The exesheet class and package

Antoine Missier
antoine.missier@ac-toulouse.fr

2023/01/16, v2.0

Contents

1 Introduction

The exesheet *package* is used for typesetting exercise or exam sheets. In addition, the exesheet *class* loads the schooldocs package. This one makes adjustments for margins and title and defines various layout styles with particular header and footer, appropriate for exercise sheets (among others). See the schooldocs documentation for more details. The exesheet *class* is based on article and passes to it its unknown options.

Many other packages are dedicated to exercise sheets. Most propose to encapsulate each exercise in an environment whereas exesheet begins each exercise with `\exercise`, which works like a subsection (with the same features) and is suitable for documents consisting exclusively of exercises. The package provides also alternative formatting, more relevant for short exercises.

Another specificity of the exesheet package is the particular settings for enumeration lists, useful for the numbering of questions or answers inside an exercise.

Other packages provide often more or less elaborate mechanisms to manage the placement of answers. exesheet has no such ambitions, however, for all exercises of the sheet, you can display questions only, answers only or both, but always at the place they are inserted in the source file. On the other hand this choice may be very flexible: you can do a correct version for all exercises together, or a correction per exercise, per part (subpart of exercise), per question, per sub-question.

Finally this package enables to display a detailed marking scheme in the margin, with optional explanations or remarks, and with consistency control.

Many settings can be changed and several options manage the output document. These options are based on the keyval mechanism: `key=value` (thanks to Maxime Chupin and Denis Bitouzé for their wise ideas to improve this package). The options can be passed to class or package, e.g.

```
\documentclass[a4paper,11pt,output=answers,display=pts]{exesheet}
```

or later with the `\exesheetset{`⟨*options*⟩`}` command. In the example above, `a4paper,11pt` are options passed to the underlying class article.

In the current document, a frame is used to highlight examples output.

2 Titles

2.1 The `\exercise` command

`\exercise` Each exercise begins with the `\exercise[`⟨*opt*⟩`]` command. This command typesets **Exercise**, as a document subsection, followed by automatic numbering, unique for the whole document. The optional parameter ⟨*opt*⟩ is used to put additional text on the same title line, for example to precise a subject or a marking scheme. `\exercise[(to begin)]` yields:

> **Exercise 1 (to begin)**
>
> Try to use this first command now, it's easy.

To bring additional text closer to the exercise number, we can use `\unskip` which eliminates preceding space, and also `\hrulefill` can be put in the optional

argument to produce an horizontal rule. See the following example, obtained with `\exercise[\unskip*** (difficult)]`:

Exercise 2* (difficult)**

Calculate $1 + 1$.

Several settings can be changed by refining the following commands.

`\exercisename` The word "Exercise" has automatic translation in a few languages[1] according to the loaded language (by babel or polyglossia). It can be redefined, with `\renewcommand`, or better you can use (in the preamble) macros from the translations package (allowing dynamic language change), e.g. `\DeclareTranslation{Swedish}{exesheet-exercise}{\"Ovning}}`.

`\labelexercise` This command calls `\exercisename` following by the exercise number. It can be redefined. For example, to add a period after the exercise number: `\renewcommand{\labelexercise}{\exercisename~\theexercise.}`

`\theexercise` To change only the numbering type, redefine the `\theexercise` command, based on the `exercise` counter.

`\labelexercisestyle` This macro (which is empty by default) allows to define a particular style for exercise titles. In the present document, we defined in the preamble: `\renewcommand{\labelexercisestyle}{\rmfamily\color{black}}`[2].

`\exercise*` The starred version `\exercise*[`⟨*opt*⟩`]{`⟨*label*⟩`}` allows to choose another ⟨*label*⟩ for a particular exercise and removes the numbering. For instance: `\exercise*[(Fermat's theorem)]{Problem}` yields:

Problem (Fermat's theorem)

Prove that there are no positive integers x, y, z such that $x^n + y^n = z^n$ for any integer n greater than 2.

2.2 The `\subpart` command

`\subpart` An exercise may contain several parts that we obtain with the `\subpart[`⟨*opt*⟩`]` command, typeset like a sub-subsection.

Exercise 3

Part A (preliminary)

First of all, prepare your cup of tea.

Part B

Now you are ready to make the current exercise.

The following macros manage formatting in the same way as for `\exercise`.

`\thesubpart` By default, the subpart numbering uses letters : A, B, C, etc. This numbering

[1] Translation is currently integrated into the package for the following languages: French, German, Spanish, Italian, Portuges.

[2] In the present document, to highlight real sections and subsections titles, their color and font have been modified with the `\allsectionsfont` macro from the sectsty package.

type can be redefined with the \thesubpart command based on the subpart counter, for instance \renewcommand\thesubpart{\arabic{subpart}}.

\subpartname \labelsubpart \labelsubpartstyle

The \subpart command uses \subpartname (with automatic translation in a few languages according to the selected language), \labelsubpart and \labelsubpartstyle, which can be changed.

\subpart*

Like \exercise*, the starred version \subpart*[⟨*opt*⟩]{⟨*label*⟩} allows to freely typeset the subpart {⟨*label*⟩}, for instance \subpart*{First part}.

2.3 The \annex command

\annex

The \annex[⟨*opt*⟩] command typesets the title **ANNEX**, in uppercase letters, centered and in the subsection style, with an optional parameter, added on the same line.

ANNEX (to return)

\annexname

The word "Annex" has automatic translation in a few languages. It can be extended to other languages or modified by redefining \annexname or with macros from the translations package.

\annexstyle

The annex title style is set by the \annexstyle macro, defined as follows: \newcommand\annexstyle{\MakeUppercase}. This command may be redefined as one wants.

2.4 Exercise titles in table of contents

[exetoc=⟨*bool*⟩]

By default, the titles **Exercise**, **Part** or **Annex**, appear in the table of contents (or in the pdf file summary when hyperref package is used). To avoid this, you can set the option exetoc=false (default is true). But notice that title optional arguments will always be ignored in the table of contents.

2.5 Short exercises: the \exe command

\exe

The \exe command starts an exercise by the abbreviation **Ex.** followed by the exercise number, without using sectioning commands, and the exercise body begins on the same line. An exercise starts a new paragraph without indentation.

Ex. 4 — This is a short exercise who can contain several paragraphs or questions however.

Here for example starts a new paragraph.

Ex. 5 — This is another short exercise.

\exname \exlabel \exsepmark

The abbreviation **Ex** may be changed by redefining \exname or with macros from the translations package. The \exlabel macro calls \exname following by a period then the exercise number, and \exsepmark typesets a long dash. One can change these features by redefining these commands.

\exe*

The starred version prints no separator as shown below:

Ex. 6 Another short exercise without separator.

3 Enumerations and lists

3.1 List settings

`enumerate` `\item`

Enumeration lists are intended to represent questions and sub-questions inside exercises. For a good highlight, labels are typeset in bold. Moreover, they are left aligned, at the start of the line, without indentation, and the vertical space between items is increased compared with LaTeX standard lists. These settings are done by the `\setlist` command[3] from the enumitem package of Javier Bezos. Lists with `itemize` environment are kept in their default configuration[4].

Exercise 7

1. First question
   - (a) First sub-question
   - (b) Second sub-question
2. Second question

`[setlist=⟨bool⟩]`

One can avoid enumeration list alterations and restore LaTeX default settings with the option `[setlist=false]` (default value is `true` of course).

3.2 List of exercises : the `exenumerate` environment

`exenumerate`

When an exercise sheet is made of short independent questions, it would be ill-advised to display the complete title **Exercise** for each. In addition to the `\exe` command, previously presented, we provide an even lighter solution with the `exenumerate` environment. It's only an enumeration list in which spaces between items are further increased compared to those of `enumerate`. Below is an example (the main list is an `exenumerate` environment but the sub-list is produced with common `enumerate` environments):

1. Translate the following sentences in English:
   - (a) Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi.
   - (b) Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus.
2. Translate the following sentence in German:

   Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi.
3. Translate the following sentences in French: Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus.

The environment takes an optional parameter, like `enumerate`, which enables, among others, to typeset alternative list labels, e.g. `\begin{exenumerate}[A.]`. There are many other options (see the enumitem package documentation).

[3] Labels may also be changed occasionally in an optional argument e.g. `\begin{enumerate}[A.]`, or globally thanks to `\labelenumi` and `\labelenumii` commands.

[4] The `french` option of the babel package alters `itemize` lists behavior and uses long dashes as labels for each list level. This behavior is problematic when mathematics follows the dash symbol because the latter may be confused with the minus sign. Default `itemize` lists are restored.

3.3 Items aligned by row: `tablenum1|a`, `tablitem`

`tablenum1` `tablenuma` `tablitem`

These three environments are used to typeset short questions (`tablenum1`), subquestions (`tablenuma`) or `itemize` lists (`tablitem`) on the same line. They have the same syntax: `\begin{tablenum1}[`⟨*opt*⟩`](`⟨*cols*⟩`)`. The ⟨*cols*⟩ parameter is the number of columns used by the environment. It must be *in parentheses*. This parameter can be omitted, then its value is 2. As for classic lists, each item begins with the `\item` command.

These three environments are defined by the `\NewTasksEnvironment` macro, from the tasks package by Clemens Niederberger. They take an optional argument ⟨*opt*⟩ explained in the documentation of this package, e.g. `label=\arabic*)` produces an Arabic numbering following by a closing parenthesis (like in `enumitem` package). There are also many possibilities to place items in an original way, for instance, the `\item*` command allows to specify the number of columns the item is supposed to span. In the following example, the five `\item` are placed in order between `\begin{tablenum1}(3)` and `\end{tablenum1}`. Notice that numbering is made line by line.

