In [2]:
```
import pandas as pd
import numpy as np
```

In [10]:
```
letter_list=np.array(["Anthony","Brutus","Caesar","Decimus"])
letters=pd.Series(letter_list)
print(letters)
print("Series object after all names are capitalised=")
print(letters.str.capitalize())#str class has all vectorized python string functions
```

```
0    Anthony
1     Brutus
2     Caesar
3    Decimus
dtype: object
Series object after all names are capitalised=
0    Anthony
1     Brutus
2     Caesar
3    Decimus
dtype: object
```

In [36]:
```
jc=pd.Series(["Anthony","Brutus","Caesar","Cassius","Octovian","Portia","Decimus"])
#trying out all string fucntions
print(jc)
print("\n")
#len
print("len=")
print(jc.str.len())
print("\n")
#ljust
print("ljust=")
print(jc.str.ljust(10,"$"))
#allign string to left,fist argument is length and second argument is the character used to fill blank characters
print("\n")
print("rjust=")
print(jc.str.rjust(10,"$"))
#allign string to right,fist argument is length and second argument is the character used to fill blank characters
print("\n")
#centre
print("center=")
print(jc.str.center(10,"$"))
#allign string to center,fist argument is length and second argument is the character used to fill blank characters
print("\n")
#zfill
print("zfill=")
print(jc.str.zfill(10))
#adds zeores until string reaches specified length
print("\n")
#lstrip
print("lstrip=")
print(jc.str.lstrip("CasBruAn"))
print("\n")
#lstrip removes leading characters from left side by default it removes white spaces
#rstrip
print("rstrip=")
print(jc.str.rstrip("arus"))
#rstrip removes leading characters from left side by default it removes white spaces
print("\n")
#find
print("find=")
print(jc.str.find("es"))
print("\n")
#finds index from which given substring is found,returns -1 if string not found
```

```
0     Anthony
1      Brutus
2      Caesar
3     Cassius
4    Octovian
5      Portia
6     Decimus
dtype: object


len=
0    7
1    6
2    6
3    7
4    8
5    6
6    7
dtype: int64


ljust=
0    Anthony$$$
1    Brutus$$$$
2    Caesar$$$$
3    Cassius$$$
4    Octovian$$
5    Portia$$$$
6    Decimus$$$
dtype: object


rjust=
0    $$$Anthony
1    $$$$Brutus
2    $$$$Caesar
3    $$$Cassius
4    $$Octovian
5    $$$$Portia
6    $$$Decimus
dtype: object


center=
0    $Anthony$$
1    $$Brutus$$
2    $$Caesar$$
3    $Cassius$$
4    $Octovian$
5    $$Portia$$
6    $Decimus$$
dtype: object


zfill=
0    000Anthony
1    0000Brutus
2    0000Caesar
3    000Cassius
4    00Octovian
5    0000Portia
6    000Decimus
dtype: object


lstrip=
0       thony
1         tus
2        esar
3         ius
4    Octovian
5      Portia
6     Decimus
dtype: object


rstrip=
0     Anthony
1        Brut
2         Cae
3       Cassi
4    Octovian
5       Porti
6       Decim
dtype: object
```

```
find=
0   -1
1   -1
2    2
3   -1
4   -1
5   -1
6   -1
dtype: int64
```

In [48]:
```
#rfind
print("rfind=")
print(jc.str.find("a"))
print("\n")
#finds index from which last occurence of given substring is found,returns -1 if string not found
#swapcase
print("swapcase=")
print(jc.str.swapcase())
print("\n")
#makes lowercase characters as upper and uppercase characters as lower
print("translate=")
d={97:35,115:46}#replacing all 'a' with '#' and replacing all 's' with '.'
print(jc.str.translate(d))
print("\n")
#translate takes dictionary with ASCII values as argument and maps characters as per dictionary
#startswith
print("startswith=")
print(jc.str.startswith("C"))
print("\n")
#returns boolean value for the condition if string is starting with the particular character
#endswith
print("endswith=")
print(jc.str.endswith("s"))
print("\n")
#returns boolean value for the condition if string is ending with the particular character
s="Julius Caesar Is The King Of Rome"
print("is title?=",s.istitle())#istitle returns true if all words start with uppercase
print("split function=",s.split())#splits string to words by default splits strings by blank spaces
print("partition function=",s.partition("Is"))#partition returns tuple with three elements:before match,match and after match
```

```
rfind=
0   -1
1   -1
2    1
3    1
4    6
5    5
6   -1
dtype: int64


swapcase=
0     aNTHONY
1      bRUTUS
2      cAESAR
3     cASSIUS
4    oCTOVIAN
5      pORTIA
6     dECIMUS
dtype: object


translate=
0     Anthony
1      Brutu.
2      C#e.#r
3     C#..iu.
4    Octovi#n
5      Porti#
6     Decimu.
dtype: object


startswith=
0    False
1    False
2     True
3     True
4    False
5    False
6    False
dtype: bool


endswith=
0    False
1     True
2    False
3     True
4    False
5    False
6     True
dtype: bool


is title?= True
split function= ['Julius', 'Caesar', 'Is', 'The', 'King', 'Of', 'Rome']
partition function= ('Julius Caesar ', 'Is', ' The King Of Rome')
```

In [67]:
```
#vectorised slicing
print("vectorised slicing=")
print(jc.str[0:3])
print("get function=")
print(jc.str.get(-1))#getting last character from each string
#get dummies
shakespeare=pd.DataFrame({"names":jc,"info":["A/B","B/C","A/D","C/D","A/D","A/C","B/D"]})
print(shakespeare)
print("get dummies=")
print(shakespeare["info"].str.get_dummies("/"))
```

```
vectorised slicing=
0    Ant
1    Bru
2    Cae
3    Cas
4    Oct
5    Por
6    Dec
dtype: object
get function=
0    y
1    s
2    r
3    s
4    n
5    a
6    s
dtype: object
      names info
0   Anthony  A/B
1    Brutus  B/C
2    Caesar  A/D
3   Cassius  C/D
4  Octovian  A/D
5    Portia  A/C
6   Decimus  B/D
get dummies=
   A  B  C  D
0  1  1  0  0
1  0  1  1  0
2  1  0  0  1
3  0  0  1  1
4  1  0  0  1
5  1  0  1  0
6  0  1  0  1
```