Welcome to The Carpentries Etherpad!

This pad is synchronized as you type, so that everyone viewing this page sees the same text. This allows you to collaborate seamlessly on documents.

Use of this service is restricted to members of The Carpentries community; this is not for general purpose use (for that, try https://etherpad.wikimedia.org).

Users are expected to follow our code of conduct: https://docs.carpentries.org/topic_folders/policies/code-of-conduct.html

All content is publicly available under the Creative Commons Attribution License:
https://creativecommons.org/licenses/by/4.0/

Regular Expressions

https://librarycarpentry.org/lc-data-intro/01-regular-expressions/index.htm
https://www.cs.cmu.edu/~pattis/15-1XX/common/handouts/ascii.html

ALT + 0065
https://www.ascii-code.com/
https://home.unicode.org/

1)
what is ^[Oo]rgani.e\b going to match?
Organize
organize, Organize, Organise, organise, Organi7e,
organise
organize
Organise
organise
organi3e
Organize
organize
organite

Organize, organize

organize

Organi(any 1 char)e(end) | organi(any 1 char)e(end)

organize
organi

Organize
organi3e
Organite
Organipe
(upper or lower case o)rgani(any character)e
orgazize Organize
Organize
organise
organized
organize, Organize
organised
organixe
organi_e [any characters at underscore]
organike
organise
organize
Organize
Organite
organize, Organize

organiie
organice
organyze
organyse
organys
organyz
Organise, organise, organize, Organize
organize, Organize

2)
Speed check? (write + on the next line)
go-faster:
go-slower: ++++++++++
okay: ++++++++

Excercise - 10 min
https://librarycarpentry.org/lc-data-intro/01-regular-expressions/index.html#:~:text=Exercise

https://librarycarpentry.org/lc-data-intro/02-match-extract-strings/index.html

Excercise - 10 min
https://librarycarpentry.org/lc-data-intro/02-match-extract-strings/index.html#:~:text=Exercise

Extracting a substring in Google Sheets using regex
https://docs.google.com/spreadsheets/d/1x8kFB4kZ1LazdTWzfpLzJEbfTLI0ryoXCu30TEqPK9s/edit#gid=1508806902
REGEXEXTRACT(G2,"-?\d+\.\d+, -?\d+\.\d+")

Tryouts:

https://regexcrossword.com/
https://regex101.com/
https://regexr.com/

Quizes
https://librarycarpentry.org/lc-data-intro/03-quiz/index.html

https://librarycarpentry.org/lc-data-intro/04-exercises/index.html

- Need help? Post your questions below.

Shell lessons

https://swcarpentry.github.io/shell-novice/

1. **Commands**
2. ls
3. cd
4. pwd

https://swcarpentry.github.io/shell-novice/02-filedir/index.html

23-10-2021

Python Setup:

https://swcarpentry.github.io/python-novice-inflammation/setup.html

Anaconda Installation:

https://www.anaconda.com/products/individual

Lesson Materials:

https://swcarpentry.github.io/python-novice-inflammation/data/python-novice-inflammation-data.zip
https://swcarpentry.github.io/python-novice-inflammation/code/python-novice-inflammation-code.zip

1. Download

2. Create a folder called swc-python on your Desktop.
3. Move downloaded files to swc-python.
4. Unzip the files.

>> You should see two folders called data and code in the swc-python directory on your Desktop.

To download files using terminal

curl https://swcarpentry.github.io/python-novice-inflammation/data/python-novice-inflammation-data.zip -O -J -L

curl https://swcarpentry.github.io/python-novice-inflammation/code/python-novice-inflammation-code.zip -O -J -L

Python Lessons

Running jupyter notebook
https://pythonforundergradengineers.com/opening-a-jupyter-notebook-on-windows.html

https://swcarpentry.github.io/python-novice-inflammation/

Visualizing Data
https://swcarpentry.github.io/python-novice-inflammation/03-matplotlib/index.html

```
fig = matplotlib.pyplot.figure (figsize = (10.0, 3.0))

axes1 = fig.add_subplot(1, 3, 1)
axes2 = fig.add_subplot(1, 3 ,2)
axes3 = fig.add_subplot(1, 3, 3)

axes1.set_ylabel('average')
axes1.plot(numpy.mean(data,axis=0))

axes2.set_ylabel('max')
axes2.plot(numpy.max(data,axis=0))

axes3.set_ylabel('min')
axes3.plot(numpy.min(data,axis=0))

fig.tight_layout()
```

```
matplotlib.pyplot.savefig('inflammation.png')
matplotlib.pyplot.show()
```

