

Il pacchetto esami*

Grazia Messineo <grazia.messineo@unicatt.it>
Salvatore Vassallo <salvatore.vassallo@unicatt.it>

24 aprile 2013

Indice

1	Introduzione e motivazioni	2
2	Pacchetti richiesti	2
3	Installazione	2
4	Come scrivere un esame	3
4.1	Il preambolo	3
4.2	Il documento	3
5	Le opzioni del pacchetto	4
6	Gli esercizi	5
6.1	Le varianti	5
6.2	L'ambiente test	5
6.3	Gli ambienti problem e problem*	8
6.4	Altre tipologie di esercizi	9
6.5	Altri comandi	11
7	I comandi per lavorare con i parametri e con le liste	12
7.1	Comandi per lavorare con i parametri (casuali)	12
7.2	Comandi per la scelta da una lista	15
8	I file ausiliari	16
8.1	Il file di localizzazione	16
8.2	Il file di configurazione	16
8.3	I file master e master-sol	16
8.4	Il file totale-versioni	16

*Questo documento corrisponde al pacchetto **esami** versione 1.0, con data 2013/03/08.

1 Introduzione e motivazioni

Il pacchetto **esami** è stato creato per produrre i testi e le soluzioni per gli esami di Matematica nella Facoltà di Economia dell'Università Cattolica di Milano e nasce dall'esigenza di poter generare molte versioni (da 12 a 50 solitamente) di una stessa prova d'esame contenenti esercizi simili, ma diversi e di stampare le soluzioni e, se fossero presenti domande a risposta chiusa, la stringa delle soluzioni.

Il pacchetto consente di scrivere esercizi di vario tipo (a risposta multipla con risposte che variano in ordine casuale, a risposta chiusa o aperta, *matching*, problemi, ecc.). Le domande di ogni gruppo ruotano in modo casuale all'interno del compito e le varianti di ciascun esercizio vengono scelte in modo casuale da un file che le contiene tutte e, a nostra conoscenza, è l'unico che permetta di ottenere tali risultati senza utilizzare programmi esterni a L^AT_EX.

Lo sviluppo del pacchetto è iniziato nel 2008 con l'intento di estendere e unificare alcune proprietà dei pacchetti **exerquiz** e **eqexam** di D.P. Story [4] e **probsoln** di N.Talbot [5].

2 Pacchetti richiesti

Il pacchetto **esami** dipende da alcuni altri pacchetti non standard:

1. **fp**: per le operazioni matematiche;
2. **currfile**: per ottenere il nome del file e della cartella di lavoro;
3. **xargs**: per avere più di un parametro opzionale;
4. **pstricks**, **pstricks-add** e **auto-pst-pdf**: per i grafici.

Inoltre i seguenti pacchetti sono opzionali, ma raccomandati:

1. **lmodern** and **amsfonts**: necessari per un buon output in pdf;
2. **numprint**: per un output migliore dei numeri e per avere formati numerici diversi a seconda del linguaggio;
3. **eurosym**: per usare la valuta
4. **icomma**: dal pacchetto **ws** per una corretta formattazione delle virgole.

Tutti i pacchetti sono disponibili sia in T_EXLive che in MikT_EX.

3 Installazione

Il pacchetto è composto dal file **esami.sty**, da alcuni file di configurazione ***.cfg**, da 3 template per la compilazione e dalla documentazione. Il file **sty** e i file **cfg** e **lng** devono essere copiati in una cartella dove possano essere trovati da L^AT_EX (ad esempio **texmf-local/tex/latex/esami**) e in seguito deve essere aggiornato il database.

4 Come scrivere un esame

In questa sezione si vedrà come costruire una prova d'esame usando il pacchetto `esami`.

4.1 Il preambolo

```
\documentclass[italian]{article}
\usepackage[<opzioni>]{esami}
```

Le opzioni sono discusse nella sezione 5

```
\def\numcompiti{10}
\date{2013/04/21}%% THE DATE IN THE FORMAT YYYY/MM/DD
\def\Data{\longdate}%% or \shortdate: the date in the heading
```

Il comando `\date` è quello solito di L^AT_EX, ma la data deve essere scritta nel formato YYYY/MM/DD, mentre il comando `\numcompiti` serve a definire il numero di versioni che devono essere generate. Il comando `\Data` stamperà la data in formato lungo (D MMMM YYYY `\longdate`) o corto (D/M/YYYY `\shortdate`).

