Bilgisayar Programlama 2

Ders Notu: 3

Strings

- Stringler, Python'da metinlerle çalışmak için kullanılan bir veri türüdür. Python, dizeleri işlemek için bir dizi güçlü özelliğe sahiptir.

Temel Bilgiler

- Tek tırnak (') veya çift tırnak ("") kullanılır. Çok satırlı dizeler (kelimeler) için üçlü tırnak kullanılabilir.

```
s = 'Hello'
t = "Hello"
m = """This is a long string that is
spread across two lines."""
```

- Boş dize '', 0 sayısının dize karşılığıdır. İçinde hiçbir şey olmayan bir dizedir. Bunu daha önce, print ifadesinin isteğe bağlı bağımsız değişkeni olan sep='' içinde görmüştük.
- Bir dizenin uzunluğunu (kaç karakterden oluştuğunu) elde etmek için, yerleşik `len` işlevi kullanılır. Örneğin, `len('Hello')` 5'tir.

Birleştirme ve Tekrarlama

- + ve * operatörleri stringler üzerinde kullanılabilir. + operatörü iki stringi birleştirir. Bu işleme birleştirme (concatenation) denir. * operatörü ise bir dizeyi belirli sayıda tekrarlar. Örnekler:

Expression	Result
`'AB'+'cd'`	`'ABcd'`
`'A'+'7'+'B'`	`'A7B'`
`'Hi'*4`	`'HiHiHiHi'`

- **Örnek 1 :** Uzun bir çizgi dizisi yazdırmak istiyorsak, yandaki işlemi yapabiliriz.

```
1  print('-'*75)
```

- **Örnek 2:** Kullanıcıdan tekrar tekrar 10 harf girmesini isteyen ve yalnızca kullanıcının girdiği sesli harflerden oluşan bir metin oluşturan bir uygulama yazınız.

```
1  s = ''
2  for i in range(10):
3      t = input('Bir harf giriniz: ')
4      if t=='a' or t=='e' or t=='ı' or t=='i' or t=='o' or t=='ö' or t=='u' or t=='ü':
5          s = s + t
6  print(s)
```

in operatörü

- `in` operatörü, bir stringin bir şey içerip içermediğini belirtmek için kullanılır. Örnek:

```
1   metin = "ybs"
2   if 'y' in metin:
3       print('Metin a harfini içeriyor.')
```

- `not in` operatörü, bir stringin tam tersine bir şey içermediğini kontrol için kullanılır. Örnek:

```
1   metin = "ybs"
2   if 'k' not in metin:
3       print('Metin k harfini içermiyor.')
```

- **Örnek**: Önceki bölümde uzun bir if koşulu ele almıştık:

```
if t=='a' or t=='e' or t=='ı' or t=='i' or t=='o'  or t=='ö' or t=='u'or t=='ü':
```

- `in` operatörünü kullanarak, bu ifadeyi aşağıdakiyle değiştirebiliriz:

```
if t in 'aeıioöuü':
```

Indeksleme (Indexing) ve Dilimleme (Slices)

- Bir stringden tek tek karakterleri seçmek isteyeceğimiz durumlar sıklıkla yaşanır. Python bunu yapmak için köşeli parantezler kullanır. Aşağıdaki tablo, s="**Python**" dizesinin indekslenmesine ilişkin bazı örnekler vermektedir.

Statement	Result	Description
`s[0]`	`P`	first character of s
`s[1]`	`y`	second character of s
`s[-1]`	`n`	last character of s
`s[-2]`	`o`	second-to-last character of s

- Bir stringin bir bölümünü seçmek için dilimleme (slice) kullanılır. İndeksleme ve aralık (range) fonksiyonunun bir kombinasyonu gibi davranır. Aşağıda s= "**abcdefghij**" dizesiyle ilgili bazı örnekler bulunmaktadır.

```
index:     0 1 2 3 4 5 6 7 8 9
letters:   a b c d e f g h i j
```

Code	Result	Description
`s[2:5]`	`cde`	characters at indices 2, 3, 4
`s[ :5]`	`abcde`	first five characters
`s[5: ]`	`fghij`	characters from index 5 to the end
`s[-2:]`	`ij`	last two characters
`s[ : ]`	`abcdefghij`	entire string
`s[1:7:2]`	`bdf`	characters from index 1 to 6, by twos
`s[ : :-1]`	`jihgfedcba`	a negative step reverses the string

- Temel yapı aşağıdaki gibidir:
- ***stringAdı[başlangıçKonumu: bitişKonumu+1]***
- Dilimlemeler, *range* fonksiyonuyla aynı şekilde çalışır; bitiş konumunu içermezler. Örneğin, yukarıdaki örnekte, s[2:5], 2, 3 ve 4. indekslerdeki karakterleri verir, ancak 5. indeksteki karakteri vermez.

