



Python in Data Sci (1)



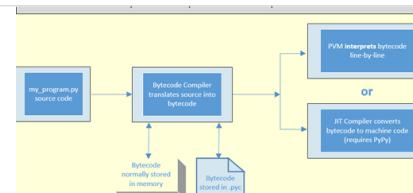
▼ Links

Python Programlama dili ve Development Asamaları:

Python Programlama Dili

Python Programlama Dilinin Tarihsel Gelişimi, Python'ın Avantajları ve Dezavantajları, Program Geliştirme Aşamaları, Python'ın Uygulama...

<https://cabircelik.medium.com/python-programlama-dili-8cb025eb12c1>

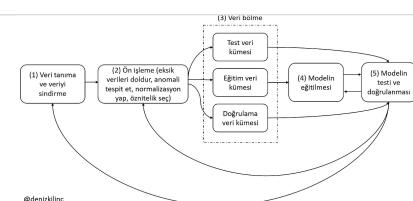


Python in Data Science;

Python ile Veri Bilimine Dalış

Veri Bilimi ve Python

[https://medium.com/deep-learning-turkiye/python-ile-veri-bilimine-dalı-3f069260eda](https://medium.com/deep-learning-turkiye/python-ile-veri-bilimine-dal%C4%91-3f069260eda)



Python Documentation Webpage;

5. Data Structures

This chapter describes some things you've learned about already in more detail, and adds some new things as well. More on Lists: The list data type has some more methods. Here are all of the method...

<https://docs.python.org/3/tutorial/datastructures.html>



▼ Functions in Python

Python da Fonksiyonlar Nedir, Nasıl Kullanılır? #3

"Herkese merhaba ben Nurcan, SistersLab'in Toplum Gönüllüleri Vakfı
(link:<https://www.tog.org.tr/en/>) tarafından desteklenen Women in Tech..."

📹 <https://medium.com/@nrcntpkr/python-da-fonksiyonlar-nedir-nasıl-kullanılır-eaf104f75e7>



▼ Presentation topics



Why Python is essential for data analysis and data science?



How does it differ from other programming languages?

Assignment Description:

In this task, you are asked to investigate why Python is important for data analysis and data science and how it differs from other programming languages, and prepare your findings in a short presentation or report.

▼ Expectations for Your Presentation:

▼ 1. Python basics.



Python, genel amaçlı bir programlama dilidir ve hem başlangıç düzeyindeki kullanıcılar hem de deneyimli programcılar tarafından kullanılabilir. Python'un temellerini anlamak, dilin sözdizimini, veri tiplerini, kontrol yapılarını ve fonksiyonları içerir. İşte Python'un temel kavramlarının bir açıklaması:



Sözdizimi ve Okunabilirlik: Python'un sözdizimi oldukça basittir ve insanların doğal dilde yazılmış gibi görünmesini sağlar. Bu, Python kodunun okunabilirliğini artırır ve hata ayıklamayı kolaylaştırır.



Veri Tipleri: Python'da temel veri tipleri şunlardır:

- **Integer (int):** Tam sayıları temsil eder.
- **Float (float):** Ondalık sayıları temsil eder.
- **String (str):** Metin verilerini temsil eder.
- **List (list):** Birden çok öğeyi saklamak için kullanılan değiştirilebilir bir veri yapısıdır.
- **Tuple (tuple):** Değiştirilemez bir veri yapısıdır ve parantez içinde elemanları saklar.
- **Dictionary (dict):** Anahtar-değer çiftlerini saklar.



Kontrol Yapıları: Python'da kontrol yapıları şunlardır:

- **If ifadesi:** Belirli bir koşulun doğru olup olmadığını kontrol eder ve eylemleri buna göre belirler.
- **For döngüsü:** Bir dizi veya başka bir yinelenebilir nesne üzerinde döngü yapmak için kullanılır.
- **While döngüsü:** Belirli bir koşul doğru olduğu sürece bir döngüyü yürütür.
- **Break ve Continue:** Döngülerde çalışmayı durdurmak veya bir sonraki yinelemeye geçmek için kullanılır.