https://swcarpentry.github.io/python-novice-inflammation/04-lists/index.html

https://swcarpentry.github.io/python-novice-inflammation/05-loop/index.html

https://swcarpentry.github.io/python-novice-inflammation/06-files/index.html

```
filenames = sorted(glob.glob('inflammation*.csv'))
filenames = filenames[0:3]
for filename in filenames:
    print(filename)

    data = numpy.loadtxt(fname=filename, delimiter=',')
    fig = matplotlib.pyplot.figure(figsize=(10.0,3.0))

    axes1 = fig.add_subplot(1, 3, 1)
    axes2 = fig.add_subplot(1, 3 ,2)
    axes3 = fig.add_subplot(1, 3 ,3)

    axes1.set_ylabel('average')
    axes1.plot(numpy.mean(data, axis=0))

    axes2.set_ylabel('max')
    axes2.plot(numpy.max(data, axis=0))

    axes3.set_ylabel('min')
    axes3.plot(numpy.min(data, axis=0))

    fig.tight_layout()
    matplotlib.pyplot.show()
```

```
def visualize(filename):

    data = numpy.loadtxt(fname=filename, delimiter=',')

    fig = matplotlib.pyplot.figure(figsize=(10.0, 3.0))

    axes1 = fig.add_subplot(1, 3, 1)
```

```
    axes2 = fig.add_subplot(1, 3, 2)
    axes3 = fig.add_subplot(1, 3, 3)

    axes1.set_ylabel('average')
    axes1.plot(numpy.mean(data, axis=0))

    axes2.set_ylabel('max')
    axes2.plot(numpy.max(data, axis=0))

    axes3.set_ylabel('min')
    axes3.plot(numpy.min(data, axis=0))

    fig.tight_layout()
    matplotlib.pyplot.show()




def detect_problems(filename):

    data = numpy.loadtxt(fname=filename, delimiter=',')

    if numpy.max(data, axis=0)[0] == 0 and numpy.max(data, axis=0)[20] == 20:
        print('Suspicious looking maxima!')
    elif numpy.sum(numpy.min(data, axis=0)) == 0:
        print('Minima add up to zero!')
    else:
        print('Seems OK!')
```

08-11-2021

Lexical Dispersion Plot

```
============================================================
import nltk
nltk.download("inagural")
============================================================


============================================================
from nltk.corpus import inaugural
```

```
from nltk.text import Text
inaugural_tokens = inaugural.words()
inaugural_texts = Text(inaugural_tokens)
============================================================


============================================================
print(inaugural_tokens)
============================================================


============================================================
print(inaugural_texts)
============================================================


============================================================
from nltk.draw.dispersion import dispersion_plot
import matplotlib.pyplot as plt

plt.figure(figsize = (12, 9))
targets = ['great', 'good', 'tax', 'work', "change"]
dispersion_plot(inaugural_texts, targets, ignore_case = True, title = "Lexical Dispersion Plot")
============================================================
```

Plotting frequency over time

```
============================================================
============
from nltk.probability import ConditionalFreqDist

plt.rcParams['figure.figsize'] = (12, 9)

# targets=['great', 'good', 'tax', 'work', "change", 'wom[ae]n']
targets=['great', 'good', 'tax', 'work', "change", 'woman', 'women']

cfd = nltk.ConditionalFreqDist((target, fileid[:4])
    for fileid in inaugural.fileids()
    for word in inaugural.words(fileid)
    for target in targets
    if word.lower().startswith(target))
cfd.plot()
============================================================
============
```

```
==========
plt.rcParams["figure.figsize"] = (12, 9)
targets=['m[ea]n']
cfd = nltk.ConditionalFreqDist((target, fileid[:4])
    for fileid in inaugural.fileids()
    for word in inaugural.words(fileid)
    for target in targets
    if word.lower().startswith(target))
cfd.plot()
==========
```

