# Package 'icd'

May 15, 2017

**Title** Tools for Working with ICD-9 and ICD-10 Codes, and Finding Comorbidities

**Description** Calculate comorbidities, Charlson scores, perform fast and accurate validation, conversion, manipulation, filtering and comparison of ICD-9 and ICD-10 codes. Common ambiguities and code formats are handled. This package enables a work flow from raw lists of ICD codes in hospital billing databases to comorbidities. ICD-9 and ICD-10 comorbidity mappings from Quan (Deyo and Elixhauser versions), Elixhauser and AHRQ included. This package replaces 'icd9', which should be uninstalled.

**Classification/ACM-2012** Social and professional topics~Medical records, Applied computing~Health care information systems, Applied computing~Health informatics, Applied computing~Bioinformatics

**Version** 2.2

**Date** 2017-05-01

**Maintainer** Jack O. Wasey <jack@jackwasey.com>

**URL** https://github.com/jackwasey/icd

**Depends** R (>= 3.3.0)

**Imports** checkmate (>= 1.7.0), magrittr, stats, Rcpp (>= 0.12.3), utils

**Suggests** knitr, RODBC, roxygen2 (>= 5.0.0), rmarkdown, testthat, xml2

**LazyData** true

**LazyDataCompression** xz

**ByteCompile** true

**BugReports** https://github.com/jackwasey/icd/issues

**License** GPL-3

**Copyright** See file (inst/)COPYRIGHTS

**VignetteBuilder** knitr

**LinkingTo** Rcpp, testthat

**RoxygenNote** 6.0.1

**NeedsCompilation** yes

**Author** Jack O. Wasey [aut, cre, cph],
William Murphy [ctb] (Van Walraven scores),
R Core Team [ctb, cph] (m4 macro for OpenMP detection in configure)

**Repository** CRAN

**Date/Publication** 2017-05-15 00:10:19 UTC

# R topics documented:

icd-package *icd: Tools for Working with ICD-9 and ICD-10 Codes, and Finding Comorbidities*

## Description

Calculate comorbidities, Charlson scores, perform fast and accurate validation, conversion, manipulation, filtering and comparison of ICD-9 and ICD-10 codes. Common ambiguities and code formats are handled. This package enables a work flow from raw lists of ICD codes in hospital billing databases to comorbidities. ICD-9 and ICD-10 comorbidity mappings from Quan (Deyo and Elixhauser versions), Elixhauser and AHRQ included. This package replaces 'icd9', which should be uninstalled.

## Details

**Comorbidities** `icd_comorbid` determines comorbidities for a set of patients with one or more ICD-9 codes each. `icd_charlson` calculates Charlson score (Comorbidity Index). If you already calculated the comorbidities, you can use `icd_charlson_from_comorbid`.

- AHRQ comorbidity mapping is provided, and a function to read the raw SAS code from AHRQ into R data structures. The data is available by lazy-loading in `icd9_map_ahrq`. AHRQ releases new mappings annually.
- Quan revised both Deyo/Charlson and Elixhauser ICD-9 to comorbidity mappings. These are presented as: `icd9_map_quan_deyo` (which is also derived from the original SAS code used in his publication, referenced in the data documentation), `icd10_map_quan_deyo`, `icd9_map_quan_elix`, and `icd10_map_quan_elix` which was transcribed directly from the same paper.
- The original Elixhauser mapping is provided, with codes transcribed from the original publication. See `icd9_map_elix`.

**Validation** `icd_is_valid` checks whether ICD-9 codes are syntactically valid (although not necessarily genuine ICD-9 diagnoses). In contrast, `icd_is_defined` checks whether ICD-9 codes correspond to diagnoses in the current ICD-9-CM definition from CMS.

**Conversion** There are many functions to convert ICD-9 codes or their components between different formats and structures. The most commonly used are: `icd_decimal_to_short`, `icd_short_to_decimal` to convert, e.g., 002.3 to 0023 and back again. See convert for other options.

**Manipulation** You can find children of a higher-level ICD-9 code with `icd_children` and find a common parent to a set of children (or arbitrary list of ICD-9 codes) with `icd_condense`. `icd_sort` sorts in hierarchical, then numerical order, so 100.0 comes before 100.00, for example. `icd_wide_to_long` and `icd_long_to_wide` convert the two most common data structures containing patient disease data. This is more sophisticated and tailored to the problem than base reshaping or extension packages, although these could no doubt be used.

**Explanation, or decoding** Use `icd_explain` to convert a list of codes into human-readable descriptions. This function can optionally reduce the codes to a their top-level groups if all the child members of a group are present. `icd_diff_comorbid` allows summary of the differences between comorbidity mappings, e.g. to find what has changed from year-to-year or between revisions by different authors. `icd9cm_hierarchy` is a `data.frame` containing the

full ICD-9 classification for each diagnosis. icd9_chapters contains definitions of chapters, sub-chapters and three-digit groups.

## Author(s)

**Maintainer**: Jack O. Wasey <jack@jackwasey.com> [copyright holder]

Other contributors:

- William Murphy <WMurphy@eatright.org> (Van Walraven scores) [contributor]
- R Core Team (m4 macro for OpenMP detection in configure) [contributor, copyright holder]

Jack O. Wasey <jack@jackwasey.com>

## References

http://www.hcup-us.ahrq.gov/toolssoftware/comorbidity/comorbidity.jsp

## See Also

Useful links:

- https://github.com/jackwasey/icd
- Report bugs at https://github.com/jackwasey/icd/issues

---

# combine *combine ICD codes*

---

## Description

These function implement combination of lists or vectors of codes, while preserving ICD classes. Base R c just drops all user defined classes and casts down to lowest common denominator, e.g. if mixing numbers and characters. No attempt here to catch all possible combinations of feeding in mixed ICD types and other types. Let R do what it normally does, but just try to keep classes of the first item in the list.

