubproblem@disable\exf@scansolution@disabletrue\fi%
2012 \else%
2013 \ifexf@scanproblem@disable\exf@scansolution@disabletrue\fi%
2014 \fi%

```

Circumvent solution display:

```

2015 \ifexf@scansolution@disable%
2016 \exf@clear@prevsubprob%
2017 \def\exf@verbatim@process{\@gobble}%
2018 \else%

```

If solution is to be displayed immediately, make sure to display points first; insert solution section heading in buffer; compose additional arguments to `printsolution`:

```

2019 \exf@ifis\exf@solutionbelow{here}{\showpoints}%
2020 \exf@generate@solprobarg%
2021 \ifdefined\exf@problem@solfirst\exf@process@solsec\fi%
2022 \global\let\exf@problem@solfirst\@undefined%
2023 \exf@clear@prevsubprob%

```

Write separator and `printsolution` environment to buffer:

```

2024 \ifexf@lineno\exf@addline\exf@solbuf{\exf@linesep}%
2025 \exf@addline\exf@solbuf{\exf@lineno}\fi%
2026 \exf@addline\exf@solbuf%
2027 {\@backslashchar begin{printsolution}{\exf@solprobarg#1}}%

```

Start verbatim processing:

```

2028 \def\exf@verbatim@process{\exf@append@buf\exf@solbuf}%
2029 \fi%
2030 \exf@verbatim#2}%

```

End verbatim processing; close `printsolution` environment:

```

2031 {\exf@endverbatim%
2032 \ifexf@scansolution@disable\else%
2033 \exf@addline\exf@solbuf{\@backslashchar end{printsolution}}%
2034 \global\exf@solbuf@cleanfalse%
2035 \fi%

```

Write buffer to file if output file open:

```

2036 \ifexf@solfile@open%
2037 \exf@write@buf\exf@solfile\exf@solbuf%
2038 \exf@clear@solbuf%
2039 \fi%

```

Drop buffer if solutions not to be displayed:

```

2040 \ifsolutions\else\exf@clear@solbuf\fi%

```

Display solution immediately in various cases:

```

2041 \exf@ifis\exf@solutionbelow{here}{%
2042 \exf@showolutions{\exf@config@composetitlesolutionsingle}{}}%
2043 \ifdefined\exf@in@subproblem\else%
2044 \exf@ifis\exf@solutionbelow{subproblem}{%
2045 \exf@showolutions{\exf@config@composetitlesolutionsingle}{}}%
2046 \exf@ifis\exf@solutionbelow{subproblem*}{%
2047 \exf@showolutions{\exf@config@composetitlesolutionsingle}{}}\fi%
2048 \ifdefined\exf@in@problem\else%
2049 \exf@ifis\exf@solutionbelow{problem}{%
2050 \exf@showolutions{\exf@config@composetitlesolutionmulti}{}}%
2051 \exf@ifis\exf@solutionbelow{problem*}{%
2052 \exf@showolutions{\exf@config@composetitlesolutionmulti}{}}\fi%

```

Done:

```

2053 \ignorespacesafterend}

```


solution Define solution environment (potentially using custom name) to choose between direct and buffered version:

```

2054 \newenvironment{\exf@solutionname}%
2055 {\ifexf@solutionbuf\let\exf@tmp\exf@solution@scan%
2056 \else\let\exf@tmp\exf@solution@direct\fi%
2057 \exf@tmp}%
2058 {\ifexf@solutionbuf\let\exf@tmp\endexf@solution@scan%
2059 \else\let\exf@tmp\endexf@solution@direct\fi%
2060 \exf@tmp}

```

C.15 Solution Blocks

Solution Block Handling.

\exf@solutionssection Output solution block section:

```

2061 \newcommand{\exf@solutionssection}[6]{%

```

Check whether title is empty:

```

2062 \protected@edef\exf@solutionstitleexp{#2}%
2063 \ifx\exf@solutionstitleexp\exf@empty\else%

```

Define a label:

```

2064 \ifdefined#5%
2065 \exf@csdo\def{the\exf@solutioncounter}%
2066 {\exf@config@composeitemssolutionlabel{\exf@solprob}{\exf@solsubprob}}%
2067 \refstepcounter{\exf@solutioncounter}\label{#5}%
2068 \fi%

```

Output section line:

