The GOST package

Igor A. Kotelnikov[*]

2012/08/31, v.1.2a

Abstract

GOST is a bundle of BibTeX styles designed to meet State Standards (GOST) on information, librarianship and publishing issued by The Russian Federation and Interstate Committee of former USSR States.

It comprises 16 BibTeX styles to format bibliography in English, Russian and Ukrainian according to GOST 7.0.5-2008 and GOST 7.1-2003. Both 8-bit and Unicode (UTF-8) versions of each BibTeX style, in each case offering a choice of sorted and unsorted. Further, 2 more styles, `gost780` and `gost780s` styles (not conforming to effective standards) are retained for backwards compatibility.

1 Introduction

The package was initially developed by Maksym Polyakov. It was later updated by Igor Kotelnikov to the present status and some code was borrowed from disser package developed by Stanislav Kruchinin and unpublished work by Artem Petrenkov.

Nowdays, GOST is a bundle of BibTeX styles designed to meet State Standards (GOST) on information, librarianship and publishing issued by Russian Federation and interstate committee of former USSR States.

The System of Standards includes:

GOST 7.0.5-2008 Bibliographic reference. General requirements and rules of making.

GOST 7.1 -2003 Bibliographic record. Bibliographic description. General requirements and rules.

GOST 7.80 -2000 Bibliographic record. Heading. General requirements and rules.

GOST 7.11 -2004 Bibliographic description and references. Rules for the abbreviation of words and word combinations in foreign European languages.

GOST 7.83 -2001 Electronic editions. Basic types and imprint.

ect.

Currently, GOST contains 16 BibTeX styles to format bibliography in English, Russian and Ukrainian according to GOST 7.0.5-2008 and GOST 7.1-2003. Both 8-bit and Unicode (UTF-8)

[*] `kia999@mail.ru`

versions of each BibTeX style, in each case offering a choice of sorted and unsorted. Further, 2 styles, gost780 and gost780s styles (which do not conform to effective standards) are retained for backwards compatibility.

All styles in the GOST bundle are derived from single master file gost.dtx by applying different set of options as shown in the table below.

Style	utf8	strict	modern	eprint	long	sort	natbib
gost780							
gost780s						+	
gost2003		+		+			
gost2003s		+		+		+	
gost2008			+	+			
gost2008n			+	+			+
gost2008l			+	+	+		
gost2008s			+	+		+	
gost2008ns			+	+		+	+
gost2008ls			+	+	+	+	
ugost2003	+	+		+			
ugost2003s	+	+		+		+	
ugost2008	+		+	+			
ugost2008n	+		+	+			+
ugost2008l	+		+	+	+		
ugost2008s	+		+	+		+	
ugost2008ns	+		+	+		+	+
ugost2008ls	+		+	+	+	+	
Style	utf8	strict	modern	eprint	long	sort	natbib

Gost2008 style is recommended for most applications. It corresponds to the currently effective Standard 7.0.5-2008. Librarians should use the style gost2003 instead of gost2008 to compile a library catalog to meet the Standard 7.1-2003. Use of other styles is best explained through the meaning of options used to compile those styles from the master source.

Two styles, gost780 and gost780s, compiled without modern and strict options, are retained for backward compatibility. They do not conform the Standards 7.0.5-2008 and 7.1-2003 cited above.

The strict option provides conformance to the Standard 7.1-2003. The styles compiled with that option bear the name gost2003 with possible suffixes s, l, n as explained below. These styles are intended primarily for the librarians who compose a library catalog.

The modern option meets the Standard 7.0.5-2008 which can be thought off as a relaxed version of the Standards 7.1-2003. The styles compiled with that option bear the name gost2008 with possible suffixes s, l, n. The strict option has precedence over modern so that a style compiled with both options will resemble mainly the gost2003 style rather than gost2008.

If the number of authors exceeds 4, modern styles cut the list of authors to at most 4 persons as prescibed by the Standards. The option `long` overrides this rule to provide backward compatibility with the package disser by Stanislab Kruchinin. Two styles, `gost2008l` and `gost2008ls`, compiled with the option `long` mimic behavior of the styles `gost705` and `gost705s` from the disser package. Major effect of the `long` option is that the list of authors always precedes book or article title no matter how long is it. Modern styles compiled without `long` place long list of authors behind the title. The names of styles compiled with the option `long` has the suffix `l`. Recall that those styles do not conform effective Standards and their use is discouraged.

The `eprint` option enables formatting electronic publications. In particular, it enables `eprint`, `eprinttype`, `eprintclass`, and `doi` fields for a bibliographic entry. The styles generated without the `eprint` option, ignore the these fields. Starting from the version 1.2 of the GOST package, all modern styles are compiled with this option included, and the suffix `e` which designated this option in earlier versions is not appended to the name of style any more.

The `natbib` option provides compatibility with the `natbib` package. The names of styles compiled with the option `natbib` bear the suffix `n`. Currently 4 styles with that option are available for beta testing.

The `sort` option enables sorting bibliographic references by author names and references titles. The names of styles compiled with the option `sort` bear the suffix `s`.

Finally, the `utf8` option produces bibliographic styles in unicode rather that in 8-bit encoding. Names of those styles bear the prefix `u`.

Beyond bibliographic style, GOST bundle contains CS files (codepage and sorting order).

Encoding	CSF	Sorting order
cp866	ruscii.csf	Cyrillic first, Latin
cp1251	cp1251.csf	Cyrillic first, Latin
koi8-u	koi8u.csf	Cyrillic first, Latin
utf8	utf8cyrillic.csf	Cyrillic first, Latin

In addition, BibTeX8 distribution comes with few more CSFs.

Encoding	CSF	Sorting order
cp866	cp866rus.csf	Latin first, Cyrillic

1.1 How to use

1. Select bibliography style by adding appropriate `\bibliographystyle` declaration to your source file `<filename>.tex`, e.g.

```
\bibliographystyle{gost2008}
\bibliography{database}
```

2. Add the field `language="ukrainian"` or `language="russian"` to the bibliographic entries in Ukrainian or Russian languages in your database; English is the default language. German, Italian and French are partially supported.

3. To compile list of references from your database use `bibtex8.exe` rather than `bibtex.exe`. Depending on the codepage of your bibliographic database, indicate one of the CS files listed above as option to `bibtex8.exe`. Run LaTeX, then run `bibTeX8` and LaTeX again:

   ```
   latex <filename>.tex
   bibtex8 -B -c <csf_file>.csf <filename>.aux
   latex <filename>.tex
   ```

4. For details on preparing bibliographic database see examples in `gost*.pdf`.

5. `ugost*` styles are primarily intended for use with unicode compilers (`xelatex` and `lualatex`). They should be preferred as well when using 8bit compilers (`latex` and `pdflatex`) if source file is in utf8 encoding.

6. Neither `bitex.exe` nor `bitex8.exe` provides correct sorting order of unicode text. It means that using `ugost2008s` or `ugost2008ns` may produce unexpected result for documents in `utf8` encoding.

7. `Bibtex8` fails to change case of a string if it contains Cyrillic letter in unicode. Therefore `ugost2008*` styles do not change case of titles and other parts of bibliographic record while 8-bit styles do the case change where appropriate.

8. Either `bibtex8` or `Bibtex8` fail to cut Cyrillic names to initials. Therefore `ugost2008*` styles do not modify name of authors.

9. Package `natbib` is required when choosing styles with suffix `n` int their names.

1.2 Customization

Every GOST style defines few commands to format some parts of a reference. You can redefine these commands prior to the `\bibliography{<bibtex_style>}` command. Initial definitions are listed below.

```
\providecommand*{\url}[1]{{\small #1}}
\providecommand*{\BibUrl}[1]{\url{#1}}
\providecommand{\BibAnnote}[1]{}
\providecommand*{\BibEmph}[1]{#1}
```

By default, gost styles separate logical parts of a bibliography record by a period and cyrdash (`. "---`). It is legitimate to drop that dash by overriding the command `\BibDash` as follows

```
\providecommand*{\BibDash}{}
```

By default, `\BibDash` is equivalent to the shorthand `"---` defined by the babel package with the option `russian`. It prints a so called Cyrillic dash (`\cyrdash`), which is 20 shorter then ordinary LaTeX dash (`---`), and puts unbreakable space before `\cyrdash` so that dash never appears in the beginning of a line.

1.3 Where to get

1. CTAN:biblio/bibtex/contrib/gost.
2. CTAN:pkg/gost.

1.4 What's new in version 1.2 (2012.02.22)

1. Code refactoring. All styles are now generated from single source file.
2. Support for GOST-7.1-2003. The field `medium` is added to reflect type of material. For most entry types `medium` defaults to `text`.
3. Support for `natbib` package.
4. All modern styles are now compiled with the `eprint` option.

1.5 What's new in version 1.1 (2012.01.21)

1. Support for GOST 7.0.5-2008 and GOST 7.1-2003 is provided.
2. `@Online` entry is added to format a reference to electronic resource on Internet.
3. `@MastersThesis` entry is added to format a reference to master's thesis.
4. `@DSciThesis` entry is added to format a reference to doctor of sciences thesis.
5. `Urldate`, `eprint`, `eprintclass`, `eprinttype` fields are added.

1.6 Version history

2012.02.22 Support for `natbib` package.

2012.02.02 Adaptation to GOST 7.0.5, electronic publishing.

2005.08.12 First version uploaded to CTAN.

2003.06.06 First public version.

2 Implementation

We need Russian fonts to produce documentation of the code below. Therefore we switch current language to Russian by issuing the command `\selectlanguage{russian}`.

```
⟨*bst⟩
%%
%% This bibstyle attempts to format bibliography according to
⟨!modern⟩%% GOST 7.80-2000 for bibliographic records.
⟨modern⟩%% GOST 7.0.5-2008 for bibliographic reference.
⟨*natbib⟩%%
%%-------------------------------------------------------------------
%% This is an author-year citation style bibliography.
%% It requires a special package file to function properly
%% such as natbib.sty by Patrick W. Daly.
%% The form of the \bibitem entries is
%%   \bibitem[Jones et al.(1990)]{key}...
%%   \bibitem[Jones et al.(1990)Jones, Baker, and Smith]{key}...
%% where the label part (in brackets) consists of the author names,
%% as they should appear in the citation, with the year in parentheses following.
%% There must be no space before the opening parenthesis!
%% A full list of authors may also follow the year.
%% In natbib.sty, it is possible to define the type of enclosures that is
%% really wanted (brackets or parentheses), but in either case, there must
%% be parentheses in the label.
%% The \cite command functions as follows:
%%   \citet{key}              => Jones et al. (1990)
%%   \citet*{key}             => Jones, Baker, and Smith (1990)
%%   \citep{key}              => (Jones et al., 1990)
%%   \citep*{key}             => (Jones, Baker, and Smith, 1990)
%%   \citep[chap. 2]{key}     => (Jones et al., 1990, chap. 2)
%%   \citep[e.g.][]{key}      => (e.g. Jones et al., 1990)
%%   \citep[e.g.][p. 32]{key} => (e.g. Jones et al., p. 32)
%%   \citeauthor{key}         => Jones et al.
%%   \citeauthor*{key}        => Jones, Baker, and Smith
%%   \citeyear{key}           => 1990
%%---------------------------------------------------------------------
⟨/natbib⟩

```

2.1 Fields

Enlist all entry types allowed in a bibliographic database. Most entries are common for many standard bst styles.

```
ENTRY
  { address
    annote
    author
    booktitle
    bookauthor
    chapter
    edition
    editor
    compiler
    howpublished
    institution
    journal
    key
    month
    note
    number
    organization
    pages
    publisher
    school
    series
    title
    medium % new in v.2.
    type
    volume
    year
    language
    booklanguage
```

Remaining entries are borrowed from biblatex.

```
    numpages
    url
    urldate
    isbn
    doi
%    archive
    eprinttype   % = archivePrefix
    eprintclass  % = primaryClass
    eprint
  }
  {}
⟨!natbib⟩  { label }
⟨natbib⟩  { label extra.label sort.label short.list }

```

Declare internal variables and constants used in to format references.

```
INTEGERS { output.state before.all mid.sentence after.sentence after.block
after.dblslash after.slash after.column after.semicolumn }

```

init.state.consts

```
FUNCTION {init.state.consts}
{ #0 'before.all :=
  #1 'mid.sentence :=
  #2 'after.sentence :=
  #3 'after.block :=
  #4 'after.dblslash :=
  #5 'after.slash :=
  #6 'after.column :=
  #7 'after.semicolumn :=
}

STRINGS { s t }

STRINGS { curlanguage }

```

2.2 Formatting functions

change.language

Declare function to switch language.

```
FUNCTION {change.language}
{ booklanguage empty$
    { "" }
    { booklanguage  'curlanguage :=
      "\selectlanguageifdefined{"
      curlanguage *
      "}" *
    }
  if$
}

```

output.nonnull

Declare functions to output various parts of bibliographic record.

```
FUNCTION {output.nonnull}
{
  swap$
  output.state mid.sentence =
    { ", " * write$ }
    { output.state after.block =
        { add.period$ write$
⟨!modern⟩         " " quote$ "--- " * *  write$
⟨modern⟩          " \BibDash " write$
          newline$
          "\newblock " write$
        }
        { output.state before.all =
            'write$
            { output.state after.dblslash =
                { "~//" * change.language * " " * write$ }
                { output.state after.slash =
                    { "~/ " * write$ }
                    { output.state after.column =
⟨!(strict | modern)⟩                   { ": " * write$ }
⟨strict | modern⟩                    { "~: " * write$ }
                        { output.state after.semicolumn =
⟨!(strict | modern)⟩                       { "; " * write$ }
⟨strict | modern⟩                        { "~; " * write$ }
```

```
                        { add.period$ " " * write$ }
                       if$
                        }
                      if$
                    }
                  if$
                }
              if$
            }
          if$
        }
      if$
      mid.sentence 'output.state :=
    }
  if$
}