## Usage

```
## S3 method for class 'icd9'
c(..., warn = FALSE)

## S3 method for class 'icd10'
c(..., warn = FALSE)
```

## Arguments

| | |
|---|---|
| ... | elements to combine |
| warn | single logical value, if TRUE, will give warnings when incompatible types are combined using c |

## Examples

```
## Not run:
# throw an error? or assign type according to first argument?
c(as.icd9("E998"), as.icd10("A10"))

# benchmark subsetting to justify using .subset2 (5% faster)
library(microbenchmark)
j <- list(as.icd9cm("E990"), as.icd9cm("10010"))
k <- list(rep(as.icd9cm("E990"), times = 500))
microbenchmark(j[[1]], .subset2(j, 1),
               k[[1]], .subset2(k, 1),
               times = 1e6)

# logical list to vector
a <- list(T,T)
microbenchmark(as.logical(a), c(a, recursive = TRUE), times = 1e6)

# c(..., recursive = TRUE) vs unlist
l = list(c("100", "440", "999"), c("123", "234"))
microbenchmark::microbenchmark(c(l, recursive = TRUE),
                               c(unlist(l)),
                               times = 1e6)
stopifnot(identical(c(l, recursive = TRUE), c(unlist(l))))


## End(Not run)
```

---

`icd10cm2016` *ICD-10-CM*

---

## Description

The public domain modified ICD-10 classification as published in the public domain by the US CDC. Currently this has a slightly different structure to `icd9cm_hierarchy` because the published data helpfully has a *leaf* flag indicating whether a code is a *billable* leaf node, or a code higher in the hierarchy which nevertheless will have a description.

## Format

data frame, with columns for code, leaf status (0 or 1), short and long descriptions.

## Details

There are annual revisions to this data. Currently, the 2016 edition is included.

## Source

http://www.cdc.gov/nchs/icd/icd10cm.htm

## icd10_chapters *ICD-10 chapters*

### Description

The WHO ICD-10 scheme chapters. The chapter level is the highest in the hierarchy, each chapter containing sets of codes which span multiple three-digit 'major' codes, and in some cases also span codes across two alphabetic initial characters. E.g. Chapter I spans A00 to B99.

### Format

list with chapter names stored in list names, each with two element named character vector with start and end codes.

### Details

2017 ICD-10-CM does not have any U codes (codes for special purposes). U00-U49 - Provisional assignment of new diseases of uncertain etiology or emergency use U82-U85 - Resistance to antimicrobial and anti-neoplastic drugs

### Source

http://apps.who.int/classifications/icd10/browse/2016/en

## icd10_sub_chapters *ICD-10 sub-chapters*

### Description

The WHO ICD-10 scheme sub-chapters. N.b. there may be WHO vs CM differences: please file bug if noted. In the XML definition of ICD-10-CM there are some intermediate hierarchical levels, e.g. for neoplasms. Sub-chapter here is defined as the lowest-level grouping of three-digit codes, e.g. C00-C14 "Malignant neoplasms of lip, oral cavity and pharynx", not C00-C96 "Malignant neoplasms" which itself is a subset of the chapter C00-D49 "Neoplasms"

### Format

list with sub-chapter or major names stored in list names, each with two element named character vector with start and end codes.

### Source

http://apps.who.int/classifications/icd10/browse/2016/en

## icd9cm_billable *list of annual versions of billable leaf nodes of ICD-9-CM*

### Description

These are derived from the CMS published updates, with versions 23 to 32 currently available going back to 2004/5. The source files back to version 27 have short and long descriptions. The short descriptions are in ASCII with no special characters, whereas the long descriptions contain accented characters which seem to be interpreted as Unicode, `latin-1` or `cp1252`. This all done during package creation, but can be repeated by package users, including pulling the data from the web pages directly. Despite my best efforts, current locale can give different results, but this packaged data is correct, with some `UTF-8` encoded strings.

### Format

list of data frames. Each list item is named by the version as a string, e.g. "32". The constituent data frames have columns `icd9`, `shortDesc`, and `longDesc`.

### Source

http://www.cms.gov/Medicare/Coding/ICD9ProviderDiagnosticCodes/codes.html

## icd9cm_hierarchy *Latest ICD-9-CM diagnosis codes, in flat* `data.frame` *format*

### Description

Short-form ICD-9 codes with short and long descriptions, and description of each hierarchy level containing each code.

### Format

data frame

### Source

http://wonder.cdc.gov/wonder/sci_data/codes/icd9/type_txt/icd9cm.asp

Rich text descriptions here: http://www.cdc.gov/nchs/icd/icd9cm.htm http://www.cms.gov/Medicare/Coding/ICD9ProviderDiagnosticCodes/codes.html This page has versions 23 to 32 (2005 to 2014). At present, only the 2014 data is included in this package.

http://wonder.cdc.gov/wonder/sci_data/codes/icd9/type_txt/icd9abb.asp

http://wonder.cdc.gov/wonder/sci_data/codes/icd9/type_txt/icd9cm.asp

http://wonder.cdc.gov/wonder/sci_data/codes/icd9/type_txt/icdcm.asp

http://wonder.cdc.gov/wonder/sci_data/codes/icd9/type_txt/icd9abb.asp

`icd9_chapters` *ICD-9 chapters*

## Description

`icd9_chapters`, `icd9_chapters_sub` and `icd9_majors` contain mappings from the higher level descriptions of ICD-9 codes to the ranges of ICD-9 codes they describe. Helpful in summarizing codes or grouping for human-readable output. These can easily be converted to a co-morbidity mapping, as shown in the vignette.

## Format

list with chapter/sub-chapter or major names stored in list names, each with two element named character vector with start and end codes.

## Details

- 001-139 Infectious And Parasitic Diseases
- 140-239 Neoplasms
- 240-279 Endocrine, Nutritional And Metabolic Diseases, And Immunity Disorders
- 280-289 Diseases Of The Blood And Blood-Forming Organs
- 290-319 Mental Disorders
- 320-389 Diseases Of The Nervous System And Sense Organs
- 390-459 Diseases Of The Circulatory System
- 460-519 Diseases Of The Respiratory System
- 520-579 Diseases Of The Digestive System
- 580-629 Diseases Of The Genitourinary System
- 630-679 Complications Of Pregnancy, Childbirth, And The Puerperium
- 680-709 Diseases Of The Skin And Subcutaneous Tissue
- 710-739 Diseases Of The Musculoskeletal System And Connective Tissue
- 740-759 Congenital Anomalies
- 760-779 Certain Conditions Originating In The Perinatal Period
- 780-799 Symptoms, Signs, And Ill-Defined Conditions
- 800-999 Injury And Poisoning
- V01-V91 Supplementary Classification Of Factors Influencing Health Status And Contact With Health Services
- E000-E999 Supplementary Classification Of External Causes Of Injury And Poisoning

## Source

http://www.cms.gov/Medicare/Coding/ICD9ProviderDiagnosticCodes/codes.html

## icd9_is_n *do ICD-9 codes belong to numeric, V or E sub-types?*

### Description

For each code, return TRUE if numeric or FALSE if a V or E code.