Fonksiyonlar: Python'da fonksiyonlar, belirli bir görevi yerine getiren ve adlandırılan bir blok kodu temsil eder. Fonksiyonlar, kodun tekrar kullanılabilirliğini artırır ve karmaşık işlemleri parçalara ayırarak daha yönetilebilir hale getirir.

▼ Data structures can help you to focus on the bigger picture rather than getting lost in the details.

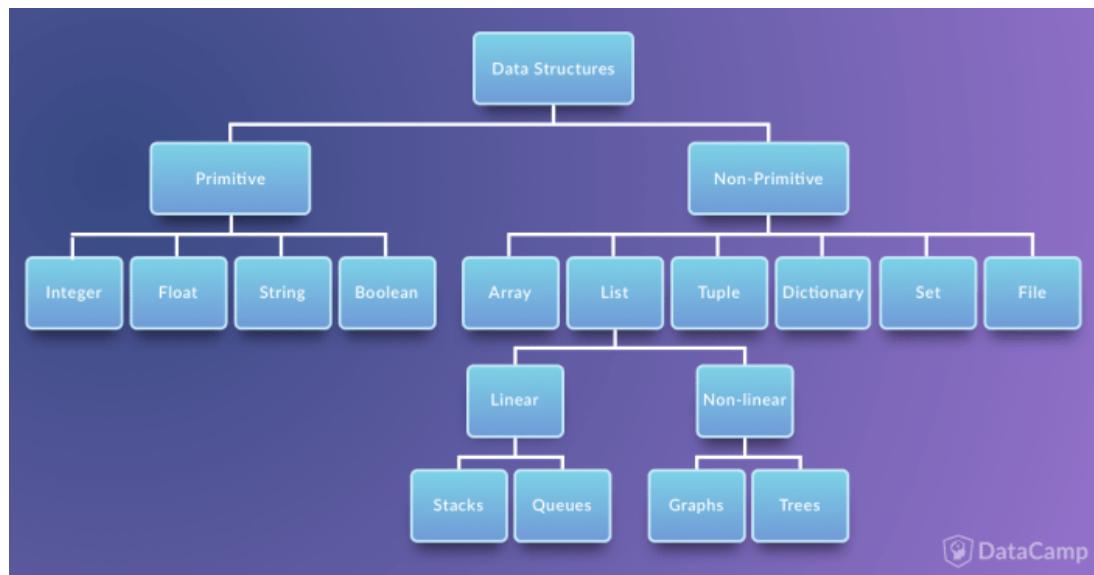
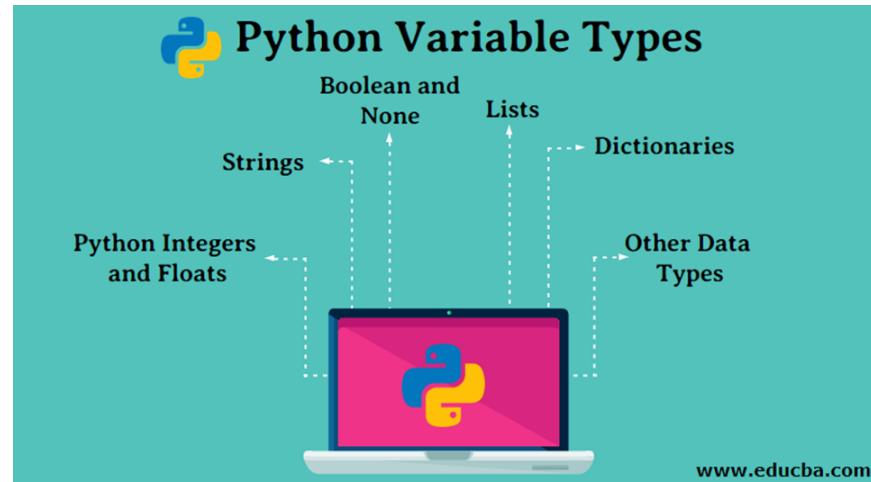
This is known as data abstraction. Data structures are actually an implementation of **Abstract Data Types or ADT**.