Exercise 8

Give the derivative of the following functions:

1. $f(x) = \dfrac{1-x^2}{\mathrm{e}^x + \mathrm{e}^{-x}}$, **2.** $g(x) = \ln\left(\dfrac{1-x}{1+x^2}\right)$, **3.** $h(x) = \displaystyle\int_0^1 \mathrm{e}^{xy}\,\mathrm{d}y$,

4. $k(x) = \displaystyle\sum_{i=1}^{\infty} \frac{1}{x^i}$, **5.** $l(x) = \displaystyle\int_{\frac{1}{x}}^{x} \frac{1}{\ln t}\,\mathrm{d}t$.

For `tablenuma`, labels are letters (a, b, c, ...) surrounded by parentheses. This cannot be modified globally, except by redefining the environment with `\RenewTasksEnvironment`. If the exesheet package is called with the option `setlist=false`, labels of `tablenum1` and `tablenuma` environments are displayed with indentation and in normal font instead of bold.

3.4 Items aligned by column: `colsenum`, `colsitem`

`colsenum`

To get numbering of items by column, we provide the `colsenum` environment: `\begin{colsenum}[`⟨*opt*⟩`]{`⟨*cols*⟩`}`. The mandatory parameter is the number of columns and the optional one will be passed to `enumerate`, allowing, for example, to change the type of numbering (a, A, etc.). To use this environment, we have to load the multicol package in the preamble. Here an example with `\begin{colsenum}{3}`:

Exercise 9

Give the derivative of the following functions:

1. $f(x) = \dfrac{1-x^2}{\mathrm{e}^x + \mathrm{e}^{-x}}$, **3.** $h(x) = \displaystyle\int_0^1 \mathrm{e}^{xy}\,\mathrm{d}y$, **5.** $l(x) = \displaystyle\int_{\frac{1}{x}}^{x} \frac{1}{\ln t}\,\mathrm{d}t$.

2. $g(x) = \ln\left(\dfrac{1-x}{1+x^2}\right)$, **4.** $k(x) = \displaystyle\sum_{i=1}^{\infty} \frac{1}{x^i}$,

colsenum*

We will notice that, on each line, items are not necessarily well aligned, which can produce inelegant effects. On the other hand, the `colsenum` environment doesn't try to align columns from the bottom by adjusting space between items. If we want to get this (which is the default option in multicol), we have the `colsenum*` environment (same syntax than `colsenum`). Here what we get in that case, with the same exercise:

Exercise 10

Give the derivative of the following functions:

1. $f(x) = \dfrac{1-x^2}{\mathrm{e}^x + \mathrm{e}^{-x}}$,

2. $g(x) = \ln\left(\dfrac{1-x}{1+x^2}\right)$,

3. $h(x) = \displaystyle\int_0^1 \mathrm{e}^{xy}\,\mathrm{d}y$,

4. $k(x) = \displaystyle\sum_{i=1}^{\infty} \frac{1}{x^i}$,

5. $l(x) = \displaystyle\int_{\frac{1}{x}}^{x} \frac{1}{\ln t}\,\mathrm{d}t$.

We can see that these alignments are less good than those obtained by row numbering. The column numbering may nevertheless be preferable when there are many items with variable heights, and a number of items which can be different from one column to the other. Moreover, an advantage of `colsenum` is that the choice of labels is automatic depending on the list level (and the language), unlike `tablenum1` or `tablenuma`.

colsitem
colsitem*

For `itemize` lists, the environment `colsitem` produces items aligned by column rather than by lines as for `tablitem`: `\begin{colsitem}[`⟨*opt*⟩`]{`⟨*cols*⟩`}`. The optional parameter, which is passed to the underlying `itemize` environment, allows to change the item label (bullet by default). And, as for `colsenum*`, the `colsitem*` environment produces an alignment of columns from the bottom.

4 Questions and solutions

4.1 Environments `questions` and `answers`

questions
answers

The package provides the two environments `questions` and `answers` to make optionally appear or disappear questions and answers of exercises.

[output=⟨*opt*⟩]

The output is controlled by the `output` key option: it takes three recognized values: `questions`, `answers`, `both`. The `questions` value allows to display questions without answers, `answers` displays answers without questions, `both` (the default option) displays both.

\correctionstyle
correctioncolor
\correctionname

In the case (by default) where questions and answers are displayed both, answers are then typeset in the style `\correctionstyle`, which uses the color `correctioncolor`. This color may be changed with the `\definecolor` macro[5] (by default `\definecolor{correctioncolor}{rgb}{0,0.2,0.6}` = kind of dark blue). Moreover, with `output=both` the title **Correction** is displayed at the beginning of `answers` environments. It is defined by the `\correctionname` macro

[5] from the xcolor package by Uwe Kern which is loaded by exesheet.

(with translation in a few languages and that can changed: for example you may prefer "Solution" than "Correction"). The style defined by `\correctionstyle` will apply to the title as for the whole environment. Here an example.

Exercise 11

1. Is the exesheet package useful ?

2. Isn't there any other packages that deal with exercises ?

Correction

1. Yes, the exesheet package is useful for teachers.

2. There are many other packages that deal with exercises, and give the ability to produce separately questions and solutions, for example exercise by Paul Pichaureau, exercises by Roger Jud, exsheets (superseded by xsim) by Clemens Niederberger, exframe by Niklas Beisert, exam by Philip Hirschhorn, answers by Mike Piff and Joseph Wright, probsoln by Nicola Talbot, etc.

When only answers are displayed, the text color remains black and the word "Correction" is not displayed.

4.2 About the title "Correction" in `answers` environments

Internally, we have used the `\comment` and `\endcomment` macros from the versions package by Uwe Lück. Other excellent packages allow to manage selectively piece of code. Let us mention verbatim by Rainer Schöpf, comment by Victor Eijkhout, version by Donald Arseneau and Stephen Bellantoni, optional by Donald Arseneau and codesection by Matthias Pospiech.

The versions package provides furthermore the `\includeversion{`⟨*env*⟩`}` and `\excludeversion{`⟨*env*⟩`}` macros who allow to make appear or disappear any environment ⟨*env*⟩ and these optional environments may be nested[6].

However the `questions` and `answers` environments perform another task, not only making appear or disappear piece of text. In what format should the output yield the title **Correction** and at which level should it be put in the table of contents (or in the summary of the pdf file)? In fact it depends on which level the environment has been nested. We can make a single `answers` environment for the whole sheet or an `answers` environment for each exercise, for each exercise part, for each question or sub-question. In fact, the typeset of the title **Correction** and his level in the table of contents will be computed by the environment.

`answers[`⟨*level*⟩`]`

Nevertheless, one can imagine twisted situations in which the title level will not be correct. Moreover, the user may want to change the level of the title. It is then possible to force the level of the title "Correction" with `\begin{answers}[`⟨*level*⟩`]`. The optional ⟨*level*⟩ parameter is defined as follows: 1 for section level titles, 2 for subsections (like **Exercise**), 3 for sub-subsections (like **Part**), another number for lower levels (that will not appear in the table of contents).

[6] The codesection package also allows such nesting, including in the preamble, as well as the optional package, but the latter manages only short optional code.

answers* The starred version `answers*` makes the title **Correction** completely disappear.

4.3 Commands \question, \answer and \answerspace

\question
\answer
Instead of `questions` and `answers` environments, we can also use simple `\question{`⟨*ques*⟩`}` and `\answer{`⟨*ans*⟩`}` macros in which display of ⟨*ques*⟩ and ⟨*ans*⟩ arguments are controlled by the same previous key option `output=`⟨*opt*⟩. It may be more suitable when questions and answers are short, for instance if you want to display the answer after each question item. The title "Correction" will not appear on the beginning of each answer; answers are typeset with `\correctionstyle` if `output=both`. But these commands don't work with `verbatim` text inside, whereas `questions` and `answers` environments allow that.

\answerspace
Some teachers are used to give to their students documents in which questions are typeset but instead of answers you have blank space, so that the paper is intended to be fulfilled by the students. For this purpose, thanks to a suggestion of Maxime Chupin, we provide the macro `\answerspace[`⟨*height*⟩`]`, in which the parameter height is a valid length, e.g. `\answerspace{3cm}`.

[answerspace=⟨*bool*⟩]
This vertical space display is optional and is controlled by the key `answerspace` which can be `true` or `false` (by default). Of course this macro is not intended to be placed inside `answers` environments, but anywhere else.

5 Marginal notes for marking scheme

The exesheet package allows to display in margins a marking scheme with comments and explanations about answers.

5.1 The \points command

\points
The `\points{`⟨*pts*⟩`}` command displays the number of points awarded for an exercise. It is intended to be entered in the optional argument of the `\exercise` command[7]. For instance `\exercise[\points{5}]` yields:

> **Exercise 12** 5 points
>
> Try to read this document to the end without drinking tea and you get five points.

When only correction is displayed in an exercise, the `\points` macro is patched to suppress the display of points. An extended solution for printing answers with the scale will be presented in section 5.5, including a new `\totalpoints` macro.

\pointsname
\pointname
The word "points" (or in the singular "point" if ⟨*pts*⟩ is inferior to 2), is added and, like before, gets automatic translation in a few languages (and can be changed).

\pointsstyle
pointscolor
The style used by the `\points` command may be redefined with `\pointsstyle`. The color is set by `pointscolor` with the `\definecolor` command, and you can change it, e.g.: `\definecolor{pointscolor}{named}{blue}`.

[7] Using this command in the optional argument of `\exercise` is incompatible with the memoir class which redefines section commands.