4.2 Il documento

```
\begin{document}
\pagestyle{esame}
% \immediate\openout\sols=\thenomefile.sol.tex
```

Il comando `\pagestyle` è definito in un file di configurazione (si veda la sezione 8.2) e definisce gli header e i footer, il comando commentato si usa nel file delle soluzioni per generare il file che contiene la stringa delle soluzioni.

```
\whiledo{\thevers<\numcompiti}{\stepvers
```

inizia la routine di compilazione delle varianti che terminerà alla fine del documento.

```
\FPeval\seme{round((\thenomefile+\thevers):0)}
\randomi=\seme
```

genera il seme per i processi di randomizzazione e lo assegna anche a `\randomi`

```
%\immediate\write\sols{\string\begin{minipage}{.33\string\textwidth}
\string\large\string\textbf{Soluzione Versione n. \thevers}}
\testa
\istruzionii
```

`\testa`, `\istruzionii` vengono definiti nel file di configurazione e contengono le istruzioni per la testata e delle eventuali istruzioni per lo svolgimento dell'esame, mentre l'istruzione commentata serve nel file soluzioni per la stringa delle soluzioni.

```
\begin{test}[punti]
\begin{questions}
\esercizi{test1,test2,test3,test4}
\end{questions}
\end{test}
%\immediate\write\sols{\string\par\string\end\string{enumerate\string}\string\end{minipage}
% \string\vspace{1.5cm}}
```

In questo modo si definisce un ambiente per un test tipicamente con domande a risposta multipla, il comando `\esercizi` serve per scegliere casualmente gli esercizi dai file indicati e per rimescolarli. Come sopra i comandi commentati servono per il file soluzioni per generare la stringa delle soluzioni.

```
\esercizi{problema1, problema2}
```

sceglie tra due esercizi a risposta aperta.

```
\closevers
}
%\immediate\closeout\sols
%\stringasol
\end{document}
```

Termina la routine di compilazione, e, nel file delle soluzioni, chiude il file della stringa delle risposte.

5 Le opzioni del pacchetto

Per usare il pacchetto bisogna usare il solito comando `\usepackage[options]{esami}`. Le opzioni possibili sono:

- *allowrandomize* e *norandomize*: la prima permette di permutare in modo casuale l'ordine delle risposte nelle domande a risposta chiusa, la seconda di non permutarlo (di default le risposte vengono permutate in modo casuale).
- *shuffle*, *shufflerandom* e *noshuffle*: la prime due permettono di permutare gli esercizi, la prima in modo deterministico se il numero degli esercizi è minore di 9 e casuale se è maggiore, la seconda del tutto casualmente, la terza di non permutarli (l'opzione di default è *shuffle*).
- *xyz*: Fa caricare un file di configurazione (`esami-xyz.cfg`) con nome dell'esame, istruzioni, ecc. relative a questo esame. Nel file `esami.sty` sono predefiniti alcune opzioni; è possibile caricare altre configurazioni definendole in `esami.sty` o creare un proprio file di configurazione e farlo leggere senza modificare il file `esami.sty`: è sufficiente scrivere un'opzione sconosciuta (ad esempio `math`) e scrivere il file `esami-math.cfg`.
- *pointsonright*: fa apparire la casella con il punteggio sulla destra dell'esercizio. Se l'opzione non è specificata, la casella non appare.
- *nosolutions*: questa opzione genera il compito senza soluzioni (default).
- *solutions*: genera il file delle soluzioni.
- *prova*: si usa solo con il file `totale-versioni` (si veda 8.4) e serve per visualizzare tutte le varianti di un esercizio. Fa visualizzare automaticamente le risposte esatte negli esercizi a risposta multipla e la soluzione nei problemi da svolgere.
- *param*: serve per visualizzare le lettere che rappresentano i parametri casuali anziché il valore numerico attribuito. Stampa anche una frase iniziale, nella quale sono contenuti il valore iniziale e finale attribuiti ai parametri usati nell'esercizio. Può essere usata solo insieme all'opzione *prova*.

- *correzione*: stampa solo i testi di tutti gli esercizi, senza soluzioni (funziona solo nel file `totale-versioni`).
- *fillb*: da usare se nel compito ci sono esercizi di tipo `fillin`, per poter inserire le risposte corrette nella stringa delle soluzioni.
- *twocolumns*: per scrivere il testo delle domande a risposta chiusa su due colonne (ma si veda anche 6.2).
- *sansserif*: usa caratteri sans serif: utile soprattutto se si usano le due colonne.
- *autopst* e *autopstoff*: caricano il pacchetto `auto-pst-pdf` e permettono di compilare il file direttamente con `pdflatex` (anche se contiene codice di `pstricks`). La prima opzione crea e include direttamente nel documento i file delle immagini, la seconda non crea i file delle immagini (e rende più veloce la compilazione)¹.
- *language*: legge il file di localizzazione `es-language.lng` (si veda ??). Nel file sono contenute le traduzioni delle etichette (esercizio, soluzione, ...), dei messaggi d'errore, delle intestazioni contenute nel file (`esami-xyz.cfg`). Di default viene caricato il file della lingua italiana. Poiché le opzioni linguistiche hanno lo stesso nome di quelle usate in `babel` è possibile passare l'opzione in `\documentclass`.