```

output
output.check

```
FUNCTION {output}
{ duplicate$ empty$
    'pop$
    'output.nonnull
  if$
}

FUNCTION {output.check}
{ 't :=
  duplicate$ empty$
    { pop$ "empty " t * " in " * cite$ * warning$ }
    'output.nonnull
  if$
}

```

fin.entry

fin.entry finalizes current entry. It writes dot, if no dot is found in stack, and starts new line.

```
FUNCTION {fin.entry}
{ add.period$
  write$
  newline$
}

```

new.block

Declare family of functions to put punctuation marks depending of current status of output stack. The just check output state and revert it another state if required. Checking output state prevents occasional doubling of punctuation marks.

```
FUNCTION {new.block}
{ output.state before.all =
    'skip$
    { after.block 'output.state := }
  if$
}

```

new.dblslash

```
FUNCTION {new.dblslash}
{ output.state before.all =
    'skip$
    { after.dblslash 'output.state := }
  if$
}

```

new.slash

```
FUNCTION {new.slash}
{ output.state before.all =
    'skip$
    { after.slash 'output.state := }
  if$
}

```

new.column

```
FUNCTION {new.column}
{ output.state before.all =
    'skip$
    { after.column 'output.state := }
  if$
}

```

new.semicolumn

```
FUNCTION {new.semicolumn}
{ output.state before.all =
    'skip$
    { after.semicolumn 'output.state := }
  if$
}

```

new.sentence

```
FUNCTION {new.sentence}
{ output.state after.block =
    'skip$
    { output.state before.all =
        'skip$
        { after.sentence 'output.state := }
      if$
    }
  if$
}

```

add.blank

```
FUNCTION {add.blank}
{ " " * before.all 'output.state :=
}

```

not

Declare few logical functions.

```
FUNCTION {not}
{   { #0 }
    { #1 }
  if$
}

```

and

```
FUNCTION {and}
{   'skip$
    { pop$ #0 }
  if$
}

```

or

```
FUNCTION {or}
{   { pop$ #1 }
    'skip$
  if$
}

```

chop.word

The function `chop.word` in `substr len str chop.word` removes given substring `substr` of length `len` from the beginning of the string `str`.

```
⟨*sort | natbib⟩
INTEGERS { len }

FUNCTION {chop.word}
{ 's :=
  'len :=
  s #1 len substring$ =
    { s len #1 + global.max$ substring$ }
    's
  if$
}
⟨/sort | natbib⟩

```

non.stop

```
FUNCTION {non.stop}
{ duplicate$
   "}" * add.period$
   #-1 #1 substring$ "." =
}
```

new.block.checka

```
FUNCTION {new.block.checka}
{ empty$
    'skip$
    'new.block
  if$
}

```

new.block.checkb

```
FUNCTION {new.block.checkb}
{ empty$
  swap$ empty$
  and
    'skip$
    'new.block
  if$
}

```

w.sentence.checka

```
FUNCTION {new.sentence.checka}
{ empty$
    'skip$
    'new.sentence
  if$
}

```

w.sentence.checkb

```
FUNCTION {new.sentence.checkb}
{ empty$
  swap$ empty$
  and
    'skip$
    'new.sentence
  if$
}

```

w.dblslash.checka

For `online` entry.

```
FUNCTION {new.dblslash.checka}
{ empty$
    'skip$
    'new.dblslash
  if$
}

```

field.or.null

```
FUNCTION {field.or.null}
```

```
{ duplicate$ empty$
    { pop$ "" }
    ’skip$
  if$
}

```

emphasize

Declare function to emphasize last string in stack.

```
FUNCTION {emphasize}
{ duplicate$ empty$
    { pop$ "" }
    { "\BibEmph{" swap$ * "}" * }
  if$
}

```

e.square.brackets

New in v.1.2. Declare function to enclose last word in square brackets. It do so only for the `string` option.

```
⟨*!strict⟩
FUNCTION {enclose.square.brackets} { }
⟨/!strict⟩
⟨*strict⟩
FUNCTION {enclose.square.brackets}
{ duplicate$ empty$
    { pop$ "" }
    { "[" swap$ * "]" * }
  if$
}
⟨/strict⟩


% \DescribeFunction{bracify}
% \DescribeFunction{bracketise}
% \DescribeFunction{parenthesify}
% New in v.1.2. An idea borrowed from apsrev4-1.bst.
% Declare function to enclose last word in braces, square brackets and
% parenthesis.
%    \begin{macrocode}
FUNCTION {bracify}
{ duplicate$ empty$
    { pop$ "{}" }
    { "{" swap$ * "}" * }
  if$
}
FUNCTION {bracketise}
{ duplicate$ empty$
    { pop$ "[]" }
    { "[" swap$ * "]" * }
  if$
}
FUNCTION {parenthesify}
```

```
{ duplicate$ empty$
    { pop$ "()" }
    { "(" swap$ * ")" * }
  if$
}


```

space.word

`space.word` inserts space before last string in stack.

```
FUNCTION {space.word}
{ " " swap$ * " " * }

```

2.3 Standard abbreviations

bbl.edby

Declare language-sensitive abbreviations.

```
FUNCTION {bbl.edby}    %  { "\bbledby{}" }
{ curlanguage "english" =
   {"Ed.\ by"}
   { curlanguage "ukrainian" =
⟨!utf8⟩      {"{\cyr\CYRP\cyrii\cyrd\ \cyrr\cyre\cyrd.}"}
⟨utf8⟩      {"Під ред."}
      { curlanguage "russian" =
⟨!utf8⟩         {"{\cyr\CYRP\cyro\cyrd\ \cyrr\cyre\cyrd.}"}
⟨utf8⟩         {"Под ред."}
         { curlanguage "german" =
            { "ed." }
            {"language is not defined: " language "edby" * * warning$ "Ed.\ by"}
         if$}
      if$}
   if$}
if$}

```

bbl.cmplr

```
FUNCTION {bbl.cmplr}
{ curlanguage "english" =
   { "Compiler"}
   { curlanguage "german" =
      { "Hrsg." }
      { curlanguage "ukrainian" =
⟨!utf8⟩         {"{\cyr\CYRU\cyrk\cyrl.}"}
⟨utf8⟩         {"{Укл.}"}
         { curlanguage "russian" =
⟨!utf8⟩            {"{\cyr\CYRS\cyro\cyrs\cyrt.}"}
⟨utf8⟩            {"{Сост.}"}
            {"language is not defined: " language  "cmplr" * * warning$ "Compiler"}
         if$}
      if$}
   if$}
if$}
```

bbl.edition

```
FUNCTION {bbl.edition} %  { "\bbledition{}" }
{ curlanguage "english" =
   {"edition"}
   { curlanguage "ukrainian" =
⟨!utf8⟩       {"{\cyr\cyrv\cyri\cyrd.}"}
⟨utf8⟩       {"{вид.}"}
      { curlanguage "russian" =
⟨!utf8⟩          {"{\cyr\cyri\cyrz\cyrd.}"}
⟨utf8⟩          {"{изд.}"}
         { curlanguage "german" =
            {" {aus.}" } %%%% { "Aufl." } ??
            { curlanguage "italian" =
               {"edizione"}
               { curlanguage "french" =
                  {"\'{e}dition"}
                  {"language is not defined: " language  "edition" * * warning$ "edition"}
               if$}
            if$}
         if$}
      if$}
   if$}
if$}

```

bbl.vvolume

```
FUNCTION {bbl.vvolume} %  { "\bblVolume{}" }
{ curlanguage "english" = curlanguage "french" = or curlanguage "italian" = or
   {"Volume"}
   { curlanguage "ukrainian" = curlanguage "russian" = or
⟨!utf8⟩       { "\CYRT\cyro\cyrm" }
⟨utf8⟩       { "Том" }
      { curlanguage "german" =
         {"{Band}"} %%%% { "Volumen" }
         {"language is not defined: " language  "vvolume" * * warning$ "Volume"}
      if$}
   if$}
if$}

```

bbl.vvol

```
FUNCTION {bbl.vvol}    %  { "\bblVol{}" }
{ curlanguage "english" = curlanguage "french" = or curlanguage "italian" = or
   {"Vol."}
   { curlanguage "ukrainian" = curlanguage "russian" = or
⟨!utf8⟩       {"\CYRT."}
⟨utf8⟩       {"Т."}
      { curlanguage "german" =
         {"{Bd.}"} %%%% { "Vol." }
         {"language is not defined: " language  "vvol" * * warning$ "Vol."}
      if$}
   if$}
if$}

```

bbl.iissue

```
FUNCTION {bbl.iissue}  %  { "\bblIssue{}" }
{ curlanguage "english" =
   {"Issue"}
   { curlanguage "ukrainian" =
⟨!utf8⟩      {"\CYRV\cyri\cyrp\cyru\cyrs\cyrk"}
⟨utf8⟩       {"Випуск"}
      { curlanguage "russian" =
⟨!utf8⟩         {"\CYRV\cyrery\cyrp\cyru\cyrs\cyrk"}
⟨utf8⟩          {"Выпуск"}
         { curlanguage "german" =
            {"{Heft}"} %%%% { "Ausgabe" }
            {"language is not defined: " language "iissue" * * warning$ "Issue"}
         if$}
      if$}
   if$}
if$}

```

bbl.iiss

```
FUNCTION {bbl.iiss}  %  { "\bblIss{}" }
{ curlanguage "english" =
   {"Iss."}
   { curlanguage "ukrainian" =
⟨!utf8⟩      {"\CYRV\cyri\cyrp."}
⟨utf8⟩       {"Вип."}
      { curlanguage "russian" =
⟨!utf8⟩         {"\CYRV\cyrery\cyrp."}
⟨utf8⟩          {"Вып."}
         { curlanguage "german" =
            {"{H.}"}
            {"language is not defined: " language "iiss" * * warning$ "Iss."}
         if$}
      if$}
   if$}
if$}

```

bbl.of

```
FUNCTION {bbl.of}      %  { "\bblof{}" }
{ curlanguage "english" =
   {"of"}
   { curlanguage "german" =
      { "von" }
      { curlanguage "ukrainian" =
⟨!utf8⟩         { "{\cyr\cyrii\cyrz}" }
⟨utf8⟩          { "{із}" }
```

```
         { curlanguage "russian" =
⟨!utf8⟩          { "{\cyr\cyri\cyrz}" }
⟨utf8⟩           { "{из}" }
            {"language is not defined: " language "of" * * warning$ "of"}
         if$}
      if$}
   if$}
if$}

```

bbl.etal

```
FUNCTION {bbl.etal}      %  { "\bblof{}" }
{ curlanguage "english" =
   {"et~al."}
   { curlanguage "german" =
      { "u.~a." }
      { curlanguage "ukrainian" =
⟨!utf8⟩       {"{\cyr\cyrt\cyra~\cyrii\cyrn.}"}
⟨utf8⟩        {"{та~ін.}"}
         { curlanguage "russian" =
⟨!utf8⟩          {"{\cyr\cyri~\cyrd\cyrr.}"}
⟨utf8⟩           {"и~др."}
            {"language is not defined: " language  "et~al" * * warning$ "et~al."}
         if$}
      if$}
   if$}
if$}

```

bbl.and

```
⟨*natbib⟩
FUNCTION {bbl.and}      %  { "\bbland{}" }
{ curlanguage "english" =
   {"and"}
   { curlanguage "german" =
      { "und" }
      { curlanguage "ukrainian" =
⟨!utf8⟩       {"{\cyrii}"}
⟨utf8⟩        {"і"}
         { curlanguage "russian" =
⟨!utf8⟩          {"{\cyri}"}
⟨utf8⟩           {"и"}
            {"language is not defined: " language  "and" * * warning$ "and"}
         if$}
      if$}
   if$}
if$}
⟨/natbib⟩

```

bbl.nnumber

```
FUNCTION {bbl.nnumber} %  { "\bblNumber{}" }
```

```
{ curlanguage "english" =
   {"Number"}
   { curlanguage "ukrainian" = curlanguage "russian" = or
⟨!utf8⟩      { "\CYRN\cyro\cyrm\cyre\cyrr" }
⟨utf8⟩      { "{Номер}" }
      { curlanguage "german" =
         {"{Heft}"} %%% { "Anzahl" }
         {"language is not defined: " language  "nnumber" * * warning$ "Number"}
      if$}
   if$}
if$}

```

bbl.number

```
FUNCTION {bbl.number}  %  { "\bblnumber{}" }
{ curlanguage "english" =
   {"number"}
   { curlanguage "ukrainian" = curlanguage "russian" = or
⟨!utf8⟩      {"{\cyr\cyrn\cyro\cyrm\cyre\cyrr}"}
⟨utf8⟩      {"{номер}"}
      { curlanguage "german" =
         {"{Heft}"} %%% { "anzahl" }???
         {"language is not defined: " language  "number" * * warning$ "number"}
      if$}
   if$}
if$}

```

bbl.nr

```
FUNCTION {bbl.nr}     %   { "\bblno{}" }
{ curlanguage "english" =
   {"no."}
   { curlanguage "italian" =
      { "no" }
      { curlanguage "ukrainian" = curlanguage "russian" = or
⟨!utf8⟩         { "{\cyr\textnumero}" }
⟨utf8⟩         { "{№}" }
         { curlanguage "german" =
            {"{H.}"} %%% { "an." }
            {"language is not defined: " language "nr" * * warning$ "no."}
         if$}
      if$}
   if$}
if$}

