## Word Tokenization

Word Tokenization using built-in split method

```
text = """Founded in 2002, SpaceX’s mission is to enable humans to become a spacefaring civilization and a multi-planet
species by building a self-sustaining city on Mars. In 2008, SpaceX’s Falcon 1 became the first privately developed
liquid-fuel launch vehicle to orbit the Earth."""
tokens = text.split()
print(tokens)
```

```
['Founded', 'in', '2002,', 'SpaceX’s', 'mission', 'is', 'to', 'enable', 'humans', 'to', 'become', 'a', 'spacefaring', 'civilization', 'and', '
```

Word Tokenization using Regular Expression

```
import re
text = """Founded in 2002, SpaceX’s mission is to enable humans to become a spacefaring civilization and a multi-planet
species by building a self-sustaining city on Mars. In 2008, SpaceX’s Falcon 1 became the first privately developed
liquid-fuel launch vehicle to orbit the Earth."""
tokens = re.findall("[\w']+", text)
print(tokens)
```

```
['Founded', 'in', '2002', 'SpaceX', 's', 'mission', 'is', 'to', 'enable', 'humans', 'to', 'become', 'a', 'spacefaring', 'civilization', 'and',
```

## Regional language filtration

```
mixed_text = "Google's service, offered निःशुल्क of charge, instantly अनुवाद words, phrases, and web pages between अंग्रेजी and over 100 other languages."
mixed_text_tokens = mixed_text.split()
print(mixed_text_tokens)

for token in mixed_text_tokens:
  if not token.isascii():
    mixed_text_tokens.remove(token)

print(mixed_text_tokens)
```

```
["Google's", 'service,', 'offered', 'निःशुल्क', 'of', 'charge,', 'instantly', 'अनुवाद', 'words,', 'phrases,', 'and', 'web', 'pages', 'between', '
["Google's", 'service,', 'offered', 'of', 'charge,', 'instantly', 'words,', 'phrases,', 'and', 'web', 'pages', 'between', 'and', 'over', '100'
```

## Stop Word Filtration

Some common stopwords in English, for example, include:

articles (a, an, the) conjunctions (and, but, or) prepositions (in, on, at) pronouns (he, she, it, they) auxiliary verbs (is, are, was, were)

```
stop_words = ['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "you'd", 'your', 'yours', 'yourself'
tokens_after_stop_word_filteration = [word for word in tokens if word.lower() not in stop_words]
print(tokens_after_stop_word_filteration)
```

```
['Founded', '2002', 'SpaceX', 'mission', 'enable', 'humans', 'become', 'spacefaring', 'civilization', 'multi', 'planet', 'species', 'building'
```

## Punctuation Filtration

Remove Punctuation from a String with Translate

```
import string
punc_filtered_str_1 = text.translate(str.maketrans('', '', string.punctuation))
print(punc_filtered_str_1)
```

```
Founded in 2002 SpaceX’s mission is to enable humans to become a spacefaring civilization and a multiplanet
species by building a selfsustaining city on Mars In 2008 SpaceX’s Falcon 1 became the first privately developed
liquidfuel launch vehicle to orbit the Earth
```

Remove Punctuation from a String with a Python loop

```
punc = '''!()-[]{};:'"\,<>./?@#$%^&*_~'''
punc_filtered_str_2 = text
for ele in punc_filtered_str_2:
```

```
    if ele in punc:
        punc_filtered_str_2 = punc_filtered_str_2.replace(ele, "")
print(punc_filtered_str_2)
```

```
Founded in 2002 SpaceX’s mission is to enable humans to become a spacefaring civilization and a multiplanet
species by building a selfsustaining city on Mars In 2008 SpaceX’s Falcon 1 became the first privately developed
liquidfuel launch vehicle to orbit the Earth
```

Using filter() and lambda function to filter out punctuation characters

```
punc_filtered_str_3 = ''.join(filter(lambda x: x.isalpha() or x.isdigit() or x.isspace(), text))
print(punc_filtered_str_3)
```

```
Founded in 2002 SpaceXs mission is to enable humans to become a spacefaring civilization and a multiplanet
species by building a selfsustaining city on Mars In 2008 SpaceXs Falcon 1 became the first privately developed
liquidfuel launch vehicle to orbit the Earth
```

## Email Validation

```
regex = r'\b[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Z|a-z]{2,7}\b'
def check_email(email):
    if(re.fullmatch(regex, email)):
        print("Valid Email")
    else:
        print("Invalid Email")

email = "ankitrai326@gmail.com"
check_email(email)
email = "my.ownsite@our-earth.org"
check_email(email)
email = "ankitrai326.com"
check_email(email)
```

```
Valid Email
Valid Email
Invalid Email
```

## Phone Number Validation

```
regex = r'\+?(0|91)?[6-9][0-9]{9}'
def check_phno(phno):
  if(re.fullmatch(regex, phno)):
    print('Valid phone number')
  else:
    print('Invalid phone number')

phno = "+919876543210"
check_phno(phno)
phno = "911234567890"
check_phno(phno)
phno = "916234567890"
check_phno(phno)
phno = "8796372841"
check_phno(phno)
```

```
Valid phone number
Invalid phone number
Valid phone number
Valid phone number
```

## Name Validation

```
def check_name(name):
  if name.replace(" ", "").isalpha():
    print("Name is valid")
  else:
    print("Name is invalid")

#Driver code
name = "tejas"
check_name(name)
name = " tejas "
check_name(name)
check_name(name)
name = "tej@s"
check_name(name)
```

```
Name is valid
Name is valid
Name is valid
Name is invalid
```