5.2 The \pts command

\pts When exercises are typeset with the \exe macro, or as a list with the exenumerate environment, the marking scheme can be displayed in the margin, on the line where we put the \pts{⟨num⟩} command (in general the first line of the exercise). The ⟨num⟩ parameter is the number of points assigned to the exercise. Below what we obtain with with \exe\pts{3}... \exe\pts{1.5}...

(3 pts) **Ex. 13** — The first exercise with a marking scheme.

(1.5 pt) **Ex. 14** — The second one.

\ptsname \ptname ptscolor \ptsstyle The abbreviation "pts" (or "pt" when the number of points is inferior to 2) is automatically added with macros \ptsname or \ptname (translated in a few languages if babel ore polyglossia are loaded). The display color of the points is defined by ptscolor, which can be changed with \definecolor: by default \definecolor{ptscolor}{named}{red}. The display style is defined by \ptsstyle: among other things, it adds parenthesis around.

[display=⟨opt⟩] The display of the marking scheme is controlled by the option key display. By default display=none and the marking scheme will not be displayed. Use display=pts to make it appear. More details about this key are given in 5.4.

[marginpos=⟨opt⟩] The side where to place the scale is controlled by the marginpos key option whose possible values are (first) left or right. The default is left (although the default behavior of LaTeX is to place marginal notes on the right side). This option has no effect when display=none.

For a *two side* document, the default behavior is to typeset text in the outer margin which will be enlarged relatively to the inner margin (that contains the binding). The outer margin is on the right hand side for odd pages and on the left hand side for even pages. So the marginpos key option can take those two additional values. When you give a left or right value with a two side document, it will be converted to default value, outer, with a warning message.

twoside mode For a document is in twoside mode, marginal notes sometimes appear on the wrong side of a page. This is a known bug of LaTeX for which the solution is to call the package mparhack (what is done by exesheet for a twoside mode document) and to *run LaTeX twice*. If necessary you get a warning message to rerun.

5.3 Commands \totalexe, \note* and \note

To give a more detailed marking scheme, we get the following commands.

\totalexe \totalexe{⟨num⟩} displays the total number of points of an exercise, by default inside an oval box, with the word "pts" (or "pt") added and in bold red. In the next example, the exercise title has been obtained with \exercise[\totalexe{4}].

\note* For each answer or solution in the correct version, the command \note*{⟨num⟩} states the number of points of that question. The appearance is somewhat different from the one obtained with \pts: by default the number is displayed without being followed by "pts" or "pt", without parenthesis, and in bold. In the answer 3 of the next example, just after \item, we used: \note*{1.5}.

`\note` The `\note{`⟨*comment*⟩`}` macro is used to give details about the marking scheme and to indicate how points are awarded. In a comment argument you can use `\\` to get a line break, or even `\\[`⟨*height*⟩`]` to increase the line break from ⟨*height*⟩.

`\note[`⟨*num*⟩`]` It is often convenient to put `\note*{`⟨*num*⟩`}\note{`⟨*comment*⟩`}` at the beginning of an answer. In that case LaTeX will place the margin notes one under the other and will protest with: `LaTeX Warning: Marginpar on page ... moved.` This is not a real problem because generally LaTeX is able to manage the placement of these marginal notes, one below the other. Nevertheless, one can avoid unnecessary protests by compacting both commands in a single one and typing the number of points as an optional argument of note: `\note[`⟨*num*⟩`]{`⟨*comment*⟩`}`.

The first comment on the following example is obtained (just after `\item`) with `\note[1]{0.5 for the anti-derivative\\0.5 for simplifying}`.

4 pts

Exercise 15

For each following question, say if the assertion is true or false. Justify the answer carefully.

1. $\displaystyle\int_0^{\sqrt{3}} \frac{1}{x+\sqrt{3}}\,\mathrm{d}x = \ln 2,$ **2.** $\displaystyle\int_2^{\mathrm{e}} \frac{1}{x\ln x}\,\mathrm{d}x = -\ln 2,$

3. The function F defined on $\mathbf{R}$ by $F(x) = \displaystyle\int_0^x \frac{1}{t^2+t+1}\,\mathrm{d}t$ is increasing on $\mathbf{R}$.

Correction

1
0.5 for the anti-derivative
0.5 for simplifying

1. We calculate:

$$\int_0^{\sqrt{3}} \frac{1}{x+\sqrt{3}}\,\mathrm{d}x = \Big[\ln\left(x+\sqrt{3}\right)\Big]_0^{\sqrt{3}} = \ln\left(2\sqrt{3}\right) - \ln\sqrt{3} = \ln\left(\frac{2\sqrt{3}}{\sqrt{3}}\right) = \ln 2.$$

TRUE.

1.5
1 for the anti-derivative
0.5 for final value

other method: $\frac{1}{x\ln x} > 0$ on $[2,\mathrm{e}]$ whereas $-\ln 2 < 0$

2. We have $\displaystyle\frac{1}{x\ln x} = \frac{\frac{1}{x}}{\ln x} = \frac{u'(x)}{u(x)}$ with $u(x) = \ln x$, which is positive on $[2,\mathrm{e}]$. Hence

$$\int_2^{\mathrm{e}} \frac{1}{x\ln x}\,\mathrm{d}x = \Big[\ln(\ln x)\Big]_2^{\mathrm{e}} = \ln(\ln \mathrm{e}) - \ln(\ln 2) = \ln 1 - \ln(\ln 2) = -\ln(\ln 2).$$

FALSE.

1.5

3. The function F defined on $\mathbf{R}$ by

$$F(x) = \int_0^x \frac{1}{t^2+t+1}\,\mathrm{d}t$$

0.5 for F'
1 for the sign of F' and conclusion

is derivable on $\mathbf{R}$ and its derivative is such that $F'(x) = \frac{1}{x^2+x+1}$. The denominator is a quadratic polynomial, always positive because its discriminant is $\Delta = -3 < 0$. Thus F is increasing on $\mathbf{R}$.
TRUE.

In the comment of answer 2, a wider vertical space is produced at line break with the optional argument `\\[2ex]`. The last comment, not placed beside the number of points of answer 3, has been produced by placing on the first line after the formula: `\note{0.5 for $F'$\\1 for the sign of $F'$ and conclusion}`.

`markingcolor` `\markingstyle` `\ptsboxlength`

The points display color, in `\totalexe` and `\note*`, is defined by `markingcolor` and the style by `\markingstyle`, which are modifiable. The oval box produced by `\totalexe` is obtained with the `\ovalbox` command of the fancybox package (by Timothy Van Zandt), with corner arcs set by `\cornersize{1}`. The length of the box is not adjusted to content but depends on the value of `\ptsboxlength` in order to keep uniform appearance from one exercise to the other.

`notecolor` `\notestyle`

Comment notes are typeset by default in a dark green color defined by `\definecolor{notecolor}{rgb}{0.0,0.4,0.0}`. The style of the comment is set by the `\notestyle` macro.

5.4 Margin notes options

`[display=`⟨*opt*⟩`]`

The `display` key option controls how the marking scheme will be displayed: as seen previously (subsection 5.2), `display=none` displays nothing. With `display=pts` the numbers passed as argument to `\pts`, `\totalexe`, `\note*` or as optional argument of `\note[`⟨*num*⟩`]{...}` will be displayed. The last possible value is `display=notes` which displays the full marginal notes with points and comments (the mandatory argument of `\note`) as seen in the previous example.

`[marginpos=`⟨*opt*⟩`]`

As seen previously in subsection 5.2, the side where to place the scale is controlled by the `marginpos` key option whose possible values are `left` and `right` (or `inner` and `outer` if the document is in two side mode).

`[marginwidth=`⟨*opt*⟩`]`

The key option `marginwidth` controls margins layout. The possible values are `standard`, `expand` or `unset`.

This option has no effect when `display=none`. In that case, left and right margins have same width, except for a two side mode document for which the ratio between left and right margin is 2:3. Otherwise the key `marginwidth` acts as follow:

`standard` The left margin is enlarged and the right one is reduced, with a ratio of 3:2 (or 2:3 if `marginpos=left`). The text body is shifted without altering the text width. The margin paragraph width remains relatively short; it depends also on page geometry. This option is not suitable for verbose comments.

`expand` It is the default value. The behavior is the same than with the `standard` value if `display=pts`, but if `display=notes` the margin is expanded with a ratio of 3:1 (or 1:3) and margin paragraph width is increased.

`unset` The previous settings may not suit to anyone, so you have this other option. In that case, no setting will be proceeded on the margin width and you can make your own setting. For that, you have the convenient macro `\geometry` from the geometry package (by Hideo Umeki). For example you can put in the preamble

`\geometry{hmarginratio=2:1,marginparwidth=2.5cm}`.

If `marginpos=right`, you must invert the ratio, e.g. 1:2 instead of 2:1. If `marginwidth` is not set to `unset`, such a command will have no effect.

The margin settings are valid for the whole document and must be set in the preamble.

`[noteragged=⟨opt⟩]` For the mandatory argument of `\note`, text alignment in margins is controlled by the package option `noteragged`, which can take the following values: `left`, `right`, `center`, `justify` or `twoside`. The default value is `noteragged=left`. It means that the text is right aligned, which is a common behavior for text in the left margin; `noteragged=right` yields a left aligned text; `justify` means that the text is justified, which is the default setting of LaTeX marginal notes. Finally `noteragged=twoside` is equivalent to `noteragged=left` for odd pages and `noteragged=right` for even pages, if the document is in two side mode. It has no effect otherwise (the default value `noteragged=left` will be taken and a warning message appears in the shell).

