REGular EXpressions (REGEX)

In [1]:
```
import re
```

In [2]:
```
cumle = "Mustafa Kemal Atatürk, Türk asker, devlet adamı ve Türkiye Cumhuriyeti'nin kurucusudur."
patern = "Türk"

re.search(patern, cumle) #belirlenen paterni cümle içinde arar ve çeşitli parametrelerini verir
```

Out [2]:
```
<re.Match object; span=(23, 27), match='Türk'>
```

In [3]:
```
durum = re.search(patern, cumle)
durum.span()
```

Out [3]:
```
(23, 27)
```

In [4]:
```
dir(durum)
```

Out [4]:
```
['__class__',
 '__class_getitem__',
 '__copy__',
 '__deepcopy__',
 '__delattr__',
 '__dir__',
 '__doc__',
 '__eq__',
 '__format__',
 '__ge__',
 '__getattribute__',
 '__getitem__',
 '__gt__',
 '__hash__',
 '__init__',
 '__init_subclass__',
 '__le__',
 '__lt__',
 '__module__',
 '__ne__',
 '__new__',
 '__reduce__',
 '__reduce_ex__',
 '__repr__',
 '__setattr__',
 '__sizeof__',
 '__str__',
 '__subclasshook__',
 'end',
 'endpos',
 'expand',
 'group',
 'groupdict',
 'groups',
 'lastgroup',
 'lastindex',
 'pos',
 're',
 'regs',
 'span',
 'start',
 'string']
```

In [6]:
```
durum.start()
```

Out [6]:
```
23
```

In [7]:
```
durum.end()
```

Out [7]:
```
27
```

In [8]:
```
durum.group()
```

Out [8]:
```
'Türk'
```

Çoklu Eşleşmelerde (match) Kullanım

In [10]:
```
for eslesme in re.findall(patern, cumle): #findall tüm eşleşmeleri verir ancak konumları görünmez
    print(eslesme)
```

```
Türk
Türk
```

In [11]:
```
for eslesme in re.finditer(patern, cumle):
    print(eslesme.span(), eslesme.group())
```

```
(23, 27) Türk
(51, 55) Türk
```

In [12]:
```
for eslesme in re.findall(patern, cumle): #findall böyle çalışmaz finditer çalışır
    print(eslesme.span(), eslesme.group())
```

```
---------------------------------------------------------------------------AttributeError                            Traceback (most recent call last)Cell In[12], 
      1 for eslesme in re.findall(patern, cumle): #findall böyle çalışmaz finditer çalışır
----> 2     print(eslesme.span(), eslesme.group())
AttributeError: 'str' object has no attribute 'span'
```

Dinamik Kullanım

Ifade ########## Aciklama ######## Örnek ########### Patern ########### ######## ---------- ######## ######## \d ######### rakam ######## base42 ######### base\d\d ######### ######## \w ######### karakter ###### R2-D2 ######### \w\w\w\w\w ######### ######## \s ######### bosluk ###### Ping Pong ######### Ping\sPong ######### ######## \D ######## rakam degil ###### base ######### \D\D\D\D ######### ######## \W ###### karakter degil ### R2D2 ######### \W\W\W\W ########## ######## \S ###### bosluk degil ###### PingPong ### \S\S\S\S\S\S\S\S ######## ######## ---------- ########

In [17]:
```
ornek = "En sevdiğim kanal base42."
patern = r"base\d\d" #base ile başlayıp sonrasında 2 rakam gelen elemanlar aranır, r kullanımı tavsiye edilir pythonista st
re.search(patern, ornek)
```

Out [17]: <re.Match object; span=(18, 24), match='base42'>

In [20]:
```
cumle = "Selam, benim telefon numaram 0535-9998877"
patern = r"\d\d\d\d-\d\d\d\d\d\d\d"

re.search(patern, cumle)
```