```

bbl.nnr

```
FUNCTION {bbl.nnr}     %   { "\bblno{}" }
{ curlanguage "english" =
   {"No."}
   { curlanguage "ukrainian" = curlanguage "russian" = or
⟨!utf8⟩      { "{\cyr\textnumero}" }
⟨utf8⟩       { "{№}" }
      { curlanguage "german" =
         {"{H.}"} %%% { "an." }
         {"language is not defined: " language  "nnr" * * warning$ "No."}
      if$}
   if$}
if$}

```

bbl.in

```
FUNCTION {bbl.in}    %    { "\bblin{}" }
{ curlanguage "english" = curlanguage "german" = or
   {"in"}
   { curlanguage "ukrainian" = curlanguage "russian" = or
⟨!utf8⟩       { "{\cyr\cyrv}" }
⟨utf8⟩       { "{в}" }
      {"language is not defined: " language  "in" * * warning$ "in"}
   if$}
if$}

```

bbl.iin

```
FUNCTION {bbl.iin}  %     { "\bblIn{}" }
{ curlanguage "english" = curlanguage "german" = or
   {"In"}
   { curlanguage "ukrainian" = curlanguage "russian" = or
⟨!utf8⟩       { "\CYRV" }
⟨utf8⟩       { "{В}" }
      {"language is not defined: " language  "iin" * * warning$ "In"}
   if$}
if$}

```

bbl.pages

```
FUNCTION {bbl.pages}    % { "\bblpp." }
{ curlanguage "english" = curlanguage "french" = or curlanguage "italian" = or
   {"p."} %%% {"pp."}
   { curlanguage "ukrainian" = curlanguage "russian" = or
⟨!utf8⟩       {"{\cyr\cyrs.}"}
⟨utf8⟩       {"{с.}"}
      { curlanguage "german" =
         {"S."} %%%% { "s." }
         {"language is not defined: " language  "pages" * * warning$ "p."}
      if$}
   if$}
if$}

```

bbl.page

```
FUNCTION {bbl.page}   %   { "\bblp."
{ curlanguage "english" = curlanguage "french" = or curlanguage "italian" = or
   {"p."}
```

```
  { curlanguage "ukrainian" = curlanguage "russian" = or
⟨!utf8⟩      {"{\cyr\cyrs.}"}
⟨utf8⟩      {"{c.}"}
    { curlanguage "german" =
      {"S."} %%%% { "s." }
      {"language is not defined: " language  "page" * * warning$ "p."}
    if$}
  if$}
if$}

```

bbl.ppages

```
FUNCTION {bbl.ppages}%    { "\bblPp." }
{ curlanguage "english" = curlanguage "french" = or curlanguage "italian" = or
  {"P."} %%%% { "Pp." }
  { curlanguage "ukrainian" = curlanguage "russian" = or
⟨!utf8⟩      {"{\cyr\CYRS.}"}
⟨utf8⟩      {"{C.}"}
    { curlanguage "german" =
      {"S."}
      {"language is not defined: " language "ppages" * * warning$ "P."}
    if$}
  if$}
if$}

```

bbl.ppage

```
FUNCTION {bbl.ppage}  %   { "\bblP." }
{ curlanguage "english" = curlanguage "french" = or curlanguage "italian" = or
  {"P."}
  { curlanguage "ukrainian" = curlanguage "russian" = or
⟨!utf8⟩      {"{\cyr\CYRS.}"}
⟨utf8⟩      {"{C.}"}
    { curlanguage "german" =
      {"S."}
      {"language is not defined: " language  "ppage" * * warning$ "P."}
    if$}
  if$}
if$}

```

bbl.urldate Next function was added in version 2012.01.15.

```
FUNCTION {bbl.urldate}
{ curlanguage "english" =
  {"online; accessed"}
  { curlanguage "ukrainian" =
⟨!utf8⟩      { "{\cyrd\cyra\cyrt\cyra\ \cyrz\cyrv\cyre\cyrr\cyrn\cyre\cyrn\cyrn\cyrya}" }
⟨utf8⟩      { "{дата звернення}" }
    { curlanguage "russian" =
⟨!utf8⟩        { "{\cyrd\cyra\cyrt\cyra\ \cyro\cyrb\cyrr\cyra\cyrshch\cyre\cyrn\cyri\cyrya}" }
⟨utf8⟩        { "{дата обращения}" }
      { curlanguage "german" =
            { "{online; abgerufen}" }
            { "language is not defined: " language "urldate" * * warning$ "online; accessed" }
         if$}
      if$}
   if$}
if$}

```

bbl.techreport

```
FUNCTION {bbl.techreport} % rename to bbl.techreport
{ curlanguage "english" =
   { "Rep." }
   { curlanguage "german" =
      { "Bericht" }
      { curlanguage "russian" =
⟨!utf8⟩       { "{\cyr\CYRO\cyrt\cyrch\cyre\cyrt}" }
⟨utf8⟩        { "{Отчет}" }
         { "language is not defined: " language "techrep" * * warning$ "Rep." }
      if$}
   if$}
if$}

```

bbl.mthesis

```
FUNCTION {bbl.mthesis}
{ curlanguage "english" =
   { "Master's thesis" }
   { curlanguage "german" =
      { "Diss.~Mag." }
      { curlanguage "russian" =
⟨!utf8⟩      { "{\cyr\CYRK\cyrv\cyra\cyrl\cyri\cyrf\cyri\cyrk\cyra\cyrc\cyri"
⟨!utf8⟩        "\cyro\cyrn\cyrn\cyra\cyrya\ \cyrr\cyra\cyrb\cyro\cyrt\cyra\ " *
⟨!utf8⟩        "\cyrm\cyra\cyrg\cyri\cyrs\cyrt\cyrr\cyra}" * }
⟨utf8⟩        { "{Квалификационная работа магистра}" }
         { "language is not defined: " language "mthesis" * * warning$ "Master's thesis" }
      if$}
   if$}
if$}

```

bbl.phdthesis

```
FUNCTION {bbl.phdthesis}
{ curlanguage "english" =
   { "Ph.\,D. thesis" }
   { curlanguage "german" =
      { "Diss.~Ph.\,D." }
      { curlanguage "russian" =
⟨!utf8⟩       { "{\cyr\CYRD\cyri\cyrs\cyrs\ldots\ \cyrk\cyra\cyrn\cyrd\cyri"
⟨!utf8⟩         "\cyrd\cyra\cyrt\cyra\ \cyrn\cyra\cyru\cyrk}" * }
⟨utf8⟩        { "{Дисс\ldots\ кандидата наук}" }
         { curlanguage "french" =
           { "Th\`{e}se de doctorat" }
           { "language is not defined: " language "phdthesis" * * warning$ "Ph.\,D. thesis" }
         if$}
      if$}
   if$}
if$}

```

bbl.dscithesis

```
FUNCTION {bbl.dscithesis}
{ curlanguage "english" =
   { "Dr.\,Sci. dissertation" }
   { curlanguage "german" =
      { "Diss.~Dr." }
      { curlanguage "russian" =
⟨!utf8⟩         { "{\cyr\CYRD\cyri\cyrs\cyrs\ldots\ \cyrd\cyro\cyrk\cyrt\cyro"
⟨!utf8⟩           "\cyrr\cyra\ \cyrn\cyra\cyru\cyrk}" * }
⟨utf8⟩          { "{Дисс\ldots\ доктора наук}" }
         { "language is not defined: " language "dscithesis" * * warning$ "Dr.\,Sci. dissertation" }
      if$}
   if$}
if$}

```

bbl.nnoaddress

```
FUNCTION {bbl.nnoaddress}
{ curlanguage "english" =
   { "S.\ l." }
   { curlanguage "russian" =
⟨!utf8⟩      { "{\cyr\CYRB.\ \cyrm.}" }
⟨utf8⟩       { "{Б.\ м.}" }
      { "language is not defined: " language "nnoaddress" * * warning$ "S.\ l." }
   if$}
if$}

```

bbl.nopublisher

```
FUNCTION {bbl.nopublisher}
{ curlanguage "english" =
   { "s.\ n." }
   { curlanguage "russian" =
⟨!utf8⟩      { "{\cyr\cyrb.\ \cyri.}" }
⟨utf8⟩       { "{б.\ и.}" }
      { "language is not defined: " language "nnopublisher" * * warning$ "s.\ n." }
   if$}
if$}

```

bbl.nnopublisher

```
FUNCTION {bbl.nnopublisher}
{ curlanguage "english" =
   { "S.\ n." }
   { curlanguage "russian" =
⟨!utf8⟩       { "{\cyr\CYRB.\ \cyri.}" }
⟨utf8⟩       { "{Б.\ и.}" }
      { "language is not defined: " language "nnopublisher" * * warning$ "S.\ n." }
   if$}
if$}

```

bbl.medium.text

```
FUNCTION {bbl.medium.text}
{ curlanguage "english" =
   { "Text" }
   { curlanguage "russian" = curlanguage "ukrainian" = or
⟨!utf8⟩       { "{\cyr\CYRT\cyre\cyrk\cyrs\cyrt}" }
⟨utf8⟩       { "{Текст}" }
      { "language is not defined: " language "medium" * * warning$ "Text" }
   if$}
if$}

```

bbl.medium.elres

```
FUNCTION {bbl.medium.elres}
{ curlanguage "english" =
   { "Electronic resource" }
   { curlanguage "russian" =
⟨!utf8⟩       { "{\cyr\CYREREV\cyrl\cyre\cyrk\cyrt\cyrr\cyro\cyrn\cyrn\cyrery\cyrishrt\ \cyrr\cyre\cyrs\cyru\cy
⟨utf8⟩       { "{Электронный ресурс}" }
      { curlanguage "ukrainian" =
⟨!utf8⟩         { "{\cyr\CYRE\cyrl\cyre\cyrk\cyrt\cyrr\cyro\cyrn\cyrn\cyri\cyrishrt\ \cyrr\cyre\cyrs\cyru\cyrr\
⟨utf8⟩         { "{Електронний ресурс}" }
        { "language is not defined: " language "medium" * * warning$ "Electronic resource" }
      if$}
   if$}
if$}


```

bbl.chief

```
FUNCTION {bbl.chief}
{ curlanguage "english" =
   { "chief" }
   { curlanguage "russian" =
⟨!utf8⟩       { "\cyrr\cyru\cyrk." }
⟨utf8⟩       { "{рук.}" }
      { curlanguage "ukrainian" =
⟨!utf8⟩         { "\cyrr\cyru\cyrk." }
⟨utf8⟩         { "{рук.}" }
        { "language is not defined: " language "chief" * * warning$ "chief" }
      if$}
   if$}
if$}

```

bbl.executor

```
FUNCTION {bbl.executor}
{ curlanguage "english" =
   { "Executor" }
   { curlanguage "russian" =
⟨!utf8⟩      { "{\cyr\cyri\cyrs\cyrp\cyro\cyrl\cyrn.}" }
⟨utf8⟩      { "{исполн.}" }
      { curlanguage "ukrainian" =
⟨!utf8⟩        { "{\cyr\cyrv\cyri\cyrk\cyro\cyrn\cyra\cyrv\cyre\cyrc\cyrsftsn}" }
⟨utf8⟩        { "{виконавець}" }
        { "language is not defined: " language "medium" * * warning$ "Executor" }
      if$}
   if$}
if$}

```

bbl.medium

```
FUNCTION {bbl.medium}
{ type$ "online" =
   { bbl.medium.elres }
   { bbl.medium.text  }
if$}

```

2.4 Formatting functions

Declare functions to format separate elements of bibliographic reference.

```
INTEGERS { nameptr namesleft numnames }


```

format.names

Function `format.names` has 2 version. First is for bibliographic records rather then for bibliographic references. It is used for `.bst` styles compiled without the `modern` option. It format every name as 'LastName, F. S.'. Historically, this version was used first for earlier styles included into GOST bundle.

