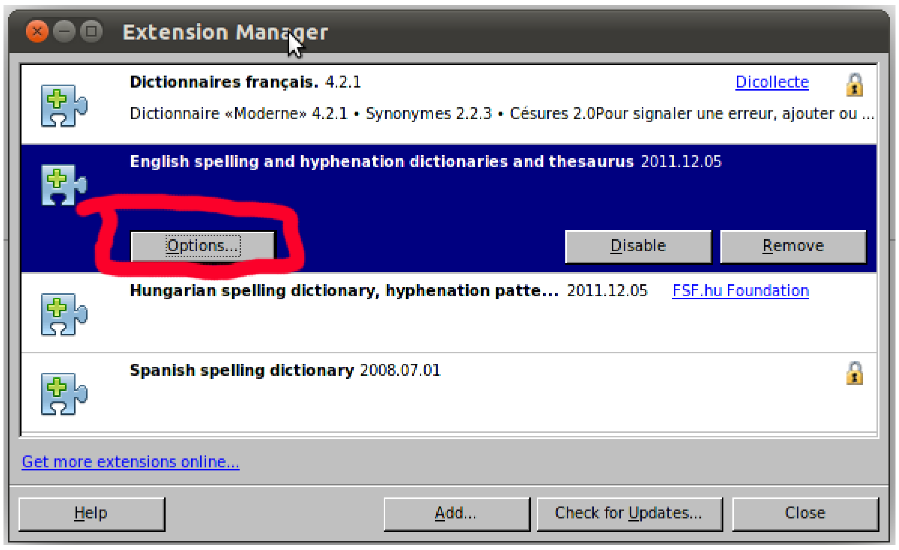
English sentence checking in LibreOffice

Default checks

- Some word duplications: *and and or or for for the the.*
- Simple grammar checking: *Her's is a better idea.*
- Articles: *a hour, an one-way* etc. It doesn't check ambiguous (for example *a/an hotel*) and unknown (missing from the Hunspell dictionary) words.
- capitalization of paragraphs. Condition: two or more sentences in the paragraph.
- Punctuation: (parentheses), comma , colon : semicolon ; period . exclamation mark ! Question mark ?
- Typewriter dashes: foo - bar ⟶ foo – bar, foo--bar ⟶ foo—bar or foo–bar
- Missing space: *one,two*
- Multiplication sign: *4x4* ⟶ *4×4.*
- Double or triple spaces between   words.

Settings for optional features

Tools » Extension manager » English dictionary extension » Options:



Extension Manager

Dictionnaires français. 4.2.1 Dicollecte

Dictionnaire «Moderne» 4.2.1 • Synonymes 2.2.3 • Césures 2.0Pour signaler une erreur, ajouter ou ...