Out [20]: <re.Match object; span=(29, 41), match='0535-9998877'>

######## Ifade ######## Aciklama ######## Örnek ############## Patern ############ ######## ---------- ######## ######## {5} ######## adet ######## aaaaa ######### \w{5} ########## ######## {3,4} ######## veya ######## abc ########## \w{3,4} ######### ######## {2,} ######## en az ###### 198721321 ######## \d{2,} ######### ######## * ### 0 ya da fazla ######## x ########## \w* ######## ######## + ### 1 ya da fazla ####### Ahmet1 ######## \w+\d+ ######### ######## ? ###### ya 1 ya hic ######## Mura ######### Murat? ####### ######## ---------- ######## ##

In [23]:
```
cumle = "Selam, benim telefon numaram 0535-9998877"
patern = r"\d{3,4}-\d{7}" #3 veya 4 - 7

re.search(patern, cumle)
```

Out [23]: <re.Match object; span=(29, 41), match='0535-9998877'>

In [24]:
```
cumle = "Selam, benim telefon numaram 0535-9998877"
patern = r"\d{3}-\d{7}" #3 - 7 dediğimizde baştaki sıfırı yok etti ki işimize gelebilir

re.search(patern, cumle)
```

Out [24]: <re.Match object; span=(30, 41), match='535-9998877'>

In [26]:
```
cumle = "En sevdigim kanal base42."
patern = r"\s\w{5,}" #boşlukla başlayacak (kelimeyi başından al) ve 4 karakter takip edecek
re.search(patern, cumle) #ilkini alır
```

Out [26]: <re.Match object; span=(2, 11), match=' sevdigim'>

In [31]:
```
cumle = "En sevdigim kanal base42."
patern = r"\s\w{5,}"
for match in re.finditer(patern, cumle):
    print(match.span(), match.group())
```

```
(2, 11)  sevdigim
(11, 17)  kanal
(17, 24)  base42
```

In [34]:
```
patern = r"\d?"
for match in re.finditer(patern, cumle):
    print(match.span(), match.group())
```

```
(0, 0)
(1, 1)
(2, 2)
(3, 3)
(4, 4)
(5, 5)
(6, 6)
(7, 7)
(8, 8)
(9, 9)
(10, 10)
(11, 11)
(12, 12)
(13, 13)
(14, 14)
(15, 15)
(16, 16)
(17, 17)
(18, 18)
(19, 19)
(20, 20)
(21, 21)
(22, 23) 4
(23, 24) 2
(24, 24)
(25, 25)
```

In [41]:
```
patern = r"\d+" # bir ya da daha fazla sayı
for match in re.finditer(patern, cumle):
    print(match.span(), match.group())
```

```
(22, 24) 42
```

In [42]:
```
patern = r"\w*\d+" #harf olsun olmasın, bir ya da daha fazla sayı
                   #üsttekinden farkı artık bunu group ile çağırdığımızda kelime de beraberinde gelecek
```

```
for match in re.finditer(patern, cumle):
    print(match.span(), match.group())
```

```
(18, 24) base42
```

In [43]:
```
cumle = "En sevdigim kanal 42base." #42nin yerini değiştirirsek öyle olmaz ama
patern = r"\w*\d+"

for match in re.finditer(patern, cumle):
    print(match.span(), match.group())
```

```
(18, 20) 42
```

In [44]:
```
cumle = "En sevdigim kanal 42base." #böyle olabilir
patern = r"\w*\d+\w*"

for match in re.finditer(patern, cumle):
    print(match.span(), match.group())
```

```
(18, 24) 42base
```

GSM Operatörleri: # 54... -> Vodafone # 501,505,506 -> Turk Telekom # 53... -> Turkcell

In [59]:
```
def gsm_operator_bul(tel_no):
    patern= r"(\d{3,4})-(\d{7})" # \d{3,4} bize 0xxx verir ve bu if leri döndürmede sıkıntı yaşatacak
    match = re.search(patern, tel_no)

    if match:
        kod = match.group()[0] #group yazmak bize 0 veya 5 sayısını verir
        print(kod)
        if kod.startswith("54"):
            return "Vodafone"
        if kod.startswith("501") or kod.startswith("505") or kod.startswith("506"):
            return "Turk Telekom"
        if kod.startswith("53"):
            return "Turkcell"
        else:
            return "Operator bilinmiyor."
    else:
        return "Patern bulunamadı."
```

In [60]:
```
tel_no = "0535-8886622."
gsm_operator_bul(tel_no)
```

```
0
```

Out [60]: 'Operator bilinmiyor.'