Generally, data structures can be divided into two categories in computer science: **primitive** and **non-primitive** data structures.

Python Data Structures with Primitive & Non-Primitive Examples

Learn how to use Python Data Structures to store your data. Understand primitive and non-primitive data structures, such as strings, lists and stacks today!

<https://www.datacamp.com/tutorial/data-structures-python>



Primitive types are the most primitive or basic data structures. They are the building blocks for data manipulation and contain pure, simple values of data.

Python has four primitive variable types:

- Integers
- Float
- Strings
- Boolean

Non-primitive types are the sophisticated members of the data structure family. They don't just store a value, but rather a collection of values in various formats.

In the traditional computer science world, the non-primitive data structures are divided into:

- Arrays
- Lists
- Files

▼ 2. Python development process



Gereksinim Analizi/ User Requirements Analysis:

Herhangi bir yazılım projesinde olduğu gibi, Python geliştirme süreci de gereksinim analiziyle başlar. Bu aşamada, projenin hedefleri belirlenir, kullanıcı gereksinimleri belirlenir ve projenin kapsamı netleştirilir.



Development Environment:

Python'u kullanarak yazılım geliştirmek için geliştirme ortamını (IDE veya metin düzenleyici), Python yürütme ortamını ve gereksinim duyulan paketleri yüklemek önemlidir. Bu adım, geliştirme sürecinin başında yapılır.



Kod Yazma / Development:

Geliştirme ortamını kurduktan sonra, kod yazma aşamasına geçilir. Python'un basit ve okunabilir sözdizimi, kodun yazılmasını kolaylaştırır. Geliştirme sürecinin bu aşamasında, gereksinimlere uygun kod parçaları oluşturulur.



Test Etme ve Hata Ayıklama / Testing and Debugging:

Kod yazıldıktan sonra, yazılım test edilir ve hata ayıklama yapılır. Python'un test yazma ve hata ayıklama süreçlerini kolaylaştırın araçları vardır. Bu aşamada, yazılımın doğru çalıştığından ve beklenen sonuçları ürettiğiinden emin olmak için farklı test senaryoları çalıştırılır.



Optimizasyon ve Performans İyileştirmesi:

Geliştirme sürecinin bu aşamasında, kodun performansını artırmak için gerekli optimizasyonlar yapılır. Bu, kodun daha hızlı çalışmasını sağlamak veya daha az bellek kullanmasını sağlamak için yapılan iyileştirmeleri içerir.



Belgeleme / Reports:

Yazılım geliştirme sürecinin önemli bir parçası, kodun ve projenin belgelenmesidir. Python'un içinde bulunan ve dışarıdan yüklenen kütüphanelerle birlikte projeyi kullanacak diğer geliştiricilerin veya kullanıcıların kodu ve projeyi anlamasına yardımcı olmak için belgeler oluşturulur.



Paketleme ve Dağıtım /Distribution:

Yazılım, kullanıcılar tarafından kullanılabilir hale getirilir. Python'un paketleme ve dağıtım araçları, yazılımın farklı platformlara dağıtılmasını kolaylaştırır. İşte bunlardan bazıları:

1. **pip**: Python'un standart paket yönetici olan `pip`, Python paketlerini kurmak, kaldırmak ve yönetmek için kullanılır. `pip` aracılığıyla birçok üçüncü taraf kütüphanesi kolayca indirip yüklenebilir.
2. **setuptools**: Python paketlerini paketlemek ve dağıtmak için kullanılan bir kütüphanedir. `setup.py` adında bir dosya oluşturularak paketin yapılandırması ve gereksinimleri belirlenir. Daha sonra `setuptools` aracılığıyla bu paket, `pip` aracılığıyla yüklenip dağıtılabılır.
3. **cx_Freeze**: Python betiklerini Windows, Linux ve macOS gibi işletim sistemlerinde çalıştırılabilir uygulamalara dönüştürmek için kullanılır. Bu, Python yazılımının dağıtımını ve kullanımını kolaylaştırır.