When `display` is not set to `notes`, the `noteragged` option has no effect because it concerns only text put in mandatory argument of `\note`.

5.5 The `\totalpoints` command

`\totalpoints` The `\totalpoints{⟨num⟩}` macro is intended to replace `\points` when using a detailed marking scheme. When the scale is not displayed, it is equivalent to `\points` and when the scale is displayed, it is equivalent to `\totalexe`. For example, in the exercise 15, we should use `\totalpoints` rather than `\totalexe`, because, when the detailed marking scheme is not displayed, the total points will be typeset as in the exercise 5.1 rather than in the margin.

5.6 Marking scheme consistency checking

`[checkpts=⟨bool⟩]` The marking scheme can be checked[8] with the keyval option `[checkpts=true]`, the default value is `false`.

For each exercise, the sum of points awarded for each question (with `\pts`, `\note*` or `\note[ ]` is compared to the total of the exercise given in `\points`, `\totalexe` or `totalpoints`. A warning message appear in the shell to indicate if the scale is valid or not for the exercise. For example:

```
Package exesheet warning:  Exercise 3:  sum of points is 4.5pt
                           instead of 5pt.
```

Depending on your language, you can use comma notation numbers (4,5 as well as 4.5). The checking takes place at the beginning of the next exercise. If no points are specified for the questions, no warning message will be displayed at this level.

`\totalsheet` At the end of the sheet, the last exercise is checked, then a global checking is made on the whole sheet. For that, the total points of the sheet must be specified in the preamble with the `\totalsheet{⟨total⟩}` macro (else you get a warning message). If subtotal points have been awarded for exercises, the global comparison is between the sum of these subtotals and the total of points recorded with the `\totalsheet` macro. If not, the audit covers the sum of points awarded for each individual question. A warning message indicates the result of this last checking. Finally a last message indicate if all the scale controls have been successfully passed or not.

[8] Thanks to Denis Bitouzé for his suggestion about this feature.

6 Options

6.1 Summary of available options

Here we present a summary table of available options. Explanations of use are given in the corresponding sections. The default value is typeset in bold.

Key	Possible values	See section
exetoc	**true**, false	2.4
setlist	**true**, false	3.1
output	questions, answers, **both**	4.1
answerspace	true, **false**	4.3
display	**none**, pts, notes	5.2, 5.4
marginpos	**left** (inner), right (**outer**)	5.2, 5.4
marginwidth	standard, **expand**, unset	5.4
noteragged	**left**, right, center, justify, twoside	5.4
checkpts	true, **false**	5.6
correct	true, **false**, conditional	see below

When an invalid key is passed, an error is produced, but when a value is not recognized, a warning message occur:
`Value '⟨value⟩' is not supported by option '⟨key⟩' on input line ...`
For each option, you can set them with class or package calling: e.g.
`\usepackage[output=answers,display=notes,noteragged=right]{exesheet}`

`\exesheetset` You can also use the `\exesheetset`{list of ⟨*key*⟩=⟨*value*⟩ options} command. Nevertheless, only the options `output`, `answerspace`, `display`, and `noteragged`, can be changed dynamically, even in the document body, although this is not really expected. The others are usable in the preamble only. Dynamic options are processed at each call, the others will be processed once, at begin document.

`[correct=`⟨*opt*⟩`]` A special option, `correct`, can be used only when calling the exesheet class or in combination with the schooldocs package. Its effect is to add "Correct version" (or its translation) in the document title and headers. Possible values are: `true`, `false` (by default) or `conditional`. The value `correct=conditional`, means `true` if answers are displayed, `false` otherwise.

6.2 Alternative (deprecated) commands

Previously to version 2.0, we used some special commands to set output and display options. Thanks to a suggestion of Maxime Chupin, keyval options have been implemented in the package. Although the latter is more convenient, the old commands will be presented here and maintained for now, for compatibility reasons. When using them, a warning message will appear, but these commands still work. However previous options `nosetlist` and `notoc` are no longer supported.

`\questionsonly` `\answersonly` These two commands are equivalent to respectively `output=questions` and `output=answers`.

`\displaypts` These two commands are now equivalent to `display=pts`.

`\displaypoints` `\displaynotes` `\displaynotesright` These two commands mean `display=notes` and moreover `marginpos=right` for the latter.

7 Implementation

7.1 Options and required packages

The `exesheet` class is based on the `article` class and passes to it all its unknown options. `\ProcessKeyvalOptions*` is useless in class, will be done by the package.

```
⟨*class⟩
\RequirePackage{kvoptions}
\DeclareBoolOption[true]{exetoc}
\DeclareBoolOption[true]{setlist}
\DeclareStringOption[both]{output}
\DeclareStringOption[none]{display}
\DeclareBoolOption[false]{answerspace}
\DeclareStringOption[left]{marginpos}
\DeclareStringOption[expand]{marginwidth}
\DeclareStringOption[left]{noteragged}
\DeclareBoolOption[false]{checkpts}
\DeclareStringOption[false]{correct}
\DeclareOption*{\PassOptionsToClass{\CurrentOption}{article}}
\ProcessOptions \relax
\LoadClass{article}
\RequirePackage{exesheet}
\RequirePackage{schooldocs}
⟨/class⟩
```

Then options are defined thanks to kvoptions package (based on keyval). String options are processed in separate macros defined in the corresponding sections of each. These process macros will be executed when package is loaded (at end of package because `\exs@process...` aren't recognized at the beginning), or at begin document for options whose effect cannot be changed dynamically and must be set in the preamble (they will be processed once).

A special case is `setlist` when used with babel-french. It will then be processed immediately and disabled (see below for explanations).

```
⟨*package⟩
\@ifclassloaded{exesheet}{}{
    \RequirePackage{kvoptions}
    \DeclareBoolOption[true]{exetoc}
    \DeclareBoolOption[true]{setlist}
    \DeclareStringOption[both]{output}
    \DeclareStringOption[none]{display}
    \DeclareBoolOption[false]{answerspace}
    \DeclareStringOption[left]{marginpos}
    \DeclareStringOption[expand]{marginwidth}
    \DeclareStringOption[left]{noteragged}
    \DeclareBoolOption[false]{checkpts}
    \DeclareStringOption[false]{correct}
}

\ProcessKeyvalOptions*

\def\exs@process@dynoptions{
    \exs@process@output
    \exs@process@display
    \exs@process@noteragged
```

```
}

\AtEndOfPackage{\exs@process@dynoptions}
\AtBeginDocument{
    \newif\ifexesheet@multicol
    \@ifpackageloaded{multicol}{
        \exesheet@multicoltrue}{\exesheet@multicolfalse}
        % to set the rule to the right color in answers environments
    \exs@process@setlist
    \exs@process@marginpos
    \exs@process@marginwidth
    \exs@process@checkpts
    \exs@process@correct
    \DisableKeyvalOption[action=warning,package=exesheet]{exesheet}{setlist}
    \DisableKeyvalOption[action=warning,package=exesheet]{exesheet}{marginpos}
    \DisableKeyvalOption[action=warning,package=exesheet]{exesheet}{marginwidth}
    \DisableKeyvalOption[action=warning,package=exesheet]{exesheet}{checkpts}
    \DisableKeyvalOption[action=warning,package=exesheet]{exesheet}{correct}
}

```

`\exesheetset` The macro `\exesheetset` can receive keyval options. It can be used anywhere in the document to modify some settings, but has no effect on non dynamic options if called outside the preamble; thanks to `\DisableKeyValOption` a warning message occur in that case.

```
\def\exesheetset#1{\setkeys{exesheet}{#1}\exs@process@dynoptions}

```

Now we load several packages. The `shortlabel` option in the enumitem package allows to use labels like in the enumerate package e.g. 1., a), A. etc. The mparhack package (by Tom Sgouros and Stefan Ulrich) is loaded only for two side mode documents.

```
\RequirePackage{ifthen}
\RequirePackage{geometry}
\RequirePackage{xcolor}
\RequirePackage[shortlabels]{enumitem}
\RequirePackage{tasks}
\RequirePackage{versions}
\RequirePackage{fancybox}
\RequirePackage{translations}
\RequirePackage{ragged2e}
\ifthenelse{\boolean{@twoside}}{\RequirePackage{mparhack}}{}

```

7.2 Internationalization

Here we define keywords and their translation in French, German, Spanish Italian, Portuguese, thanks to macros of the translations package by Clemens Niederberger. It detects the used language loaded by babel or polyglossia.