### Usage

```
icd9_is_n(x)

icd9_is_v(x)

icd9_is_e(x)
```

### Arguments

| | |
|---|---|
| x | vector of strings or factor to test |

### Value

logical vector

### Functions

- icd9_is_v: are the given codes V type?
- icd9_is_e: are the given codes E type?

## icd9_map_ahrq *AHRQ comorbidities*

### Description

This mapping of comorbidities to ICD-9 codes is derived directly from SAS code provided by AHRQ, and translated into this R data structure. This is a revision of the Elixhauser system, notably excluding cardiac arrythmia.

### Format

list of character vectors

### Source

http://www.hcup-us.ahrq.gov/toolssoftware/comorbidity/comorbidity.jsp http://www.hcup-us.ahrq.gov/toolssoftware/comorbidityicd10/comorbidity_icd10.jsp

## icd9_map_elix *Elixhauser comorbidities*

### Description

The original mapping of Elixhauser's ICD-9-CM to 30 comorbidities. According to Sharabiani, this mapping provides the best long-term mortality prediction. The weaknesses of this mapping are that it is based on slightly out-dated ICD-9 codes. I have not yet verified what changes to the ICD-9-CM specification between 1998 and now would impact this mapping.

### Format

list of character vectors, each named by co-morbidity

### References

Sharabiani, Mansour T. A., Paul Aylin, and Alex Bottle. "Systematic Review of Comorbidity Indices for Administrative Data." Medical Care December 2012 50, no. 12 (2012): 1109-18. doi:10.1097/MLR.0b013e31825f64d0. http://www.ncbi.nlm.nih.gov/pubmed/22929993

Elixhauser, Anne, Claudia Steiner, D. Robert Harris, and Rosanna M. Coffey. "Comorbidity Measures for Use with Administrative Data." Medical Care January 1998 36, no. 1 (1998): 8-27.

## icd9_map_hcc *Medicare Hierarchical Condition Categories*

### Description

Medicare HCC model was developed to use current year diagnoses and demographics predict current year healthcare expenditure. This classification has been used for additional risk adjustment models. ICD codes are first assigned to numeric Condition Categories ('CCs'). A hierarchy rule is then applied so that each patient is coded for only the most severe of the Condition Categories in a group. For example, if a patient has metastatic lung cancer, they will only be assigned the 'CC' for "Metastatic Cancer and Acute Leukemia", and will not be assigned the 'CC' for "Lung and other Severe Cancers". Once the hierarchy rules are applied, the codes are referred to as HCCs. This mapping can change over time. It remained the same from 2007-10

### Format

`dataframe` with 3 columns (`icd_code`, `cc`, and `year`)

## References

Pope, Gregory C., et al. "Diagnostic cost group hierarchical condition category models for Medicare risk adjustment." Health Economics Research, Inc. Waltham, MA (2000). https://www.cms.gov/Research-Statistics-Data-and-Systems/Statistics-Trends-and-Reports/Reports/Downloads/Pope_2000_2.pdf

Risk Adjustment, Centers for Medicare and Medicaid Services https://www.cms.gov/Medicare/Health-Plans/MedicareAdvtgSpecRateStats/Risk-Adjustors.html

---

`icd9_map_quan_deyo` *Quan adaptation of Deyo/Charlson comorbidities*

---

## Description

Derived automatically from the SAS code used in the original publication. According to the referenced study, this provides the best predictor of in-patient to <30d mortality. Of note, Deyo drops the distinction between leukemia, lymphoma and non-metastatic cancer. As far as I have looked into this, in the rare cases where someone had two or three of leukemia, lymphoma and non-metastatic cancer, the Quan adaptation would give a lower Charlson score than the original scheme. The Deyo original Charlson to ICD-9-CM groups does include distinct categories for these things.

## Format

list of character vectors, each named by co-morbidity

## References

Quan, Hude, Vijaya Sundararajan, Patricia Halfon, Andrew Fong, Bernard Burnand, Jean-Christophe Luthi, L. Duncan Saunders, Cynthia A. Beck, Thomas E. Feasby, and William A. Ghali. "Coding Algorithms for Defining Comorbidities in ICD-9-CM and ICD-10 Administrative Data." Medical Care 43, no. 11 (November 1, 2005): 1130-39. http://www.ncbi.nlm.nih.gov/pubmed/16224307 http://web.archive.org/web/20110225042437/http://www.chaps.ucalgary.ca/sas

---

`icd9_map_quan_elix` *Quan adaptation of Elixhauser comorbidities*

---

## Description

These were transcribed directly from the Quan paper referenced.

## Format

list of character vectors, each named by co-morbidity

## References

Quan, Hude, Vijaya Sundararajan, Patricia Halfon, Andrew Fong, Bernard Burnand, Jean-Christophe Luthi, L. Duncan Saunders, Cynthia A. Beck, Thomas E. Feasby, and William A. Ghali. "Coding Algorithms for Defining Comorbidities in ICD-9-CM and ICD-10 Administrative Data." Medical Care 43, no. 11 (November 1, 2005): 1130-39. http://www.ncbi.nlm.nih.gov/pubmed/16224307 http://web.archive.org/web/20110225042437/http://www.chaps.ucalgary.ca/sas

---

`icd9_sources` *ICD-9 data sources*

---

## Description

List of ICD-9 data sources for different versions of ICD-9-CM

---

`icd_charlson` *Calculate Charlson Comorbidity Index (Charlson Score)*

---

## Description

Charlson score is calculated in the basis of the Quan revision of Deyo's ICD-9 mapping. (peptic ulcer disease no longer warrants a point.) Quan published an updated set of scores, but it seems most people use the original scores for easier comparison between studies, even though Quan's were more predictive.