Python geliştirme süreci, genellikle bu adımları izler, ancak projenin özelliklerine ve gereksinimlerine göre farklılık gösterebilir. Her aşama, yazılımın kalitesini artırmak ve istenilen sonuçları elde etmek için önemlidir.

▼ 3. Why Python is important for data analysis and data science

Python, veri analizi ve veri bilimi için öncelikli bir dil haline gelmiştir çünkü kullanımı kolay, esnek, güçlü ve yaygın bir topluluğa sahiptir;



Kolay Okunabilir ve Kullanılabilir:

Python'un basit ve okunabilir sözdizimi, kullanıcıların kodları hızlıca anlamalarına ve yazmalarına yardımcı olur. Bu, veri bilimcilerin ve analistlerin daha verimli çalışmalarını sağlar.



Geniş Kütüphane Desteği:

Python, veri analizi ve bilimi için zengin bir kütüphane ekosistemine sahiptir. NumPy, Pandas, Matplotlib, SciPy gibi kütüphaneler, veri manipülasyonu, analizi ve görselleştirmesi için gerekli araçları sağlar.



Ucretsiz ve Açık kaynaklı olması / Open Resource;

bagimsiz kullanıcılar tarafından geliştirilebilir



Veri Yönetimi ve Manipülasyonu:

Pandas kütüphanesi, veri çerçevelerini (data frames) etkili bir şekilde yönetmek ve işlemek için kullanılır. Bu, veri analizi süreçlerini kolaylaştırır ve hızlandırır.



Gelişmiş Görselleştirme:

Matplotlib, Seaborn gibi kütüphaneler, veriyi grafikler, histogramlar ve diğer görsel öğeler aracılığıyla etkili bir şekilde görselleştirmeyi sağlar. Bu, verilerin anlaşılmasını kolaylaştırır ve analiz sürecini geliştirir.



Makine Öğrenimi ve Derin Öğrenme:

Python, makine öğrenimi ve derin öğrenme için popüler kütüphaneler olan Scikit-learn, TensorFlow ve PyTorch gibi araçları destekler. Bu, veri bilimi projelerinin karmaşıklığını yönetmeyi ve geliştirmeyi kolaylaştırır.

▼ 5. Data analysis and data science applications with Python



Veri Manipülasyonu ve Temizleme:

Pandas kütüphanesi, veri manipülasyonu ve temizleme için yaygın olarak kullanılır. Veri setlerini yüklemek, dönüştürmek, filtrelemek, birleştirmek ve özetlemek gibi işlemleri kolayca gerçekleştirebilirsiniz.



Veri Görselleştirme:

Matplotlib, Seaborn ve Plotly gibi kütüphanelerle veri görselleştirme yapılabilir. Bu kütüphanelerle grafikler, histogramlar, kutu grafikleri, dağılım grafikleri ve ısı haritaları gibi görsel öğeler oluşturulabilir. Verilerin daha anlaşılabilir hale gelmesine yardımcı olur.



Makine Öğrenimi:

Python, makine öğrenimi için çeşitli kütüphanelere sahiptir. Scikit-learn, makine öğrenimi modellerinin uygulanması ve değerlendirilmesi için yaygın olarak kullanılan bir kütüphanedir. Ayrıca TensorFlow ve PyTorch gibi derin öğrenme kütüphaneleriyle karmaşık sinir ağları oluşturabilir ve eğitebilirsiniz.



Veri Madenciliği:

Python, veri madenciliği için de kullanılabilir. Veri madenciliği, büyük veri kümelerinde desenler ve ilişkiler bulmayı içerir. Python'daki kütüphaneler ve araçlar, veri madenciliği algoritmalarının uygulanması için birçok seçenek sunar.