Accented characters cannot be used here because they are not recognized if inputenc is loaded after exesheet. So we have used basic LaTeX control sequences to produce them.

```
\DeclareTranslationFallback{exesheet-exercise}{Exercise}
\DeclareTranslationFallback{exesheet-subpart}{Part}
\DeclareTranslationFallback{exesheet-annex}{Annex}
\DeclareTranslationFallback{exesheet-ex}{Ex}
\DeclareTranslationFallback{exesheet-points}{points}
\DeclareTranslationFallback{exesheet-point}{point}
\DeclareTranslationFallback{exesheet-correction}{Correction}
\DeclareTranslationFallback{exesheet-pts}{pts}
\DeclareTranslationFallback{exesheet-pt}{pt}

\DeclareTranslation{English}{exesheet-exercise}{Exercise}
\DeclareTranslation{English}{exesheet-subpart}{Part}
\DeclareTranslation{English}{exesheet-annex}{Annex}
\DeclareTranslation{English}{exesheet-ex}{Ex}
\DeclareTranslation{English}{exesheet-points}{points}
\DeclareTranslation{English}{exesheet-point}{point}
\DeclareTranslation{English}{exesheet-correction}{Correction}
\DeclareTranslation{English}{exesheet-pts}{pts}
\DeclareTranslation{English}{exesheet-pt}{pt}

\DeclareTranslation{French}{exesheet-exercise}{Exercice}
\DeclareTranslation{French}{exesheet-subpart}{Partie}
\DeclareTranslation{French}{exesheet-annex}{Annexe}
\DeclareTranslation{French}{exesheet-ex}{Ex}
\DeclareTranslation{French}{exesheet-points}{points}
\DeclareTranslation{French}{exesheet-point}{point}
\DeclareTranslation{French}{exesheet-correction}{Correction}
\DeclareTranslation{French}{exesheet-pts}{pts}
\DeclareTranslation{French}{exesheet-pt}{pt}

\DeclareTranslation{German}{exesheet-exercise}{\"Ubung}
\DeclareTranslation{German}{exesheet-subpart}{Teil}
\DeclareTranslation{German}{exesheet-annex}{Anhang}
\DeclareTranslation{German}{exesheet-ex}{\"Ub}
\DeclareTranslation{German}{exesheet-points}{Punkte}
\DeclareTranslation{German}{exesheet-point}{Punkt}
\DeclareTranslation{German}{exesheet-correction}{Verbesserung}
\DeclareTranslation{German}{exesheet-pts}{P.}
\DeclareTranslation{German}{exesheet-pt}{P.}

\DeclareTranslation{Spanish}{exesheet-exercise}{Ejercicio}
\DeclareTranslation{Spanish}{exesheet-subpart}{Parte}
\DeclareTranslation{Spanish}{exesheet-annex}{Anexo}
\DeclareTranslation{Spanish}{exesheet-ex}{Ej}
\DeclareTranslation{Spanish}{exesheet-points}{puntos}
\DeclareTranslation{Spanish}{exesheet-point}{punto}
\DeclareTranslation{Spanish}{exesheet-correction}{Correcci\'on}
\DeclareTranslation{Spanish}{exesheet-pts}{ptos}
\DeclareTranslation{Spanish}{exesheet-pt}{pto}

\DeclareTranslation{Italian}{exesheet-exercise}{Esercizio}
\DeclareTranslation{Italian}{exesheet-subpart}{Parte}
\DeclareTranslation{Italian}{exesheet-annex}{Annesso}
\DeclareTranslation{Italian}{exesheet-ex}{Es}
```

```
\DeclareTranslation{Italian}{exesheet-points}{punti}
\DeclareTranslation{Italian}{exesheet-point}{punto}
\DeclareTranslation{Italian}{exesheet-correction}{correzione}
\DeclareTranslation{Italian}{exesheet-pts}{pti}
\DeclareTranslation{Italian}{exesheet-pt}{pt}

\DeclareTranslation{Portuges}{exesheet-exercise}{Exerc\'icio}
\DeclareTranslation{Portuges}{exesheet-subpart}{Parte}
\DeclareTranslation{Portuges}{exesheet-annex}{Anexo}
\DeclareTranslation{Portuges}{exesheet-ex}{Ex}
\DeclareTranslation{Portuges}{exesheet-points}{pontos}
\DeclareTranslation{Portuges}{exesheet-point}{ponto}
\DeclareTranslation{Portuges}{exesheet-correction}{Corre\c c\~ao}
\DeclareTranslation{Portuges}{exesheet-pts}{pts}
\DeclareTranslation{Portuges}{exesheet-pt}{pt}

\newcommand*\exercisename{\GetTranslation{exesheet-exercise}}
\newcommand*\subpartname{\GetTranslation{exesheet-subpart}}
\newcommand*\annexname{\GetTranslation{exesheet-annex}}
\newcommand*\exname{\GetTranslation{exesheet-ex}}
\newcommand*\pointsname{\GetTranslation{exesheet-points}}
\newcommand*\pointname{\GetTranslation{exesheet-point}}
\newcommand*\correctionname{\GetTranslation{exesheet-correction}}
\newcommand*\ptsname{\GetTranslation{exesheet-pts}}
\newcommand*\ptname{\GetTranslation{exesheet-pt}}

```

7.3 Titles

The `exercise` counter numbers exercises for the whole document regardless of any section. To reset the counter at some point, just write `\setcounter{exercise}{0}` and for an automatic reset at each section, add in the preamble `\makeatletter \@addtoreset{exercise}{section} \makeatother`.

The parts counter depends on `exercise` and is reset at each new exercise.

Commands `\labelexercisestyle` and `\labelsubpartstyle` are empty, but allow to personalize the style, for instance: `\renewcommand\labelexercisestyle{\sffamily}`.

The `\exe@check` macro checks the marking scheme, it will be defined in section 7.6. By default, the table of contents displays the titles of exercises and parts, `\ifexesheet@exetoc` is `true`. To display in it only exercise titles but not parts, place in the preamble `\setcounter{tocdepth}{2}`.

`\exercise`

```
\newcounter{exercise}

\newcommand{\labelexercise}{\exercisename\space \theexercise}
\newcommand{\labelexercisestyle}{}
\newcommand*{\@exercise}[1][]{%
    \ifexesheet@checkpts \exe@check{\labelexercise} \fi
    % curiously the \execheck must be done before \refstepcounter !
    \refstepcounter{exercise}
    \subsection*{\labelexercisestyle\labelexercise\enskip #1}
    \ifexesheet@exetoc
```

```
        \addcontentsline{toc}{subsection}{\labelexercise}
    \fi
}
\newcommand*{\@@exercise}[2][]{%
    \ifexesheet@checkpts \exe@check{#2} \fi
    \subsection*{\labelexercisestyle #2\enskip #1}
    \setcounter{subpart}{0} % resets the parts counter
    \ifexesheet@exetoc
        \addcontentsline{toc}{subsection}{#2}
    \fi
}
\newcommand{\exercise}{\@ifstar{\@@exercise}{\@exercise}}
```

\subpart

```
\newcounter{subpart}[exercise] %
\renewcommand{\thesubpart}{\Alph{subpart}}

\newcommand{\labelsubpart}{\subpartname~\thesubpart}
\newcommand{\labelsubpartstyle}{}
\newcommand*{\@subpart}[1][]{%
    \refstepcounter{subpart}%
    \subsubsection*{\labelsubpartstyle\labelsubpart\enskip #1}
    \ifexesheet@exetoc
        \addcontentsline{toc}{subsubsection}{\labelsubpart}
    \fi
}
\newcommand*{\@@subpart}[2][]{%
    \subsubsection*{\labelsubpartstyle #2\enskip #1}
    \ifexesheet@exetoc
        \addcontentsline{toc}{subsubsection}{#2}
    \fi
}
\newcommand{\subpart}{\@ifstar{\@@subpart}{\@subpart}}
```

\annex

```
\newcommand{\annexstyle}{\MakeUppercase}
\newcommand*{\annex}[1][]{%
    \subsection*{\mbox{}\hfill\annexstyle{\annexname} #1\hfill\mbox{}}
    \ifexesheet@exetoc
        \addcontentsline{toc}{subsection}{\annexname}
    \fi
}
```

\exe

```
\newcommand{\exlabel}{\exname.~\theexercise}
\newcommand{\exsepmark}{---}
\newcommand{\@exe}{\bigskip\refstepcounter{exercise}
    \par\noindent\textbf{\exlabel~\exsepmark}~}
\newcommand{\@@exe}{\bigskip\refstepcounter{exercise}
    \par\noindent\textbf{\exlabel}~}
\newcommand{\exe}{\@ifstar{\@@exe}{\@exe}}
```

7.4 Enumerations and lists

\exenumerate

The \setlist command comes from the enumitem package (\setenumerate is obsolete). By default itemsep=1ex for lists of first level, and leftmargin=1.5em allows to align labels on the start of lines.

```
\newenvironment{exenumerate}[1][]{%
    \setlist[enumerate]{font=\bfseries}
    \setlist[enumerate,1]{leftmargin=1.5em,
        itemsep=3ex plus 1ex minus 1ex,topsep=3ex plus 1ex minus 1ex}
    \setlist[enumerate,3]{noitemsep,nolistsep}
    \setlist[itemize]{noitemsep,nolistsep}
    \begin{enumerate}[#1]
        }{\end{enumerate}}

```

When using babel with the option french, itemize lists are modified with the same dash label for each list level. These modifications are canceled here to restore default LaTeX itemize lists (labels and spaces). We have create the \standardfrenchlists command which must be called into AtBeginDocument or not, depending on whether exesheet is loaded before babel or after.

```
\newcommand\standardfrenchlists{%
    \@ifpackagewith{babel}{french}{
        \frenchsetup{StandardLists=true}
    }{}
}
\ifexesheet@setlist
    \standardfrenchlists % necessary when exesheet is loaded after babel
    \DisableKeyvalOption[action=warning,package=exesheet]{exesheet}{setlist}
\fi

\def\exs@process@setlist{% must be executed at begin document
  \ifexesheet@setlist
    \standardfrenchlists % if exesheet is loaded before babel package
    \setlist[enumerate]{font=\bfseries}
    \setlist[enumerate,1]{topsep=1.5ex plus 1ex minus 1ex,leftmargin=1.5em}
  \fi
```

tablenum1
tablenuma

The \NewTasks command comes from the tasks package. It allows to define environments tablenum1, tablenuma and tablitem. Horizontal spaces are adjusted to get good alignments with items of other enumerate (or itemize) environments.

```
  \ifexesheet@setlist
    \NewTasksEnvironment[label=\arabic*.,label-format=\bfseries,
        column-sep=1em,label-align=right,
        item-indent=1.5em,label-width=1em,label-offset=0.5em,
        after-item-skip=0.5ex plus 0.5ex minus 0.5ex]{tablenum1}[\item](2)
    \NewTasksEnvironment[label=(\alph*),label-format=\bfseries,
        column-sep=1em,label-align=right,
        item-indent=2.15em,label-width=1.6em,label-offset=0.5em,
        after-item-skip=0.5ex plus 0.5ex minus 0.5ex]{tablenuma}[\item](2)
  \else
    \NewTasksEnvironment[label=\arabic*.,
        column-sep=1em,
        after-item-skip=0.5ex plus 0.5ex minus 0.5ex]{tablenum1}[\item](2)
    \NewTasksEnvironment[label=(\alph*),
```

```
        column-sep=1em,label-align=right,
        item-indent=2.15em,label-width=1.6em,label-offset=0.5em,
        after-item-skip=0.5ex plus 0.5ex minus 0.5ex]{tablenuma}[\item](2)
  \fi
} % end of macro \exs@process@setlist

