Veri Tipleri

Prof.Dr. Bahadır AKTUĞ
JFM212 Python ile Mühendislik Uygulamaları

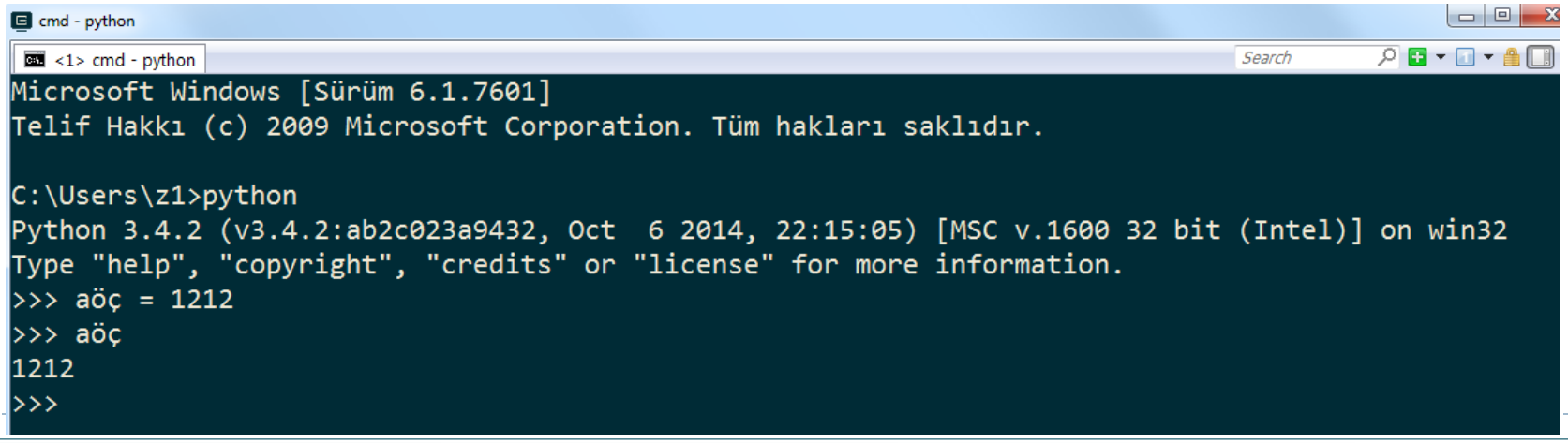
**Kaynakça bölümünde verilen kaynaklardan derlenmiştir.*

Python

- Python dinamik tipli bir dildir.
- Dinamik tipli bir dildeki değişkenlerin tipi, kuvvetli tipli (C/C++/Pascal vb.) dillerin aksine veriye göre değişebilir.
- Örneğin aynı değişken aynı program bloğu içerisinde hem metin (string) hem de tamsayı (integer) tipinde değişken saklayabilir.
- Buna karşın, değişkenin o anki tipi ile işlem uyumlu olmalıdır.
- a = "Python Dil"
- a=1457
- a = input()
- print(int(a))

Python Değişken Adları

- Python 3 ile birlikte değişken isimlendirme oldukça esnek hale gelmiştir.
- Python 3'de değişkenler ile ilgili kurallar genel olarak aşağıdaki şekilde verilebilir:
- Değişkenin ilk karakteri "_" veya büyük/küçük harf olmak zorundadır.
- Harfler Unicode olarak herhangi bir harf olabilir



Python Değişken Adları

- Python büyük-küçük harf duyarlı bir dildir. Bu kural değişken adları için de geçerlidir.
- Python anahtar sözcükleri (komut vs.) değişken adı olamaz. Python anahtar sözcükleri
  - *and, as, assert, break, class, continue, def, del, elif, else, except, False, finally, for, from, global, if, import, in, is, lambda, None, nonlocal, not, or, pass, raise, return, True, try, while, with, yield*
- Değişken adlarında ilk karakterden sonra herhangi bir harf veya rakam gelebilir.