Part-of Speech Tagging Text

```
==========
import nltk
nltk.download('averaged_perceptron_tagger')
from nltk.tag import pos_tag
nltk.download('inaugural')
from nltk.corpus import inaugural
==========
```

```
==========
inaugural_tokens=inaugural.words()
print(inaugural_tokens)
==========
```

```
==========
inaugural_tokens_trump = inaugural.words(inaugural.fileids()[0:-1])
print(inaugural_tokens_trump)
==========
```

```
==========
tagged_inaugural_tokens = nltk.pos_tag(inaugural_tokens)
```

```
tagged_inaugural_tokens[:20]
==========================================================
==========


==========================================================
==========
nouns = []
nouns = [word for (word, pos) in tagged_inaugural_tokens if (pos == 'NN' or pos ==
'NNS')]
nouns[:20]
==========================================================
==========


==========================================================
==========
from nltk.probability import FreqDist
fdist = FreqDist(nouns)
fdist.plot(30,title='Frequency distribution for 30 most common nouns in the
inaugural corpus')
==========================================================
==========


==========================================================
==========
from wordcloud import WordCloud
%matplotlib inline
import matplotlib.pyplot as plt
cloud = WordCloud(max_font_size=60,colormap="hsv").generate(' '.join(nouns))
plt.rcParams["figure.figsize"] = (16,12)
plt.imshow(cloud, interpolation='bilinear')
plt.axis('off')
plt.show()
==========================================================
==========
```

POS Tags:

CC Coordinating Conjunction
CD Cardinal Digit
DT Determiner
EX Existential There. Example: “there is” … think of it like “there exists”)
FW Foreign Word.
IN Preposition/Subordinating Conjunction.

JJ Adjective.
JJR Adjective, Comparative.
JJS Adjective, Superlative.
LS List Marker 1.
MD Modal.
NN Noun, Singular.
NNS Noun Plural.
NNP Proper Noun, Singular.
NNPS Proper Noun, Plural.
PDT Predeterminer.
POS Possessive Ending. Example: parent’s
PRP Personal Pronoun. Examples: I, he, she
PRP$ Possessive Pronoun. Examples: my, his, hers
RB Adverb. Examples: very, silently,
RBR Adverb, Comparative. Example: better
RBS Adverb, Superlative. Example: best
RP Particle. Example: give up
TO to. Example: go ‘to’ the store.
UH Interjection. Example: errrrrrrrm
VB Verb, Base Form. Example: take
VBD Verb, Past Tense. Example: took
VBG Verb, Gerund/Present Participle. Example: taking
VBN Verb, Past Participle. Example: taken
VBP Verb, Sing Present, non-3d take
VBZ Verb, 3rd person sing. present takes
WDT wh-determiner. Example: which
WP wh-pronoun. Example: who, what
WP$ possessive wh-pronoun. Example: whose
WRB wh-abverb. Example: where, when

WEB Scraping

Introduction

$x("/html/head/title/text()")

$x("/html/body/div/article/h1")

$x("/html/body/div/h1[@id='introduction']")

$x("/html/body/div/article/blockquote")

$x("//blockquote")

$x("//h2[@id = 'select-this-challenge-box']/..")[0]

html/body/div/h3/@id='exercises-2'

html/body/div/h3/@id='exercises-4'

//h1/@id='references' or @id='introduction'

//author[contains(.,"Matt")]

Up next: "*Manually scrape data using browser extensions*"

08-11-2021

```
import nltk
import numpy
import string
import matplotlib.pyplot as plt

from nltk.tokenize import word_tokenize

nltk.download("punkt")

humpty_string = "Humpty Dumpty sat on a wall, Humpty Dumpty had a great fall; All the king's horses and all the king's men couldn't put Humpty together again."
humpty_tokens = word_tokenize(humpty_string)

humpty_tokens[0:10]

lower_hompty_tokens = [word.lower() for word in humpty_tokens]
lower_humpty_tokens[0:6]

file = open('nls-text-indiaPapers/74457530.txt','r')
india_raw = file.read()
india_tokens = word_tokenize(india_raw)
```

```
lower_india_tokens = [word.lower() for word in india_tokens]
lower_india_tokens[0:10]

len(lower_india_tokens)

from nltk.corpus import PlaintextCorpusReader
corpus_root = 'nls-text-indiaPapers/'
wordlists = PlaintextCorpusReader(corpus_root, '.*txt', encoding = 'latin1') # (.*
regular expression to match anything and followed by "txt" at the end)
corpus_tokens = wordlists.words()
print(corpus_tokens[:10])

len(corpus_tokens)

lower_corpus_tokens = [str(word).lower() for word in corpus_tokens]
lower_corpus_tokens[0:10]

print(corpus_tokens[:30])

from nltk.text import Text
t = Text(lower_india_tokens)
t.concordance('woman')
```

https://librarycarpentry.org/lc-webscraping/04-scrapy/index.html

```
def parse(self, response):
    for url in response.xpath('//td[@class="views-field views-field-field-full-name-by-last-name is-
active"]/a/@href').extract()[:5]:
        print(response.urljoin(url))
```