Doğal Dil İşleme (NLP):

Doğal dil işleme, metin verilerini analiz etmek ve anlamak için kullanılır. Python'daki NLTK (Doğal Dil İşleme Kütüphanesi) ve spaCy gibi kütüphanelerle, metin sınıflandırma, duygusal analiz, dil çevirisisi ve kelime öbekleme gibi NLP görevlerini gerçekleştirebilirsiniz.



Zaman Serisi Analizi:

Python, zaman serisi verilerini analiz etmek için de kullanılabilir. Pandas ve Statsmodels gibi kütüphanelerle, zaman serisi modelleri oluşturabilir, tahminler yapabilir ve zaman serisi verilerinin trendlerini ve desenlerini inceleyebilirsiniz.

▼ 4. Differences of Python from other programming languages



Basit sözdizimi: Python'un okunması ve yazılması kolaydır.



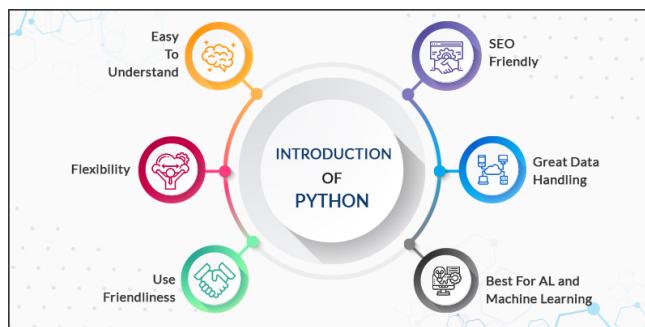
Çok yönlülük: Web geliştirme, veri bilimi, oyun geliştirme ve daha fazlası gibi çeşitli alanlarda kullanılabilir.



Geniş kütüphane desteği: Neredeyse her görev için bir kütüphane mevcuttur.



Açık kaynak kodlu: Ücretsiz ve herkes tarafından kullanılabilir ve geliştirilebilir.



▼ Example Questions

- ▼ Write a program which will find all such numbers which are divisible by 7 but are not a multiple of 5, between 2000 and 3200 (both included).The numbers obtained should be printed in a comma-separated sequence on a single line.

```
result = []

for num in range(2000, 3201):
    if num % 7 == 0 and num % 5 != 0:
        result.append(str(num))

print(','.join(result))
```

- ▼ Write a program which can compute the factorial of a given numbers. The results should be printed in a comma-separated sequence on a single line. Suppose the following input is supplied to the program: 8 Then, the output should be:40320

```
# Kullanıcıdan bir sayı al
num = int(input("Bir sayı girin: "))

# Faktöriyel hesaplama
faktoriyel = 1
for i in range(1, num + 1):
    faktoriyel *= i

# Sonucu virgülle ayrılmış bir dizi olarak yazdır
result = []
for i in range(1, faktoriyel + 1):
    result.append(str(i))
print(','.join(result))
```

- ▼ Write a program that accepts a comma separated sequence of words as input and prints the words in a comma-separated sequence after sorting them alphabetically.

Suppose the following input is supplied to the program:

without, hello, bag, world

Then, the output should be:

bag, hello, without, world

?

▼ With a given integral number n, write a program to generate a dictionary that contains $(i, i \times i)$ such that i is an integral number between 1 and n (both included). and then the program should print the dictionary. Suppose the following input is supplied to the program: 8

Then, the output should be:

```
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64}
```

```
# Kullanıcıdan bir tam sayı al
n = int(input("Bir tam sayı girin: "))

# Sözlüğü oluştur
square_dict = {}

# 1'den n'a kadar olan tamsayıların karesini hesapla ve sözlüğe ekle
for i in range(1, n + 1):
    square_dict[i] = i * i

# Sözlüğü yazdır
print(square_dict)
```