Python'da Sayılar

Tamsayılar

- Onluk sistemdeki normal tamsayılar
- Sekizlik tabandaki tamsayılar: (başında sıfır ve "o" harfi olmalıdır)

```
>>> a = 0o20
>>> print(a)
>>> 16
```

- 16'lık tabandaki tamsayılar (başında sıfır ve "x" veya "X" harfi olmalıdır)

```
>>> a = 0x10
>>> print(a)
>>> 16
```

- 2'lik tabandaki tamsayılar (başında sıfır ve "b" veya "B" harfi olmalıdır)

```
>>> a = 0b110
>>> print(a)
>>> 6
```

Python'da Sayılar

Tamsayılar

- Onluk tabanda ifade edilen sayılar diğer tabanlara aşağıdaki şekilde dönüştürülebilir:
- Onluk tabandan 8'lik tabana

  >>> a = 16

  >>> print(oct(a))

  >>> '0x20' (string olarak dönüştürüldüğüne dikkat edilmelidir)
- 10'luk tabandan 16'lık tabana
- >>> a = 16

  >>> print(hex(a))

  >>> '0x10'
- 10'luk tabandan 2'lik tabana
- >>> a = 16

  >>> print(bin(a))

  >>> '0b10000'

Python'da Sayılar

Tamsayılar

- Tamsayılar için herhangi bir sınırlama yoktur:

```
>>> x = 787366098712738903245678234782358292837498729182728
>>> x * x * x
488123970070638215986770162105731315538827586091948617997871122950228891123960901918308618286311523282239313708275589787123005317148968569797875581092352
```

Kesirli ve Kayan Noktalı Sayılar

```
>>> a = 14.56
>>> a = 2.4583e-8
```

Karmaşık Sayılar

- Python'da karmaşık sayılar doğrudan tanımlanarak üzerlerinde işlem yapılabilir.

```
>>> a = 3 – 5j
>>> b = 4 +7j
>>> a+b
```

Python'da Metin Tipi Değişken (string)

- Bilgisayar dünyasında ardışık harf gruplarının temsil edilmesi için string tipine ihtiyaç vardır.
- ASCII kodlama 256 (2^8) farklı karakterin gösterilmesine izin verir.
- Buna karşın, tüm dillerin ve matematiksel sembollerin gösterimi için yeterli değildir. Bu nedenle ASCII yerine Unicode standardı oluşturulmuştur.
- Unicode ASCII'nin 1 byte'lık gösterimi yerine 4 byte'lık gösterime olanak tanır.
- Unicode'un 4 byte'lık gösterimi ile $(2^8)^4$ > 4 Milyon karakter gösterilebilir.
- Unicode'un 4 byte'lık karakter haritalaması, 1 byte ile temsil edilebilecek birçok harf için 4 byte ayırdığından farklı Unicode kodlamaları geliştirilmiştir (UF-8, UTF-16 ve UTF-32)

Python'da Metin Tipi Değişken (string)

- Python'da karakterle herhangi bir ilave kodlama olmaksızın Unicode olarak ifade edilir
- Metin tipi değişkenler tek veya çift tırnak ile gösterilebilir:

```
>>> a = 'JFM212'
>>> a = "JFM212"
```

- Kullanılacak metin içinde tırnak işareti varsa, öncesinde backslash (\) kullanılmalıdır. Değişken tanımlaması tek tırnak ile yapılmışsa, içinde çift tırnak, çift tırnak ile yapılmışsa içinde tek tırnak doğrudan kullanılabilir.

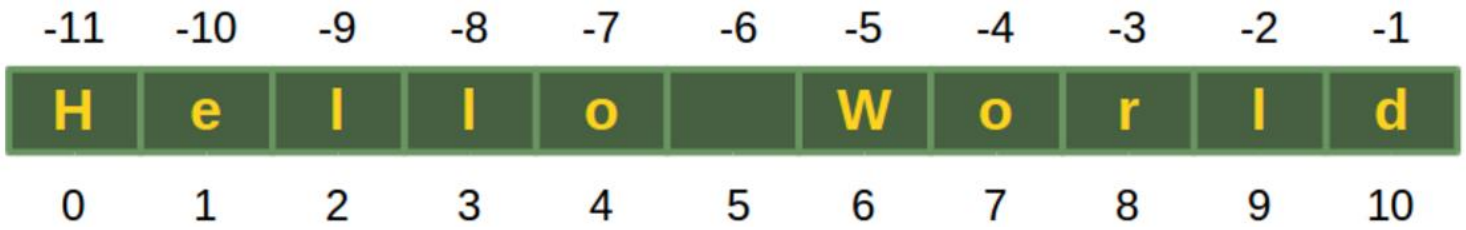
```
>>> a = 'JFM212\'nin içeriği'
>>> a = "JFM212 \"nin içeriği "
```

- Birden fazla "comment" ve "açıklama" için ayrıca ''' operatörü de kullanılabilir.