English spelling and hyphenation dictionaries and thesaurus 2011.12.05

Options... Disable Remove

Hungarian spelling dictionary, hyphenation patte... 2011.12.05 FSF.hu Foundation

Spanish spelling dictionary 2008.07.01

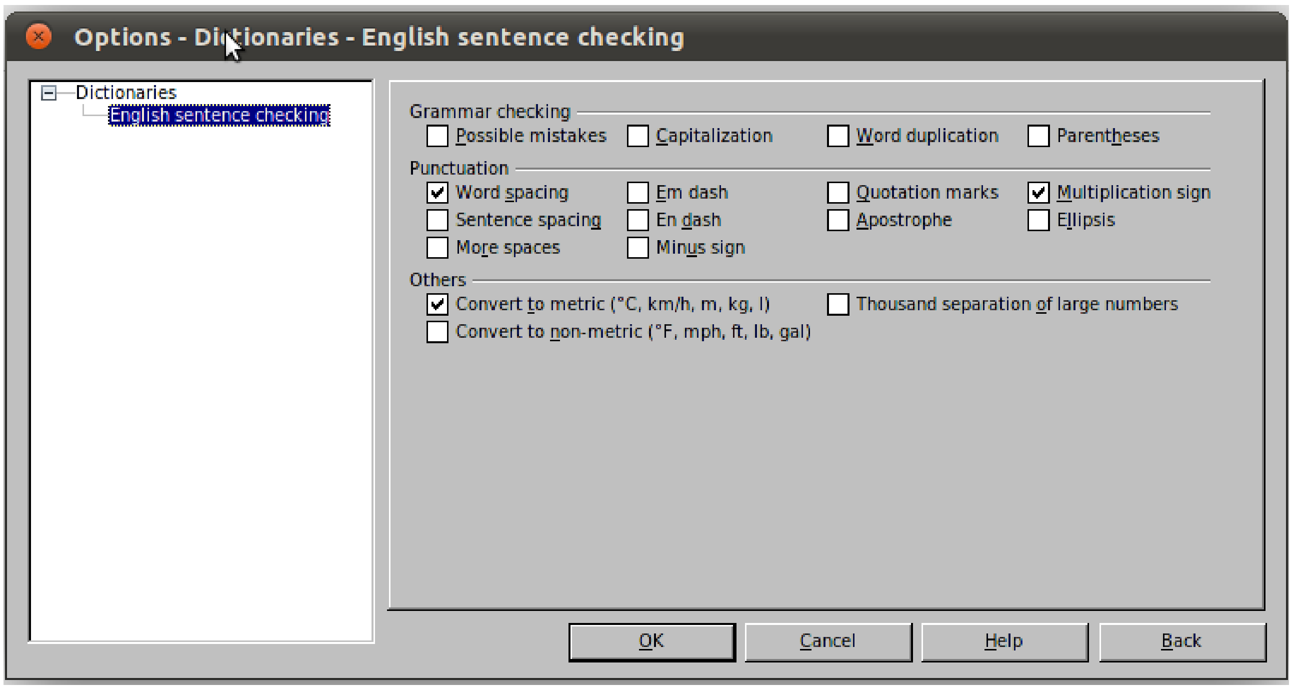
Get more extensions online...

Help Add... Check for Updates... Close

Optional features

- Other grammar checks: *with it's, he don't, this things* etc.
- Capitalization of sentences. the sentence boundary detection depends on abbreviations.
- All word duplication duplication.
- Pair of parentheses) and quotation" marks.
- Sentence spacing. More spaces   replaced with a single space or a tabulator.
- Unspaced em dash and spaced en dash conversion: "xxx—xxx" ↔ "xxx – xxx".
- Typographical signs: minus (-5 ⟶ −5), "quotation mark", apostrophe', ellipsis...

- Measurement conversion: 10 gallons, 22,000 lbs, 45 °F, 100.5 mph, 5 ft 1½ in ↔ 38 l, 9979 kg, 7 ℃, 162 km/h, 156 cm.
- Thousand separation: 1000000 ⟶ 1,000,000 or 1 000 000.


Options - Dictionaries - English sentence checking

Dictionaries
English sentence checking

Grammar checking
Possible mistakes | Capitalization | Word duplication | Parentheses

Punctuation
[✓ Word spacing] | Em dash | Quotation marks | [✓ Multiplication sign]
Sentence spacing | En dash | Apostrophe | Ellipsis
More spaces | Minus sign

Others
[✓ Convert to metric (°C, km/h, m, kg, l)] | Thousand separation of large numbers
Convert to non-metric (°F, mph, ft, lb, gal)