```

2069 \exf@section{#3}{\exf@config@styletitle\exf@config@styletitlesolution#1%
2070 \ifexf@solutionhref\exf@href{#6}{\exf@solutionstitleexp}%
2071 \else\exf@solutionstitleexp\fi}#4%
2072 \fi}

```

\exf@solutionstitle Compose the title for a solution block:

```

2073 \newcommand{\exf@solutionstitle}{\exf@solutionssection%
2074 {\exf@config@styletitlesolutions}{%
2075 \exf@config@composetitlesolutions}{\exf@config@skipsolutionstitle}%
2076 {\exf@addcontentsline{\exf@config@toclevelsolutions}%
2077 {\exf@config@composetocsolutions}}{\@undefined}{\@undefined}}

```

\exf@showssolutionsin Output solution block intro:

```

2078 \newcommand{\exf@showssolutionsin}{%

```

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

```

2079 \par\exf@config@styletext\addvspace{\exf@config@skipsolutionsabove}%
2080 \exf@config@styletextsolution%
2081 \exf@config@insertsolutionsbefore}

```

\exf@showssolutionsout Output solution block outro:

```

2082 \newcommand{\exf@showssolutionsout}{%

```

Insert hook code; close paragraph; add vertical space:

```
2083 \exf@config@insertsolutionsafter%
2084 \par\exf@config@styletext\advspace{\exf@config@skipsolutionsbelow}}
```

`\exf@showsolutions` Output solution block in buffer:

```
2085 \newcommand{\exf@showsolutions}[2]{%
```

Do nothing if buffer is empty (avoid titles):

```
2086 \ifexf@solbuf@clean\else\begin{group}%
```

Execute output solution block intro:

```
2087 \exf@showsolutionsin%
2088 \let\exf@composetitle#1%
2089 #2%
```

Source and clear buffer:

```
2090 \exf@source@buf\exf@solbuf%
2091 \exf@clear@solbuf%
```

Execute output solution block outro:

```
2092 \exf@showsolutionsout%
2093 \endgroup\fi}
```

Solutions Buffer Interface.

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

```
2094 \newcommand{\writesolutions}[1][\jobname]{%
2095 \exf@close@solfile\exf@start@solfile{#1}}
```

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

```
2096 \newcommand{\closesolutions}{\exf@close@solfile}
```

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

```
2097 \newcommand{\readsolutions}[1][\jobname]{\exf@close@solfile%
2098 \ifsolutions\begin{group}%
2099 \exf@config@styletext\exf@config@styletextsolution%
2100 \let\exf@composetitle\exf@config@composetitlesolutionmulti%
2101 \exf@solutionstitle%
2102 \input{#1\exf@config@extsolutions}%
2103 \endgroup\fi}
```

`\insertsolutions` Show solutions buffer:

```
2104 \newcommand{\insertsolutions}{\exf@showsolutions%
2105 {\exf@config@composetitlesolutionmulti}{\exf@solutionstitle}}
```

C.16 Interaction with metastr

This package transfers some functionality to the package `metastr` when loaded previously (preferably with package option `course`) by the package option `metastr`. The following changes apply:

- The terms specified by the configuration options `term...`, see section C.3, are transferred to the corresponding term registers `\metaterm{...}` in `metastr`. This can facilitate internationalisation, and suitable words for English, German, French and Spanish are provided.
- Metadata as described in section 2.3 should be filled via `\metaset` rather than `\exercisedata`. The present package will read it from the `metastr` registers.
- Basic pdf metadata is written automatically or manually as described in section 2.3. This is done via `\metawritepdfinfo`, consequently, automatic writing of these basic metadata is disabled in `metastr`.
- Sheet-specific metadata (package option `pdfdata=sheet`) is handled via the `metastr` registers `sheettitle` and `sheetdata` rather than `composemetasheet`.

Apply modifications only when package `metastr` is loaded:

```
2106 \ifdefined\metaset
```

Transfer Metadata. Import basic data from `metastr`:

```
2107 \exercisedata{author}={\metapick[] {author}}
2108 \exercisedata{title}={\metapick[] {title}}
2109 \exercisedata{subject}={\metapick[] {subject}}
2110 \exercisedata{keyword}={\metapick[] {keyword}}
2111 \exercisedata{date}={\metapick[] {date}}
```

Import course data from `metastr`:

```
2112 \ifdefined\mstr@def@course
2113 \exercisedata{course}={\metapick[] {course}}
2114 \exercisedata{instructor}={\metapick[] {instructor}}
2115 \exercisedata{institution}={\metapick[] {institution}}
2116 \exercisedata{period}={\metapick[] {period}}
2117 \exercisedata{material}={\metapick[] {material}}
2118 \fi
```

Sheet Data.

`sheettitle` Registers for sheet title and author:

`sheetauthor`