Döngü (Looping)

- Bir metin dizisini karakter karakter taramak gerekebilir. Aşağıdaki gibi bir for döngüsü bunu yapmak için kullanılabilir. Bu döngü, s adlı bir metin dizisini karakter karakter, her birini ayrı bir satıra yazdırarak işler:

```
1   s="python"
2
3   for i in range(len(s)):
4       print (s[i])
```

- «range» ifadesinde «len(s)» bulunmakta ve bu, «s»'nin uzunluğunu döndürür. Yani, «s» 6 karakter uzunluğundaysa, bu «range(6)» gibi olur ve döngü değişkeni «i» 0'dan 5'e kadar çalışır. Bu, «s[i]»'nin «s»'nin karakterleri boyunca çalışacağı anlamına gelir. Bu döngü yöntemi, döngü sırasında dizedeki konumumuzu takip etmemiz gerektiğinde kullanışlıdır.
- Eğer konumumuzu takip etmemiz gerekmiyorsa, kullanabileceğimiz daha basit bir döngü türü vardır:

```
1   s="python"
2
3   for c in s:
4       print(c)
```

String Fonksiyonları

- Stringler, metin hakkında bilgi döndüren veya orijinal dizenin değiştirilmiş bir versiyonu olan yeni bir metin döndüren çok sayıda yöntem ve fonksiyonla birlikte gelir. En kullanışlı olanlardan bazıları:

Method	Description
`lower()`	returns a string with every letter of the original in lowercase
`upper()`	returns a string with every letter of the original in uppercase
`replace(x,y)`	returns a string with every occurrence of `x` replaced by `y`
`count(x)`	counts the number of occurrences of `x` in the string
`index(x)`	returns the location of the first occurrence of `x`
`isalpha()`	returns `True` if every character of the string is a letter

- **NOT:** Yukarıdaki staring fonksiyonlarından örnek olarak eğer bir metni (s) tamamen küçük harfe dönüştürmek istiyorsanız, sadece s.lower() kullanmak yeterli değildir. Şunları yapmanız gerekir: *s = s.lower()*
- **Kısa Örnekler:**

Statement	Description
`print(s.count(' '))`	prints the number of spaces in the string
`s = s.upper()`	changes the string to all caps
`s = s.replace('Hi','Hello')`	replaces each `'Hi'` in `s` with `'Hello'`
`print(s.index('a'))`	prints location of the first `'a'` in `s`

String Fonksiyonları

- **isalpha:** `isalpha` metodu, bir karakterin harf olup olmadığını belirlemek için kullanılır. Karakter harf ise `True`, aksi takdirde `False` döndürür. Tam bir metin ile kullanıldığında, yalnızca metnin her karakteri harf ise `True` döndürür. `isalpha` metodunu `if` koşullarında kullanılabilir. Kullanım örneği:

```
1   s = input('Bir cumle giriniz: ')
2
3   if s[0].isalpha():
4       print('Cumlen bir harf ile basliyor')
5   if not s.isalpha():
6       print('Cumlen harf disinda bir karakter içeriyor.')
```

- **Diğer Fonksiyonlar:** Daha birçok string metodu vardır. Örneğin, isalpha'ya benzer **isdigit** ve **isalnum** metotları mevcuttur. Daha sonra öğreneceğimiz diğer kullanışlı metotlar arasında **join** ve **split** bulunur. Tüm string metotlarının listesini görmek için Python komut satırına **dir(str)** yazılarak listelenir. Metotlardan birinin, örneğin isdigit metodunun Python dokümantasyonunu okumak için **help(str.isdigit)** yazılır.