Appendix

Source code

http://www.numbertext.org/lightproof

Rules

```
# English sentence checking

# word-level rules (case-sensitive)

[word]

# basic syntax of the rules:
#
# pattern -> suggestion # warning message
# pattern <- condition -> suggestion # warning message
#

# duplicates

and and -> and # Did you mean:
or or -> or # Did you mean:
for for -> for # Did you mean:
the the -> the # Did you mean:

# word-level rules (case-insensitive)

[Word]

# multiword expressions

ying and yang -> yin and yang # Did you mean:

# multiple suggestions separated by "\n"

scot free -> scot-free\nscotfree # Did you mean:

# possessive pronouns

# Your's -> Yours
(your|her|our|their)['’]s -> \1s # Possessive pronoun:

# a or an (rules for articles)#
###############################

[word]

# pattern "a" matches "a" or "A":
```

```
a [Aa]

# pattern "_" matches space and optional quotation marks:
_ [ ]['‘"“]?

# pattern "vow" matches words beginning with vowels:
vow [aeiouAEIOU]\w*

# pattern "con" matches words beginning with consonants:
con [bcdfghj-np-tv-zBCDFGHJ-NP-TV-Z]\w*

# pattern "etc" matches other word parts separated by hyphen, endash or apostrophes:
etc [-–'’\w]*

# rules ("aA", "aAN", "aB" sets are defined at the end of the file)

{a}n{_}{vow}{etc} <- {vow} in aA or {vow}.lower() in aA -> {a}{_}{vow}{etc} # Did you mean:

a{_}{vow}{etc} <- ({vow} <> {vow}.upper()) and not ({vow} in aA or
          {vow}.lower() in aA) and spell({vow}) -> an{_}{vow}{etc} # Bad article?

a{_}{con}{etc} <- {con} in aAN or {con}.lower() in aAN -> an{_}{con}{etc} # Did you mean:

{a}n{_}{con}{etc} <- ({con} <> {con}.upper()) and not ({con} in aA or
          {con}.lower() in aAN) and not {con} in aB and spell({con}) -> {a}{_}{con}{etc} # Bad article?

# rules for sentences beginning with "A"

^A{_}{vow}{etc} <- ({vow} <> {vow}.upper()) and not ({vow} in aA or
          {vow}.lower() in aA) and spell({vow}) -> An{_}{vow}{etc} # Bad article?

^A{_}{con}{etc} <- {con} in aAN or {con}.lower() in aAN -> An{_}{con}{etc} # Did you mean:

# check numbers

nvow (8[0-9]*|1[18](000)*)(th)? # 8, 8th, 11, 11th, 18, 18th, 11000, 11000th...

a{_}{nvow}{etc} -> an{_}{nvow}{etc} # Did you mean:
^A{_}{nvow}{etc} -> An{_}{nvow}{etc} # Did you mean:

ncon [0-79][0-9]*

{a}n{_}{ncon}{etc} <- not {ncon}[:2] in ["11", "18"] -> {a}{_}{ncon}{etc} # Did you mean:

# paragraph capitalization

[code]
# pattern matching for common English abbreviations
abbrev = re.compile("(?i)\b([a-z]|acct|approx|appt|apr|apt|assoc|asst|aug|ave|avg|co(nt|rp)?|ct|dec|defn|dept|dr|eg|equip|esp|est|
etc|excl|ext|feb|fri|ft|govt?|hrs?|ib(id)?|ie|in(c|t)?|jan|jr|jul|lit|ln|mar|max|mi(n|sc)?|mon|Mrs?|mun|natl?|neg?|no(rm|s|v)?|nw|
obj|oct|org|orig|pl|pos|prev|proj|psi|qty|rd|rec|rel|reqd?|resp|rev|sat|sci|se(p|pt)?|spec(if)?|sq|sr|st|subj|sun|sw|temp|thurs|tot|
tues|univ|var|vs)\.")

[word]

# condition: the paragraph begins with a lowercase letter and it contains real sentence boundaries.

low [a-z]+

^{low} <- re.match("[a-z].*[.?!] [A-Z]", TEXT) and not abbrev.search(TEXT) -> = {low}.capitalize() # Missing capitalization?

# optional sentence capitalization

^{low} <- option("cap") and not abbrev.search(TEXT) -> = {low}.capitalize() # Missing capitalization?

# punctuation

[code]

punct = { "?": "question mark", "!": "exclamation mark",