###

```
    def parse(self, response):
        for url in response.xpath('//td[@class="views-field views-field-field-full-name-by-last-name is-
active"]/a/@href').extract()[:5]:
            full_url = response.urljoin(url)
            print("Found url: "+full_url)
            yield scrapy.Request(full_url, callback=self.get_details)



    def get_details(self, response):
        print("Visited URL: "+response.url)
```

Examples of Xpath Query for extracting data not structured in a table

$x("//div[2]/div[2]/div[1]/h3[1]/following-sibling::text()")

XPATH : //div[2]/div[2]/div

./h2
concat(./h3[1],./h3[1]/following-sibling::text()[1],./h3[1]/following-sibling::text()[2])
concat(./h3[2],./p)

Contents of items.py

```
import scrapy

class OntariomppsItem(scrapy.Item):
    # define the fields for your item here like:
    name = scrapy.Field()
    url = scrapy.Field()
    party = scrapy.Field()
```

Contents of ontariompps/spiders/mppaddresses.py

```
import scrapy
from ontariompps.items import OntariomppsItem

class MppaddressesSpider(scrapy.Spider):
    name = 'mppaddresses'
    allowed_domains = ['www.ola.org/en/']
    start_urls = ['http://www.ola.org/en/members/current/']

    def parse(self, response):
        for data in response.xpath('//tbody/tr'):
            yield self.get_details(data,response)

    def get_details(self, data, response):
        item = OntariomppsItem()
        item['name']=data.xpath('normalize-space(//td[@class="views-field views-field-field-full-name-by-last-name is-active"]/a/text())').extract_first()
        item['url']= response.urljoin(data.xpath('normalize-space(//td[@class="views-field views-field-field-full-name-by-last-name is-active"]/a/@href)').extract_first())
        item['party']=data.xpath('normalize-space(//td[@class="views-field views-field-field-party"]/text())').extract_first()
        data.remove()

        return item
```

24-11-2021

URLS:
https://www.nltk.org/data.html
https://www.nltk.org/book/
https://www.nltk.org/book/ch01.html
https://www.nltk.org/

```
import nltk
nltk.download()

from nltk.book import *

text4.dispersion_plot(["citizens", "democracy", "freedom", "duties", "America"])

text3.generate()

len(text3)

len(set(text3))

len(set(text3)) / len(text3)

text3.count("smote")

100 * text4.count("a") / len(text4)
```

27-11-2021

```
from nltk.book import *
import nltk
def lexical_diversity(text):
    return len(set(text)) / len(text)

def percentage(count, total):
    return 100*count/total

lexical_diversity(text3)
lexical_diversity(text5)
percentage(4, 5)
percentage(text4.count('a'), len(text4))

sent1 = ['call', 'me', 'Ishmael', '.']
```

```
sent1

len(sent1)

lexical_diversity(sent1)

sent2

sent3

sent4

sent4 + sent1

sent1.append("Some")

sent1

text4[173]

text1.count("awaken")

text4.count('awaken')

text2.count('awaken')

text2.count('awytjygaken')
text1.count('heaven')

text5[16715:16735]

vocab = set(text1)
vocab_size = len(vocab)

vocab_size

fdist1 = FreqDist(text1)

print(fdist1)

fdist1.most_common(50)

fdist1.plot(50, cumulative=True)

fdist1['whale']

fdist1.hapaxes()
```

```
len(fdist1.hapaxes())

V = set(text1)

long_words = [w for w in v if len(w) > 15]

sorted(long_words)

fdist5 = FreqDist(text5)
sorted(w for w in set(text5) if len(w) > 7 and fdist5[w] > 7)

list(bigrams(['more', 'is', 'said', 'than', 'done']))

text4.collocations()

text8.collocations()
```