Escape Karakterleri

- Ters slash, \, belirli özel karakterleri (kaçış karakterleri olarak da adlandırılır) metninize eklemek için kullanılır. Çeşitli kaçış karakterleri vardır ve en kullanışlı olanları:
- **\n** Yeni satır karakteri. Sonraki satıra geçmek için kullanılır. Örnek:

```
print('Hi\n\nthere!')
```

```
Hi

There!
```

- **\'** Metinlere kesme işareti eklemek için '\' kullanılır. Örneğin, aşağıdaki metne sahip olduğunuzu varsayalım:

```
s = 'I can\'t go'
```

- **** Bu, ters eğik çizginin kendisini elde etmek için kullanılır. Örnek:

```
filename = 'c:\\programs\\file.py'
```

Örnekler

- **Örnek 1:** Boş bir satır yazdırmanın kolay bir yolu `print()` fonksiyonudur. Ancak, on tane boş satır yazdırılmanın bir yolu olarak:

```
1    print('\n'*9)
```

- **Örnek 2:** Kullanıcıdan bir cümle isteyen ve bu metindeki her 'a' harfinin yerini yazdıran bir program yazın.

```
1    s = input('Bir kelime giriniz.: ')
2    for i in range(len(s)):
3        if s[i]=='a':
4            print(i)
```

- Metinde karakter karakter taramak için bir döngü kullanıyoruz. Döngü değişkeni i, dizedeki konumumuzu takip eder ve s[i], o konumdaki karakteri verir. Dolayısıyla, üçüncü satır her karakterin 'a' olup olmadığını kontrol eder ve eğer varsa, dizideki 'a'nın konumunu (i) yazdırır.

- **Örnek 3:** Kullanıcıdan bir metin isteyen ve orijinal metin dizesinin her karakterini ikiye katlayan yeni bir metin dizesi oluşturan bir program yazın. Örneğin, kullanıcı "Hello" girerse, çıktı "HHeellllloo" olmalıdır.

```
1    s = input("Bir metin giriniz: ")
2
3    kelime = ""
4    for i in range(len(s)):
5        kelime = kelime + s[i]*2
6
7    print(kelime)
```

Örnekler

- **Örnek 4:** Kullanıcıdan adını isteyen ve bu adı aşağıdaki eğlenceli düzende yazdıran bir program yazın, (E El Elv Elvi Elvis)

```
isim = input("isminizi yaziniz: ")

yeniKelime = ""
sayac = 1

for i in range(len(isim)):
    yeniKelime = yeniKelime +  isim[0:sayac] + ' '
    sayac= sayac + 1
```

- **Örnek 5:** Bir metin olan `s`'ten tüm büyük harfleri ve yaygın noktalama işaretlerini kaldıran bir program yazınız.

```
s = input ("Metin: ")

s = s.lower()
for c in ',.;:-?!()\'"':
    s = s.replace(c, '')
print (s)
```

- Bu işlem şu şekilde çalışır: Noktalama işaretleri dizisindeki her karakter için, s içindeki her geçişini boş karakter '' ile değiştiririz. Burada teknik bir not: Noktalama karakterlerinde ' karakterine ihtiyacımız var. Önceki bölümde açıklandığı gibi, bu karakteri \' kaçış karakterini kullanarak metne dahil ederiz.

Örnekler

- **Örnek 6:** Ondalık sayı içeren bir dize verildiğinde, sayının ondalık kısmını yazdıran bir program yazın. Örneğin, 3.14159 verildiğinde, program .14159 yazdırmalıdır.

```
1    s = "3.1432532423232334"
2
3    index_s = s.index('.')
4    ondalikSonrasiSayi = s[index_s:]
5    print(ondalikSonrasiSayi)
```

- Matematiksel farklı bir yol:

```
1    from math import floor
2
3    num = float(input('Ondalikli bir sayi giriniz: '))
4    print(num - floor(num))
```

- İki yöntem arasındaki farklardan biri, birincisinin bir metin, ikincisinin ise bir sayı tipinde çıktı üretmesidir.