          ",": "comma", ":": "colon", ";": "semicolon",
          "(": "opening parenthesis", ")": "closing parenthesis",
          "[": "opening square bracket", "]": "closing square bracket",
          u"“": "opening quotation mark", u"”": "closing quotation mark"}

[char]

" ([.?!,:;)”\[\]])\b" -> "\1 "          # Reversed space and punctuation?
" +[.]" <- LOCALE.Country == "US" -> . # Extra space before the period?
" +[.]" <- LOCALE.Country != "US" -> . # Extra space before the full stop?
" +([?!,:;)”\]])" -> \1      # = "Extra space before the " + punct[\1] + "?"
"([([“]) " -> \1    # = "Extra space after the " + punct[\1] + "?"

TEST: ( item ) -> (item)
TEST: A small - but important - example. -> A small – but important – example.

# En dash and em dash

\b(---?| --? )\b <- not option("ndash") and not option("mdash") -> " – \n—" # En dash or em dash:
\b(---?| --? |—)\b <- option("ndash") and not option("mdash") -> " – " # En dash:
\b(---?| --? | – )\b <- option("mdash") -> — # Em dash:

# multiplication sign

number \d+([.]\d+)?

{number}(x| x ){number} <- option("times") -> {number}×{number} # Multiplication sign.
TEST: 800x600 -> 800×600

# missing space

abc [a-z]+
ABC [A-Z]+
Abc [a-zA-Z]+
pun [?!,:;%‰‱°“”‘]

{Abc}{pun}{Abc} -> {Abc}{pun} {Abc}    # Missing space?
{abc}[.]{ABC} -> {abc}. {ABC}          # Missing space?
TEST: missing,space -> missing, space
TEST: missing.Space -> missing. Space

[)] <- option("pair") and not "(" in TEXT -> # Extra closing parenthesis?
[(] <- option("pair") and TEXT[-1] in u"?!;:”’" and not ")" in TEXT -> # Extra opening parenthesis?
(?<![0-9])” <- option("pair") and not u"“" in TEXT -> # Extra quotation mark?
(?<=[0-9])” <- option("apostrophe") and not u"“" in TEXT -> ″\n # Bad double prime or extra quotation mark?
“ <- option("pair") and TEXT[-1] in u"?!;:”’" and not u"”" in TEXT -> # Extra quotation mark?

"[.]{3}" <- option("ellipsis") -> "…"  # Ellipsis.

\b {2,3}(\b|$) <- option("spaces") -> "\1 " # Extra space.
TEST: Extra  space -> Extra space
TEST: End... -> End…

(^|\b|{pun}|[.]) {2,3}(\b|$) <- option("spaces2") -> "\1 " # Extra space.
TEST: Extra  space -> Extra space
TEST: End... -> End…

(^|\b|{pun}|[.]) {4,}(\b|$) <- option("spaces3") -> "\1 \n " # Change multiple spaces to a single space or a tabulator:

# quotation

# Using typographic quotation marks is the

(?i)[\"“”‟„]({abc}[^\"“”‟„]*)[\"“‟] <- option("quotation") -> “\1” # Quotation marks.
(?i)[\"”‟„]({abc}[^\"“”‟„]*)[\"“”‟] <- option("quotation") -> “\1” # Quotation marks.

(?i)'{abc}' <- option("apostrophe") -> ‘{abc}’ # Quotation marks.
(?i)[\"”‟„]({abc}[^\"“”‟„]*)[\"“”‟] <- option("apostrophe") -> “\1” # Quotation marks.

# apostrophe

w \w*
(?i){Abc}'{w} <- option("apostrophe") -> {Abc}’{w}         # Replace typewriter apostrophe or quotation mark:
TEST: o'clock -> o’clock
TEST: singers' voices -> singers’ voices

(?<= )'{Abc} <- option("apostrophe") -> ‘{Abc}\n’{Abc} # Replace typewriter quotation mark or apostrophe:
^'{Abc} <- option("apostrophe") -> ‘{Abc}\n’{Abc} # Replace typewriter quotation mark or apostrophe:

# formats

# Thousand separators: 10000 -> 10,000  (common) or 10 000 (ISO standard)

# definitions
d         \d\d\d             # name definition: 3 digits
d2        \d\d               # 2 digits
D         \d{1,3}            # 1, 2 or 3 digits

# ISO thousand separators: space, here: narrow no-break space (U+202F)
\b{d2}{d}\b          <- option("numsep") -> {d2},{d}\n{d2} {d}                          # Use thousand separator (common or ISO).
\b{D}{d}{d}\b        <- option("numsep") -> {D},{d},{d}\n{D} {d} {d}           # Use thousand separators (common or ISO).
\b{D}{d}{d}{d}\b <- option("numsep") -> {D},{d},{d},{d}\n{D} {d} {d} {d}       # Use thousand separators (common or ISO).
TEST: 1234567890 -> 1,234,567,890\n1 234 567 890

# word duplication

[word]

{Abc} \1 <- option("dup") -> {Abc} # Word duplication?

# Optional grammar checking

([Tt])his {abc} <- option("grammar") and morph({abc}, "Ns") -> \1hese {abc}\n\1his, {abc} # Did you mean:

with it['’]s <- option("grammar") -> with its\nwith, it’s # Did you mean:

[Word]

(it|s?he) don['’]t <- option("grammar") -> \1 doesn’t # Did you mean:

################### measurements #########################

[word]

# Temperature

([-−]?\d+(?:[,.]\d+)*) (°F|Fahrenheit) <- option("metric") -> = measurement(\1, "F", "C", u" °C", ".", ",") # Convert to Celsius:
([-−]?\d+(?:[,.]\d+)*) (°C|Celsius) <- option("nonmetric") -> = measurement(\1, "C", "F", u" °F", ".", ",") # Convert to Fahrenheit:

# Length

([-−]?\d+(?:[,.]\d+)*(?: 1/2| ?½)?) (ft|foot|feet)(?! [1-9]) <- option("metric") -> =
          measurement(\1, "ft", "cm", " cm", ".", ",") + "\n" +
          measurement(\1, "ft", "m", " m", ".", ",") # Convert to metric:

([-−]?\d+(?:[,.]\d+)*(?: 1/2| ?½)?) ft[.]? ([0-9]+(?: 1/2| ?½)?) in <- option("metric") -> =
          measurement(\1 + "*12+" + \2, "in", "cm", " cm", ".", ",") + "\n" +
          measurement(\1 + "*12+" + \2, "in", "m", " m", ".", ",") # Convert to metric:

([-−]?\d+(?:[,.]\d+)*(?: 1/2| ?½)?) in <- option("metric") -> =
          measurement(\1, "in", "mm", " mm", ".", ",") + "\n" +
          measurement(\1, "in", "cm", " cm", ".", ",") + "\n" +
          measurement(\1, "in", "m", " m", ".", ",") # Convert to metric:

([-−]?\d+(?:[,.]\d+)*) mm <- option("nonmetric") -> =
          measurement(\1, "mm", "in", " in", ".", ",") # Convert from metric:

([-−]?\d+(?:[,.]\d+)*) cm <- option("nonmetric") -> =
          measurement(\1, "cm", "in", " in", ".", ",") + "\n" +
          measurement(\1, "cm", "ft", " ft", ".", ",") # Convert from metric:

([-−]?\d+(?:[,.]\d+)*) (m|meter|metre) <- option("nonmetric") -> =
          measurement(\1, "m", "in", " in", ".", ",") + "\n" +
          measurement(\1, "m", "ft", " ft", ".", ",") + "\n" +
          measurement(\1, "m", "mi", " mi", ".", ",") # Convert from metric:

([-−]?\d+(?:[,.]\d+)*(?: 1/2| ?½)?) miles? <- option("metric") -> =
          measurement(\1, "mi", "m", " m", ".", ",") + "\n" +
          measurement(\1, "mi", "km", " km", ".", ",") # Convert to metric:

([-−]?\d+(?:[,.]\d+)*) km <- option("nonmetric") -> =
          measurement(\1, "km", "mi", " mi", ".", ",") # Convert to miles:

([-−]?\d+(?:,\d+)?) (yd|yards?) <- option("metric") -> = measurement(\1, "yd", "m", " m", ".", ",") # Convert to metric:

# Volume

([-−]?\d+(?:,\d+)?) (gal(lons?)?) <- option("metric") -> =
          measurement(\1, "gal", "l", " l", ".", ",") + "\n" +
          measurement(\1, "uk_gal", "l", " l (in UK)", ".", ",") # Convert to metric:

([-−]?\d+(?:,\d+)?) (pint) <- option("metric") -> =
          measurement(\1, "pt", "dl", " dl", ".", ",") + "\n" +