In [61]:
```
def gsm_operator_bul(tel_no):
    patern= r"(\d{3})-(\d{7})" # \d{3} yaparsak çalışır
    match = re.search(patern, tel_no)

    if match:
        kod = match.groups()[0]
        print(kod)
        if kod.startswith("54"):
            return "Vodafone"
        if kod.startswith("501") or kod.startswith("505") or kod.startswith("506"):
            return "Turk Telekom"
        if kod.startswith("53"):
            return "Turkcell"
        else:
            return "Operator bilinmiyor."
    else:
        return "Patern bulunamadı."
```

In [62]:
```
tel_no = "0535-8886622."
gsm_operator_bul(tel_no)
```

```
535
```

Out [62]: 'Turkcell'

In [63]:
```
tel_no = "Selam, benim telefon numaram 0535-8886622." #cümle içinde de bulabilir çünkü biz sayıları arıyoruz sadece
gsm_operator_bul(tel_no)
```

```
535
```

Out [63]: 'Turkcell'

######## Ifade ######## Aciklama ######## Örnek ############## Patern ############ ######## ---------- ######## ######## | ######## veya ######## slm ########## selam|slm ########## ######## ^ ######## baslar ######## Ahmet ########## ^\w+ ######### ######## $ ######## biter ###### base42 ######### \d$ ######### ######## . ####### herhangi ###### abcdef ######## .* ######### ######## \ ######## esc ###### (not) ######## \(\w{3}\) ######### ######## ---------- ########
##

In [64]:
```
import re

def mesaj_hissi_bul(mesaj):
```

```
    hisler = []

    pozitif_patern = r"(merhaba|selam|ask|sevgi|dost|kardes|:\)+)"
    negatif_patern = r"(lan|aptal|abv|yeter|birak)"

    heyecanli_patern = r"!|[!|?]{2,}$"
    sakin_patern = r"^[Tabi+|Hayhay]"

    emin_patern = r"[K|k]esin|[T|t]abi|[E|e]lbet"
    kararsiz_patern = r"[B|b]elki|[S|s]anirim"

    if re.search(pozitif_patern, mesaj):
        hisler.append("Pozitif")
    if re.search(negatif_patern, mesaj):
        hisler.append("Negatif")
    if re.search(heyecanli_patern, mesaj):
        hisler.append("Heyecanli")
    if re.search(sakin_patern, mesaj):
        hisler.append("Sakin")
    if re.search(emin_patern, mesaj):
        hisler.append("Emin")
    if re.search(kararsiz_patern, mesaj):
        hisler.append("Kararsiz")

    return hisler
```

In [65]:

```
cumle1 = "Naber abi? :)             "
cumle2 = "Tabiii ki buyrun          "
cumle3 = "Sacmalamayi birak artik!  "
cumle4 = "Belki yarindan da yakin..."
cumle5 = "Elbet birgün bulusacagiz  "
cumleler = [cumle1, cumle2, cumle3, cumle4, cumle5]
for cumle in cumleler :
    print(cumle, '\t', mesaj_hissi_bul(cumle))
```

```
Naber abi? :)                   ['Pozitif']
Tabiii ki buyrun                ['Sakin', 'Emin']
Sacmalamayi birak artik!        ['Negatif', 'Heyecanli']
Belki yarindan da yakin...      ['Kararsiz']
Elbet birgün bulusacagiz        ['Emin']
```

In [ ]: