1. **<u>Indexering van tekens in string/ substrings:</u>**
    a. s[i] met i : 0 .. len(s) – 1
       Voorbeeld: “python”[1] → “y”
    b. s[i] met i : -1 … -len(s)
       Voorbeeld: “python”[-2] → “o”
    c. **substring**: s[i:j]
       Voorbeeld:
            “banaan”[2:5] → “naa”
            “banaan”[2:] → “naan”
            “banaan”[:3] → “ban”

2. **<u>Bewerkingen met strings</u>**
    a. Optellen van strings: **+** operator
       Voorbeeld: “abc” + “def” → “abcdef”
    b. Vermenigvuldigen van string met int: ***** operator
       Voorbeeld: 3 * “abc” → “abcabcabc”
    c. String komt voor in andere string: **in** operator
       Voorbeeld: “ab” in “bhdabhj” → True
            “ab” in “kdlskdqmaklb” → False

3. **<u>Eenvoudige functies</u>**
    a. Aantal tekens in een string: **len()**
       Voorbeeld1: len(“abc) → 3
       Voorbeeld2: len(“”) → 0
    b. Ordening van tekens: **ord()** en **chr()**

       ASCII tekens : 256 tekens met ordening

       ord(c) geeft getal van 0..255 (voorbeeld: ord(“a”) is 97)

       chr(i) met i: 0..255 geeft teken c waarvan ord(c) == i (voorbeeld chr(97) == “a”)

4. **<u>Vergelijken van strings</u>**
   Vergelijkingsoperatoren: <, >, <=, >=, ==, != → alfabetische volgorde
   Voorbeeld: “abc” < “de”

5. **String methods** : aanroep s.m(); opmerking: s blijft ongewijzigd, resultaat is nieuwe string

s.isalpha()	True als alle tekens letters zijn
s.isdigit()	True als alle tekens cijfers zijn
s.isspace()	True als alle tekens spatie of newline zijn
s.strip()	string zonder de spaties (ook newline) aan begin en einde van s
s.rstrip()	string zonder de spaties (ook newline) aan einde van s
s.lstrip()	string zonder de spaties (ook newline) aan begin van s
s.upper()	string waarbij alle kleine letters uit s worden vervangen door equivalente grote letter alle andere tekens zijn zelfde als deze uit s
s.lower()	string waarbij alle grote letters uit s worden vervangen door equivalente kleine letter alle andere tekens zijn zelfde als deze uit s
s1.find(s2)	geeft startindex (0..len(s)-1) van eerste voorkomen van de substring s2 in s1, -1 als s2 niet als substring in s1 voorkomt
s1.rfind(s2)	geeft startindex (0..len(s)-1) van laatste voorkomen van de substring s2 in s1, -1 als s2 niet als substring in s1 voorkomt
s1.replace(s2,s3)	geeft nieuwe string waarbij elk voorkomen van de substring s2 in s1 wordt vervangen door de string s3
s.split()	zie later nadat lijsten zijn behandeld
s.join()	zie later nadat lijsten zijn behandeld

en nog meer : https://www.w3schools.com/python/python_ref_string.asp

6. **format() versus f-strings voor het formatteren van strings**

	format()	f string
Python versie	Python 2.6	Python 3.6
	x = 11 s = “oneven”	
Invullen	“{1} is een {0} getal”.format(s,x) “{} is een {} getal”.format(x,s)	f”{x} is een {s} getal”
Aantal cijfers na decimale punt	"{:.2f}".format(x/3)	f"{x/3:.2f}"
uitlijning	"{:11}".format(s) "{:<11}".format(s)	f"{s:11}"
	"{:>11}".format(s)	f"{s:>11}"
	"{:^11}".format(s)	f"{s:^11}"
Uitlijning met vul-karakter anders dan spatie	"{:*^11}".format(s)	f"{s:*^11}"
Met conditie	'Is {} even? {}'.format(x, "Ja" if x%2 == 0 else "Nee")	f'Is {x} even? {"Ja" if x%2 == 0 else "Nee"}'