          measurement(\1, "uk_pt", "dl", " dl (in UK)", ".", ",") + "\n" +
          measurement(\1, "pt", "l", " l", ".", ",") + "\n" +
          measurement(\1, "uk_pt", "l", " l (in UK)", ".", ",") # Convert to metric:

([-−]?\d+(?:,\d+)?) (l|L|litres?|liters?) <- option("nonmetric") -> =
          measurement(\1, "l", "gal", " gal", ".", ",") + "\n" +
          measurement(\1, "l", "gal", " gal (in UK)", ".", ",") # Convert to gallons:

# Weight

([-−]?\d+(?:[,.]\d+)*) lbs?[.]? <- option("metric") -> =
          measurement(\1, "lbm", "kg", " kg", ".", ",") # Convert to metric:
([-−]?\d+(?:[,.]\d+)*) kg[.]? <- option("nonmetric") -> =
          measurement(\1, "kg", "lbm", " lb", ".", ",") # Convert to pounds:

# Speed

([-−]?\d+(?:[,.]\d+)*) mph <- option("metric") -> = measurement(\1, "mph", "km/h", " km/h", ".", ",") # Convert to km/hour:
([-−]?\d+(?:[,.]\d+)*) km/h <- option("nonmetric") -> = measurement(\1, "km/h", "mph", " mph", ".", ",") # Convert to miles/hour:

[code]

aA = set(["eucalypti", "eucalyptus", "Eucharist", "Eucharistic",
"euchre", "euchred", "euchring", "Euclid", "euclidean", "Eudora",
"eugene", "Eugenia", "eugenic", "eugenically", "eugenicist",
"eugenicists", "eugenics", "Eugenio", "eukaryote", "Eula", "eulogies",
"eulogist", "eulogists", "eulogistic", "eulogized", "eulogizer",
"eulogizers", "eulogizing", "eulogy", "eulogies", "Eunice", "eunuch",
"eunuchs", "Euphemia", "euphemism", "euphemisms", "euphemist",
"euphemists", "euphemistic", "euphemistically", "euphonious",
"euphoniously", "euphonium", "euphony", "euphoria", "euphoric",
"Euphrates", "euphuism", "Eurasia", "Eurasian", "Eurasians", "eureka",
"eurekas", "eurhythmic", "eurhythmy", "Euridyce", "Euripides", "euripus",
"Euro", "Eurocentric", "Euroclydon", "Eurocommunism", "Eurocrat",
"eurodollar", "Eurodollar", "Eurodollars", "Euromarket", "Europa",
"Europe", "European", "Europeanisation", "Europeanise", "Europeanised",
"Europeanization", "Europeanize", "Europeanized", "Europeans", "europium",
"Eurovision", "Eustace", "Eustachian", "Eustacia", "euthanasia",
"Ewart", "ewe", "Ewell", "ewer", "ewers", "Ewing", "once", "one",
"oneness", "ones", "oneself", "onetime", "oneway", "oneyear", "u",
"U", "UART", "ubiquitous", "ubiquity", "Udale", "Udall", "UEFA",
"Uganda", "Ugandan", "ugric", "UK", "ukase", "Ukraine", "Ukrainian",
"Ukrainians", "ukulele", "Ula", "ululated", "ululation", "Ulysses",
"UN", "unanimity", "unanimous", "unanimously", "unary", "Unesco",
"UNESCO", "UNHCR", "uni", "unicameral", "unicameralism", "Unicef",
"UNICEF", "unicellular", "Unicode", "unicorn", "unicorns", "unicycle",
"unicyclist", "unicyclists", "unidimensional", "unidirectional",
"unidirectionality", "unifiable", "unification", "unified", "unifier",
"unifilar", "uniform", "uniformally", "uniformed", "uniformer",
"uniforming", "uniformisation", "uniformise", "uniformitarian",
"uniformitarianism", "uniformity", "uniformly", "uniformness", "uniforms",
"unify", "unifying", "unijugate", "unilateral", "unilateralisation",
"unilateralise", "unilateralism", "unilateralist", "unilaterally",
"unilinear", "unilingual", "uniliteral", "uniliteralism", "uniliteralist",
"unimodal", "union", "unionism", "unionist", "unionists", "unionisation",
"unionise", "unionised", "unionising", "unionization", "unionize",