## Usage

```
icd_charlson(x, visit_name = NULL, scoring_system = c("original",
  "charlson", "quan"), return_df = FALSE,
  stringsAsFactors = getOption("stringsAsFactors"), ...)

## S3 method for class 'data.frame'
icd_charlson(x, visit_name = NULL,
  scoring_system = c("original", "charlson", "quan"), return_df = FALSE,
  stringsAsFactors = getOption("stringsAsFactors"), ...)

icd_charlson_from_comorbid(x, visit_name = NULL, hierarchy = FALSE,
  scoring_system = c("original", "charlson", "quan"))
```

## Arguments

| | |
|---|---|
| x | data frame containing a column of visit or patient identifiers, and a column of ICD-9 codes. It may have other columns which will be ignored. By default, the first column is the patient identifier and is not counted. If visit_name is not specified, the first column is used. |
| visit_name | The name of the column in the data frame which contains the patient or visit identifier. Typically this is the visit identifier, since patients come leave and enter hospital with different ICD-9 codes. It is a character vector of length one. If left empty, or NULL, then an attempt is made to guess which field has the ID for the patient encounter (not a patient ID, although this can of course be specified directly). The guesses proceed until a single match is made. Data frames may be wide with many matching fields, so to avoid false positives, anything but a single match is rejected. If there are no successful guesses, and visit_id was not specified, then the first column of the data frame is used. |
| scoring_system | One of original, charlson, or quan. The first two will give the original Charlson weights for each comorbidity, whereas quan uses the updated weights from Quan 2001. |
| return_df | single logical value, if true, a two column data frame will be returned, with the first column named as in input data frame (i.e. visit_name), containing all the visits, and the second column containing the Charlson Comorbidity Index. |
| stringsAsFactors | single logical, passed on when constructing data.frame if return_df is TRUE. If the input data frame x has a factor for the visit_name, this is not changed, but a non-factor visit_name may be converted or not converted according to your system default or this setting. |
| ... | further arguments to pass on to icd9_comorbid_quan_deyo, e.g. icd_name |
| hierarchy | single logical value, default is FALSE. If TRUE, will drop DM if DMcx is present, etc. |

## Details

When used, hierarchy is applied per Quan, "The following comorbid conditions were mutually exclusive: diabetes with chronic complications and diabetes without chronic complications; mild liver disease and moderate or severe liver disease; and any malignancy and metastatic solid tumor." The Quan scoring weights come from the 2011 paper (dx.doi.org/10.1093/aje/kwq433). The comorbidity weights were recalculated using updated discharge data, and some changes, such as Myocardial Infarction decreasing from 1 to 0, may reflect improved outcomes due to advances in treatment since the original weights were determined in 1984.

## Methods (by class)

- data.frame: Charlson scores from data frame of visits and ICD-9 codes. ICD-10 Charlson can be calculated simply by getting the Charlson (e.g. Quan Deyo) comorbidities, then calling icd_charlson_from_comorbid.

## Examples

```
mydf <- data.frame(visit_name = c("a", "b", "c"),
                   icd9 = c("441", "412.93", "044.9"))
cmb <- icd9_comorbid_quan_deyo(mydf)
cmb
icd_charlson(mydf)
# can specify short_code directly instead of guessing
icd_charlson(mydf, short_code = FALSE, return_df = TRUE)
icd_charlson_from_comorbid(cmb)
```

---

# icd_comorbid_df_to_mat

*convert comorbidity matrix to data frame*

---

## Description

convert matrix of comorbidities into data frame, preserving visit_name information

## Usage

```
icd_comorbid_df_to_mat(x, visit_name = get_visit_name(x),
  stringsAsFactors = getOption("stringsAsFactors"))
```

## Arguments

| | |
|---|---|
| `x` | data frame, with a `visit_name` column (not necessarily first), and other columns with flags for comorbidities, as such column names are required. |
| `visit_name` | The name of the column in the data frame which contains the patient or visit identifier. Typically this is the visit identifier, since patients come leave and enter hospital with different ICD-9 codes. It is a character vector of length one. If left empty, or `NULL`, then an attempt is made to guess which field has the ID for the patient encounter (not a patient ID, although this can of course be specified directly). The guesses proceed until a single match is made. Data frames may be wide with many matching fields, so to avoid false positives, anything but a single match is rejected. If there are no successful guesses, and `visit_id` was not specified, then the first column of the data frame is used. |
| `stringsAsFactors` | Single logical value, describing whether the resulting data frame should have strings, e.g. `visit_id` converted to factor. Default is to follow the current session option. This is identical to the argument used in, among other base functions `as.data.frame`. |

## Examples

```
longdf <- icd_long_data(
            visit = c("a", "b", "b", "c"),
            icd9 = c("441", "4424", "443", "441")
            )
cmbdf <- icd9_comorbid_elix(longdf, return_df = TRUE)
class(cmbdf)
rownames(cmbdf)
mat.out <- icd_comorbid_df_to_mat(cmbdf)
stopifnot(is.matrix(mat.out))
mat.out[, 1:4]
```

---

# icd_comorbid_mat_to_df

*convert comorbidity data frame from matrix*

---

## Description

convert matrix of comorbidities into data frame, preserving visit_name information

## Usage

```
icd_comorbid_mat_to_df(x, visit_name = "visit_id",
  stringsAsFactors = getOption("stringsAsFactors"))
```

## Arguments

| | |
|---|---|
| x | Matrix of comorbidities, with row and columns names defined |
| visit_name | Single character string with name for new column in output data frame. Everywhere else, visit_name describes the input data, but here it is for output data. |
| stringsAsFactors | Single logical value, describing whether the resulting data frame should have strings, e.g. visit_id converted to factor. Default is to follow the current session option. This is identical to the argument used in, among other base functions as.data.frame. |

## Examples

```
longdf <- icd_long_data(
            visit_id = c("a", "b", "b", "c"),
            icd9 = as.icd9(c("441", "4424", "443", "441")))
mat <- icd9_comorbid_elix(longdf)
class(mat)
typeof(mat)
rownames(mat)
df.out <- icd_comorbid_mat_to_df(mat)
stopifnot(is.data.frame(df.out))
```

```
# output data frame has a factor for the visit_name column
stopifnot(identical(rownames(mat), as.character(df.out[["visit_id"]])))
df.out[, 1:4]
# when creating a data frame like this, stringsAsFactors uses
# the system-wide option you may have set e.g. with
# options("stringsAsFactors" = FALSE).
is.factor(df.out[["visit_id"]])
```

---

`icd_count_codes` *Count ICD codes or comorbidities for each patient*

---

## Description

`icd_count_codes` takes a data frame with a column for `visit_name` and another for ICD-9 code, and returns the number of distinct codes for each patient.

## Usage

```
icd_count_codes(x, visit_name = get_visit_name(x), return_df = FALSE)
```

## Arguments

| | |
|---|---|
| `x` | data frame with one row per patient, and a true/false or 1/0 flag for each column. By default, the first column is the patient identifier and is not counted. If `visit_name` is not specified, the first column is used. |
| `visit_name` | The name of the column in the data frame which contains the patient or visit identifier. Typically this is the visit identifier, since patients come leave and enter hospital with different ICD-9 codes. It is a character vector of length one. If left empty, or `NULL`, then an attempt is made to guess which field has the ID for the patient encounter (not a patient ID, although this can of course be specified directly). The guesses proceed until a single match is made. Data frames may be wide with many matching fields, so to avoid false positives, anything but a single match is rejected. If there are no successful guesses, and `visit_id` was not specified, then the first column of the data frame is used. |
| `return_df` | single logical, if `TRUE`, return the result as a data frame with the first column being the `visit_name`, and the second being the count. If `visit_name` was a factor or named differently in the input, this is preserved. |

## Details

The `visit_name` field is typically the first column. If there is no column called `visit_name` and `visit_name` is not specified, the first column is used.

## Value

vector of the count of comorbidities for each patient. This is sometimes used as a metric of comorbidity load, instead of, or in addition to metrics like the Charlson Comorbidity Index (aka Charlson Score)

## Examples

```
mydf <- data.frame(visit_name = c("r", "r", "s"),
                   icd9 = c("441", "412.93", "044.9"))
icd_count_codes(mydf, return_df = TRUE)
icd_count_codes(mydf)

cmb <- icd9_comorbid_quan_deyo(mydf, isShort = FALSE, return_df = TRUE)
icd_count_comorbid(cmb)

wide <- data.frame(visit_name = c("r", "s", "t"),
                   icd9_1 = c("0011", "441", "456"),
                   icd9_2 = c(NA, "442", NA),
                   icd9_3 = c(NA, NA, "510"))
icd_count_codes_wide(wide)
# or:
library(magrittr)
wide %>% icd_wide_to_long %>% icd_count_codes
```

---

# icd_count_codes_wide *Count ICD codes given in wide format*

---

## Description

For `icd_count_codes`, it is assumed that all the columns apart from `visit_name` represent actual or possible ICD-9 codes. Duplicate `visit_names` are repeated as given and aggregated.

## Usage

```
icd_count_codes_wide(x, visit_name = get_visit_name(x), return_df = FALSE,
  aggr = FALSE)
```

## Arguments

| | |
|---|---|
| `x` | `data.frame` with one row per patient, hospital visit, encounter, etc., and multiple columns containing any ICD codes attributed to that encounter or patient. i.e. data frame with ICD codes in wide format. |
| `visit_name` | The name of the column in the data frame which contains the patient or visit identifier. Typically this is the visit identifier, since patients come leave and enter hospital with different ICD-9 codes. It is a character vector of length one. If left empty, or `NULL`, then an attempt is made to guess which field has the ID for the patient encounter (not a patient ID, although this can of course be specified directly). The guesses proceed until a single match is made. Data frames may be wide with many matching fields, so to avoid false positives, anything but a single match is rejected. If there are no successful guesses, and `visit_id` was not specified, then the first column of the data frame is used. |
| `return_df` | single logical value, if `TRUE`, return the result as a data frame with the first column being the `visit_id`, and the second being the count. If `visit_id` was a factor or named differently in the input, this is preserved. |

aggr — single logical, default is FALSE. If TRUE, the length (or rows) of the output will no longer match the input, but duplicate visit_names will be counted together.

---

## icd_count_comorbid *Count number of comorbidities per patient*

---

### Description

icd_count_comorbid differs from the other counting functions in that it counts *comorbidities*, not individual diagnoses. It accepts any data.frame with either logical or binary contents, with a single column for visit_name. No checks are made to see whether visit_name is duplicated.

### Usage

```
icd_count_comorbid(x, visit_name = get_visit_name(x), return_df = FALSE)
```

### Arguments

| | |
|---|---|
| x | data frame with one row per patient, and a true/false or 1/0 flag for each column. By default, the first column is the patient identifier and is not counted. If visit_name is not specified, the first column is used. |
| visit_name | The name of the column in the data frame which contains the patient or visit identifier. Typically this is the visit identifier, since patients come leave and enter hospital with different ICD-9 codes. It is a character vector of length one. If left empty, or NULL, then an attempt is made to guess which field has the ID for the patient encounter (not a patient ID, although this can of course be specified directly). The guesses proceed until a single match is made. Data frames may be wide with many matching fields, so to avoid false positives, anything but a single match is rejected. If there are no successful guesses, and visit_id was not specified, then the first column of the data frame is used. |
| return_df | single logical value, if TRUE, return the result as a data frame with the first column being the visit_id, and the second being the count. If visit_id was a factor or named differently in the input, this is preserved. |

---

## icd_diff_comorbid *show the difference between two comorbidity mappings*

---

### Description

Compares two comorbidity to ICD code mappings. The results are returned invisibly as a list. Only those comorbidities with (case sensitive) overlapping names are compared.

## Usage

```
icd_diff_comorbid(x, y, all_names = NULL, x_names = NULL, y_names = NULL,
  show = TRUE, explain = TRUE)

## S3 method for class 'list'
icd_diff_comorbid(x, y, all_names = NULL, x_names = NULL,
  y_names = NULL, show = TRUE, explain = TRUE)
```

## Arguments

| | |
|---|---|
| x | list of character vectors |
| y | list of character vectors |
| all_names | character vector of the comorbidity names |
| x_names | character vector of the comorbidity names from x to compare |
| y_names | character vector of the comorbidity names from y to compare |
| show | single logical value. The default is TRUE which causes a report to be printed. |
| explain | single logical value. The default is TRUE which means the differing codes are attempted to be reduced to their parent codes, in order to give a more succinct summary. |

## Value

A list, each item of which is another list containing the intersections and both asymmetric differences.

## Methods (by class)

- list: Show difference between comorbidity maps with ICD-9 codes

## Examples

```
# compare CHF for ICD-10 mappings from Elixhauser and AHRQ
icd_diff_comorbid(icd10_map_elix, icd10_map_ahrq, show = FALSE)[["CHF"]]
## Not run:
# default is to show the results in a human readable manner:
diff_result <- icd_diff_comorbid(icd9_map_elix, icd9_map_ahrq)[["CHF"]]
# show differences for
# give full report on all comorbidities for these mappings
diff_result <- icd_diff_comorbid(icd9_map_elix, icd9_map_ahrq, show = FALSE)

# the following outputs a summary to the console:
icd_diff_comorbid(icd9_map_elix, icd9_map_ahrq)

## End(Not run)
```

`icd_filter_poa` *Filters data frame based on present-on-arrival flag*

## Description

Present On Arrival (POA) is not a simple flag, since many codes are exempt, unspecified, or unknown. Therefore, two options are given: get all the comorbidities where the POA flag was definitely negative, coded as 'N' or definitely positive and coded as 'Y'. Negating one set won't give the other set unless all codes were either Y or N.

## Usage

```
icd_filter_poa(x, poa_name = "poa", poa = icd_poa_choices)

icd_filter_poa_yes(x, poa_name = "poa")

icd_filter_poa_no(x, poa_name = "poa")

icd_filter_poa_not_no(x, poa_name = "poa")

icd_filter_poa_not_yes(x, poa_name = "poa")
```

## Arguments

| | |
|---|---|
| `x` | input vector of ICD codes |
| `poa_name` | The name of column in the data frame which contains the Present On Arrival (POA) flag. The flag itself is a single character, typically one of "Y", "N", "E", "X", "U" or empty. |
| `poa` | single character value, being one of `Yes`, `No`, `NotYes`, and `NotNo`, indicating whether to account for comorbidities flagged as present-on-arrival. This is not a simple flag, because many codes are exempt, unspecified, or unknown. The intermediate codes, such as "exempt", "unknown" and `NA` mean that "yes" is not the same as "not no." |

## Functions

- `icd_filter_poa_yes`: Select rows where Present-on-Arrival flag is explicitly 'Yes.'
- `icd_filter_poa_no`: Select rows where Present-on-Arrival flag is explicitly 'No.'
- `icd_filter_poa_not_no`: Select rows where Present-on-Arrival flag is anything but 'No.' This includes unknown, exempt, other codes, and of course all those marked 'Yes.'
- `icd_filter_poa_not_yes`: Select rows where Present-on-Arrival flag is anything but 'Yes.' This would group exempt, unknown and other codes under 'Not POA' which is unlikely to be a good choice, since exempt codes, of which there are a quite large number, tend to describe chronic or out-of-hospital characteristics.

## Examples

```
## Not run:
library(magrittr, warn.conflicts = FALSE, quietly = TRUE)
myData <- data.frame(
  visit_id = c("v1", "v2", "v3", "v4"),
  diag = c("39891", "39790", "41791", "4401"),
  poa = c("Y", "N", NA, "Y"),
  stringsAsFactors = FALSE
)
myData %>% icd_filter_poa_not_no() %>% icd_comorbid_ahrq()
# can fill out named fields also:
myData %>% icd_filter_poa_yes(poa_name="poa") %>%
  icd_comorbid_ahrq(icd_name = "diag", visit_name = "visit_id", short_code = TRUE)
# can call the core icd_comorbid() function with an arbitrary mapping
myData %>%
  icd_filter_poa_yes %>%
  icd_comorbid(icd_name = "diag", visit_name = "visit_id",
    map = icd_map_quan_elix, short_mapping = TRUE)

## End(Not run)
```

---

# icd_filter_valid *Filter ICD codes by validity.*

---

## Description

Filters a data.frame of patients for valid or invalid ICD-9 codes

## Usage

```
icd_filter_valid(x, icd_name = get_icd_name(x),
  short_code = icd_guess_short(.subset2(x, icd_name)), invert = FALSE)

icd_filter_invalid(x, icd_name = get_icd_name(x),
  short_code = icd_guess_short(x[[icd_name]]), invert = FALSE)

icd9_filter_valid(x, icd_name = get_icd_name(x),
  short_code = icd_guess_short(x[[icd_name]]), invert = FALSE)

icd10_filter_valid(x, icd_name = get_icd_name(x),
  short_code = icd_guess_short(x[[icd_name]]), invert = FALSE)

icd9_filter_invalid(x, icd_name = get_icd_name(x),
  short_code = icd_guess_short(x[[icd_name]]), invert = FALSE)

icd10_filter_invalid(x, icd_name = get_icd_name(x),
  short_code = icd_guess_short(x[[icd_name]]), invert = FALSE)
```

### Arguments

| | |
|---|---|
| x | input vector of ICD codes |
| icd_name | The column in the data.frame which contains the ICD codes. This is a character vector of length one. If it is NULL, icd9 will attempt to guess the column name, looking for progressively less likely possibilities until it matches a single column. Failing this, it will take the first column in the data frame. Specifying the column using this argument avoids the guesswork. |
| short_code | single logical value which determines whether the ICD-9 code provided is in short (TRUE) or decimal (FALSE) form. Where reasonable, this is guessed from the input data. |
| invert | Single logical value. Returns the inverse of the result. E.g. if seeking valid ICD-9 codes, the invalid ones are returned. |
| ... | arguments passed to the class-specific functions |

### Functions

- icd_filter_invalid: Filter invalid rows from data frame of patients with ICD codes. This can also be achieved with icd_filter_valid and invert = TRUE
- icd9_filter_valid: Filter data frame for valid ICD codes

---

## icd_get_defined *Select only defined ICD codes*

---

### Description

Return only those codes which are heading or leaf (billable), specifying whether codes are all short-form or all decimal-form

### Usage

```
icd_get_defined(x, short_code = icd_guess_short(x), billable = FALSE)
```

### Arguments

| | |
|---|---|
| x | input vector or factor, possibly with an ICD class |
| short_code | logical value, whether short-form ICD code |
| billable | single logical value, whether to limit return codes also by whether they are billable, i.e. leaf nodes. This is really only designed for use with ICD-9-CM, ICD-10-CM etc, since the WHO versions are not designed for billing, but for public health and death reporting. |

`icd_long_to_wide` *Convert ICD data from long to wide format*

## Description

This is more complicated than reshape or `reshape2::dcast` allows. This is a reasonably simple solution using built-in functions.

## Usage

```
icd_long_to_wide(x, visit_name = get_visit_name(x),
  icd_name = get_icd_name(x), prefix = "icd_", min_width = 0,
  aggr = TRUE, return_df = FALSE)
```

## Arguments

| | |
|---|---|
| `x` | data.frame of long-form data, one column for visit_name and one for ICD code |
| `visit_name` | The name of the column in the data frame which contains the patient or visit identifier. Typically this is the visit identifier, since patients come leave and enter hospital with different ICD-9 codes. It is a character vector of length one. If left empty, or `NULL`, then an attempt is made to guess which field has the ID for the patient encounter (not a patient ID, although this can of course be specified directly). The guesses proceed until a single match is made. Data frames may be wide with many matching fields, so to avoid false positives, anything but a single match is rejected. If there are no successful guesses, and `visit_id` was not specified, then the first column of the data frame is used. |
| `icd_name` | The column in the `data.frame` which contains the ICD codes. This is a character vector of length one. If it is `NULL`, `icd9` will attempt to guess the column name, looking for progressively less likely possibilities until it matches a single column. Failing this, it will take the first column in the data frame. Specifying the column using this argument avoids the guesswork. |
| `prefix` | character, default `icd_` to prefix new columns |
| `min_width,` | single integer, if specified, writes out this many columns even if no patients have that many codes. Must be greater than or equal to the maximum number of codes per patient. |
| `aggr` | single logical value, if TRUE (the default) will take more time to find out-of-order visit_names, and combine all the codes for each unique visit_name. If `FALSE`, then out-of-order visit_names will result in a row in the output data per contiguous block of identical visit_names. |
| `return_df` | single logical value, if `TRUE`, return a data frame with a field for the visit_name. This may be more convenient, but the default of `FALSE` gives the more natural return data of a matrix with row names being the visit IDs from `visit_names`. |

## See Also

Other ICD-9 convert: icd9PartsToShort, icd9_chapters_to_map, icd9_drop_leading_zeroes, icd_wide_to_long

## Examples

```
longdf <- data.frame(visit_name = c("a", "b", "b", "c"),
  icd9 = c("441", "4424", "443", "441"))
icd_long_to_wide(longdf)
icd_long_to_wide(longdf, prefix = "ICD10_")
```

---

# icd_names_elix *Comorbidity names*

---

## Description

These lists provide correctly sorted names of the comorbidities and their particular permutations in both full and abbreviated forms.

## Format

list, with character/numeric code. 'Hypertension, uncomplicated' and 'Hypertension, complicated' are labelled '6a' and '6b'. Diabetes, cancer, and metastasis are counted independently, as in the original paper, giving the original 30 groups. "01" to "30"

## Details

In the Elixhauser derived mappings, uncomplicated and complicated hypertension are listed separately, but are always combined in the final analyses. Uncomplicated and complicated hypertension are list separately and as "Hypertension, combined." _abbrev suffix indicates a very short space-free description. Quan's version of Elixhauser is identical. AHRQ updates drops the arrythmia field. The naming convention is a root, e.g. `icd9_map_elix` or `icd10_map_elix`, with neither/either/both suffixes `_htn` and `_abbrev`. The Charlson derived mappings do not include hypertension. _abbreviated comorbidity names are helpful for interactive work, whereas the full names might be preferred for plotting.

---

# icd_poa_choices *Present-on-admission flags*

---

## Description

See icd_filter_poa for more details.

## Usage

```
icd_poa_choices
```

### Format

An object of class `character` of length 4.

### Examples

```
icd_poa_choices
```

---

`icd_van_walraven` *Calculate van Walraven Elixhauser Score*

---

### Description

van Walraven Elixhauser score is calculated from the Quan revision of Elixhauser's ICD-9 mapping. This function allows for the hierarchical exclusion of less severe versions of comorbidities when their more severe version is also present via the `hierarchy` argument. For the Elixhauser comorbidities, this is diabetes v. complex diabetes and solid tumor v. metastatic tumor

### Usage

```
icd_van_walraven(x, visit_name = NULL, return_df = FALSE,
  stringsAsFactors = getOption("stringsAsFactors"), ...)

## S3 method for class 'data.frame'
icd_van_walraven(x, visit_name = NULL,
  return_df = FALSE, stringsAsFactors = getOption("stringsAsFactors"), ...)

icd_van_walraven_from_comorbid(x, visit_name = NULL, hierarchy = FALSE)
```

### Arguments

| | |
|---|---|
| `x` | data frame containing a column of visit or patient identifiers, and a column of ICD-9 codes. It may have other columns which will be ignored. By default, the first column is the patient identifier and is not counted. If `visit_name` is not specified, the first column is used. |
| `visit_name` | The name of the column in the data frame which contains the patient or visit identifier. Typically this is the visit identifier, since patients come leave and enter hospital with different ICD-9 codes. It is a character vector of length one. If left empty, or `NULL`, then an attempt is made to guess which field has the ID for the patient encounter (not a patient ID, although this can of course be specified directly). The guesses proceed until a single match is made. Data frames may be wide with many matching fields, so to avoid false positives, anything but a single match is rejected. If there are no successful guesses, and `visit_id` was not specified, then the first column of the data frame is used. |
| `return_df` | single logical value, if true, a two column data frame will be returned, with the first column named as in input data frame (i.e. `visit_name`), containing all the visits, and the second column containing the Charlson Comorbidity Index. |

stringsAsFactors
: Single logical value, describing whether the resulting data frame should have strings, e.g. visit_id converted to factor. Default is to follow the current session option. This is identical to the argument used in, among other base functions as.data.frame.

...
: arguments passed on to other functions

hierarchy
: single logical value that defaults to TRUE, in which case the hierarchy defined for the mapping is applied. E.g. in Elixhauser, you can't have uncomplicated and complicated diabetes both flagged.

## Methods (by class)

- data.frame: van Walraven scores from data frame of visits and ICD-9 codes

## Author(s)

wmurphyrd

## References

van Walraven C, Austin PC, Jennings A, Quan H, Forster AJ. A Modification to the Elixhauser Comorbidity Measures Into a Point System for Hospital Death Using Administrative Data. Med Care. 2009; 47(6):626-633. http://www.ncbi.nlm.nih.gov/pubmed/19433995

## Examples

```
mydf <- as.icd9(data.frame(visit_name = c("a", "b", "c"),
                  icd9 = c("412.93", "441", "044.9")))

cmb <- icd9_comorbid_quan_elix(mydf, short_code = FALSE, hierarchy = TRUE, return_df=TRUE)
cmb

icd_van_walraven_from_comorbid(cmb)

icd_van_walraven(mydf)
icd_van_walraven(mydf, return_df = TRUE)
```

---

# icd_wide_to_long *Convert ICD data from wide to long format*

---

## Description

Reshaping data is a common task, and is made easier here by knowing more about the underlying structure of the data. This function wraps the reshape function with specific behavior and checks related to ICD codes. Empty strings and NA values will be dropped, and everything else kept. No validation of the ICD codes is done.