Important note

Neither `bibtex` nor `bibtex8` can handle unicoded text without troubles. In particular, they fail to reduce a Cyrillic name to initials. Therefore we avoid using `f.` primitive when option `utf8` is in effect.

```
⟨*!modern⟩
FUNCTION {format.names}
{
⟨!utf8⟩  #1 "{vv~}{ll}{~jj}{,~f.}" format.name$
⟨utf8⟩  #1 "{vv~}{ll}{~jj}{,~ff}" format.name$
}
⟨/!modern⟩
```

Second version drops comma from output so that every name is formatted as 'LastName F. S.'. It also substitutes 4th and following names by localized term 'et al'.

```
⟨*modern⟩
```

```
FUNCTION {format.names}
{
  's :=
  #1 'nameptr :=
  s num.names$ 'numnames :=
  numnames 'namesleft :=
    { namesleft #0 > }
    { s nameptr
⟨!utf8⟩       "{vv~}{ll}{~jj}{~f.}" format.name$ 't :=
⟨utf8⟩       "{vv~}{ll}{~jj}{~ff}" format.name$ 't :=
      nameptr #1 >
        { nameptr #4 =
          numnames #4 > and
            { "others" 't :=
              #1 'namesleft := }
            'skip$
          if$
          namesleft #1 >
            { ", " * t * }
            { t "others" =
          t "~others" =
          or
⟨!strict⟩                 { " " * bbl.etal * }
⟨strict⟩                 { " " * bbl.etal enclose.square.brackets *}
                { ", " * t * }
              if$
            }
          if$
        }
        't
      if$
      nameptr #1 + 'nameptr :=
      namesleft #1 - 'namesleft :=
    }
  while$
}
⟨/modern⟩

```

`format.lab.names` Declare function to go to optional argument of `\bibitem` in the styles generated with the option `natbib`.

```
⟨*natbib⟩
FUNCTION {format.lab.names}
{ 's :=
  language empty$
    { "english" 'curlanguage := }
    { language  'curlanguage := }
  if$
  s #1 "{vv~}{ll}" format.name$
  s num.names$ duplicate$
  #2 >
```

```
    %{ pop$ " et~al." * }
    { pop$ " " bbl.etal * * }
    { #2 <
        'skip$
        { s #2 "{ff }{vv }{ll}{ jj}" format.name$ "others" =
            %{ " et~al." * }
            { " " bbl.etal * * }
            %{ " and " * s #2 "{vv~}{ll}" format.name$ * }
            { " " bbl.and " " * * * s #2 "{vv~}{ll}" format.name$ * }
          if$
        }
      if$
    }
  if$
}
⟨/natbib⟩

```

format.names.rev

Declare function to format names for authors/bookauthors list after title and etc. Note that `format.names.rev` cuts list of names to at most 4 persons. We do not cut names to initials in this list.

```
FUNCTION {format.names.rev}
{
  's :=
  #1 'nameptr :=
  s num.names$ 'numnames :=
  numnames 'namesleft :=
    { namesleft #0 > }
    { s nameptr
⟨!utf8⟩    %"{f.}{~vv}{~ll}{, jj}" format.name$ 't :=
⟨!utf8⟩    "{ff}{~vv}{~ll}{, jj}" format.name$ 't :=
⟨utf8⟩     "{ff}{~vv}{~ll}{, jj}" format.name$ 't :=
      nameptr #1 >
        { nameptr #4 =
          numnames #4 > and
            { "others" 't :=
              #1 'namesleft := }
            'skip$
          if$
          namesleft #1 >
            { ", " * t * }
            { t "others" =
          t "~others" =
          or
⟨!strict⟩             { " " * bbl.etal * }
⟨strict⟩              { " " * bbl.etal enclose.square.brackets * }
                { ", " * t * }
              if$
            }
          if$
        }
        't
      if$
      nameptr #1 + 'nameptr :=
      namesleft #1 - 'namesleft :=
    }
  while$
}

```

format.key Function to substitute empty field (usually, author name) with the key field.

```
⟨*natbib⟩
FUNCTION {format.key}
{ empty$
    { key field.or.null }
    { "" }
  if$
}
⟨/natbib⟩

```

format.authors

```
FUNCTION {format.authors}
{ author empty$
    { "" }
    { author format.names emphasize}
  if$
}

```

author.key.label

```
⟨*natbib⟩
FUNCTION {author.key.label}
{ author empty$
    { key empty$
        { cite$ #1 #3 substring$ }
        'key
      if$
    }
    { author format.lab.names }
  if$
}

FUNCTION {author.editor.key.label}
{ author empty$
    { editor empty$
        { key empty$
            { cite$ #1 #3 substring$ }
            %'key          %% causes lost of year
            { "{}" key * } %% Bug in bibtex8 ??
          if$
        }
        { editor format.lab.names }
      if$
    }
    { author format.lab.names }
  if$
}

FUNCTION {author.key.organization.label}
{ author empty$
    { key empty$
        { organization empty$
            { cite$ #1 #3 substring$ }
            { "The " #4 organization chop.word #3 text.prefix$ }
          if$
        }
        'key
      if$
    }
    { author format.lab.names }
  if$
}

FUNCTION {editor.key.organization.label}
{ editor empty$
    { key empty$
        { organization empty$
            { cite$ #1 #3 substring$ }
            { "The " #4 organization chop.word #3 text.prefix$ }
          if$
        }
        'key
      if$
    }
    { editor format.lab.names }
  if$
}

FUNCTION {calc.short.authors}
{ type$ "book" =
  type$ "inbook" =
  or
    'author.editor.key.label
    { type$ "proceedings" =
        'editor.key.organization.label
        { type$ "manual" =
            'author.key.organization.label
            'author.key.label
          if$
        }
      if$
    }
  if$
  'short.list :=
}

FUNCTION {calc.label}
{ calc.short.authors
  short.list
  "("
  *
  year duplicate$ empty$
  short.list key field.or.null = or
     { pop$ "" }
     'skip$
  if$
  *
  'label :=
}

⟨/natbib⟩

```

ormat.bookauthors

This function is used only once, in `bookauthor.before`, and the latter is used only in `inbook` entry.

```
FUNCTION {format.bookauthors}
{ bookauthor empty$
    { "" }
    { bookauthor format.names}
  if$
}

```

rmat.authors.after

```
FUNCTION {format.authors.after}
{ author empty$
    { "" }
    { author format.names.rev}
  if$
}

```

bookauthors.after

```
FUNCTION {format.bookauthors.after}
{ bookauthor empty$
    { "" }
    { bookauthor format.names.rev}% always cuts to 4 persons
  if$
}

```

rmat.editors.after

```
FUNCTION {format.editors.after}
{ editor empty$
```

```
    { "" }
    { bbl.edby "\ " * editor format.names.rev * }
  if$
}

```

ormat.chief.after

```
FUNCTION {format.chief.after}
{ editor empty$
    { "" }
    { bbl.chief "\ " * editor format.names.rev * }
  if$
}

```

at.executor.after

```
FUNCTION {format.executor.after}
{ author empty$
    { "" }
    { bbl.executor ": " * author format.names.rev * }
  if$
}

```

at.compiler.after

```
FUNCTION {format.compiler.after}
{ compiler empty$
    { "" }
    { bbl.cmplr "\ " * compiler format.names.rev * }
  if$
}

```

format.title

Important note Neither bibtex nor bibtex8 can handle unicoded text without troubles. In particular, bibtex8 fails to change case of a string if it contains Cyrillic letter. Therefore we avoid using change.case$ when option utf8 is in effect.

```
FUNCTION {format.title}
{ title empty$
    { "" }
⟨!utf8⟩    { title "t" change.case$ }
⟨utf8⟩    { title }
  if$
}

```

format.date

```
FUNCTION {format.date}
{ year empty$
    { month empty$
        { "" }
        { "there's a month but no year in " cite$ * warning$
          month
        }
      if$
    }
    { month empty$
        'year
⟨!modern⟩         { year ". " quote$ "--- " month * * * * }
⟨modern⟩         { year ". \BibDash " month * * }
      if$
    }
  if$
⟨natbib⟩  extra.label * % new in v.1.2
}

```

address.publisher

dress.publisher.date

```
⟨*!strict⟩
FUNCTION {output.address.publisher}
{
  address empty$
    'skip$
    { address output
      publisher empty$
        'skip$
        { new.column }
      if$
    }
  if$
  publisher output
}
⟨/!strict⟩
⟨*strict⟩
%FUNCTION {output.address.publisher}
%{
%  address empty$
%    {
%      bbl.nnoaddress
%      publisher empty$
%        { "~: " * bbl.nopublisher *  }
%        { }
%      if$
%      enclose.square.brackets
%    }
%    {
%      address output
%      new.column
%      publisher empty$
%        { bbl.nopublisher enclose.square.brackets }
%        { publisher }
%      if$
%    }
%  if$
%   output
%}
FUNCTION {output.address.publisher}
{
  address empty$
    {
      bbl.nnoaddress
      publisher empty$
        { "~: " * bbl.nopublisher * enclose.square.brackets }
        { enclose.square.brackets "~: " * publisher * }
      if$
    }
    {
      address output
      new.column
      publisher empty$
        { bbl.nopublisher enclose.square.brackets }
        { publisher }
      if$
    }
  if$
  output
}

⟨/strict⟩

```

dress.publisher.date `Otput.address.publisher.date` is used in old styles. New styles use `output.address.publisher`.

```
⟨*!(modern | strict)⟩
FUNCTION {output.address.publisher.date}
{
  output.address.publisher
  format.date output
}
⟨/!(modern | strict)⟩

```

output.bibitem

```
⟨*!natbib⟩
FUNCTION {output.bibitem}
{ newline$
  "\bibitem" write$
  cite$ bracify write$
  newline$
  language empty$
    { "english" 'curlanguage := }
    { language  'curlanguage := }
  if$
  "\selectlanguageifdefined" curlanguage bracify * write$
  newline$
  ""
  before.all 'output.state :=
}
⟨/!natbib⟩
```

format.full.names
author.full
editor.full
uthor.editor.full
make.full.names
output.bibitem

In case of `natbib` option, we need `make.full.names` to compose `output.bibitem`, and the latter in its turn requires some more functions.

```
⟨*natbib⟩
FUNCTION {format.full.names}
{'s :=
  language empty$
    { "english" 'curlanguage := }
    { language  'curlanguage := }
  if$
  #1 'nameptr :=
  s num.names$ 'numnames :=
  numnames 'namesleft :=
    { namesleft #0 > }
    { s nameptr
      "{vv~}{ll}" format.name$ 't :=
      nameptr #1 >
        {
          namesleft #1 >
            { ", " * t * }
            {
              numnames #2 >
              curlanguage "english" =
              and
                { "," * }
                'skip$
              if$
              t "others" =
          %t "~others" =
          %or
                %{ " et~al." * }
                { " " bbl.etal * * }
                %{ " and " * t * }
                { " " bbl.and " " * * * t * }
              if$
            }
          if$
        }
        't
      if$
      nameptr #1 + 'nameptr :=
      namesleft #1 - 'namesleft :=
    }
  while$
}

FUNCTION {author.full}
{ author empty$
```

```
    { "" }
    { author format.full.names }
  if$
}

FUNCTION {editor.full}
{ editor empty$
    { "" }
    { editor format.full.names }
  if$
}

FUNCTION {author.editor.full}
{ author empty$
    { editor empty$
        { "" }
        { editor format.full.names }
      if$
    }
    { author format.full.names }
  if$
}

FUNCTION {make.full.names}
{ type$ "book" =
  type$ "inbook" =
  or
    'author.editor.full
    { type$ "proceedings" =
        'editor.full
        'author.full
      if$
    }
  if$
}

% ========================================
FUNCTION {output.bibitem}
{ newline$
  "\bibitem[" write$
  label write$
  ")" make.full.names duplicate$ short.list =
     { pop$ }
     { * }
   if$
  "]{" * write$
  cite$ write$
  "}" write$
  language empty$
    { "english" 'curlanguage := }
    {language  ’curlanguage := }
  if$
  "\selectlanguageifdefined" curlanguage bracify * write$
  newline$
  ""
  before.all ’output.state :=
}
% ========================================
%FUNCTION {output.bibitem}
%{ newline$
%  "\bibitem" write$
%%  author.key.label
%%  year parenthesify *
%%  "; lbl:" label * *
%%  "; mfn:"  make.full.names * *
%  label
%  make.full.names *
%  bracketise write$
%  cite$ bracify write$
%  newline$
%  language empty$
%    { "english" ’curlanguage := }
%    {language  ’curlanguage := }
%  if$
%  "\selectlanguageifdefined" curlanguage bracify * write$
%  newline$
%  ""
%  before.all ’output.state :=
%}
% ========================================
⟨/natbib⟩

```

n.dashify

```
FUNCTION {n.dashify}
{ ’t :=
  ""
    { t empty$ not }
    { t #1 #1 substring$ "-" =
        { t #1 #2 substring$ "--" = not
            { "--" *
              t #2 global.max$ substring$ ’t :=
            }
            {   { t #1 #1 substring$ "-" = }
                { "-" *
                  t #2 global.max$ substring$ ’t :=
                }
              while$
            }
          if$
        }
        { t #1 #1 substring$ *
          t #2 global.max$ substring$ 't :=
        }
      if$
    }
  while$
}

```

word.in

```
FUNCTION {word.in}
{ bbl.iin
  " " * }

```

format.btitle

```
FUNCTION {format.btitle}
{ title
}

```

.or.space.connect

```
FUNCTION {tie.or.space.connect}
{ duplicate$ text.length$ #3 <
    { "~" }
    { " " }
  if$
  swap$ * *
}

```

tie.connect Declare function to insert unbreakable space before last word in stack.

```
FUNCTION {tie.connect}
 {"~"
  swap$ * *
 }


```

either.or.chec

```
FUNCTION {either.or.check}
{ empty$
    'pop$
    { "can't use both " swap$ * " fields in " * cite$ * warning$ }
  if$
}

```

format.bvolume

```
FUNCTION {format.bvolume}
{ volume empty$
    { "" }
    { bbl.vvol volume tie.connect
```

```
   series empty$
    'skip$
    { bbl.of space.word * series emphasize * }
   if$
   "volume and number" number either.or.check
  }
 if$
}

```

rmat.number.series

```
FUNCTION {format.number.series}
{ volume empty$
  { number empty$
    { series field.or.null }
    { series empty$
      { "there's a number but no series in " cite$ * warning$
       bbl.nnr }
      {
       %new.dblslash
       new.sentence
       series
       bbl.nr
       tie.or.space.connect}
     if$
     number tie.or.space.connect
    }
   if$
  }
  { "" }
 if$
}

```

is.num

```
FUNCTION {is.num}
{ chr.to.int$
 duplicate$ "0" chr.to.int$ < not
 swap$ "9" chr.to.int$ > not and
}

```

extract.num

```
FUNCTION {extract.num}
{ duplicate$ 't :=
 "" 's :=
 { t empty$ not }
 { t #1 #1 substring$
  t #2 global.max$ substring$ 't :=
  duplicate$ is.num
   { s swap$ * 's := }
   { pop$ "" 't := }
```

```
    if$
  }
  while$
  s empty$
    'skip$
    { pop$ s }
  if$
}

⟨*debug⟩
```

eng.ord

```
FUNCTION {eng.ord}
{ duplicate$ "1" swap$ *
  #-2 #1 substring$ "1" =
     { bbl.th * }
     { duplicate$ #-1 #1 substring$
       duplicate$ "1" =
         { pop$ bbl.st * }
         { duplicate$ "2" =
             { pop$ bbl.nd * }
             { "3" =
                 { bbl.rd * }
                 { bbl.th * }
               if$
             }
           if$
          }
       if$
     }
   if$
}
⟨/debug⟩

```

convert.edition

```
FUNCTION {convert.edition}
{ edition
% edition extract.num "l" change.case$ 's :=
%  s "first" = s "1" = or
%    { bbl.first 't := }
%    { s "second" = s "2" = or
%        { bbl.second 't := }
%        { s "third" = s "3" = or
%            { bbl.third 't := }
%            { s "fourth" = s "4" = or
%                { bbl.fourth 't := }
%                { s "fifth" = s "5" = or
%                    { bbl.fifth 't := }
%                    { s #1 #1 substring$ is.num
%                        { s eng.ord 't := }
%                        { edition ’t := }
%                      if$
%                    }
%                  if$
%                }
%              if$
%            }
%          if$
%        }
%      if$
%    }
%  if$
%  t
}

```

format.edition

```
FUNCTION {format.edition}
{ edition empty$
    { "" }
    { output.state mid.sentence =
⟨!utf8⟩        { convert.edition "l" change.case$ " " * bbl.edition * }
⟨!utf8⟩        { convert.edition "t" change.case$ " " * bbl.edition * }
⟨utf8⟩        { convert.edition " " * bbl.edition * }
⟨utf8⟩        { convert.edition " " * bbl.edition * }
      if$
    }
  if$
}

INTEGERS { multiresult }

```

multi.page.check

```
FUNCTION {multi.page.check}
{ ’t :=
  #0 ’multiresult :=
    { multiresult not
      t empty$ not
      and
    }
    { t #1 #1 substring$
      duplicate$ "-" =
      swap$ duplicate$ "," =
      swap$ "+" =
      or or
        { #1 ’multiresult := }
        { t #2 global.max$ substring$ ’t := }
      if$
    }
  while$
  multiresult
}

```

format.pages

```
FUNCTION {format.pages}
{ pages empty$
    { "" }
    { pages multi.page.check
        { bbl.ppages pages n.dashify tie.connect }
        { bbl.ppage pages tie.connect }
      if$
    }
  if$
}

```

format.pages.page

```
FUNCTION {format.pages.page}
{ pages empty$
    { numpages empty$
      { "" }
      { numpages bbl.pages tie.connect }
    if$}
    { format.pages}
  if$
}

```

rmat.vol.num.pages

```
FUNCTION {format.vol.num.pages}
{ volume field.or.null
  number empty$
    'skip$
    {
      ", no." number tie.or.space.connect *
      volume empty$
        { "there's a number but no volume in " cite$ * warning$ }
        'skip$
      if$
    }
  if$
  pages empty$
    'skip$
    { duplicate$ empty$
        { pop$ format.pages }
        { ": " * pages n.dashify * }
      if$
    }
  if$
}

```

format.volume

```
FUNCTION {format.volume}
{ volume empty$
    { "" }
    { bbl.vvol volume tie.or.space.connect }
  if$
}

```

format.number

```
FUNCTION {format.number}
{ number empty$
    { "" }
    { bbl.nr number tie.or.space.connect }
  if$
}

⟨*debug⟩
```

rmat.chapter.pages

```
FUNCTION {format.chapter.pages}
{ chapter empty$
    'format.pages
    { type empty$
        { bbl.chapter }
        { type "l" change.case$ }
      if$
      chapter tie.or.space.connect
      pages empty$
        'skip$
        { ", " * format.pages * }
      if$
    }
  if$
}
⟨/debug⟩

```

empty.misc.check

```
FUNCTION {empty.misc.check}
{ author empty$ title empty$ howpublished empty$
  month empty$ year empty$ note empty$
  and and and and and
  key empty$ not and
    { "all relevant fields are empty in " cite$ * warning$ }
    'skip$
  if$
}

```

ormat.thesis.type

```
FUNCTION {format.thesis.type}
```

```
{ type empty$
    'skip$
    { pop$
⟨!utf8⟩      type "t" change.case$
⟨utf8⟩       type
    }
  if$
}

```

chrep.type.number

Function to format report type and number.

```
%FUNCTION {format.techrep.type.number}
%{ type empty$
%    { bbl.techreport }
%    'type
%  if$
%  number empty$
%%<!utf8>    { "t" change.case$ }
%%<utf8>    { "" }
%    { number tie.or.space.connect }
%  if$
%}

FUNCTION {format.techreport.type}
{ type empty$
   { bbl.techreport }
   'type
  if$
}

```

author.before

Declare the function `author.before` to format list of authors in heading of a bibliographic record. If the number of authors is 4 or larger, some styles skip the list of authors in the beginning of the bibliographic record, while other styles always print that list. So, we need two version of `author.before`.

First version is used if `.bst` style is compiled without option `long`. It skips authors if their number is greater than or equal to 4 or if the author field is empty. Note that GOST requires for a long list of authors to be reduced. Hence this first version is preferable. Note also that `format.names` cuts list of names to 4 person at most in case if `modern` option is used. and `format.authors` just emphasizes `format.names`.

```
⟨*!long⟩
FUNCTION {author.before}
{
  author empty$
    'skip$
    {author num.names$ #4 <
      {format.authors output
       new.sentence}
      'skip$
    if$}
  if$
}
⟨/!long⟩
```

Second version is used if `.bst` style is compiled with the option `long`. It skips only if the author field is empty.

```
⟨*long⟩
FUNCTION {author.before}
{
  author empty$
    'skip$
    { format.authors output
      new.sentence
    }
  if$
}
⟨/long⟩

```

`bookauthor.before`

There are also 2 version of the function `bookauthor.before`. Not used anymore!

```
%%<*!long>
%%FUNCTION {bookauthor.before}
%%{
%%  bookauthor empty$
%%    'skip$
%%    {bookauthor num.names$ #4 <
%%      {format.bookauthors output
%%       new.sentence}
%%      'skip$
%%    if$}
%%  if$
%%}
%%</!long>
%%<*long>
%%FUNCTION {bookauthor.before}
%%{
%%  bookauthor empty$
%%    'skip$
%%    { format.bookauthors output
%%      new.sentence
%%    }
%%  if$
%%}
%%</long>

```

`author.after`

Functions `author.after` and `bookauthor.after` also have by 2 versions. They are used to write authors list after the title followed by a slash. In modern styles, compiled with option `modern`, the list of authors is always cut to at most 4 persons. The cut is performed first by `format.names.rev`, which is called by `format.authors.after`. For old styles, `author.after` just outputs formatted string whereas for new style it skips the string if the number of authors exceeds 3 (and author list is not printed before the title).

```
⟨*!modern⟩
FUNCTION {author.after}
{
  author empty$
    'skip$
    {format.authors.after output
       new.semicolumn }
  if$
}
⟨/!modern⟩
⟨*modern⟩
⟨*!long⟩
FUNCTION {author.after}
{
  author empty$
    'skip$
    {author num.names$ #3 >
      {format.authors.after output
       new.semicolumn }
      'skip$
    if$}
  if$
}
⟨/!long⟩
⟨*long⟩
FUNCTION {author.after} { }
⟨/long⟩
⟨/modern⟩

```

`bookauthor.after` This function is used only in `inbook` entry. It always cuts list to 4 persons since `format.bookauthors.after` does that.

```
FUNCTION {bookauthor.after}
{
  bookauthor empty$
    'skip$
    {format.bookauthors.after output
       new.semicolumn }
  if$
}

```

`rganization.after`

```
FUNCTION {editor.organization.after}
{
  compiler empty$
    {}
    { format.compiler.after  output
    new.semicolumn
    }
  if$
  editor empty$
    {}
    { format.editors.after  output
    new.semicolumn
    }
  if$
  organization empty$
    {}
    {organization output
    new.semicolumn
    }
  if$
}

```

format.url

```
FUNCTION {format.url}
{ url empty$
    { "" }
    {
⟨!(modern | strict)⟩        "\BibUrl{ " url * "}" *
⟨modern | strict⟩       "URL: \BibUrl{" url * "}" *
      urldate empty$
        { "" }
        { " (" bbl.urldate * ": " * urldate * ")" * }
      if$ *
    }
  if$
}

```

format.annote

```
FUNCTION {format.annote}
{ annote empty$
    { "" }
{ after.sentence 'output.state :=
  "\BibAnnote{" annote add.period$ * "}" *
}
  if$
}

```

format.isbn Do we really need to provide electronic search for ISNB?

```
FUNCTION {format.isbn}
{
  isbn empty$
    { "" }
    { "ISBN:~\href{http://isbndb.com/search-all.html?kw=" isbn *
      "}{" * isbn * "}" *
    }
  if$
}

```

add.doi

The Digital Object Identifier (DOI) System is for identifying content objects in the digital environment. DOI names are assigned to any entity for use on digital networks. They are used to provide current information, including where they (or information about them) can be found on the Internet. Information about a digital object may change over time, including where to find it, but its DOI name will not change.

Function `add.doi` embraces last string in stack into hyperlink that links it to specified `doi` identificator at http://dx.doi.org/ web-site.

```
⟨*eprint⟩
FUNCTION {add.doi}
{ duplicate$ empty$
    'skip$
    { doi empty$
        'skip$
        { "\href{http://dx.doi.org/" doi * "}{" * swap$ * "}" * }
      if$
    }
  if$
}
⟨/eprint⟩
```

If `.bst` style is compiled without `eprint` option, we just ignore `doi` field.

```
⟨*!eprint⟩
FUNCTION {add.doi} { }
⟨/!eprint⟩

```

add.medium

New in version 2.

```
⟨*!strict⟩
FUNCTION {add.medium} { }
⟨/!strict⟩
⟨*strict⟩
FUNCTION {add.medium}
{ duplicate$ empty$
    'skip$
    { medium empty$
        { " " * bbl.medium enclose.square.brackets * }
        { " " * medium enclose.square.brackets * }
%%        { bbl.medium enclose.square.brackets * }
%%        { medium enclose.square.brackets * }
      if$
    }
  if$
}
⟨/strict⟩

```

2.5 Electronic Publishing Information

The biblatex package provides three fields for electronic publishing information: `eprint`, `eprinttype`, and `eprintclass`. The `eprint` field is a verbatim field similar to `doi` which holds the identifier of the item. The `eprinttype` field holds the resource name, i. e., the name of the site or electronic archive. The optional `eprintclass` field is intended for additional information specific to the resource indicated by the `eprinttype` field. This could be a section, a path, classification information, etc. If the `eprinttype` field is available, the standard styles will use it as a literal label. In the following example, they would print “Resource: identifier” rather than the generic “eprint: identifier”:

```
eprint = {identifier},
eprinttype = {Resource},
```

`format.eprint` The electronic identifier of an online publication. This is roughly comparable to a `doi` but specific to a certain archive, repository, service, or system. Also see eprinttype and eprintclass.

This function should use url. TO BE DONE YET.

```
⟨*eprint⟩
%FUNCTION {format.eprint}
%{ eprint empty$
%   { "" }
%   { eprintclass empty$
%      { " \href{http://arxiv.org/abs/" eprint * "}" * "{" * eprint * "}" * }
%      { eprinttype empty$
%         { " \href{http://arxiv.org/abs/" eprint * "}" *
%          "{" * eprintclass * "/" * eprint * "}" *
%         }
%         { " \href{http://arxiv.org/abs/" eprint * "}" *
%          "{" * eprinttype * ":" * eprintclass * "/" * eprint * "}" *
%         }
%      if$}
%   if$}
%if$}

%FUNCTION {format.eprint}
%{ eprint empty$
%   { "" }
%   { eprinttype empty$
%       { "" }
%       { eprinttype "~: " *}
%     if$
%     eprintclass empty$
%       { }
%       { eprintclass * "/" *}
%    if$
%    eprint *
%   }
%  if$
%  url empty$
%    { }
%    { "\href{" url * "}{" * swap$ * "}" *}
%  if$
%}

FUNCTION {format.eprint}
{ eprint empty$
   { "" }
   { eprinttype empty$
       { "" }
       { eprinttype "~: " *}
     if$
     eprintclass empty$
       { }
       { eprintclass * "/" *}
     if$
     url empty$
      { eprint * }
      { "\href{" * url * "}{" * eprint * "}" *}
    if$
   }
  if$
}

FUNCTION {output.eprint.url}
{
  eprint empty$
    { format.url output }
    { format.eprint output }
  if$
}

⟨/eprint⟩

⟨*!eprint⟩
FUNCTION {output.eprint.url}
{
    format.url output
}
⟨/!eprint⟩

```

2.6 Entry types

Text below in this section is borrowed from biblatex manual. Not every field listed below is actually supported by GOST styles. So description below should be considered as a goal or a feature request.