"unionized", "unionizing", "unions", "unipolar", "uniprocessor",
"unique", "uniquely", "uniqueness", "uniquer", "Uniroyal", "unisex",
"unison", "Unisys", "unit", "Unitarian", "Unitarianism", "Unitarians",
"unitary", "unite", "united", "unitedly", "uniter", "unites", "uniting",
"unitize", "unitizing", "unitless", "units", "unity", "univ", "Univac",
"univalent", "univalve", "univariate", "universal", "universalisation",
"universalise", "universalised", "universaliser", "universalisers",
"universalising", "universalism", "universalist", "universalistic",
"universality", "universalisation", "universalization", "universalize",
"universalized", "universalizer", "universalizers", "universalizing",
"universally", "universalness", "universe", "universes", "universities",
"university", "univocal", "Unix", "uracil", "Urals", "uranium", "Uranus",
"uranyl", "urate", "urea", "uremia", "uremic", "ureter", "urethane",
"urethra", "urethral", "urethritis", "Urey", "Uri", "uric", "urinal",
"urinalysis", "urinary", "urinated", "urinating", "urination", "urine",
"urogenital", "urokinase", "urologist", "urologists", "urology",
"Uruguay", "Uruguayan", "Uruguayans", "US", "USA", "usable", "usage",
"usages", "use", "used", "useful", "usefulness", "usefully", "useless",
"uselessly", "uselessness", "Usenet", "user", "users", "uses", "using",
"usual", "usually", "usurer", "usurers", "usuress", "usurial", "usurious",
"usurp", "usurpation", "usurped", "usurper", "usurping", "usurps",
"usury", "Utah", "utensil", "utensils", "uterine", "uterus", "Utica",
"utilitarian", "utilitarianism", "utilities", "utility", "utilizable",
"utilization", "utilize", "utilized", "utilizes", "utilizing", "utopia",
"utopian", "utopians", "utopias", "Utrecht", "Uttoxeter", "uvula",
"uvular"])

aAN = set(["f", "F", "FBI", "FDA", "heir", "heirdom", "heired",
"heirer", "heiress", "heiring", "heirloom", "heirship", "honest",
"honester", "honestly", "honesty", "honor", "honorable", "honorableness",
"honorably", "honorarium", "honorary", "honored", "honorer", "honorific",
"honoring", "honors", "honour", "honourable", "honourableness",
"honourably", "honourarium", "honourary", "honoured", "honourer",
"honourific", "honouring", "Honours", "hors", "hour", "hourglass", "hourlong",
"hourly", "hours", "l", "L", "LCD", "m", "M", "MBA", "MP", "mpg", "mph",
"MRI", "MSc", "MTV", "n", "N", "NBA", "NBC", "NFL", "NGO", "NHL", "r",
"R", "s", "S", "SMS", "sos", "SOS", "SPF", "std", "STD", "SUV", "x",
"X", "XML"])

aB = set(["H", "hallucination", "haute", "hauteur", "herb", "herbaceous", "herbal",
"herbalist", "herbalism", "heroic", "hilarious", "historian", "historic", "historical",
"homage", "homophone", "horrendous", "hospitable", "horrific", "hotel", "hypothesis", "Xmas"])

def measurement(mnum, min, mout, mstr, decimal, remove):
    if min == "ft" or min == "in" or min == "mi":
          mnum = mnum.replace(" 1/2", ".5").replace(u" ½", ".5").replace(u"½",".5")
    m = calc("CONVERT_ADD", (float(eval(mnum.replace(remove, "").replace(decimal, ".").replace(u"−", "-"))), min, mout))
    a = list(set([str(calc("ROUND", (m, 0)))[:-2], str(calc("ROUND", (m, 1))), str(calc("ROUND", (m, 2))), str(m)])) # remove
duplicated rounded items
    a.sort(lambda x, y: len(x) - len(y)) # sort by string length
    return join(a, mstr + "\n").replace(".", decimal).replace("-", u"−") + mstr
```