

Lær Python dag 2 - modul 2

Institut for Matematik og Datalogi, Syddansk Universitet

Team IMADA

<https://moggel12.github.io/PythonWorkshop2022/>

15. maj, 2022



Indhold

Streng



Streng - en sekvens af tegn

I har set tekststreng mange gange allerede.

```
name = str(input("Intast dit navn: "))  
print("Hej " + name + "!!")
```



Strengte - en sekvens af tegn

Men strengte kan bruges til alt muligt, f.eks. til at repræsentere en DNA-sekvens eller en bog.

```
dna = "ATTAGCC"  
book = "Once upon a time ..."
```



Streng - en sekvens af tegn

En streng er i virkeligheden bare en liste af enkelte tegn.

```
dna = "ATTAGCC"
```

| | | | | | | | |
|-------|---|---|---|---|---|---|---|
| Tegn | A | T | T | A | G | C | C |
| Index | 0 | 1 | 2 | 3 | 4 | 5 | 6 |



Streng - en sekvens af tegn

En streng er i virkeligheden bare en liste af enkelte tegn.

```
dna = "ATTAGCC"
```

| | | | | | | | |
|-------|---|---|---|---|---|---|---|
| Tegn | A | T | T | A | G | C | C |
| Index | 0 | 1 | 2 | 3 | 4 | 5 | 6 |

Så derfor kan vi bruge mange af de samme funktioner som vi lærte til lister!



Streng - en sekvens af tegn

Indeksering

```
dna = "ATTAGCC"  
print(dna[1])
```



Streng - en sekvens af tegn

Indeksering

```
dna = "ATTAGCC"  
print(dna[1])
```

T



Streng - en sekvens af tegn

Indeksering

```
dna = "ATTAGCC"  
print(dna[1])
```

T

Slicing

```
dna = "ATTAGCC"  
print(dna[2:5])
```



Strengte - en sekvens af tegn

Indeksering

```
dna = "ATTAGCC"  
print(dna[1])
```

T

Slicing

```
dna = "ATTAGCC"  
print(dna[2:5])
```

TAG



Streng - en sekvens af tegn

Indeksering

```
dna = "ATTAGCC"  
print(dna[1])
```

T

Slicing

```
dna = "ATTAGCC"  
print(dna[2:5])
```

TAG

Længde

```
dna = "ATTAGCC"  
print(len(dna))
```



Streng - en sekvens af tegn

Indeksering

```
dna = "ATTAGCC"  
print(dna[1])
```

T

Slicing

```
dna = "ATTAGCC"  
print(dna[2:5])
```

TAG

Længde

```
dna = "ATTAGCC"  
print(len(dna))
```

7

Streng - en sekvens af tegn

Gennemløb af liste (iterate)

```
dna = "ATTAGCC"  
for c in dna:  
    print(c)
```



Streng - en sekvens af tegn

Gennemløb af liste (iterate)

```
dna = "ATTAGCC"  
for c in dna:  
    print(c)
```

A
T
T
A
G
C
C

SDU



Strengene - en sekvens af tegn

Lister er mutable, hvad med strenge?

```
mylist = [1, 2, 3, 4]
```

```
mylist[0] = 5
```

```
print( mylist )
```

```
dna = "ATTAGCC"
```

```
dna[0] = "G"
```

```
print(dna)
```



Strengte - en sekvens af tegn

Lister er mutable, hvad med strengte?

```
mylist = [1, 2, 3, 4]
mylist[0] = 5
print( mylist )
```

```
dna = "ATTAGCC"
dna[0] = "G"
print(dna)
```

```
[5, 2, 3, 4]
```

Traceback (most recent call last):

File "test.py", line 6, **in** <module>

 dna[0] = "G"

TypeError: 'str' **object** does **not** support item assignment

Strengene - en sekvens af tegn

En løsning på immutability er at bygge nye strenge

```
dna = "ATTAGCC"  
newdna = "G" + dna[1:]  
print(newdna)
```



Strengene - en sekvens af tegn

En løsning på immutability er at bygge nye strenge

```
dna = "ATTAGCC"  
newdna = "G" + dna[1:]  
print(newdna)
```

GTTAGCC



Streng - en sekvens af tegn

Hvad med .append?

```
dna = "ATTAGCC"  
dna.append("A")  
print(dna)
```



Streng - en sekvens af tegn

Hvad med `.append`?

```
dna = "ATTAGCC"  
dna.append("A")  
print(dna)
```

Traceback (most recent call last):

File "test.py", line 2, in <module>

 dna.append("A")

AttributeError: 'str' **object** has no attribute 'append'



Streng - en sekvens af tegn

Heldigvis ved vi at vi kan sætte sammen (concatenation) med "+"

```
dna = "ATTAGCC"  
dna = dna + "A"  
print(dna)
```



Streng - en sekvens af tegn

Heldigvis ved vi at vi kan sætte sammen (concatenation) med "+"

```
dna = "ATTAGCC"  
dna = dna + "A"  
print(dna)
```

ATTAGCCA



Strengte - en sekvens af tegn

Der er også visse funktioner (metoder) som er lavet til strengte, f.eks. `.lower()` og `.upper()` som ændrer alle bogstaver til henholdsvis små og store.

```
dna = "ATTAGCC"  
print(dna.lower())
```



Strengte - en sekvens af tegn

Der er også visse funktioner (metoder) som er lavet til strengte, f.eks. `.lower()` og `.upper()` som ændrer alle bogstaver til henholdsvis små og store.

```
dna = "ATTAGCC"  
print(dna.lower())
```

```
attagcc
```