```

tablitem

```
\NewTasksEnvironment[label=\labelitemi,
    label-align=right,
    item-indent=2.3333em,label-offset=0.5em,
    after-item-skip=0.5ex plus 0.5ex minus 0.5ex]{tablitem}[\item](2)

```

colsenum

```
\newenvironment{colsenum}[2][]{%
    \setlength{\multicolsep}{2ex}
    \raggedcolumns % default is \flushcolumns
    \begin{multicols}{#2} % #2 = number of columns
    \begin{enumerate}[#1] % #1 = options of enumerate
    }{
    \end{enumerate}
    \end{multicols}
}

```

colsenum*

```
\newenvironment{colsenum*}[2][]{%
    \setlength{\multicolsep}{2ex}
    \begin{multicols}{#2} % #2 = number of columns
    \begin{enumerate}[#1] % #1 = options of enumerate
    }{
    \end{enumerate}
    \end{multicols}
}

```

colsitem

```
\newenvironment{colsitem}[2][]{%
    \setlength{\multicolsep}{2ex}
    \raggedcolumns
    \begin{multicols}{#2}
    \begin{itemize}[#1]
    }{
    \end{itemize}
    \end{multicols}
}

```

colsitem*

```
\newenvironment{colsitem*}[2][]{%
    \setlength{\multicolsep}{2ex}
    \begin{multicols}{#2}
    \begin{itemize}[#1]
    }{
    \end{itemize}
```

```
    \end{multicols}
}
```

7.5 Questions and answers

\exs@process@output

exesheet@questions and exesheet@answers booleans control the display of corresponding environments. They are set by the output key option in the \exs@process@output macro.

```
\newboolean{exesheet@questions}\setboolean{exesheet@questions}{true}
\newboolean{exesheet@answers}\setboolean{exesheet@answers}{true}

\def\exs@process@output{
    \ifthenelse{\equal{\exesheet@output}{questions}}{
        \setboolean{exesheet@questions}{true}
        \setboolean{exesheet@answers}{false}
    }{% else if
    \ifthenelse{\equal{\exesheet@output}{answers}}{
        \setboolean{exesheet@questions}{false}
        \setboolean{exesheet@answers}{true}
    }{% else if
    \ifthenelse{\equal{\exesheet@output}{both}}{
        \setboolean{exesheet@questions}{true}
        \setboolean{exesheet@answers}{true}
    }{% else
    \PackageWarning{exesheet}{Value `\exesheet@output'
         is not supported by option `output'}
    }}}
}
```

questions

We use the versions package by Uwe Lück who provides \comment and \endcomment macros, that allow the magic of conditional displays (we can also find them in verbatim or version packages). The noteworthy codesection package, allows to encapsulate optional code between the macros \BeginCodeSection{⟨*skip*⟩} and \EndCodeSection{⟨*skip*⟩}, both in the text body and in the preamble, but these macros cannot be used inside an environment as we did here for \comment and \endcomment.

Some tests are made by \ifthenelse{\bolean{...}} because \comment and \endcomment causes some hassle with the TEX structure \if ... \else ... \fi. The two counters exe@ini and subpart@ini are used in the following \set@toclevel macro.

```
\newcounter{exe@ini}
\newcounter{subpart@ini}

\newenvironment{questions}{
    \ifthenelse{\boolean{exesheet@questions}}{%
        \setcounter{exe@ini}{\value{exercise}}
        \setcounter{subpart@ini}{\value{subpart}}
    }{\comment}
}{\ifthenelse{\boolean{exesheet@questions}}{}{\endcomment}}

```

`answers` The internal macro `\set@toclevel` calculates the title level (counter `toc@level`) to get a correct typesetting of the word "Correction" at the start of an `answers` environment (when `questions` and `answers` are displayed together). The principle is to compare the state of counters `exercise` and `subpart` with those saved at the time of the call of the `questions` environment. The `\@enumdepth` counter indicates the `enumerate` list level in which we are (0 = out of lists). The optional parameter of the `answers` environment allows to force this title level.

```
\newcounter{@toclevel}
\newcommand{\set@toclevel}[1][]{
    \ifthenelse{\equal{#1}{}}{
        \ifthenelse{\value{exercise} > \value{exe@ini}}{
            \setcounter{@toclevel}{1}
        }{% else
        \ifthenelse{\equal{\the\@enumdepth}{0}}{
            % we're not in an enumerate environment
            \ifthenelse{\(\value{subpart} > \value{subpart@ini}\)
                \or \(\value{subpart} = 0\)}{
              \setcounter{@toclevel}{2}
            }{\setcounter{@toclevel}{3}}
        }{\setcounter{@toclevel}{4}}}
    }{\setcounter{@toclevel}{#1}}}

```

The internal macro `\typeset@correctionname`, typesets the word "Correction" at the right level.

```
\definecolor{correctioncolor}{rgb}{0,0.2,0.6} % kind of dark blue
\newcommand{\correctionstyle}{\color{correctioncolor}}

\newcommand{\typeset@correctionname}{
    \ifthenelse{\value{@toclevel} = 1}{
        \section*{\correctionstyle\correctionname}
        \ifexesheet@exetoc
            \addcontentsline{toc}{section}{\correctionname}
        \fi
        \setcounter{exercise}{0}
    }{% else if
    \ifthenelse{\value{@toclevel} = 2}{%
        \subsection*{\correctionstyle\correctionname}
        \ifexesheet@exetoc
            \addcontentsline{toc}{subsection}{\correctionname}
        \fi
        \setcounter{subpart}{0}
    }{% else if
    \ifthenelse{\value{@toclevel} = 3}{%
        \subsubsection*{\correctionstyle\correctionname}
        \ifexesheet@exetoc
            \addcontentsline{toc}{subsubsection}{\correctionname}
        \fi
    }{% else
    \par\textbf{\correctionstyle\correctionname}\par
    }}}
}

```

Then we can write the `answers` environment.

```
\newenvironment{answers}[1][]{% #1 is the optional level
    \ifthenelse{\boolean{exesheet@answers}}{%
        \ifthenelse{\boolean{exesheet@questions}}{
            \set@toclevel[#1]
            \typeset@correctionname
            \correctionstyle%
            \ifexesheet@multicol
                \renewcommand{\columnseprulecolor}{\color{correctioncolor}}
            \fi
        }{}
    }{\comment}
}{\ifthenelse{\boolean{exesheet@answers}}{}{\endcomment}}

\newenvironment{answers*}{
    \ifthenelse{\boolean{exesheet@answers}}{}{\comment}
}{\ifthenelse{\boolean{exesheet@answers}}{}{\endcomment}}

```

In the `answers` environment, when placing `\correctionstyle` before `\subsubsection` (case of `\typeset@correctionname`), the preceding vertical space may be too wide.

`\question`

```
\newcommand{\question}[1]{\ifexesheet@questions #1\fi}

```

`\answer`

```
\newcommand{\answer}[1]{%
    \ifexesheet@answers%
        \ifexesheet@questions \correctionstyle #1\else #1\fi
    \fi
}

```

`\answerspace` The `answerspace` macro was a suggestion of Maxime Chupin to permit students to write answers on the given paper.

```
\newcommand\answerspace[1]{
    \ifexesheet@answerspace \par\vspace{#1} \fi}

```

7.6 Marking scheme options processing

Options `display`, `marginpos`, `marginwidth` and `noteragged` are processed with the following internal commands.

The `display` key option sets the value of the two booleans `exesheet@pts` and `exesheet@notes`. `exesheet@pts` controls the display of the content of `\pts` and of optional arguments of `\note`, whereas `exesheet@notes` controls mandatory arguments of `\note`.