6 Gli esercizi

6.1 Le varianti

`\newproblem` Ogni esercizio (con tutte le sue varianti) va scritto in un file separato. Ogni variante è racchiusa nel comando `\newproblem{ }` (che non ha alcun argomento opzionale).

Esempio 1.

```
%% Questo è il file file1.tex con 2 varianti di un esercizio
\newproblem{ Testo della prima variante}
\newproblem{Testo della seconda variante}
```

Il comando `\newproblem` è una versione molto modificata dell'analogo comando del pacchetto `probsoln`.

6.2 L'ambiente test

`test` L'ambiente `test` consente di scrivere esercizi a risposta multipla o a risposta aperta breve, in cui ciascuno dei quesiti è tratto casualmente da uno dei file nel quale sono contenuti tutti gli esercizi con le rispettive varianti.

L'ambiente viene creato dal comando

```
\begin{test}[punti]
...
\end{test}
```

punti Il parametro opzionale [*punti*] può essere solo un numero intero maggiore o uguale a 1, che indica il punteggio **totale** attribuito all'intero gruppo di domande.

questions I quesiti del gruppo, nel documento principale, sono racchiusi nell'ambiente *questions* che è una versione modificata dell'ambiente omonimo del pacchetto **exerquiz**:

```
\begin{questions}
....
\end{questions}
```

È possibile inserire all'interno di un unico ambiente **test** più ambienti **questions**. Gli esercizi di ogni gruppo saranno mescolati indipendentemente.

Esempio 2.

```
\begin{test}
  \begin{questions}
    ....
  \end{questions}
  ....
  \begin{questions}
    ....
  \end{questions}
\end{test}
```

\esercizi All'interno di tale ambiente, i quesiti vengono “caricati” dal comando

```
\esercizi{nome file sorgente 1, nome file sorgente 2,...}
```

Il parametro obbligatorio è il nome di ciascuno degli esercizi che si vogliono inserire nel test, separati da una virgola².

È possibile sia inserire tutti gli esercizi con un unico comando, sia usare più comandi **\esercizi** o **\estraies** (si veda 7.2). Questa possibilità è utile ad esempio se si vuole che ci siano esercizi da due sottoinsiemi diversi (5 esercizi sui limiti scelti tra 7 e 3 esercizi sulle derivate scelti tra 5) o se si vuole per esigenze di impaginazione avere alcuni esercizi su due colonne ed altri su una colonna: in tale caso si deve introdurre la parte che deve essere su due colonne con **\begin{multicols}{2}** e concluderla con **\end{multicols}**. **Non** si usa l'opzione *twocolumns*.

Esempio 3.

```
\begin{test}[punti]
  \begin{multicols}{2}
    \esercizi{es1,es2,es3}
  \end{multicols}
  \esercizi{es4,es5,es6}
\end{test}
```

In questo modo gli esercizi **es1,es2,es3** sono formattati su due colonne mentre gli esercizi **es4,es5,es6** sono formattati su una colonna.

Nel file degli esercizi, la sintassi da usare all'interno di **\newproblem** è la seguente:

```
\item \PTs{punteggio}
```

¹Per usare **auto-pst-pdf** è necessario abilitare **shell-escape** (T_EXLive) o **write18** (MiK_TE_X) nell'esecuzione di **pdflatex**.

²In seguito verranno introdotti altri comandi per il caricamento di esercizi

```

...testo... \begin{answers}{numero colonne}
    \bChoices[random]
\Ans0 risposta errata \eAns
\Ans0 risposta errata \eAns
\Ans1 risposta esatta \eAns
    \eFreeze
\Ans0 nessuna delle precedenti \eAns
    \eChoices
\end{answers}

```

nella quale:

- `\item \PTs{punteggio}`

`\PTs` introduce una domanda con punteggio indicato in `\PTs` (può essere anche un numero decimale e il separatore può essere la virgola).

- `\begin{answers}{numero colonne}`
`\bChoices[random]`
`...`
`\eChoices`
`\end{answers}`

`answers` introduce le risposte disposte sul numero di colonne specificate. Le risposte vengono permutate in modo casuale solo se è specificata l'opzione (*random*).

`\Ans0`

`\Ans1`

`\bChoices`

`\eChoices`

`\eFreeze`

- `\Ans0` introduce una risposta errata
- `\Ans1` introduce la risposta esatta
- `\eFreeze` introduce (se si vuole) una o più risposte che non saranno in ordine casuale.

Esempio 4.

```
\newproblem{
\item \PTs{1} Le soluzioni dell'equazione
\[x^2-5x+6=0\]
sono:
\begin{answers}{2}
  \bChoices[random]
  \Ans0 $x=3$ e $x=-2$ \eAns
  \Ans0 $x=-3$ e $x=-2$ \eAns
  \Ans1 $x=3$ e $x=2$ \eAns
  \eFreeze
  \Ans0 nessuna delle precedenti \eAns
  \eChoices
\end{answers}
}

\newproblem{
\item \PTs{1} Le soluzioni dell'equazione
\[x^2+5x+6=0\]
sono:
\begin{answers}{2}
  \bChoices[random]
  \Ans0 $x=3$ e $x=-2$ \eAns
  \Ans1 $x=-3$ e $x=-2$ \eAns
  \Ans0 $x=3$ e $x=2$ \eAns
  \eFreeze
  \Ans0 nessuna delle precedenti \eAns
  \eChoices
\end{answers}
}
```

Oltre alle domande a risposta multipla è possibile inserire nell'ambiente `test` altri tipi di domande che verranno descritti nella sezione 6.4, ma se si usano esercizi di tipo `fill-in` se ci sono più spazi bianchi da riempire la stringa delle soluzioni avrà un'utilità minore poiché non seguirà la numerazione delle risposte.

6.3 Gli ambienti `problem` e `problem*`

`problem` Servono per scrivere problemi con risposte aperte. Si usano all'interno del file che contiene le varianti dell'esercizio, annidati nel comando `\newproblem`, con la seguente sintassi:

```
\begin{problem}
...
\end{problem}
```

se si sceglie di scrivere un esercizio a risposta aperta in una sola parte; con la sintassi

```
\begin{problem*}
...
\end{problem*}
```

se invece si sceglie di scrivere un esercizio a risposta aperta in più parti. Il codice di questi ambienti è derivato, con molte modificazioni, dal codice dell'ambiente `exercise` del pacchetto `exerquiz`³.

Esercizio senza parti È introdotto, come si è detto, dall'ambiente `problem`.

```
\begin{problem} [punteggio]
.....Testo.....
\begin{solution}[spazio_soluzione]
.... testo soluzione .....
\end{solution}
\end{problem}
```

`solution` In `[<punteggio>]` va il punteggio dell'esercizio, in `[<spazio_soluzione>]` va l'eventuale spazio bianco che deve essere lasciato per la soluzione.

`parts` **Esercizio in più parti** È introdotto dall'ambiente `problem*`.

```
\begin{problem*} [punteggio]
....testo....
\begin{parts}
\item \PTs{punteggio della parte}
....testo....
    \begin{solution}[spazio_soluzione]
    .... testo soluzione .....
    \end{solution}
\item \PTs{punteggio della parte}
.....
\end{parts} \end{problem*}
```

Con `\PTs{punteggio della parte}` si indica il punteggio della singola parte.

6.4 Altre tipologie di esercizi

Sono state definite alcune nuove tipologie di esercizio:

`\fillin` **fillin:** serve per creare esercizi “a riempimento”. Parte del testo viene lasciata vuota e deve essere riempita dallo studente. Può essere usato anche per creare esercizi a risposta aperta breve. Anche questo tipo di esercizio è simile ad un tipo di esercizio introdotto da D.P. Story nel pacchetto `eqexam`.

La sintassi è

```
\fillin[u|e|b]{larghezza dello spazio}{risposta}
```

I due parametri obbligatori sono la `{<larghezza dello spazio>}` da lasciare, che deve essere espressa con una lunghezza (cm, pt, ecc.) e la `{<risposta>}` esatta (parola o numero) che lo studente dovrebbe inserire (che verrà stampata solo nel file delle soluzioni).

³L'ambiente `exercise` è rimasto e può essere usato come template per la costruzione di altri ambienti come nuovi esercizi, esempi o altro.

Il parametro opzionale definisce come deve essere segnalato lo spazio in cui inserire la risposta: *u* (*underlined*), che è l'opzione di default, fa sì che lo spazio sia evidenziato mediante una riga su cui va inserita la risposta, *e* (*empty*) crea uno spazio vuoto (non evidenziato in alcun modo), *b* (*boxed*) crea uno spazio circondato da un riquadro.

All'interno del comando `\fillin` (nello spazio per la risposta) non è possibile usare i comandi `\semplici` o `\semplici` (si veda la sezione 7.1).

problema **domanda:** questo ambiente deve contenere il testo dell'esercizio (non le soluzioni). Viene usato negli ambienti `problema` e `problema*`, con l'opzione *solutiononly* non viene stampato il testo degli esercizi, ma solo la soluzione. Nelle versioni precedenti del pacchetto per lo stesso scopo veniva usato un altro ambiente.

risposta **risposta:** serve per creare, in esercizi a risposta aperta lunga, lo spazio per inserire la risposta in un box o su righe.

La sintassi del comando è

```
\begin{risposta}[b|l]{spazio verticale}
\end{risposta}
```

Il primo parametro definisce se lo spazio per la risposta deve essere riquadrato (opzione *b*, di default) oppure diviso in righe su cui andrà scritta la risposta (opzione *l*). Il secondo parametro definisce invece l'altezza dello spazio per la risposta (in cm se con opzione *b*, in numero di righe se con opzione *l*)

\matching **matching:** è basato su un'idea del pacchetto `examdesign` [1]. Serve per creare esercizi in cui le parole o formule contenute su due colonne vanno abbinate dallo studente in modo corretto.

\pair Le coppie vengono definite dal comando `\pair`:

```
\pair{nome 1}{nome 2}
```

che va inserito tante volte quanti sono i nomi o le formule da abbinare.

L'elenco dei nomi (mischiati in ordine casuale su ciascuna colonna) viene visualizzato con il comando `\matching`.

Esempio 5.

<code>\pair{Regno Unito}{Londra}</code>	_____	Grecia	(A) Parigi
<code>\pair{Francia}{Parigi}</code>	_____	Regno Unito	(B) Berlino
<code>\pair{Italia}{Roma}</code>	_____	Francia	(C) Roma
<code>\pair{Germania}{Berlino}</code>	_____	Italia	(D) Londra
<code>\pair{Grecia}{Atene}</code>	_____	Germania	(E) Atene
<code>\matching</code>			

Nelle soluzioni viene invece visualizzato l'abbinamento corretto.

tabella **tabella:** serve per creare esercizi con più risposte aperte incolonnate.

La sintassi è

```
\begin{tabella}[numero colonne sempre visibili]
{allineamento colonne visibili}{allineamento colonna nascosta}
\dots & \dots \cr
\end{tabella}
```

Il primo parametro (il cui valore di default è 2) indica il numero di colonne della tabella il cui contenuto deve essere visibile anche nel testo dell'esercizio (e non solo nella soluzione), mentre è possibile avere solo una colonna il cui contenuto è invisibile nel testo, ma appare nella soluzione. Il secondo parametro consente di scegliere l'allineamento delle colonne il cui contenuto è sempre visibile e il terzo l'allineamento della colonna invisibile.

Esempio 6. Con il codice

```
\begin{center}
\renewcommand\arraystretch{3}
\begin{tabella}[1]{r}{c}
\hline
Il dominio della funzione è &  $D_f = (-\infty; 2]$  \cr
\hline
L'insieme immagine della funzione è &  $Im(f) = (-\infty, 0]$  \cr
\hline
\end{tabella}
\end{center}
```

si ottiene il seguente risultato

Il dominio della funzione è	$D_f = (-\infty; 2]$
L'insieme immagine della funzione è	$Im(f) = (-\infty, 0]$

workarea **workarea:** non è una tipologia di esercizi, ma serve a creare un'area di lavoro, cioè uno spazio bianco sul foglio in cui lo studente possa scrivere. La sintassi è

```
\begin{solution}{lunghezza}
\begin{workarea}[larghezza]{lunghezza}
\end{workarea}
\end{solution}
```

La lunghezza dello spazio soluzione e della workarea dovrebbero essere uguali caso contrario il testo presente nella workarea viene posizionato male nello spazio della soluzione, sovrappo-
ndosi al testo dell'esercizio se la sua lunghezza è maggiore di quella della soluzione. La larghezza della workarea è opzionale e di default è uguale alla larghezza del testo. A differenza dell'am-
biente `solution`, nella `workarea` è possibile inserire testo, grafici, ecc.; ad esempio, è possibile inserire degli assi coordinati per disegnare un grafico.

6.5 Altri comandi

Il pacchetto definisce altri due comandi utilizzati nel file `totale-versioni` (si veda 8.4)

`\selectallproblems`

- `\selectallproblems{\esercizio}`: si usa all'interno del file che serve per visualizzare tutte le varianti di un esercizio, con i parametri calcolati oppure no.

- `\esercizio`: si usa all'interno del file che serve per visualizzare tutte le varianti di un esercizio e serve per definire il nome dell'esercizio sul quale si sta lavorando, che verrà usato nel comando `\selectallproblems` visto in precedenza e verrà stampato come titolo del file pdf prodotto. La sintassi per definire il nome di un esercizio è `\def\esercizio{nome}`.

7 I comandi per lavorare con i parametri e con le liste

7.1 Comandi per lavorare con i parametri (casuali)

Sono stati definiti alcuni comandi per lavorare con i parametri (casuali).

Questi comandi si basano sul pacchetto `fp` [2], che permette di eseguire calcoli anche molto complessi all'interno di un documento \LaTeX . Il comando fondamentale che serve per definire un parametro è `\FPsetpar[⟨seme⟩]{⟨nome-parametro⟩}{⟨primo valore⟩}{⟨ultimo valore⟩}[⟨valori esclusi⟩]` che serve per dichiarare i parametri casuali all'interno di un esercizio (a risposta chiusa o aperta). Il parametro avrà nome `\nome-parametro` e varierà tra il primo valore e l'ultimo valore (inclusi). Il parametro sarà un numero intero⁴. Il parametro opzionale `[⟨seme⟩]`, serve a imporre un seme diverso per la generazione del numero casuale. Il valore di default del parametro è `\seme` ed è definito a partire dalla data dell'esame e dal numero della versione. È possibile escludere dalla scelta uno o più valori (ultimo parametro tra parentesi quadre). Se i valori sono più di uno devono essere posti tra parentesi graffe e separati tra virgole; tra i valori esclusi ci possono essere anche altri parametri purché già definiti in precedenza (anche il primo e l'ultimo valore possono essere parametri: in questo caso bisogna prestare attenzione al fatto che il primo valore deve essere minore dell'ultimo). Il meccanismo usato per soddisfare tali condizioni consiste nel ripetere la generazione più volte: il numero massimo di ripetizioni è determinato dal parametro `\maxLoopLimit`, il cui valore di default è 10, ma che è definibile anche in fase di compilazione.

Esempio 7.

```
\FPsetpar{a}{2}{10}
```

crea una variabile `\a` che può assumere un valore casuale tra 2 e 10, in questo caso, il seme è quello stabilito nel preambolo del documento. I comandi

```
\FPsetpar{a}{2}{10}[3]
\FPsetpar{b}{4}{12}{\a,6}
```

creano due variabili `\a` (che può assumere un valore casuale tra 2 e 10, escluso 3) e `\b` (che può assumere un valore tra 4 e 12, escluso 6 e il valore già assegnato ad `\a`).

Sui parametri definiti è possibile operare come su numeri e sono stati definiti comandi che permettono di operare su tali parametri mostrando il risultato numerico o le operazioni in forma parametrica.

- `\FPSv[⟨numero di cifre decimali⟩]{⟨parametro/operazione su parametri⟩}`: serve ad eseguire operazioni su parametri e restituirne il valore numerico (con il numero prescelto di

⁴Anche se è possibile definire parametri razionali o (pseudo)reali, come nel pacchetto `rangen` [3] abbiamo preferito restringere la scelta al caso intero e ottenere gli altri casi operando sui parametri

cifre decimali) o stampare l'operazione eseguita (se si sceglie l'opzione *param* nel pacchetto esami).

Esempio 8.

Istruzione	Risultato numerico	Opzione <i>param</i>
<code>\FPsv{2*k+1}</code>	5	$2 * k + 1$
<code>\FPsv[2]{(2*k+1)/2}</code>	2.50	$(2 * k + 1)/2$

Le principali caratteristiche del comando `\FPsv` sono (negli esempi, $k = 2$):

- le operazioni devono essere sempre indicate in modo esplicito

Esempio 9.

`$\FPsv{2*k}$` 4

- in via automatica, ogni operazione viene racchiusa tra parentesi tonde. La parentesi non viene visualizzata quando si vuole il risultato numerico, mentre lo è con l'opzione *param*

Esempio 10.

Istruzione	Risultato numerico	Opzione <i>param</i>
<code>\$\FPsv{2*k+1}x\$</code>	$5x$	$(2 * k + 1)x$

- Si possono usare i seguenti simboli per le operazioni: $+, -, *, / , ^$
- Occorre prestare attenzione al fatto che il pacchetto `fp` non può gestire potenze con base negativa

`\FPval`

- `\FPval{⟨nome⟩}{⟨parametro/operazione su parametri⟩}`: assegna a $\{⟨nome⟩\}$ il valore dell'operazione arrotondato (è il comando `\FPeval` di `fp` modificato) o stampare l'operazione eseguita se si sceglie l'opzione *param* nel pacchetto esami, come con `\FPsv`.

Esempio 11. Sia $k = 2$. Il codice

```
\FPsetpar{k}{1}{3}
\FPval{a}{2*k+1}
\FPsetpar{b}{2}{20}[\a]
```

genera una variabile `\b` che assume un valore casuale tra 2 e 20, escluso (in questo caso) il valore 5. Nella versione parametrica appare invece una stringa di questo tipo (per quanto riguarda la variabile `\b`)

Il parametro b varia da 2 a 20. $b \neq (2 * k + 1)$. Il seme vale 209.

`\sempli`

- `\sempli{⟨num⟩}{⟨den⟩}`: serve per scrivere delle frazioni in cui il numeratore o il denominatore contengono parametri e che potrebbero dover essere ridotte ai minimi termini. All'interno del comando `\sempli` non si deve usare `\FPsv`.

Esempio 12.

Se $k = 1$:

$$\frac{\text{\texttt{\$}\sempli{2*k}\{3*k+1}\$}}{\text{\texttt{\$}\frac{\FPsv{2*k}}{\FPsv{3*k+1}}\$}} \quad \frac{\frac{1}{2}}{\frac{2}{4}}$$

`\semplix`

- `\semplix{<num>}{<den>}`: serve per scrivere delle frazioni in cui il numeratore o il denominatore contengono parametri e che potrebbero dover essere ridotte ai minimi termini, ma nelle quali il risultato pari ad 1 non deve apparire (ad esempio perché davanti ad una incognita).

Il funzionamento è identico a quello del comando `\sempli`.

Il comando può essere utilizzato ponendo il denominatore uguale ad 1 anche per la formattazione di coefficienti affinché non appaia il valore 1 e il valore -1 appaia come $-$.

Esempio 13.

Se il parametro k assume valore 2:

$$\begin{array}{rcl} \text{\texttt{\$}\FPsv{k-1}x\$} & & 1x \\ \hline \text{\texttt{\$}\semplix{k-1}\{1\}x\$} & & x \\ \hline \text{\texttt{\$}\semplix{1-k}\{1\}x\$} & & -x \\ \hline \text{\texttt{\$}\semplix{2*k}\{k+2\}x\$} & & x \end{array}$$

`\esempli`

- `\esempli{<num>}{<den>}`: serve per scrivere delle frazioni in cui il numeratore o il denominatore contengono parametri e che potrebbero dover essere ridotte ai minimi termini, ma nelle quali il risultato pari ad 1 non deve apparire, ma deve apparire esplicitamente il valore -1 (ad esempio per gli esponenti).

Il funzionamento è identico a quello del comando `\sempli`.

Il comando può essere utilizzato ponendo il denominatore uguale ad 1 anche per la formattazione di esponenti affinché non appaia il valore 1 e il valore -1 appaia come -1 .

Esempio 14.

Se il parametro k assume valore 2:

$$\begin{array}{rcl} \text{\texttt{\$}x^{\FPsv{k-1}}\$} & & x^1 \\ \hline \text{\texttt{\$}x^{\esempli{k-1}\{1\}}\$} & & x \\ \hline \text{\texttt{\$}x^{\esempli{1-k}\{1\}}\$} & & x^{-1} \end{array}$$

`\simpliz` • `\simpliz{⟨num⟩}{⟨den⟩}`: serve per scrivere delle frazioni in cui il numeratore o il denominatore contengono parametri e che potrebbero dover essere ridotte ai minimi termini e che possono assumere il valore 0 (ad esempio nelle risposte). Mentre con gli altri comandi il risultato 0 dà luogo a un errore e al blocco della compilazione, con il comando `\simpliz` viene scritto il risultato 0.

Il funzionamento è identico a quello del comando `\sempli`.

`\simplsqrt` • `\simplsqrt{⟨ind⟩}{⟨rad⟩}`: serve per portar fuori da una radice tutti i fattori possibili. I fattori portati fuori dalla radice non possono essere semplificati con altri (ad esempio, se la radice è al numeratore di una frazione, non è possibile eseguire eventuali semplificazioni tra numeratore e denominatore). Il primo parametro obbligatorio, `{⟨ind⟩}`, è l'indice della radice; il secondo, `{⟨rad⟩}`, il radicando. Il primo può contenere un numero o un parametro; il secondo un numero, un parametro o un'operazione su parametri (anche in questo caso non si deve usare all'interno `\FPSv`).

Esempio 15.

Se a vale 2 e b vale 1

`\simplsqrt{2}{a^2+4*b}` $2\sqrt{2}$

`\RandS` • `\RandS`: serve per attribuire un segno casuale (+ o −) ad un numero. Il comando è simile a quello del pacchetto `random`.

7.2 Comandi per la scelta da una lista

Oltre al comando `\esercizi` già visto esistono altri comandi per la scelta di esercizi da una lista o, più in generale, di oggetti da una lista.

`\estrai` Il primo comando permette di scegliere casualmente $n - m$ elementi di una lista. Il comando `\estrai[⟨m⟩]{⟨lista⟩}{⟨nome⟩}` è il comando utente per estrarre gli elementi dalla lista `{⟨lista⟩}` escludendone `[⟨m⟩]`. Gli elementi estratti avranno nome `\nomei`, `\nomeii`, Gli elementi della lista devono essere separati da virgole.

Esempio 16.

Con il comando `\estrai[1]{insiemi,logica,potenze}{alpha}` si scelgono 2 elementi dell'insieme e questi due elementi si chiameranno `\alpha i` e `\alpha ii`. Se questi nomi corrispondono a nomi di file di esercizi è possibile inserire nella lista di file argomento del comando `\esercizi` anche `\alpha i` e `\alpha ii`.

`\estraialfa` Il comando `\estraialfa{⟨m⟩}{⟨lista⟩}{⟨nome⟩}` estrae `{⟨m⟩}` oggetti casuali dalla lista `{⟨lista⟩}` mantenendo l'ordine fissato. Gli elementi estratti avranno nome `\nomei`, `\nomeii`, Gli elementi della lista devono essere separati da virgole.

Esempio 17.

Con il comando `\estraialfa[2]{a,b,c,d}{alpha}` si scelgono 2 elementi dell'insieme, mantenendo l'ordine alfabetico e questi due elementi si chiameranno `\alpha i` e `\alpha ii`.

`\estraies` Il comando `\estraies[⟨m⟩]{⟨lista⟩}{⟨nome⟩}` funziona come il comando `\estrai`, ma esclusivamente su una lista di esercizi. Una volta effettuata l'estrazione, li stampa come il comando `\esercizi`.

8 I file ausiliari

8.1 Il file di localizzazione

Al momento sono presenti le localizzazioni in italiano (default), inglese (UK), inglese (US), francese, spagnolo (contributo di Maria Hernandez Cifre), tedesco (contributo di Kerstin von Kirschhausen), greco (contributo di Sotiris Hasapis). Usando la lingua greca (e forse altre localizzazioni in caratteri non latini) è necessario caricare i pacchetti `fontenc`, `inputenc` e `babel` prima del caricamento del pacchetto `esami`⁵. In questo file possono venire caricati anche alcuni pacchetti dipendenti dalla localizzazione, quali `numprint`, `geometry`, ecc. e vengono definiti i comandi `\shortdate` e `\longdate` per esprimere la data.

8.2 Il file di configurazione

Il file di configurazione `esami-xyz.cfg` contiene varie definizioni tra cui la testata (comando `\testa`), il piè di pagina, la formattazione della stringa delle soluzioni, lo stile di pagina, le istruzioni per l'esame.

8.3 I file master e master-sol

Questi due file differiscono solo perché il secondo mostra le soluzioni e contengono le istruzioni per generare effettivamente la prova d'esame. **I file vanno rinominati prima della compilazione:** in caso contrario la compilazione si arresta con un messaggio di errore. In entrambi è necessario indicare la data (comando `\date`) in formato YYYY/MM/DD, il numero di compiti da generare (comando `\numcompiti`), e, ovviamente, gli esercizi. Il tema d'esame può essere diviso in più parti e in ognuna si può usare uno o più degli ambienti introdotti e uno o più comandi per la stampa degli esercizi.

Nel file possono essere definiti anche il numero massimo di ripetizioni che il comando `\FPsetpar` deve eseguire per cercare di soddisfare le condizioni poste (comando `\maxLoopLimit`) e il meccanismo di calcolo del seme iniziale per la generazione dei numeri casuali (comando `\seme`).

È anche possibile definire il nome delle parti in cui è diviso il tema d'esame.

8.4 Il file totale-versionsi

Il file `totale-versionsi` serve per generare tutte le varianti di un esercizio. Con questo file il pacchetto ha sempre l'opzione *prova* e può essere definita l'opzione *param* per generare la versione parametrica degli esercizi.

Al suo interno si deve usare un solo comando, `\def\esercizio{nome}\`, che deve contenere il nome dell'esercizio da compilare. Se si compila la versione parametrica, i parametri e i loro intervalli di variazione vengono indicati alla fine dell'esercizio.

⁵L'opzione *greek* prevede al momento l'utilizzo dell'opzione *iso-8859-7* per il pacchetto `inputenc`

Riferimenti bibliografici

- [1] Jason Alexander. The package `examdesign`. [CTAN:/tex-archive/macros/latex/contrib/examdesign](#), 2006.
- [2] Michael Mehlich. The package `fp`. [CTAN:/macros/latex/contrib/fp](#), 1999.
- [3] D. P. Story. The package `rangen`. <http://www.math.uakron.edu/~dpstory/rangen.html>, 2009.
- [4] D. P. Story. Exerquiz & AcroTeX. <http://www.acrotex.net/>, 2012.
- [5] Nicola L.C. Talbot. The package `probsoln`. [CTAN:macros/latex/contrib/probsoln](#), 2011.