Streng - en sekvens af tegn

Hvis man er interesseret i at vide hvor i en streng et tegn forekommer, kan man bruge `.find()`.

```
dna = "ATTAGCC"  
print(dna.find("G"))
```

Streng - en sekvens af tegn

Hvis man er interesseret i at vide hvor i en streng et tegn forekommer, kan man bruge `.find()`.

```
dna = "ATTAGCC"  
print(dna.find("G"))
```

4

Streng - en sekvens af tegn

Hvis man er interesseret i at vide hvor i en streng et tegn forekommer, kan man bruge `.find()`.

```
dna = "ATTAGCC"  
print(dna.find("G"))
```

4

Hvad hvis der er flere af den samme, eller slet ikke nogen?

```
dna = "ATTAGCC"  
print(dna.find("A"))  
print(dna.find("B"))
```

Streng - en sekvens af tegn

Hvis man er interesseret i at vide hvor i en streng et tegn forekommer, kan man bruge `.find()`.

```
dna = "ATTAGCC"  
print(dna.find("G"))
```

4

Hvad hvis der er flere af den samme, eller slet ikke nogen?

```
dna = "ATTAGCC"  
print(dna.find("A"))  
print(dna.find("B"))
```

0

-1

Første forekomst og -1

Streng - en sekvens af tegn

Kan man søge på mere end bare et enkelt tegn?

```
dna = "ATTAGCC"  
print(dna.find("TAG"))
```



Streng - en sekvens af tegn

Kan man søge på mere end bare et enkelt tegn?

```
dna = "ATTAGCC"  
print(dna.find("TAG"))
```

2

Ja, svaret er index fra, hvor forekomsten starter.



Streng - en sekvens af tegn

Hvis man er ligeglad med HVOR en forekomst er, men bare vil vide OM det forekommer kan man bruge "in".

```
dna = "ATTAGCC"  
print("TAG" in dna)  
print("CAT" in dna)
```



Streng - en sekvens af tegn

Hvis man er ligeglad med HVOR en forekomst er, men bare vil vide OM det forekommer kan man bruge "in".

```
dna = "ATTAGCC"  
print("TAG" in dna)  
print("CAT" in dna)
```

```
True  
False
```



Strengte - en sekvens af tegn

Hvis man vil inddele en sætning i ord, kan man skære den i stykker og få alle ordene i en liste. Her bruges `.split()` funktionen. Man skal fortælle ved hvilket slags tegn man gerne vil skære (ofte mellemrum).

```
s = "Jeg gik mig over sø og land"  
l = s.split(" ")  
print(l)  
print(l[2])
```



Streng - en sekvens af tegn

Hvis man vil inddele en sætning i ord, kan man skære den i stykker og få alle ordene i en liste. Her bruges `.split()` funktionen. Man skal fortælle ved hvilket slags tegn man gerne vil skære (ofte mellemrum).

```
s = "Jeg gik mig over sø og land"  
l = s.split(" ")  
print(l)  
print(l[2])
```

```
['Jeg', 'gik', 'mig', 'over', 'sø', 'og', 'land']  
mig
```



Streng - en sekvens af tegn

Det modsatte af at splitte er at sætte sammen, her bruges `.join()` funktionen. Den sætter en streng ind mellem alle elementerne i listen. Det bruges ofte til hurtigt at lave en liste af strenge til en enkelt streng, ved at joine med mellemrum.

```
l = ['Jeg', 'gik', 'mig', 'over', 'sø', 'og', 'land']  
s = "-".join(l)  
print(s)
```



Strengene - en sekvens af tegn

Det modsatte af at splitte er at sætte sammen, her bruges `.join()` funktionen. Den sætter en streng ind mellem alle elementerne i listen. Det bruges ofte til hurtigt at lave en liste af strenge til en enkelt streng, ved at joine med mellemrum.

```
l = ['Jeg', 'gik', 'mig', 'over', 'sø', 'og', 'land']  
s = "-".join(l)  
print(s)
```

```
Jeg—gik—mig—over—sø—og—land
```



Streng - en sekvens af tegn

Man kan også erstatte bogstaver med `.replace()` funktionen. Her skal man fortælle, hvad der skal ændres og hvad det skal ændres til.

Vi kan f.eks. fjerne mellemrum.

```
s = "Jeg gik mig over sø og land"
```

```
# Erstatte mellemrum med ingenting = fjerner mellemrum
```

```
s2 = s.replace(" ", "")
```

```
print(s2)
```



Streng - en sekvens af tegn

Man kan også erstatte bogstaver med `.replace()` funktionen. Her skal man fortælle, hvad der skal ændres og hvad det skal ændres til.

Vi kan f.eks. fjerne mellemrum.

```
s = "Jeg gik mig over sø og land"
```

```
# Erstatte mellemrum med ingenting = fjerner mellemrum
```

```
s2 = s.replace(" ", "")
```

```
print(s2)
```

```
Jeggikmigoversøogland
```



Strengte - en sekvens af tegn

Eksempel, lad os lave en funktion som givet to ord (strengte), printer alle bogstaver der forekommer i begge ord.

```
def in_both(word1, word2):  
    #code?
```



Strengte - en sekvens af tegn

Eksempel, lad os lave en funktion som givet to ord (strengte), printer alle bogstaver der forekommer i begge ord.

```
def in_both(word1, word2):  
    #code?
```



Streng - en sekvens af tegn

```
def in_both(word1, word2):  
    for letter in word1:  
        if letter in word2:  
            print( letter )  
  
in_both("APPLE", "PEN")
```



Strengte - en sekvens af tegn

```
def in_both(word1, word2):  
    for letter in word1:  
        if letter in word2:  
            print( letter )
```

```
in_both("APPLE", "PEN")
```

P
P
E