`\exs@process@display`

```
\newboolean{exesheet@pts}
```

```
\newboolean{exesheet@notes}

\def\exs@process@display{
    \ifthenelse{\equal{\exesheet@display}{pts}}{
        \setboolean{exesheet@pts}{true}
        \setboolean{exesheet@notes}{false}
    }{% else if
    \ifthenelse{\equal{\exesheet@display}{notes}}{
        \setboolean{exesheet@pts}{true}
        \setboolean{exesheet@notes}{true}
    }{% else if
    \ifthenelse{\equal{\exesheet@display}{none}}{
        \setboolean{exesheet@pts}{false}
        \setboolean{exesheet@notes}{false}
    }{% else
    \PackageWarning{exesheet}{Value `\exesheet@display'
         is not supported by option `display'}
    }}}
}

```

`\exs@process@marginpos` The `marginpos` key option takes value `left` (the default) or `right` (or `inner` and `outer`). In practice `inner` is equivalent to `left` but, in two side mode, `left` (the default value) or `right` are converted to `outer` (the default value for two side mode).

```
\newboolean{exesheet@leftmargin}

\def\exs@process@marginpos{
    \ifthenelse{\equal{\exesheet@marginpos}{left}}{
        \if@twoside%
            \PackageWarningNoLine{exesheet}{Default `marginpos' option
                \MessageBreak
                for two side documents is `outer'.\MessageBreak
                Use `inner' to change the side}
            \def\exesheet@marginpos{outer}
            \setboolean{exesheet@leftmargin}{false}
            \normalmarginpar
        \else% default
            \setboolean{exesheet@leftmargin}{true}
            \reversemarginpar
        \fi
    }{% else if
    \ifthenelse{\equal{\exesheet@marginpos}{right}}{
        \if@twoside%
            \PackageWarningNoLine{exesheet}{Default `marginpos' option
                \MessageBreak
                for two side documents is `outer'.\MessageBreak
                Use `inner' to change the side}
            \def\exesheet@marginpos{outer}
        \fi
        \setboolean{exesheet@leftmargin}{false}
        \normalmarginpar
    }{% else if
    \ifthenelse{\equal{\exesheet@marginpos}{inner}}{
```

```
        \setboolean{exesheet@leftmargin}{true}
        \reversemarginpar
    }{% else if
    \ifthenelse{\equal{\exesheet@marginpos}{outer}}{
        \setboolean{exesheet@leftmargin}{false}
        \normalmarginpar
    }{% else
    \PackageWarningNoLine{exesheet}{Value `\exesheet@marginpos'
         is not supported by option `marginpos'}
    }}}}
}
```

`\exs@process@marginwidth` The `marginwidth` option changes the ratio between left and right margins depending on what has to be displayed in the margin (only points or full notes)[9].

When `display=notes`, the additional length `1 in` matches the default free space to the left of `\oddsidemargin`.

The macros `\standardmarginwidthfactor` and `\largemarginwidthfactor` represent the ratio between total margin width and `\marginparwidth`.

```
\def\standardmarginwidthfactor{0.6}
\def\largemarginwidthfactor{0.8}

\newcommand*{\leftnotemarginwidth}[1]{
    \setlength{\marginparwidth}{\oddsidemargin}
    \addtolength{\marginparwidth}{1in}
    \addtolength{\marginparwidth}{-\marginparsep}
    \setlength{\marginparwidth}{#1\marginparwidth}
}

\newcommand*\rightnotemarginwidth[1]{
    \setlength{\marginparwidth}{\paperwidth}
    \addtolength{\marginparwidth}{-\textwidth}
    \addtolength{\marginparwidth}{-\oddsidemargin}
    \addtolength{\marginparwidth}{-\marginparsep}
    \addtolength{\marginparwidth}{-1in}
    \setlength{\marginparwidth}{#1\marginparwidth}
}

\def\exesheet@smallmargins{
    \geometry{hmarginratio=1:1}
    \leftnotemarginwidth{\standardmarginwidthfactor} % right gives the same
}
\def\exesheet@standardmargins{
    \ifexesheet@leftmargin
        \geometry{hmarginratio=3:2}
        \leftnotemarginwidth{\standardmarginwidthfactor}
    \else
        \geometry{hmarginratio=2:3}
        \rightnotemarginwidth{\standardmarginwidthfactor}
    \fi
}
```

[9] So that the effect on the margin ratio is correct, this option is processed at begin document, after other commands that also could alter page geometry.

```
\def\exesheet@largemargins{
    \ifexesheet@leftmargin
        \geometry{hmarginratio=3:1}
        \leftnotemarginwidth{\largemarginwidthfactor}
    \else
        \geometry{hmarginratio=1:3}
        \rightnotemarginwidth{\largemarginwidthfactor}
    \fi
}

\def\exs@process@marginwidth{
    \ifthenelse{\equal{\exesheet@marginwidth}{standard}}{
        \ifthenelse{\equal{\exesheet@display}{none}}{
            \if@twoside
                \exesheet@standardmargins
            \else
                \exesheet@smallmargins
            \fi
        }{% else display=pts or notes
            \exesheet@standardmargins
        }
    }{% else if
    \ifthenelse{\equal{\exesheet@marginwidth}{expand}}{
        \ifthenelse{\equal{\exesheet@display}{none}}{
            \if@twoside
                \exesheet@standardmargins
            \else
                \exesheet@smallmargins
            \fi
        }{% else if
        \ifthenelse{\equal{\exesheet@display}{pts}}{
            \exesheet@standardmargins
        }{% else display=notes
            \exesheet@largemargins
        }}
    }{% else if
        \ifthenelse{\equal{\exesheet@marginwidth}{unset}}{
        % do nothing
    }{% else
    \PackageWarningNoLine{exesheet}{Value `\exesheet@marginwidth'
         is not supported by option `marginwidth'}
    }}}
}

```

For a two side mode document, the package geometry doesn't set the margin paragraph width correctly by default, it's too large. So we made here an explicit setting useful in the case of `marginwidth=unset`. Otherwise, the setting is made by the `marginwidth` key option.

```
\if@twoside \rightnotemarginwidth{0.5} \fi

```

`\exs@process@noteragged` The noteragged option takes one of the following values: `left`, `right`, `center`, `justify` or `twoside`.

`\marginpar` with optional parameter makes the job for a two side document. We then use `\noteraggedleft` and `\noteraggedright` instead of `\noteragged`. Commands `\RaggedLeft`, `\RaggedRight`, `\Centering` and `\justifying` come from the ragged2e package by Martin Schröder. They give better results as standard commands `\raggedleft`, `\raggedright` (or `\centering`). The default LaTeX setting for marginal notes is justifying.

```
\newcommand{\noteragged}{}
\newcommand{\noteraggedleft}{}
\newcommand{\noteraggedright}{}

\def\exs@process@noteragged{
    \ifthenelse{\equal{\exesheet@noteragged}{left}}{
        \if@twoside
            \renewcommand{\noteraggedleft}{\RaggedLeft}
            \renewcommand{\noteraggedright}{\RaggedLeft}
        \else
            \renewcommand{\noteragged}{\RaggedLeft}
        \fi
    }{% else if
    \ifthenelse{\equal{\exesheet@noteragged}{right}}{
        \if@twoside
            \renewcommand{\noteraggedleft}{\RaggedRight}
            \renewcommand{\noteraggedright}{\RaggedRight}
        \else
            \renewcommand{\noteragged}{\RaggedRight}
        \fi
    }{% else if
    \ifthenelse{\equal{\exesheet@noteragged}{center}}{
        \if@twoside
            \renewcommand{\noteraggedleft}{\Centering}
            \renewcommand{\noteraggedright}{\Centering}
        \else
            \renewcommand{\noteragged}{\Centering}
        \fi
    }{% else if
    \ifthenelse{\equal{\exesheet@noteragged}{justify}}{
          \renewcommand{\noteraggedleft}{\justifying} % equiv to nothing
          \renewcommand{\noteraggedright}{\justifying}
          \renewcommand{\noteragged}{\justifying}
    % justify is the default LaTeX setting
    }{% else  if
    \ifthenelse{\equal{\exesheet@noteragged}{twoside}}{
        \if@twoside
            \renewcommand{\noteraggedleft}{\RaggedLeft}
            \renewcommand{\noteraggedright}{\RaggedRight}
        \else
            \PackageWarning{exesheet}{Invalid option 'noteragged=twoside'
             when the document \MessageBreak is not in two side mode}
        \fi
    }{% else
    \PackageWarning{exesheet}{Value '\exesheet@noteragged'
         is not supported by option 'noteragged'}
    }}}}}
```

```
}
```

`\exs@process@checkpts`

The scale control option is based on length calculus. By default these calculus are local but we need to make them global, therefore the two first macros `\gsetlength` and `\gaddtolength`. In them we must avoid to produce a too large space at the place where `\marginpar` is called, therefore all % symbols at end of lines.