Carpentry Workshop

Web Scraping using Python: https://librarycarpentry.org/lc-webscraping/04-scrapy/index.html

```
scrapy startproject ontariompps
scrapy genspider mppaddresses https://www.ola.org/en/members/current/
```

```
    def parse(self, response):
        for data in response.xpath('//tbody/tr'):
            yield self.get_details(data, response)

    def get_details(self, data, response):
        item = OntariomppsItem()
        item['name']=data.xpath('normalize-space(//td[@class="views-field views-field-field-full-name-by-last-name is-active"]/a/text())').extract_first()
        item['url']= data.xpath('normalize-space(//td[@class="views-field views-field-field-full-name-by-last-name is-active"]/a/@href)').extract_first()
        item['url']= response.urljoin(item['url'])
        item['party']=data.xpath('normalize-space(//td[@class="views-field views-field-field-party"]/text())').extract_first()
        data.remove()
        #print(item)
        return item
```

```
##############################

import scrapy
from ontariompps.items import OntariomppsItem

class MppaddressesSpider(scrapy.Spider):
    name = 'mppaddresses'
    allowed_domains = ['www.ola.org']
    start_urls = ['http://www.ola.org/en/members/current/']

    def parse(self, response):
        #for data in response.xpath('//tbody/tr'):
        #    yield self.get_details(data, response)
        for url in response.xpath('//td[@class="views-field views-field-field-full-name-by-last-name is-active"]/a/@href').extract()[:5]:
            absoluteUrl = response.urljoin(url)
            yield scrapy.Request(absoluteUrl, callback=self.getMoreDetails)

    def get_details(self, data, response):
        item = OntariomppsItem()
        item['name']=data.xpath('normalize-space(//td[@class="views-field views-field-field-full-name-by-last-name is-active"]/a/text())').extract_first()
        item['url']= data.xpath('normalize-space(//td[@class="views-field views-field-field-full-name-by-last-name is-active"]/a/@href)').extract_first()
        item['url']= response.urljoin(item['url'])
        item['party']=data.xpath('normalize-space(//td[@class="views-field views-field-field-party"]/text())').extract_first()
        data.remove()
        #print(item)
        return item

    def getMoreDetails(self, response):
        item = OntariomppsItem()
        item['email'] = response.xpath('//div[contains(concat(" ",normalize-space(@class)," "),"  field--type-email ")]/descendant::a/text()').extract_first()
        item['url'] = response.url

        yield item
```

Programming with R

http://swcarpentry.github.io/r-novice-inflammation/
http://swcarpentry.github.io/r-novice-inflammation/01-starting-with-data/index.html

wget http://swcarpentry.github.io/r-novice-inflammation/data/r-novice-inflammation-data.zip
OR
curl -O http://swcarpentry.github.io/r-novice-inflammation/data/r-novice-inflammation-data.zip

Then,
unzip r-novice-inflammation-data.zip

```
read.csv(file="data/inflammation-01.csv", header = FALSE)
weight_kg <- 55

weight_kg
(weight_lb <- 2.2 * weight_kg)

dat <- read.csv(file="data/inflammation-01.csv", header = FALSE)
head(dat)
class(dat)
dim(dat)
```

POS tag list:

CC coordinating conjunction
CD cardinal digit
DT determiner
EX existential there (like: "there is" ... think of it like "there exists")
FW foreign word
IN preposition/subordinating conjunction
JJ adjective 'big'
JJR adjective, comparative 'bigger'
JJS adjective, superlative 'biggest'
LS list marker 1)
MD modal could, will
NN noun, singular 'desk'
NNS noun plural 'desks'
NNP proper noun, singular 'Harrison'
NNPS proper noun, plural 'Americans'
PDT predeterminer 'all the kids'
POS possessive ending parent\'s
PRP personal pronoun I, he, she

PRP$ possessive pronoun my, his, hers
RB adverb very, silently,
RBR adverb, comparative better
RBS adverb, superlative best
RP particle give up
TO to go 'to' the store.
UH interjection errrrrrrrm
VB verb, base form take
VBD verb, past tense took
VBG verb, gerund/present participle taking
VBN verb, past participle taken
VBP verb, sing. present, non-3d take
VBZ verb, 3rd person sing. present takes
WDT wh-determiner which
WP wh-pronoun who, what
WP$ possessive wh-pronoun whose
WRB wh-abverb where, when

Programming with R

```
read.csv(file="data/inflammation-01.csv", header = FALSE)
weight_kg <- 55

weight_kg
(weight_lb <- 2.2 * weight_kg)

dat <- read.csv(file="data/inflammation-01.csv", header = FALSE)
head(dat)
class(dat)
dim(dat)
```

Task:

```
best_practice <- c("Write", "programs", "for", "people", "not", "computers")
asterisk <- "***"  # R interprets a variable with a single value as a vector
                   # with one element.
highlight(best_practice, asterisk)
```

Output should be:

```
[1] "***"  "Write"  "programs"  "for"  "people"  "not"  "computers" "***"
```

Task2:

Write a function called analyze_all that takes a folder path and a filename pattern as its arguments and runs analyze for each file whose name matches the pattern.