```
2119 \metadef{sheettitle}
2120 \metadef{sheetauthor}
```

`\exf@writemetadata` Write pdf metadata via `metastr` package, let `exframe` initiate process (especially for sheet):

```
2121 \metaunset[info]{writepdf}
2122 \def\exf@writemetadata{%
2123   \exf@ifis\exf@metadata{sheet}{\metaset[use]{sheettitle}{}}%
2124   \metawritepdfinfo%
2125   \exf@ifis\exf@metadata{sheet}{\metaunset[use]{sheettitle}}}
```

Fill registers:

```
2126 \metaset{sheetauthor}{\exerciseifempty{\getsheetdata{author}}%
2127   {\metapick[#1]{instructor}}{\getsheetdata{author}}}
2128 \metaset{sheettitle}{\exerciseifempty{\getsheetdata{rawtitle}}%
2129   {\metatranslate[#1]{sheet} \thesheet}%
2130   {\getsheetdata{rawtitle}}}
2131 \metaset{author}{\exerciseifempty{\getsheetdata{author}}%
```

```

2132 {\metapick[#1]{instructor}}{\metapick[#1]{sheetauthor}}}
2133 \metaset{subtitle}{%
2134 \metaif[use]{sheettitle}%
2135 {\metapick[#1]{sheettitle}}%
2136 {\metapick[#1]{material}}}

```

Terms.

term... Transfer term definitions to metastr:

```

2137 \exerciseconfig{termsheet}{\metaterm{sheet}}
2138 \exerciseconfig{termsheets}{\metaterm{sheets}}
2139 \exerciseconfig{termproblem}{\metaterm{problem}}
2140 \exerciseconfig{termproblems}{\metaterm{problems}}
2141 \exerciseconfig{termsolution}{\metaterm{solution}}
2142 \exerciseconfig{termsolutions}{\metaterm{solutions}}
2143 \exerciseconfig{termpoint}{\metaterm{point}}
2144 \exerciseconfig{termpoints}{\metaterm{points}}

```

Translations. English:

```

2145 \ifdefined\mstr@lang@en
2146 \metasetterm[en]{sheet}{Sheet}
2147 \metasetterm[en]{sheets}{Sheets}
2148 \metasetterm[en]{problem}{Problem}
2149 \metasetterm[en]{problems}{Problems}
2150 \metasetterm[en]{solution}{Solution}
2151 \metasetterm[en]{solutions}{Solutions}
2152 \metasetterm[en]{point}{point}
2153 \metasetterm[en]{points}{points}
2154 \fi

```

German:

```

2155 \ifdefined\mstr@lang@de
2156 \metasetterm[de]{sheet}{Blatt}
2157 \metasetterm[de]{sheets}{Blätter}
2158 \metasetterm[de]{problem}{Aufgabe}
2159 \metasetterm[de]{problems}{Aufgaben}
2160 \metasetterm[de]{solution}{L\''osung}
2161 \metasetterm[de]{solutions}{L\''osungen}
2162 \metasetterm[de]{point}{Punkt}
2163 \metasetterm[de]{points}{Punkte}
2164 \fi

```

French:

```

2165 \ifdefined\mstr@lang@fr
2166 \metasetterm[fr]{sheet}{Feuille}
2167 \metasetterm[fr]{sheets}{Feuilles}
2168 \metasetterm[fr]{problem}{Probl\''eme}
2169 \metasetterm[fr]{problems}{Probl\''emes}
2170 \metasetterm[fr]{solution}{Solution}
2171 \metasetterm[fr]{solutions}{Solutions}
2172 \metasetterm[fr]{point}{point}
2173 \metasetterm[fr]{points}{points}
2174 \fi

```

Spanish:

```

2175 \ifdefined\mstr@lang@es
2176 \metasetterm[es]{sheet}{Hoja}
2177 \metasetterm[es]{sheets}{Hojas}
2178 \metasetterm[es]{problem}{Problema}
2179 \metasetterm[es]{problems}{Problemas}
2180 \metasetterm[es]{solution}{Solucion}
2181 \metasetterm[es]{solutions}{Soluciones}
2182 \metasetterm[es]{point}{punto}
2183 \metasetterm[es]{points}{puntos}
2184 \fi

```

Close metastr conditional:

```

2185 \fi

```