## Usage

```
icd_wide_to_long(x, visit_name = get_visit_name(x), icd_labels = NULL,
  icd_name = "icd_code", icd_regex = c("icd", "diag", "dx_", "dx"))
```

## Arguments

| | |
|---|---|
| x | data.frame in wide format, i.e. one row per patient, and multiple columns containing ICD codes, empty strings or NA. |
| visit_name | The name of the column in the data frame which contains the patient or visit identifier. Typically this is the visit identifier, since patients come leave and enter hospital with different ICD-9 codes. It is a character vector of length one. If left empty, or NULL, then an attempt is made to guess which field has the ID for the patient encounter (not a patient ID, although this can of course be specified directly). The guesses proceed until a single match is made. Data frames may be wide with many matching fields, so to avoid false positives, anything but a single match is rejected. If there are no successful guesses, and visit_id was not specified, then the first column of the data frame is used. |
| icd_labels | vector of column names in which codes are found. If NULL, all columns matching the regular expression icd_regex will be included. |
| icd_name | The column in the data.frame which contains the ICD codes. This is a character vector of length one. If it is NULL, icd9 will attempt to guess the column name, looking for progressively less likely possibilities until it matches a single column. Failing this, it will take the first column in the data frame. Specifying the column using this argument avoids the guesswork. |
| icd_regex | vector of character strings containing a regular expression to identify ICD-9 diagnosis columns to try (case-insensitive) in order. Default is c("icd", "diag", "dx_", "dx") |

## Value

data.frame with visit_name column named the same as input, and a column named by icd.name containing all the non-NA and non-empty codes found in the wide input data.

## See Also

Other ICD-9 convert: icd9PartsToShort, icd9_chapters_to_map, icd9_drop_leading_zeroes, icd_long_to_wide

## Examples

```
widedf <- data.frame(visit_name = c("a", "b", "c"),
  icd9_01 = c("441", "4424", "441"),
  icd9_02 = c(NA, "443", NA))
icd_wide_to_long(widedf)
```

## is.icd_short_diag *test ICD-related classes*

### Description

currently no checks on correctness of the classes for these functions

### Usage

```
is.icd_short_diag(x, must_work = FALSE)

is.icd_decimal_diag(x, must_work = FALSE)

is.icd9(x)

is.icd10(x)

is.icd9cm(x)

is.icd10cm(x)

is.icd_long_data(x)

is.icd_wide_data(x)

is.icd_comorbidity_map(x)
```

### Arguments

| | |
|---|---|
| x | Any object which may have ICD-related classes set |
| must_work | single logical value, if FALSE (the default) this may return NULL if the attribute is not present. If TRUE, then either TRUE or FALSE is returned. |

### Details

is.icd_short_diag tests for presence of an attribute, not whether the code is a valid ICD code. If must_work is TRUE then NULL (i.e. no attribute set) returns FALSE, otherwise NULL is returned.

## print.icd_comorbidity_map *Print a comorbidity map*

### Description

The default is to summarize by printing the first seven comorbidities, and the first seven codes for each. To print the whole thing, just convert it to a list.

### Usage

```
## S3 method for class 'icd_comorbidity_map'
print(x, ..., n_comorbidities = 7,
  n_codes = 7)
```

### Arguments

| | |
|---|---|
| x | a list optionally with class icd_comorbidity_map |
| ... | further arguments are passed to print |
| n_comorbidities | single integer, number of comorbidities to print |
| n_codes | single integer, number of codes per comorbidity to print |

### Examples

```
icd9_map_ahrq
## Not run:
print(icd9_map_ahrq)
print(icd9_map_ahrq, n_comorbidities = 3, n_codes = 3)
print.list(icd9_map_ahrq)
print(list(icd9_map_ahrq))

## End(Not run)
```

---

## subset_icd *extract subset from ICD data*

---

### Description

exactly the same as using x[n] or x[[n]] but preserves the ICD classes in result

### Usage

```
## S3 method for class 'icd9'
x[...]

## S3 method for class 'icd9'
x[[...]]

## S3 method for class 'icd10'
x[...]

## S3 method for class 'icd10'
x[[...]]
```

## Arguments

| | |
|---|---|
| x | input data with list, vector, factor, and class set to an ICD type. |
| ... | arguments passed on to other functions |

## Examples

```
x <- as.icd9(list(my_codes = c("V10.1", "441.1")))
x[1]
x[[1]]
x[[1]][2]
# subsetting a list should give the underlying data structure type,
# preserving the ICD class
stopifnot(!inherits(x[[1]], "list"))
stopifnot(!inherits(x[[1]][2], "list"))

y <- as.icd10(c("A01", "B0234"))
y[2]
y[[2]]
stopifnot(inherits(y[2], "icd10"))
stopifnot(inherits(y[[2]], "icd10"))
```

---

# unzip_single *unzip a single file from URL*

---

## Description

take a single file from zip located at a given URL, unzip into temporary directory, and copy to the given save_path

## Usage

```
unzip_single(url, file_name, save_path)
```

## Arguments

| | |
|---|---|
| url | URL of a zip file |
| file_name | file name of the resource within the zip file |
| save_path | file path to save the first file from the zip |

## uranium_pathology *United States Transuranium & Uranium Registries*

### Description

an ICD-10 data set (not ICD-10-CM) with mortality from the United States Transuranium & Uranium Registries, published in the public domain.

### Source

http://www.ustur.wsu.edu/database/ http://www.ustur.wsu.edu/Case_Studies/Pathology/mdb/Pathology_Office2007.zip

## vermont_dx *Hospital discharge data from Vermont*

### Description

Anonymous data from public Vermont source for 2013

### Format

CSV original, minimally processed into R data frame.

### Details

Conditions of Release Release of public use data is subject to the following conditions, which the requestor agrees to upon accepting copies of the data:

1. The data may not be used in any manner that attempts to or does identify, directly or indirectly, any individual patient or physician.

2. The requestor agrees to incorporate the following, or a substantially similar, disclaimer in all reports or publications that include public use data: "Hospital discharge data for use in this study were supplied by the Vermont Association of Hospitals and Health Systems-Network Services Organization (VAHHS-NSO) and the Vermont Department of Banking, Insurance, Securities and Health Care Administration (BISHCA). All analyses, interpretations or conclusions based on these data are solely that of [the requestor]. VAHHS-NSO and BISHCA disclaim responsibility for any such analyses, interpretations or conclusions. In addition, as the data have been edited and processed by VAHHS-NSO, BISHCA assumes no responsibility for errors in the data due to coding or processing"

### Author(s)

Vermont Division of Health Care Administration

### Source

http://healthvermont.gov/research/hospital-utilization/RECENT_PU_FILES.aspx

`[[.icd_comorbidity_map` *Extract vector of codes from an ICD comorbidity map*

## Description

Equivalent to a list, but preserves class of extracted vector.

## Usage

```
## S3 method for class 'icd_comorbidity_map'
x[[index, ...]]
```

## Arguments

| | |
|---|---|
| `x` | comorbidity map, which is a named list |
| `index` | integer |
| `...` | arguments passed on to other functions |

## Examples

```
# show that attributes are preserved when subsetting
stopifnot(is.icd_short_diag(icd10_map_ahrq[[1]]))
```

# Index