06-12-2021

```
import nltk

from nltk.book import *


fdist = FreqDist(len(w) for w in text1)

print(fdist)

fdist

fdist.most_common()

fdist.max()

fdist[3]

fdist[20]

len(text1)

fdist[3]/len(text1)*100

sent7

print([w for w in sent7 if len(w) < 4])

print([w for w in sent7 if len(w) <= 4])

print([w for w in sent7 if len(w) != 4])

print(sorted(w for w in set(text1) if w.endswith("ableness")))

print(sorted(term for term in set(text1) if 'gnt' in term))

sorted(item for item in set(text7) if item.isdigit())

sorted( w for w in set(text7) if '-' in w and 'index' in w)

sorted(wd for wd in set(text3) if wd.title() and len(wd)>10)

sorted(w for w in set(sent7) if not w.islower())
```

অত্যন্ত ধন্যবাদ অতিরিক্ত মার্কের জন্য
অপারে সুখ আমার বিশ্বাস
আকাশ-পানে ছুটবে বাঁধন-হারা
আপনি কি বাংলা ছাড়া অন্য কোনও ভাষা বলতে পারেন?
আপনি কেমন আছেন?
আমার দেশ বাংলাদেশ
আমার দেশ বাংলাদেশ
আমি এখানে শিখতে এসেছি
আমি ক্লাস করি
আমি ক্ষুধার্ত
আমি বাংলায় গান গাই
আমি বৃদ্ধ হচ্ছি
আমি ভাল আছি
আমি সবাই এই আশ্চর্যজনক সুযোগ দ্বারা প্রলুব্ধ হয়
একবার না পারিলে, দেখো শতবার |
এখন আমার ক্লাস চলতেসে
এখন কেমন
ওহে বিশ্ব!
কখন হাল ছারতে নেই |
কি খবর সবার?
ক্লাস চলছে
জাপানিস বলতে পারি হাল্কা হাল্কা।
ডাক্তার আসিবার পূর্বে রোগীটি মারা গেল
ধন ধান্যে পুষ্পে ভরা
ধন্যবাদ
ধন্যবাদ স্যার
ধন্যবাদ স্যার, এক্সট্রা পয়েন্ট দেয়ার জন্য |
নদীর ওপার কহে ছাড়িয়া নিঃশ্বাস
পাপী কে নয় পাপ কে ঘৃণা কর
ফিরে এসেছি
ফিরে এসেছি
ফিরে এসেছি বিরতির পর...
বাংলা আমার মায়ের ভাষা।
বিনা মেঘে বজ্রপাত
বিরতির পর..
মানুষের পক্ষে সব স্বপ্নই পূরণ করা সম্ভব যদি সে যথেষ্ট সাহসী হয়
যে জন দিবসে মনের হরষে জ্বালায় মোমের বাতি
রাজ্যে সরকারি চাকরি পেতে গেলে বাংলা ভাষা জানা জরুরি
রোগই সংক্রামক স্বাস্থ্য নয় ।
শোনে মায়া যন্ত্র ধ্বনি তব কলকলে
সব সাধকের বর সাধক আমার দেশের চাষা
সময় এবং স্রোত কারো জন্য অপেক্ষা করে না
সময় ও স্রোত কারো জন্য অপেক্ষা করে না

Meet Links (11-12-2021):

https://extensions.libreoffice.org/en/extensions/show/languagetool
https://languagetool.org/
https://github.com/languagetool-org/languagetool
https://www.google.com/inputtools/try/
https://avro.im/
https://www.asciitable.com/
https://www.cs.cmu.edu/~pattis/15-1XX/common/handouts/ascii.html
https://home.unicode.org/
https://www.unicode.org/charts/PDF/U0980.pdf
https://www.unicode.org/charts/PDF/U0000.pdf

https://wiki.mozilla.org/L10n:Teams:bn-BD/StyleGuide
https://pontoon.mozilla.org/
https://www.libreoffice.org/community/localization/
https://www.libreoffice.org/community/nlc
https://bn.wikipedia.org/wiki/%E0%A6%AC%E0%A6%BE%E0%A6%82%E0%A6%B2%E0%A6%BE%E0%A6%A6%E0%A7%87%E0%A6%B6
https://bn.wikipedia.org/wiki/%E0%A6%AA%E0%A7%8D%E0%A6%B0%E0%A6%A7%E0%A6%BE%E0%A6%A8_%E0%A6%AA%E0%A6%BE%E0%A6%A4%E0%A6%BE

For checking word/letter/character count:
https://databasic.io/