Örnekler

- **Örnek 7:** Gizli mesaj göndermenin basit ve çok eski bir yöntemi, ikame şifrelemesidir (cipher). Temelde, alfabenin her harfi alfabenin başka bir harfiyle değiştirilir; örneğin, her a harfi x ile, her b harfi z ile vb. değiştirilir. Bunu uygulayan bir program yazın. Aşağıdaki dönüşüm harfleri belirtilmektedir.

```
alphabet = 'abcdefghijklmnopqrstuvwxyz'
key = 'xznlwebgjhqdyvtkfuompciasr'
```

```
1   alphabet = 'abcdefghijklmnopqrstuvwxyz'
2   key = 'xznlwebgjhqdyvtkfuompciasr'
3   secret_message = input('Enter your message: ')
4   secret_message = secret_message.lower()
5
6   for c in secret_message:
7       if c.isalpha():
8           print(key[alphabet.index(c)],end='')
9   else:
10      print(c, end='')
```

- Metin anahtarı, alfabenin rastgele yeniden sıralanmasından oluşur.
- Programın önemli kısmı for döngüsüdür. Bu döngü, mesajı karakter karakter inceler ve bulduğu her harf için, onu anahtardaki karşılık gelen harfle değiştirir. Bu, geçerli harfin alfabedeki konumunu bulmak için indeks yöntemini kullanarak ve bu harfi o konumdaki anahtardaki harfle değiştirerek gerçekleştirilir. Harf olmayan tüm karakterler olduğu gibi kopyalanır. Program, geçerli karakterin harf olup olmadığını belirlemek için isalpha yöntemini kullanır.
- Mesajı deşifre etmek için kod neredeyse aynıdır. Sadece key[alphabet.index(c)] ifadesini alphabet[key.index(c)] olarak değiştirilir.

Alıştırmalar

- **Soru 1:** Kullanıcıdan bir metin girmesini isteyen bir program yazın. Program daha sonra aşağıdakileri yazdırmalıdır:
  - (a) Metnin toplam karakter sayısı
  - (b) Metnin 10 kez tekrarlanmış hali
  - (c) Metnin ilk karakteri (dize indekslerinin 0'dan başladığını unutmayın)
  - (d) Metnin ilk üç karakteri
  - (e) Metnin son üç karakteri
  - (f) Metnin tersten yazılmış hali
  - (g) Metnin yeterince uzunsa yedinci karakteri, aksi takdirde bir mesaj
  - (h) Metnin ilk ve son karakterleri çıkarılmış hali
  - (i) Metnin tamamen büyük harflerle yazılmış hali
  - (j) Metnin her "a" harfi "e" ile değiştirilmiş hali

Örnekler

- **Soru 2:** Bir metindeki kelime sayısını tahmin etmenin basit bir yolu, dizideki boşluk sayısını saymaktır. Kullanıcıdan bir metin dizisi isteyen ve dizideki kelime sayısının tahmini bir değerini döndüren bir program yazınız.

- **Soru 3:** İnsanlar formül girerken sıklıkla kapanış parantezlerini unuturlar. Kullanıcıdan bir formül girmesini isteyen ve formülün açılış ve kapanış parantezlerinin sayısının eşit olup olmadığını yazdıran bir program yazın.

- **Soru 4:** Kullanıcıdan bir kelime girmesini isteyen ve bu kelimenin sesli harf içerip içermediğini yazdıran bir program yazın.

- **Soru 5:** Kullanıcıdan bir metin girmesini isteyen bir program yazın. Program, kullanıcının girdiği dizeden ikinci karakteri yıldız işaretiyle değiştirip sonuna üç ünlem işareti ekleyerek new_string adında yeni bir metin oluşturmalıdır. Son olarak, new_string'i yazdırın.

  Tipik çıktı aşağıda gösterilmiştir:

  Metin girin: Qbert

  *Q*ert!!!*

Örnekler

- **Soru 6:** Kullanıcıdan bir metin (s) girmesini isteyen, ardından s'yi küçük harflere dönüştüren, s'deki tüm noktaları ve virgülleri kaldıran ve elde edilen metni yazdıran bir program yazın.

- **Soru 7:** Kullanıcıdan bir metin dizesi girmesini isteyen ve ardından dizenin her harfini ikiye katlayarak ayrı bir satırda yazdıran bir program yazın.

  Örneğin, kullanıcı "HEY" girerse, çıktı şu şekilde olacaktır:

  HH

  EE

  YY