The lists below indicate the fields supported by each entry type. Note that the mapping of fields to an entry type is ultimately at the discretion of the bibliography style. The lists below therefore serve two purposes. They indicate the fields supported by the standard styles which ship with this package and they also serve as a model for custom styles. Note that the required fields are not strictly required in all cases. The fields marked as optional are optional in a technical sense. Bibliographical formatting rules usually require more than just the required fields. The standard styles will generally not perform any formal validity checks, but custom styles may do so. Generic fields like abstract and annotation or label and shorthand are not included in the lists below because they are independent of the entry type.

2.6.1 Regular Types

`article` An article in a journal, magazine, newspaper, or other periodical which forms a self-contained unit with its own title. The title of the periodical is given in the journaltitle field. If the issue has its own title in addition to the main title of the periodical, it goes in the issuetitle field. Note that editor and related fields refer to the journal while translator and related fields refer to the article.

Required fields: author, title, journaltitle, year/date.

Optional fields: translator, annotator, commentator, subtitle, titleaddon, editor, editora, editorb, editorc, journalsubtitle, issuetitle, issuesubtitle, language, origlanguage, series, volume, number, eid, issue, month, pages, version, note, issn, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate.

```
FUNCTION {article}
{
  output.bibitem
  author.before
⟨natbib⟩  author format.key output
  format.title add.medium "title" output.check
  new.slash
  author.after
  new.dblslash
  journal emphasize add.doi "journal" output.check % new in v.2
  new.block
  format.date "year" output.check
  new.block
  format.volume output
  format.number output
  new.block
  format.pages.page output
  new.block
  note output
  new.sentence
%  format.url output
  output.eprint.url
  format.annote output
  fin.entry
}

```

`book` A single-volume book with one or more authors where the authors share credit for the work as a whole. In biblatex, this entry type also covers the function of the `@inbook` type of traditional BibTeX.

Required fields: author, title, year/date.

Optional fields: editor, editora, editorb, editorc, translator, annotator, commentator, introduction, foreword, afterword, subtitle, titleaddon, maintitle, mainsubtitle, maintitleaddon, language, origlanguage, volume, part, edition, volumes, series, number, note, publisher, location, isbn, chapter, pages, pagetotal, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate.

```
FUNCTION {book}
{
  output.bibitem
  author.before
⟨natbib⟩  author format.key output
  format.btitle add.doi add.medium "title" output.check
  new.slash
  author.after
  editor.organization.after
  new.sentence
  format.number.series output
  new.block
  format.edition output
  new.block
  output.address.publisher
  format.date "year" output.check
  new.block
  format.bvolume output
  new.block
  format.pages.page output
  new.block
⟨eprint⟩  format.isbn output
⟨eprint⟩  new.block
  note output
  new.sentence
%  format.url output
  output.eprint.url
  format.annote output
  fin.entry
}

```

`booklet` A book-like work without a formal publisher or sponsoring institution. Use the field howpublished to supply publishing information in free format, if applicable. The field type may be useful as well.

Required fields: author/editor, title, year/date.

Optional fields: subtitle, titleaddon, language, howpublished, type, note, location, chapter, pages, pagetotal, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate.

```
FUNCTION {booklet}
{
  output.bibitem
  author.before
⟨natbib⟩  author format.key output
  format.title add.doi add.medium "title" output.check
  new.slash
  author.after
  editor.organization.after
  new.block
  howpublished output
  address output
  format.date "year" output.check
  new.block
  note output
  new.sentence
%  format.url output
  output.eprint.url
  format.annote output
  fin.entry
}

```

inbook A part of a book which forms a self-contained unit with its own title. Note that the profile of this entry type is different from standard BibTeX.

Required fields: author, title, booktitle, year/date.

Optional fields: bookauthor, editor, editora, editorb, editorc, translator, annotator, commentator, introduction, foreword, afterword, subtitle, titleaddon, maintitle, mainsubtitle, maintitleaddon, booksubtitle, booktitleaddon, language, origlanguage, volume, part, edition, volumes, series, number, note, publisher, location, isbn, chapter, pages, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate.

```
FUNCTION {inbook}
{
  output.bibitem
  author.before
⟨natbib⟩  author format.key output
  format.btitle add.doi add.medium "title" output.check
  new.slash
  author.after
  new.dblslash
%  bookauthor.before
  booktitle "booktitle" output.check
  new.slash
  bookauthor.after
  editor.organization.after
  new.block
  format.edition output
  new.block
  format.number.series output
  new.sentence
  output.address.publisher
  format.date "year" output.check
  new.block
  format.bvolume output
  new.block
  format.pages.page output
  new.block
⟨eprint⟩  format.isbn output
⟨eprint⟩  new.block
  note output
  new.sentence
%  format.url output
  output.eprint.url
  format.annote output
  fin.entry
}

```

incollection A contribution to a collection which forms a self-contained unit with a distinct author and title. The author refers to the title, the editor to the booktitle, i. e., the title of the collection.

Required fields: author, editor, title, booktitle, year/date.

Optional fields: editora, editorb, editorc, translator, annotator, commentator, introduction, foreword, afterword, subtitle, titleaddon, maintitle, mainsubtitle, maintitleaddon, booksubtitle, booktitleaddon, language, origlanguage, volume, part, edition, volumes, series, number, note, publisher, location, isbn, chapter, pages, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate.

```
FUNCTION {incollection}
{
  output.bibitem
  author.before
⟨natbib⟩  author format.key output
  new.sentence
  format.title add.doi add.medium "title" output.check
  new.slash
  author.after
  new.dblslash
  booktitle "booktitle" output.check
  new.slash
  editor.organization.after
  new.block
  output.address.publisher
  format.date "year" output.check
  new.block
  format.bvolume output
  format.number.series output
  new.block
  format.pages.page output
  new.block
  note output
  new.sentence
%  format.url output
  output.eprint.url
  format.annote output
  fin.entry
}

```

proceedings A single-volume conference proceedings. This type is very similar to @collection. It supports an optional organization field which holds the sponsoring institution. The editor is omissible.

Required fields: editor, title, year/date.

Optional fields: subtitle, titleaddon, maintitle, mainsubtitle, maintitleaddon, eventtitle, eventdate, venue, language, volume, part, volumes, series, number, note, organization, publisher, location, month, isbn, chapter, pages, pagetotal, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate.

```
FUNCTION {proceedings}
{
  output.bibitem
⟨natbib⟩  editor format.key output
  format.btitle add.doi add.medium "title" output.check
  new.slash
  editor.organization.after
  new.block
  format.bvolume output
  format.number.series output
%  address empty$
%    { publisher output
%      format.date "year" output.check
%    }
%    { address output.nonnull
%      format.date "year" output.check
%      new.sentence
%      publisher output
%    }
%  if$
  output.address.publisher
  format.date "year" output.check
  new.block
  note output
  new.sentence
%  format.url output
  output.eprint.url
  format.annote output
  fin.entry
}

```

`inproceedings` An article in a conference proceedings. This type is similar to @incollection. It supports an optional organization field.

Required fields: author, editor, title, booktitle, year/date.

Optional fields: subtitle, titleaddon, maintitle, mainsubtitle, maintitleaddon, booksubtitle, booktitleaddon, eventtitle, eventdate, venue, language, volume, part, volumes, series, number, note, organization, publisher, location, month, isbn, chapter, pages, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate.

```
FUNCTION {inproceedings}
{ output.bibitem
  author.before
⟨natbib⟩  author format.key output
  new.sentence
  format.title add.doi add.medium "title" output.check
  new.slash
  author.after
  new.dblslash
  booktitle "booktitle" output.check
  new.slash
  editor.organization.after
  new.block
  format.bvolume output
  format.number.series output
  new.block
%  address empty$
%    { publisher output
%      format.date "year" output.check
%    }
%    { address output.nonnull
%      new.column
%      publisher output
%      format.date "year" output.check
%    }
%  if$
  output.address.publisher
  format.date "year" output.check
  new.block
  format.pages.page output
  new.block
  note output
  new.sentence
%  format.url output
  output.eprint.url
  format.annote output
  fin.entry
}

```

`manual` Technical or other documentation, not necessarily in printed form. The author or editor is omissible.

Required fields: author/editor, title, year/date.

Optional fields: subtitle, titleaddon, language, edition, type, series, number, version, note, organization, publisher, location, isbn, chapter, pages, pagetotal, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate.

```
FUNCTION {manual}
{ output.bibitem
  author empty$
    { organization empty$
        ’skip$
        { organization output.nonnull
          address output
        }
      if$
    }
    { format.authors output.nonnull }
  if$
⟨natbib⟩  author format.key output
  new.block
  format.btitle add.doi add.medium "title" output.check
  author empty$
    { organization empty$
    {
          address new.block.checka
          address output
        }
        'skip$
      if$
    }
    {
      organization address new.block.checkb
      organization output
      address output
    }
  if$
  format.edition output
  format.date "year" output.check
  new.block
  note output
  new.sentence
%  format.url output
  output.eprint.url
  format.annote output
  fin.entry
}

```

misc A fallback type for entries which do not fit into any other category. Use the field howpublished to supply publishing information in free format, if applicable. The field type may be useful as well. author, editor, and year are omissible.

Required fields: author/editor, title, year/date.

```
FUNCTION {misc}
{ output.bibitem
  format.authors output
⟨natbib⟩  author format.key output
  title howpublished new.sentence.checkb
  format.title add.medium output
  howpublished new.block.checka
  howpublished output
  new.block
  format.date "year" output.check
  new.block
  note output
  new.sentence
%  format.url output
  output.eprint.url
  format.annote output
  fin.entry
}

```

unpublished A work with an author and a title which has not been formally published, such as a manuscript or the script of a talk. Use the fields howpublished and note to supply additional information in free format, if applicable.

Required fields: author, title, year/date.

Optional fields: subtitle, titleaddon, language, howpublished, note, location, isbn, date, month, year, addendum, pubstate, url, urldate

```
FUNCTION {unpublished}
{ output.bibitem
  author.before
⟨natbib⟩  author format.key output
  format.btitle "title" output.check
  new.slash
  author.after
  editor.organization.after
  new.block
  format.date "year" output.check
  new.block
  note "note" output.check
  new.sentence
%  format.url output
  output.eprint.url
  format.annote output
  fin.entry
}

```

online An online resource. Author, editor, and year are omissible. This entry type is intended for sources such as web sites which are intrinsically online resources. Note that all entry types support the url field. For example, when adding an article from an online journal, it may be preferable to use the `@article` type and its url field.

Required fields: author/editor, title, year/date, url.

Optional fields: subtitle, titleaddon, language, version, note, organization, date, month, year, addendum, pubstate, urldate.

```
FUNCTION {online}
{ output.bibitem
  format.authors output
⟨natbib⟩  author format.key output
  title howpublished new.sentence.checkb
  format.title add.doi add.medium "title" output.check
%  howpublished new.block.checka
  howpublished new.dblslash.checka
  howpublished enclose.square.brackets output
  editor.organization.after
  new.sentence
  new.block
  output.address.publisher
  format.date output
  new.block
%  format.url output
  output.eprint.url
  new.sentence
  note output
  format.annote output
  fin.entry
}

```

internet

New in version 2012.02.15.

www webpage ielectronic

```
FUNCTION {internet}   {online}
FUNCTION {www}        {online}
FUNCTION {webpage}    {online}
FUNCTION {electronic} {online}
```

thesis

New in version 2012.02.02.

A thesis written for an educational institution to satisfy the requirements for a degree. Use the type field to specify the type of thesis.

Required fields: author, title, type, institution, year/date.

Optional fields: subtitle, titleaddon, language, note, location, month, isbn, chapter, pages, pagetotal, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate

```
⟨*!(modern | strict)⟩
FUNCTION {thesis}
{ output.bibitem
  format.authors "author" output.check
⟨natbib⟩  author format.key output
  new.sentence
  format.btitle "title" output.check
  new.column
  bbl.phdthesis format.thesis.type output.nonnull
  new.slash
  school "school" output.check
  new.block
  output.address.publisher.date
  new.block
  format.pages.page output
  note output
  new.sentence
  format.url output
  format.annote output
  fin.entry
}
⟨/!(modern | strict)⟩
⟨*modern | strict⟩
FUNCTION {thesis}
{ output.bibitem
  format.authors "author" output.check
⟨natbib⟩  author format.key output
  new.sentence
  format.btitle add.doi add.medium "title" output.check
  new.column
%  bbl.phdthesis format.thesis.type output.nonnull
  type "type" output.check
  new.column
  number output
  new.slash
  format.authors.after output
  new.semicolumn
  school "school" output.check
  new.block
  output.address.publisher
  format.date "year" output.check
  new.block
  format.pages.page output
  new.block
  note output
  new.sentence
%  format.url output
  output.eprint.url
  format.annote output
  fin.entry
}
⟨/modern | strict⟩

```

`report` New in version 2012.02.02.

A technical report, research report, or white paper published by a university or some other institution. Use the type field to specify the type of report. The sponsoring institution goes in the institution field.

Required fields: author, title, type, institution, year/date.