For each question, points assigned will be added in `\sum@pts` and, for each exercise, points are accumulated in `\sum@exe`. These lengths are compared to `\exe@total` and `\sheet@total`. `\exe@check` is called at the beginning of each exercise (when macro `\points`, `\totalexe` or `\totalpoints` is called) to check the previous one, and also in the `\exs@process@checkpts` at end of document to check the last exercise.

```
\newlength{\sheet@total}
\newlength{\sum@exe}
\newlength{\exe@total}
\newlength{\sum@pts}
\def\exe@label{none}
\newboolean{scale@valid}
\setboolean{scale@valid}{true}

\gdef\gsetlength#1#2{% to get global length values
    \begingroup
        \setlength\skip@{#2}% local assignment to a scratch register
        \global#1=\skip@%     global assignment to #1
    \endgroup               % \skip@ is restored by end of group
}

\gdef\gaddtolength#1#2{% percent symbol necessary here !
    \begingroup
        \setlength\skip@{#1}%
        \addtolength\skip@{#2}%
        \global#1=\skip@%
    \endgroup
}

\def\exe@check#1{
    \ifthenelse{\lengthtest{\sum@pts = 0pt}\or\equal{\exe@label}{none}}{
    % do not check, no \pts or first exercise begins
    }{
        \ifthenelse{\lengthtest{\exe@total = \sum@pts}}{
            \PackageWarningNoLine{exesheet}{\exe@label:
                scale \the\exe@total\space is valid}
        }{
        \PackageWarningNoLine{exesheet}{\exe@label:
            sum of points is
            \the\sum@pts\space instead of \the\exe@total}
        \setboolean{scale@valid}{false}
        }
        \gsetlength{\sum@pts}{0pt}
    }
    \def\exe@label{#1} % for the next exercise
}
```

```
631
632 \def\exs@process@checkpts{
633     \ifexesheet@checkpts
634         \ifthenelse{\lengthtest{\sheet@total = 0pt}}{
635             \PackageWarningNoLine{exesheet}{Option checkpts is true;
636                 \MessageBreak
637                 but you didn't use \string\totalsheet\space
638                 in the preamble. \MessageBreak
639                 See the documentation for more information}
640         }{}
641         \gsetlength{\sum@exe}{0pt}
642         \gsetlength{\exe@total}{0pt}
643         \gsetlength{\sum@pts}{0pt}
644         \AtEndDocument{
645             \ifthenelse{\equal{\exe@label}{none}}{
646                 \ifthenelse{\lengthtest{\sheet@total = \sum@pts}}{
647                     \PackageWarningNoLine{exesheet}{Sum of points
648                         is valid: \the\sheet@total}
649                 }{
650                     \PackageWarningNoLine{exesheet}{Inconsistent
651                         sum of points:
652                         \the\sum@pts\space instead of \the\sheet@total}
653                     \setboolean{scale@valid}{false}
654                 }
655             }{
656                 \exe@check{end}
657                 \ifthenelse{\lengthtest{\sheet@total = \sum@exe}}{
658                     \PackageWarningNoLine{exesheet}{Sum of points
659                         is valid: \the\sheet@total}
660                 }{
661                     \PackageWarningNoLine{exesheet}{Inconsistent
662                         sum of points:
663                         \the\sum@exe\space instead of \the\sheet@total}
664                     \setboolean{scale@valid}{false}
665                 }
666             }
667             \ifthenelse{\boolean{scale@valid}}{
668                 \PackageWarningNoLine{exesheet}{Scale is valid}
669             }{
670                 \PackageWarningNoLine{exesheet}{Scale is NOT valid !
671                     See above}
672             }
673         }
674     \fi
675 }
676
```

7.7 Margin notes commands

\points

```
677 \definecolor{pointscolor}{named}{red}
678 \newcommand{\pointsstyle}{%
679     \small\mdseries\sffamily\color{pointscolor}\fbox}
680 \newcommand*{\exesheet@points}[1]{\hfill
```

```
    \pointsstyle{#1~%
        \ifthenelse{\lengthtest{#1pt < 2pt}}{\pointname}{\pointsname}}
    \ifexesheet@checkpts\gaddtolength{\sum@exe}{#1pt}\fi%
}
\newcommand*{\points}[1]{%
    \ifthenelse{\boolean{exesheet@questions}}{\exesheet@points{#1}}{}}
```

Percent symbols are necessary to avoid spaces between the `\fbox` and its inner text. Without `\lengthtest`, the test `#1 < 2` doesn't work with decimal numbers but it works with lengths.

`\pts`

```
\definecolor{ptscolor}{named}{red}
\newcommand{\ptsstyle}[1]{%
    \footnotesize\centering\sffamily\color{ptscolor} (#1)}
\newcommand*{\ptsmark}[1]{%
    \ifthenelse{\lengthtest{#1pt < 2pt}}{#1 \ptname}{#1 \ptsname}}
\newcommand*{\pts}[1]{%
    \ifexesheet@pts%
        \mbox{}%
        \marginpar{\hspace{0pt}\ptsstyle{\ptsmark{#1}}}%
        \ifexesheet@checkpts%
            \gaddtolength{\sum@pts}{#1pt}%
        \fi%
    \fi%
    \ignorespaces
}
```

`\totalexe` In the following macros using `\marginpar`, percent symbols and `\ignorespaces` are necessary to avoid an enlarged blank space in the text (or the margin) where these macros are inserted.

```
\definecolor{markingcolor}{named}{red}
\newcommand{\markingstyle}[1]{\footnotesize\sffamily%
    \centering\color{markingcolor}\textbf{#1}}
    % inner arguments allow boxed styles
\newlength{\ptsboxlength}
\setlength{\ptsboxlength}{3.1em}
\cornersize{1}
\newcommand*{\totalexe}[1]{%
    \ifexesheet@pts%
        \mbox{}%
        \marginpar{\hspace{0pt}\markingstyle{\ovalbox{%
            \makebox[\ptsboxlength]{\ptsmark{#1}}}}}%
    \fi%
    \ifexesheet@checkpts%
        \gsetlength{\exe@total}{#1pt}%
        \gaddtolength{\sum@exe}{#1pt}%
    \fi%
    \ignorespaces
}
```

\totalsheet

```
\newcommand*{\totalsheet}[1]{
    \gsetlength{\sheet@total}{#1pt}
}
```

\note
\note*

Boolean `exesheet@pts` and `exesheet@notes` control the display of marginal notes. If `exesheet@pts` is `false`, `exesheet@notes` will be ignored. `\noindent` is necessary when using `\justifying` from the ragged2e package. Inside the `\note@marginpar` macro, double braces around `\markingstyle` avoid some unattended style in the mandatory argument of `\note`. A vicious error occur when using `\if` ... `\fi` structure inside the `\note@marginpar` macro instead of `\ifthenelse` (but only if `@twoside` is `true`).

```
\definecolor{notecolor}{rgb}{0.0, 0.4, 0.0} % kind of dark green
\newcommand{\notestyle}[1]{\footnotesize\sffamily\color{notecolor} #1}
\newcommand{\note@marginpar}[1]{%
    \if@twoside%
        \marginpar[\noteraggedleft #1]{\noteraggedright #1}%
    \else%
        \marginpar{\noteragged #1}%
    \fi%
}
\newcommand{\@note}[2][]{%
    \ifexesheet@pts%
        \mbox{}%
        \note@marginpar{%
            \ifthenelse{\equal{#1}{}}{}{{%
                \noindent\hspace{0pt}\markingstyle{#1}\\}}%
            \ifthenelse{\boolean{exesheet@notes}}{%
                \noindent\hspace{0pt}\notestyle #2%
            }{}%
        }%
    \fi%
    \ifexesheet@checkpts%
        \ifthenelse{\equal{#1}{}}{}{%
            \gaddtolength{\sum@pts}{#1pt}}%
    \fi%
    \ignorespaces
}
\newcommand{\@@note}[1]{%
    \ifexesheet@pts%
        \mbox{}%
        \marginpar{\noindent\hspace{0pt}\markingstyle{#1}}%
    \fi%
    \ifexesheet@checkpts%
        \gaddtolength{\sum@pts}{#1pt}%
    \fi%
    \ignorespaces
}
\newcommand{\note}{\@ifstar{\@@note}{\@note}}
```

\totalpoints

```
\newcommand{\totalpoints}{%
    \ifthenelse{\boolean{exesheet@pts}}{\totalexe}{\points}}
```

7.8 The `correct` option and other (deprecated) commands

`\exs@process@correct`

```
\def\exs@process@correct{
    \ifthenelse{\equal{\exesheet@correct}{false}}{
        % do nothing
    }{% else
    \@ifpackageloaded{schooldocs}{
        \ifthenelse{\equal{\exesheet@correct}{true}}{
            \correct
        }{% else
        \ifthenelse{\equal{\exesheet@correct}{conditional}}{
            \ifexesheet@answers \correct \fi
        }{}}
    }{
        \PackageWarningNoLine{exesheet}{Package ‘schooldocs’ must be loaded
            \MessageBreak
            to use the ‘correct’ option}
    }}
}
```

The following macros are maintained for now only for compatibility reasons.

```
\newcommand{\questionsonly}{
    \PackageWarning{exesheet}{Command \string\questionsonly\space
        is deprecated, \MessageBreak
        use ‘output=questions’ as package option instead}
    \renewcommand\exesheet@output{questions}
    \exs@process@output
}
\newcommand{\answersonly}{
    \PackageWarning{exesheet}{Command \string\answersonly\space
        is deprecated, \MessageBreak
        use ‘output=answers’ as package option instead}
    \renewcommand\exesheet@output{answers}
    \exs@process@output
}
\newcommand{\displaypts}{%
    \PackageWarning{exesheet}{Command \string\displaypts\space
        is deprecated, \MessageBreak
        use ‘display=pts’ as package option instead}
    \renewcommand\exesheet@display{pts}
    \exs@process@display
}
\newcommand{\displaypoints}{%
    \PackageWarning{exesheet}{Command \string\displaypoints\space
        is deprecated, \MessageBreak
        use ‘display=pts’ as package option instead}
    \renewcommand\exesheet@display{pts}
    \exs@process@display
```

```
}
\newcommand*{\displaynotes}[1][\RaggedLeft]{%
    % \renewcommand{\noteragged}{#1} no effect now!
    \PackageWarning{exesheet}{Command \string\displaynotes\space
        is deprecated, \MessageBreak
        use ‘display=notes’ as package option instead}
    \renewcommand\exesheet@display{notes}
    \exs@process@display
    \renewcommand{\noteragged}{#1}
}
\newcommand*{\displaynotesright}[1][\RaggedRight]{%
    % \renewcommand{\noteragged}{#1} no effect now!
    \PackageWarning{exesheet}{Command \string\displaynotes\space
        is deprecated, \MessageBreak
        use ‘display=notes, margin=right’ as package options instead}
    \renewcommand\exesheet@display{notes}
    \exs@process@display
    \renewcommand\exesheet@margin{right}
    \renewcommand{\noteragged}{#1}
}

\PackageInfo{exesheet}{Environment ‘tablenum’ is deprecated \MessageBreak
    and replaced by ‘tablenum1’. \MessageBreak
    Options ‘notoc’ and ‘nosetlist’ \MessageBreak
    are no longer supported\@gobble}
    % gobble allow to suppress line number
⟨/package⟩
```