Python'da Metin Tipi Değişken (string)

- Python'da string değişken tipinde tek tek karakterlere dizi gibi ulaşılabilir.

```
>>> s = 'Hello World'
>>> s[0]
>>> 'H'
```



- En son karaktere de iki şekilde erişilebilir:

```
>>> s[len(s)-1]
>>> 'd'
>>> s[-1]
>>> 'd'
```

Python'da Metin Tipi Değişken (string)

Birleştirme (Concatenation):

- Python'da metin birleştirme "+" operatörü ile yapılır.

```
>>> a = 'JFM212'
>>> b = " Python ile Mühendislik Uygulamaları"
>>> a+b
>>> 'JFM212 Python ile Mühendislik Uygulamaları'
```

Tekrarlama (Repetition):

- Python'da metin tekrarlama "*" operatörü ile yapılır.

```
>>> a = 'AB'
>>> 3*a
>>> 'ABABAB'
```

Python'da Metin Tipi Değişken (string)

İndisleme (Indexing):

- Python'da indisleme "[]" operatörü ile yapılır.
- Metin tipi değişkenlerde negatif indislerde kullanılabilir.

```
>>> a = 'AB'
>>> a[1]
>>> 'B'
>>> a[0]
>>> 'A'
>>> a[-1]
>>> 'B'
>>> a[-6]
>>> 'A'
>>> a[-7]
>>> Hata mesajı
```

**İndisleme 0'dan başlar!*

**Değişkenin başına geldikten sonra tekrar ilerlemez!*

Python'da Metin Tipi Değişken (string)

Dilimleme (Slicing):

- Python'da dilimleme "[:]" operatörleri ile yapılır.
- ":" operatörünün sağı ve solunda başlangıç ve bitiş indisleri yer alır.

```
>>> a = 'Ankara'
>>> a[3:5]
>>> 'ar'
```

- Dilimleme başlangıç/bitiş boş bırakılabilir. Bu durumda en baştan/en sona anlamına gelir

```
>>> a[:4]
>>> 'Anka'
>>> a[4:]
>>> 'ra'
```

**Dilimlemelerin indisleme gibi 0'dan başladığına ve de ikinci dilim indisinin dilime dahil olmadığına dikkat ediniz.*

Python'da Metin Tipi Değişken (string)

Boyut (Size/Length):

- Python'da bir metin veri tipinin boyutunun bulunması için "len" fonksiyonu kullanılır.
- "len" fonksiyonu karakter sayısını verir.
- Boşluk dahildir.
- Metin değişkeninin son karakterine ulaşmak için a[len(a)-1] kullanılabilir.

```
>>> a = 'Ankara'
>>> len(a)
>>> 6
>>> a = 'Ankara İstanbul'
>>> len(a)
>>> 15
```

Python'da Metin Tipi Değişken (string)

Değişebilirlik (mutable) ve Değişmezlik (immutable) Kavramları

- Değişebilirlik (mutable) ve Değişmezlik (immutable) kavramları fonksiyonlar ve nesne metodları ile çalışırken de karşımıza çıkacak konulardır.
- Python'da metin tipi veri tipi "değişmez (immutable)" türdendir. Bunun anlamı bir metin'in içindeki harfleri normal atama ile değiştirilemez.

```
>>> a = 'Ankara'
>>> a[0] = 'O'
>>> Hata mesajı .....
```

**"Onkara" yapılmaya çalışılıyor*

Python'da Metin Tipi Değişken (string)

Metin tipi değişkenlerin hafızada tutulması:

- Python'da her tür değişken birer nesnedir ve belirli bir hafıza adresinde tutulurlar. Değişkenlerin içeriği "==" operatörü ile karşılaştırılabilir. Aynı hafıza adresini gösterip göstermediklerini karşılaştırma için "is" operatörü kullanılır.

```
>>> a = 'Ankara'; b = "Ankara"
>>> a == b
>>> True
>>> a is b
>>> True
>>> a = 'Med-Cezir'
>>> b = "Med-Cezir"
>>> a == b
>>> True
>>> a is b
>>> False
```

**Aynı nesneyi gösteriyorlar (aynı hafıza adresini gösteriyorlar)*

**Aynı nesneyi göstermiyorlar (aynı hafıza adresini göstermiyorlar)*

Python'da Metin Tipi Değişken (string)

Metin tipi değişkenler içindeki özel karakterler (Escape Sequences):

- Metin tipi değişkenlerin içeriğinde özel karakterler bulunabilir. Bunların metin içindeki diğer karakterlerden ayrılması için başlarında "\" operatörü yer alır.