Optional fields: subtitle, titleaddon, language, number, version, note, location, month, isrn, chapter, pages, pagetotal, addendum, pubstate, doi, eprint, eprintclass, eprinttype, url, urldate.

```
%FUNCTION {report}
%{
%  output.bibitem
%  author.before
%  new.sentence
%  format.title add.doi add.medium "title" output.check
%  new.column
%%  format.techrep.type.number output.nonnull
%  type "type" output.check
%  new.slash
%  author.after
%  editor.organization.after
%  new.block
%  address output
%  new.column
%  institution "institution" output.check
%  format.date "year" output.check
%  new.block
%  note output
%  new.block                % v.2
%  format.pages.page output % v.2
%  new.sentence
%%  format.url output
%  output.eprint.url
%  format.annote output
%  fin.entry
%}
FUNCTION {report}
{
  output.bibitem
%  author.before
%  new.sentence
  format.title add.doi add.medium "title" output.check
⟨natbib⟩  title format.key output
  new.column
%  format.techrep.type.number output.nonnull
%  type "type" output.check
%  format.report.type.number "type" output.check
  type "type" output.check
  new.column
  number output
  new.slash
  institution "institution" output.check
  new.semicolumn
  format.chief.after output % from editor field
  new.semicolumn
  format.executor.after output % from author field
  new.block
  address output
  new.column
  organization output
  format.date "year" output.check
  new.block                % v.2
  format.pages.page output % v.2
  new.sentence % или new.block ?
  output.eprint.url
  new.block
  note output
  format.annote output
  fin.entry
}

```

2.6.2 Type Aliases

The entry types listed in this section are provided for backwards compatibility with traditional BibTeX styles. These aliases are resolved by BibTeX as the data is exported. Bibliography styles will see the entry type the alias points to, not the alias name. All unknown entry types are generally exported as @misc.

phdthesis Similar to @thesis except that the type field is optional and defaults to the localized term ‘PhD thesis’. You may still use the type field to override that.

```
⟨*!(modern | strict)⟩
FUNCTION {phdthesis}
{ output.bibitem
  format.authors "author" output.check
  new.sentence
  format.btitle "title" output.check
  new.column
  bbl.phdthesis format.thesis.type output.nonnull
  new.slash
  school "school" output.check
  new.block
  output.address.publisher.date
  new.block
  format.pages.page output
  note output
  new.sentence
  format.url output
  format.annote output
  fin.entry
}
⟨/!(modern | strict)⟩
⟨*modern | strict⟩
FUNCTION {phdthesis}
{ output.bibitem
  format.authors "author" output.check
⟨natbib⟩  author format.key output
  new.sentence
  format.btitle add.doi add.medium "title" output.check
  new.column
  bbl.phdthesis format.thesis.type output.nonnull
  new.column
  number output
  new.slash
  format.authors.after output
  new.semicolumn
  school "school" output.check
  new.block
  output.address.publisher
  format.date "year" output.check
  new.block
  format.pages.page output
  new.block
  note output
  new.sentence
%  format.url output
  output.eprint.url
  format.annote output
  fin.entry
}
⟨/modern | strict⟩

```

mastersthesis Similar to @thesis except that the type field is optional and defaults to the localized term ‘Master’s thesis’. You may still use the type field to override that.

```
⟨*!(modern | strict)⟩
FUNCTION {masterthesis}
{ output.bibitem
  format.authors "author" output.check
⟨natbib⟩  author format.key output
  new.sentence
  format.btitle "title" output.check
  new.column
  bbl.mthesis format.thesis.type output.nonnull
  new.slash
  school "school" output.check
  new.block
  output.address.publisher.date
  new.block
  format.pages.page output
  note output
  new.sentence
  format.url output
  format.annote output
  fin.entry
}
⟨/!(modern | strict)⟩
⟨*modern | strict⟩
FUNCTION {masterthesis}
{ output.bibitem
  format.authors "author" output.check
⟨natbib⟩  author format.key output
  new.sentence
  format.btitle add.doi add.medium "title" output.check
  new.column
  bbl.mthesis format.thesis.type output.nonnull
  new.column
  number output
  new.slash
  format.authors.after output
  new.semicolumn
  school "school" output.check
  new.block
  output.address.publisher
  format.date "year" output.check
  new.block
  format.pages.page output
  new.block
  note output
  new.sentence
%  format.url output
  output.eprint.url
  format.annote output
  fin.entry
}
⟨/modern | strict⟩

```

dscithesis Similar to `@thesis` except that the type field is optional and defaults to the localized term ‘Doctor’s of sciences thesis’. You may still use the type field to override that.

```
⟨*!(modern | strict)⟩
FUNCTION {dscithesis}
{ output.bibitem
  format.authors "author" output.check
⟨natbib⟩  author format.key output
  new.sentence
  format.btitle "title" output.check
  new.column
  bbl.dscithesis format.thesis.type output.nonnull
  new.slash
  school "school" output.check
  new.block
  output.address.publisher.date
  new.block
  format.pages.page output
  note output
  new.sentence
  format.url output
  format.annote output
  fin.entry
}
⟨/!(modern | strict)⟩
⟨*modern | strict⟩
FUNCTION {dscithesis}
{ output.bibitem
  format.authors "author" output.check
⟨natbib⟩  author format.key output
  new.sentence
  format.btitle add.doi add.medium "title" output.check
  new.column
  bbl.dscithesis format.thesis.type output.nonnull
  new.column
  number output
  new.slash
  format.authors.after output
  new.semicolumn
  school "school" output.check
  new.block
  output.address.publisher
  format.date "year" output.check
  new.block
  format.pages.page output
  new.block
  note output
  new.sentence
%  format.url output
  output.eprint.url
  format.annote output
  fin.entry
}
⟨/modern | strict⟩

```

conference

```
FUNCTION {conference} { inproceedings }

```

techreport

TechReport is similar to `@report` except that the `type` field is optional and defaults to the localized term ‘technical report’. You may still use the type field to override that.

```
%FUNCTION {techreport}
%{
%  output.bibitem
%  author.before
%  new.sentence
%  format.title add.doi add.medium "title" output.check
%  new.column
%  format.techrep.type.number output.nonnull
%  new.slash
%  author.after
%  editor.organization.after
%  new.block
%  address output
%  new.column
%  institution "institution" output.check
%  format.date "year" output.check
%  new.block
%  note output
%  new.block                % v.2
%  format.pages.page output % v.2
%  new.sentence
%%  format.url output
%  output.eprint.url
%  format.annote output
%  fin.entry
%}

FUNCTION {techreport}
{
  output.bibitem
%  author.before
%  new.sentence
  format.title add.doi add.medium "title" output.check
⟨natbib⟩  title format.key output
  new.column
%  format.techrep.type.number output.nonnull
%  type "type" output.check
%  format.report.type.number "type" output.check
%  type output
  format.techreport.type output
  new.column
  number output
  new.slash
  institution "institution" output.check
  new.semicolumn
  format.chief.after output % from editor field
  new.semicolumn
  format.executor.after output % from author field
  new.block
  address output
  new.column
  organization output
  format.date "year" output.check
  new.block                % v.2
  format.pages.page output % v.2
  new.sentence % или new.block ?
  output.eprint.url
  new.block
  note output
  format.annote output
  fin.entry
}


```

default.type

```
FUNCTION {default.type} { misc }

```

2.7 Month Abbreviations

```
MACRO {jan} {"\bbljan{}"}
MACRO {feb} {"\bblfeb{}"}
MACRO {mar} {"\bblmar{}"}
MACRO {apr} {"\bblapr{}"}
MACRO {may} {"\bblmay{}"}
MACRO {jun} {"\bbljun{}"}
MACRO {jul} {"\bbljul{}"}
MACRO {aug} {"\bblaug{}"}
MACRO {sep} {"\bblsep{}"}
MACRO {oct} {"\bbloct{}"}
MACRO {nov} {"\bblnov{}"}
MACRO {dec} {"\bbldec{}"}

```

2.8 Journal Abbreviations

2.9 Journal abbreviations

2.9.1 Physics and astronomy

Borrowed from physjour.mbs of package custom-bib.

```
MACRO {aa}{"Astron. \& Astrophys."}
MACRO {aasup}{"Astron. \& Astrophys. Suppl. Ser."}
MACRO {aj} {"Astron. J."}
MACRO {aph} {"Acta Phys."}
MACRO {advp} {"Adv. Phys."}
MACRO {ajp} {"Amer. J. Phys."}
MACRO {ajm} {"Amer. J. Math."}
MACRO {amsci} {"Amer. Sci."}
MACRO {anofd} {"Ann. Fluid Dyn."}
MACRO {am} {"Ann. Math."}
MACRO {ap} {"Ann. Phys. (NY)"}
MACRO {adp} {"Ann. Phys. (Leipzig)"}
MACRO {ao} {"Appl. Opt."}
MACRO {apl} {"Appl. Phys. Lett."}
MACRO {app} {"Astroparticle Phys."}
MACRO {apj} {"Astrophys. J."}
MACRO {apjsup} {"Astrophys. J. Suppl."}
MACRO {apss} {"Astrophys. Space Sci."}
MACRO {araa} {"Ann. Rev. Astron. Astrophys."}
MACRO {baas} {"Bull. Amer. Astron. Soc."}
MACRO {baps} {"Bull. Amer. Phys. Soc."}
MACRO {cmp} {"Comm. Math. Phys."}
MACRO {cpam} {"Commun. Pure Appl. Math."}
MACRO {cppcf} {"Comm. Plasma Phys. \& Controlled Fusion"}
MACRO {cpc} {"Comp. Phys. Comm."}
MACRO {cqg} {"Class. Quant. Grav."}
MACRO {cra} {"C. R. Acad. Sci. A"}
MACRO {fed} {"Fusion Eng. \& Design"}
MACRO {ft} {"Fusion Tech."}
MACRO {grg} {"Gen. Relativ. Gravit."}
MACRO {ieeens} {"IEEE Trans. Nucl. Sci."}
MACRO {ieeeps} {"IEEE Trans. Plasma Sci."}
MACRO {ijimw} {"Interntl. J. Infrared \& Millimeter Waves"}
MACRO {ip} {"Infrared Phys."}
MACRO {irp} {"Infrared Phys."}
MACRO {jap} {"J. Appl. Phys."}
MACRO {jasa} {"J. Acoust. Soc. America"}
MACRO {jcp} {"J. Comp. Phys."}
MACRO {jchp} {"J. Chem. Phys."}
MACRO {jetp} {"Sov. Phys.--JETP"}
MACRO {jfe} {"J. Fusion Energy"}
MACRO {jfm} {"J. Fluid Mech."}
MACRO {jmp} {"J. Math. Phys."}
MACRO {jne} {"J. Nucl. Energy"}
MACRO {jnec} {"J. Nucl. Energy, C: Plasma Phys., Accelerators, Thermonucl. Res."}
MACRO {jnm} {"J. Nucl. Mat."}
MACRO {jpc} {"J. Phys. Chem."}
MACRO {jpp} {"J. Plasma Phys."}
MACRO {jpsj} {"J. Phys. Soc. Japan"}
MACRO {jsi} {"J. Sci. Instrum."}
MACRO {jvst} {"J. Vac. Sci. \& Tech."}
MACRO {nat} {"Nature"}
MACRO {nature} {"Nature"}
MACRO {nedf} {"Nucl. Eng. \& Design/Fusion"}
MACRO {nf} {"Nucl. Fusion"}
MACRO {nim} {"Nucl. Inst. \& Meth."}
MACRO {nimpr} {"Nucl. Inst. \& Meth. in Phys. Res."}
MACRO {np} {"Nucl. Phys."}
MACRO {npb} {"Nucl. Phys. B"}
MACRO {nt/f} {"Nucl. Tech./Fusion"}
MACRO {npbpc} {"Nucl. Phys. B (Proc. Suppl.)"}
MACRO {inc} {"Nuovo Cimento"}
MACRO {nc} {"Nuovo Cimento"}
MACRO {pf} {"Phys. Fluids"}
MACRO {pfa} {"Phys. Fluids A: Fluid Dyn."}
MACRO {pfb} {"Phys. Fluids B: Plasma Phys."}
MACRO {pl} {"Phys. Lett."}
MACRO {pla} {"Phys. Lett. A"}
MACRO {plb} {"Phys. Lett. B"}
MACRO {prep} {"Phys. Rep."}
MACRO {pnas} {"Proc. Nat. Acad. Sci. USA"}
MACRO {pp} {"Phys. Plasmas"}
MACRO {ppcf} {"Plasma Phys. \& Controlled Fusion"}
MACRO {phitrsl} {"Philos. Trans. Roy. Soc. London"}
MACRO {prl} {"Phys. Rev. Lett."}
MACRO {pr} {"Phys. Rev."}
MACRO {physrev} {"Phys. Rev."}
MACRO {pra} {"Phys. Rev. A"}
MACRO {prb} {"Phys. Rev. B"}
MACRO {prc} {"Phys. Rev. C"}
MACRO {prd} {"Phys. Rev. D"}
MACRO {pre} {"Phys. Rev. E"}
MACRO {ps} {"Phys. Scripta"}
MACRO {procrsl} {"Proc. Roy. Soc. London"}
MACRO {rmp} {"Rev. Mod. Phys."}
MACRO {rsi} {"Rev. Sci. Inst."}
MACRO {science} {"Science"}
MACRO {sciam} {"Sci. Am."}
MACRO {sam} {"Stud. Appl. Math."}
MACRO {sjpp} {"Sov. J. Plasma Phys."}
MACRO {spd} {"Sov. Phys.--Doklady"}
MACRO {sptp} {"Sov. Phys.--Tech. Phys."}
MACRO {spu} {"Sov. Phys.--Uspekhi"}
MACRO {st} {"Sky and Telesc."}

```

2.9.2 Optics

Borrowed from `photjour.mbs`.

```
MACRO {appopt} {"Appl. Opt."}
MACRO {bell}   {"Bell Syst. Tech. J."}
MACRO {ell}    {"Electron. Lett."}
MACRO {jasp}   {"J. Appl. Spectr."}
MACRO {jqe}    {"IEEE J. Quantum Electron."}
MACRO {jlwt}   {"J. Lightwave Technol."}
MACRO {jmo}    {"J. Mod. Opt."}
MACRO {josa}   {"J. Opt. Soc. America"}
MACRO {josaa}  {"J. Opt. Soc. Amer.~A"}
MACRO {josab}  {"J. Opt. Soc. Amer.~B"}
MACRO {jdp}    {"J. Phys. (Paris)"}
MACRO {oc}     {"Opt. Commun."}
MACRO {ol}     {"Opt. Lett."}
MACRO {os}     {"Opt. Spectrosc."}
MACRO {phtl}   {"IEEE Photon. Technol. Lett."}
MACRO {pspie}  {"Proc. Soc. Photo-Opt. Instrum. Eng."}
MACRO {sjot}   {"Sov. J. Opt. Technol."}
MACRO {sjqe}   {"Sov. J. Quantum Electron."}
MACRO {sleb}   {"Sov. Phys.--Leb. Inst. Rep."}
MACRO {stph}   {"Sov. Phys.--Techn. Phys."}
MACRO {stphl}  {"Sov. Techn. Phys. Lett."}
MACRO {vr}     {"Vision Res."}
MACRO {zph}    {"Z. f. Physik"}
MACRO {zphb}   {"Z. f. Physik~B"}
MACRO {zphd}   {"Z. f. Physik~D"}
```

2.9.3 Physics of condensed Matter

```
MACRO {sse} {"Solid-State Electron."}
MACRO {pss} {"Phys. Sol. State"}
MACRO {sst} {"Semicond. Sci. Tech."}
MACRO {nl}  {"Nano Lett."}

READ

```

2.10 Sorting

Next chunk of code governs sorting reference list by authors' names and titles.

```
⟨*sort | natbib⟩

```

sortify

```
FUNCTION {sortify}
{ purify$
⟨!utf8⟩   "l" change.case$
}
⟨/sort | natbib⟩

```

sort.format.names

```
⟨*sort⟩
%% ====================================
%% This version from old Gost package
%%<*!natbib>
FUNCTION {sort.format.names}
{ 's :=
  #1 'nameptr :=
  ""
  s num.names$ 'numnames :=
  numnames 'namesleft :=
    { namesleft #0 > }
    { nameptr #1 >
        { "   " * }
        'skip$
      if$
      s nameptr
      "{vv{ } }{ll{ }}{  f{ }}{  jj{ }}"
      format.name$ 't :=
      nameptr numnames = t "others" = and
        { "et al" * }
        %{ bbl.etal * }
        { t sortify * }
      if$
      nameptr #1 + 'nameptr :=
      namesleft #1 - 'namesleft :=
    }
  while$
}
%%</!natbib>
%% This version from plainnat.bst
%% It ignores second and subsequent authors but include year.
%%<*natbib>
%FUNCTION {sort.format.names}
%{ 's :=
%  #1 'nameptr :=
%  ""
%  s num.names$ 'numnames :=
%  numnames 'namesleft :=
%    { namesleft #0 > }
%    {
%      s nameptr "{vv{ } }{ll{ }}{  ff{ }}{  jj{ }}" format.name$ 't :=
%      nameptr #1 >
%        {
%          "   "  *
%          namesleft #1 = t "others" = and
%            { "zzzzz" * }
%            { numnames #2 > nameptr #2 = and
%                { "zz" * year field.or.null * "  " * }
%                'skip$
%              if$
%              t sortify *
%            }
%          if$
%        }
%        { t sortify * }
%      if$
%      nameptr #1 + 'nameptr :=
%      namesleft #1 - 'namesleft :=
%    }
%  while$
%}
%%</natbib>
%% =====================================

```

sort.format.title

```
FUNCTION {sort.format.title}
{ 't :=
  "A " #2
    "An " #3
      "The " #4 t chop.word % Removes "The " if any
    chop.word               % Removes "An " if any
  chop.word                 % Removes "A " if any
  sortify
  #1 global.max$ substring$
}

```

author.sort

```
%% =====================================
%% This version from old gost package.
%%
⟨*!natbib⟩
FUNCTION {author.sort}
{ author empty$
    { key empty$
        { "to sort, need author or key in " cite$ * warning$
          ""
        }
        { key sortify }
      if$
    }
    {
      author num.names$ #4 <
        {author sort.format.names }
        {title sort.format.title}
      if$
    }
  if$
}
⟨/!natbib⟩
%% This version from plainnat.bst
⟨*natbib⟩
FUNCTION {author.sort}
{ author empty$
    { key empty$
        { "to sort, need author or key in " cite$ * warning$
          ""
        }
        { key sortify }
      if$
    }
    { author sort.format.names }
  if$
}
⟨/natbib⟩
%% ====================================

```

author.title.sort

```
⟨*!natbib | natbib⟩
FUNCTION {author.title.sort}
{ author empty$
    { title empty$
        { key empty$
            { "to sort, need author, title, or key in " cite$ * warning$
              ""
            }
            { key sortify }
          if$
        }
        { title sort.format.title }
      if$
    }
    {
      author num.names$ #4 <
        {author sort.format.names }
        {title sort.format.title}
      if$
     }
  if$
}
⟨/!natbib | natbib⟩
⟨*natbib | natbib⟩
```

```
FUNCTION {author.editor.sort}
{ author empty$
    { editor empty$
        { key empty$
            { "to sort, need author, editor, or key in " cite$ * warning$
              ""
            }
            { key sortify }
          if$
        }
        { editor sort.format.names }
      if$
    }
    { author sort.format.names }
  if$
}

FUNCTION {author.organization.sort}
{ author empty$
    { organization empty$
        { key empty$
            { "to sort, need author, organization, or key in " cite$ * warning$
              ""
            }
            { key sortify }
          if$
        }
        { "The " #4 organization chop.word sortify }
      if$
    }
    { author sort.format.names }
  if$
}

FUNCTION {editor.organization.sort}
{ editor empty$
    { organization empty$
        { key empty$
            { "to sort, need editor, organization, or key in " cite$ * warning$
              ""
            }
            { key sortify }
          if$
        }
        { "The " #4 organization chop.word sortify }
      if$
    }
    { editor sort.format.names }
  if$
}
```

```
⟨/natbib | natbib⟩


```

presort Function to compute `sort.key$`. What is the space string "␣␣␣" for?

```
⟨*!natbib⟩
FUNCTION {presort}%#1
{
  author.title.sort
  "   "
  *
  year field.or.null sortify
  *
  "   "
  *
  title field.or.null
  sort.format.title
  *
  #1 entry.max$ substring$
  'sort.key$ :=
}
⟨/!natbib⟩
⟨*natbib⟩
FUNCTION {presort}%#2
{ calc.label
  label sortify
  %author.title.sort
  "    "
  *
  % ========= plainnat.bst =========
%  type$ "book" =
%  type$ "inbook" =
%  or
%    'author.editor.sort
%    { type$ "proceedings" =
%        'editor.organization.sort
%        { type$ "manual" =
%            'author.organization.sort
%            'author.sort
%          if$
%        }
%      if$
%    }
%  if$
  author.title.sort
  "    "
  *
  year field.or.null sortify
  *
  "    "
  *
  %cite$
  title field.or.null sort.format.title
  *
  #1 entry.max$ substring$
  'sort.label :=
  sort.label *
  % ================================
  #1 entry.max$ substring$
  'sort.key$ :=
}
⟨/natbib⟩
⟨/sort⟩

⟨*!sort⟩
⟨*natbib⟩
INTEGERS { seq.num }

FUNCTION {init.seq}
{ #0 'seq.num :=}

EXECUTE {init.seq}

FUNCTION {int.to.fix}
{ "000000000" swap$ int.to.str$ *
  #-1 #10 substring$
}

FUNCTION {presort}%#3
{
  calc.label               % computes label
  label sortify            % initiates sort.label
  "    "
  *
  seq.num #1 + 'seq.num := % advance seq.num
  seq.num  int.to.fix      % prepend seq.num with 0s
  'sort.label :=           % set sort.label to seq.num
  sort.label *             % append seq.num to label
  #1 entry.max$ substring$ % cut if too long
  'sort.key$ :=            % set sort.key$
}
⟨/natbib⟩
⟨/!sort⟩

⟨*sort | natbib⟩
ITERATE {presort}

SORT

⟨/sort | natbib⟩

```

2.11 Bibliography list

We need to find longest label to put in into the argument of the `thebibliography` environment. In case of `natbib` options we also need to compute extra suffix for the `year` field if there two or more entries for given label (=author/editor/organization) in that year.

Declare global (external) strings used in calculation of the longest label.

```
⟨!natbib⟩STRINGS { longest.label }
⟨natbib⟩STRINGS { longest.label last.label next.extra }

⟨!natbib⟩INTEGERS { number.label longest.label.width }
⟨natbib⟩INTEGERS { number.label longest.label.width last.extra.num  }

```

`ize.longest.label` Initialize those string.

```
⟨*!natbib⟩
FUNCTION {initialize.longest.label}
{ "" 'longest.label :=
  #1 'number.label :=
  #0 'longest.label.width :=
}
⟨/!natbib⟩
⟨*natbib⟩
FUNCTION {initialize.longest.label}
{ "" 'longest.label :=
  #0 int.to.chr$ 'last.label :=
  "" 'next.extra :=
  #0 'longest.label.width :=
  #0 'last.extra.num :=
  #0 'number.label :=
}
⟨/natbib⟩

EXECUTE {initialize.longest.label}

```

`ize.longest.label` Iterate though the list of entries to compute `label`.

```
⟨*!natbib⟩
FUNCTION {forward.pass}
{ number.label int.to.str$ 'label :=
  number.label #1 + 'number.label :=
  label width$ longest.label.width >
    { label 'longest.label :=
      label width$ 'longest.label.width :=
    }
    'skip$
  if$
}
⟨/!natbib⟩
⟨*natbib⟩
FUNCTION {forward.pass}
{ last.label label =
    { last.extra.num #1 + 'last.extra.num :=
      last.extra.num int.to.chr$ 'extra.label :=
    }
    { "a" chr.to.int$ 'last.extra.num :=
      "" 'extra.label :=
      label 'last.label :=
    }
  if$
  number.label #1 + 'number.label :=
}
⟨/natbib⟩

ITERATE {forward.pass}

```

reverse.pass

Natbib styles require reverse iteration over all entries.

```
⟨*natbib⟩
FUNCTION {reverse.pass}
{ next.extra "b" =
    { "a" 'extra.label := }
    'skip$
  if$
  extra.label 'next.extra :=
  extra.label
  duplicate$ empty$
    'skip$
    { "{\natexlab{" swap$ * "}}" * }
  if$
  'extra.label :=
  label extra.label * 'label :=
}

REVERSE {reverse.pass}

FUNCTION {bib.sort.order}
{ sort.label  'sort.key$ :=
}

ITERATE {bib.sort.order}

SORT
⟨/natbib⟩

```

begin.bib

Within thebibliography environment we define few formatting macros for user to customize how the reference list is formatted.

```
FUNCTION {begin.bib}
{ "\begin{thebibliography}{"  longest.label  * "}" * write$ newline$
  "\def\selectlanguageifdefined#1{"                  write$ newline$
  "\expandafter\ifx\csname date#1\endcsname\relax"   write$ newline$
%  "\else\language\csname l@#1\endcsname\fi}"          write$ newline$
```

```
  "\else\selectlanguage{#1}\fi}"                         write$ newline$
  "\providecommand*{\href}[2]{{\small #2}}"              write$ newline$
  "\providecommand*{\url}[1]{{\small #1}}"               write$ newline$
  "\providecommand*{\BibUrl}[1]{\url{#1}}"               write$ newline$
  "\providecommand{\BibAnnote}[1]{}"                     write$ newline$
  "\providecommand*{\BibEmph}[1]{#1}"                    write$ newline$
⟨*modern⟩
  "\ProvideTextCommandDefault{\cyrdash}{\hbox to.8em{--\hss--}}" write$ newline$
  "\providecommand*{\BibDash}{\ifdim\lastskip>0pt\unskip\nobreak\hskip.2em\fi" write$ newline$
  "\cyrdash\hskip.2em\ignorespaces}" write$ newline$
⟨/modern⟩
⟨natbib⟩  "\providecommand{\natexlab}[1]{#1}" write$ newline$
  preamble$ empty$
    'skip$
    { preamble$ write$ newline$ }
  if$
}


EXECUTE {begin.bib}

EXECUTE {init.state.consts}

ITERATE {call.type$}

```

end.bib

```
FUNCTION {end.bib}
{ newline$
%  "\catcode`\/=11"        write$ newline$
  "\end{thebibliography}" write$ newline$
}

EXECUTE {end.bib}

⟨/bst⟩
```

That's all, Folks!

Change History

v0.8

General: \BibAnnote added 1

\BibEmph added 1

\BibUrl added 1

Entry ANNOTE added 1

v0.9

General: Bug fix in INPROCEEDINGS . . . 1

Bug fix in names and date formatting . . . 1

v1.0

General: Bug fix (long annote) 1

v1.1

General: Added German, French, Italian languages 1

Entry ONLINE 1

Gost705.dtx borrowed from Disser pkg . . 1

Upload to CTAN 1

v1.2

General: Entries eprint, eprintclass, eprinttype 1

Fix bbl.urldate for ukrainian (Andrey Shvajkoy) 1

Medium field 1

Options modern, long, eprint 1

Refactoring, Documentation 1

Strict option 1

Support for natbib package 1

Thesis entry, report entry 1

v1.2a

General: Default for \cyrdash added 1

3 Index

Numbers written in dark blue refer to the page where the corresponding entry is described; numbers in black roman refer to the code lines where the entry is used.