Escape Sequence	Meaning Notes
\newline	Ignored
\\	Backslash (\)
\'	Single quote (')
\"	Double quote (")
\a	ASCII Bell (BEL)
\b	ASCII Backspace (BS)
\f	ASCII Formfeed (FF)
\n	ASCII Linefeed (LF)
\N{name}	Character named name in the Unicode database (Unicode only)
\r	ASCII Carriage Return (CR)
\t	ASCII Horizontal Tab (TAB)
\uxxxx	Character with 16-bit hex value xxxx (Unicode only)
\Uxxxxxxxx	Character with 32-bit hex value xxxxxxxx (Unicode only)
\v	ASCII Vertical Tab (VT)
\ooo	Character with octal value ooo
\xhh	Character with hex value hh

Değişken Atamaları

- Python dinamik tipli bir dildir. Değişkenin tipini değişkenin içeriği belirler.
- Aynı değişken program içinde farklı tiplerde değişken olarak kullanılabilir.
- Bunun yanında "güçlü tipli (strongly typed)" bir dil de sayılabilir. Zira, bir değişken tipi belirlendikten sonra atandığı tipe uygun işlemlerde kullanılabilir.

```
>>> a = "Gölbaşı"
>>> a = 27e12
>>> a = 1451 * 2321
```

Değişken Atamaları

- Diğer yandan Python'da tüm değişkenlerin birer nesne olduğu dikkate alındığında, değişkenlerin birbirlerine atanmasına dikkat edilmelidir.
- Bir değişkene bir değer atandığında, hafızada bu değeri tutmak için bir adres oluşturulur.
- Değişkenlerin birbirlerine atanmasında sadece bu adres ataması yapılır, içerik kopyalanmaz.
- Adres kopyalanması, birbirlerine atanan iki değişkenden birinde yapılacak değişikliklerin diğerini de etkilemesi anlamına gelir. Ama, Python'da bir değişkene yeni bir değer atandığında aynı zamanda yeni bir adres de verilerek bu durumun önüne geçilir.

```
>>> a = "Gölbaşı"
>>> b = a
>>> b = 1451
>>> a
>>> "Gölbaşı"
```

Kaynakça

1 Wentworth, P., Elkner, J., Downey, A.B., Meyers, C. (2014). How to Think Like a Computer Scientist: Learning with Python (3nd edition).
2 Pilgrim, M. (2014). Dive into Python 3 by. Free online version: DiveIntoPython3.org ISBN: 978-1430224150.
3 Summerfield, M. (2014) Programming in Python 3 2nd ed (PIP3) : - Addison Wesley ISBN: 0-321-68056-1.
4 Summerfield, M. (2014) Programming in Python 3 2nd ed (PIP3) : - Addison Wesley ISBN: 0-321-68056-1.
5 Jones E, Oliphant E, Peterson P, et al. SciPy: Open Source Scientific Tools for Python, 2001-, http://www.scipy.org/.
6 Millman, K.J., Aivazis, M. (2011). Python for Scientists and Engineers, Computing in Science & Engineering, 13, 9-12.
7 John D. Hunter (2007). Matplotlib: A 2D Graphics Environment, Computing in Science & Engineering, 9, 90-95.
8 Travis E. Oliphant (2007). Python for Scientific Computing, Computing in Science & Engineering, 9, 10-20.
9 Goodrich, M.T., Tamassia, R., Goldwasser, M.H. (2013). Data Structures and Algorithms in Python, Wiley.
10 http://www.diveintopython.net/
11 https://docs.python.org/3/tutorial/
12 http://www.python-course.eu
13 https://developers.google.com/edu/python/
14 http://learnpythonthehardway.org/book/