Introduction

Veri bilimi yaparken, verilerinizi verimli bir şekilde kullanabilmeniz için onları organize etmenin bir yoluna ihtiyacınız vardır. Python, verilerinizi tutmak için listeler, kümeler, sözlükler ve tuple'lar gibi birçok veri yapısına sahiptir. Bu eğitimde, Python listeleriyle nasıl çalışacağınızı öğreneceksiniz.

# Motivation

Petal to the Metal yarışmasında amacınız, bir çiçeğin türünü yalnızca görüntüsüne dayanarak sınıflandırmaktır. (Bu, bilgisayarlı görüde yaygın bir görevdir ve görüntü sınıflandırması olarak adlandırılır.) Bu amaç doğrultusunda, verilerdeki çiçek türlerinin adlarını düzenlediğinizi varsayalım.

Bunu yapmanın bir yolu, adları bir Python dizesinde düzenlemektir.

flowers = "pink primrose,hard-leaved pocket orchid,canterbury bells,sweet pea,english marigold,tiger lily,moon orchid,bird of paradise,monkshood,globe thistle"

print(type(flowers))

print(flowers)

<class 'str'>

pink primrose,hard-leaved pocket orchid,canterbury bells,sweet pea,english marigold,tiger lily,moon orchid,bird of paradise,monkshood,globe thistle

Daha da iyisi, aynı verileri bir Python listesinde temsil etmektir. Bir liste oluşturmak için köşeli parantez ([, ]) kullanmanız ve her öğeyi virgülle ayırmanız gerekir. Listedeki her öğe bir Python dizesi olduğundan, her biri tırnak işaretleri içine alınır.

flowers\_list = ["pink primrose", "hard-leaved pocket orchid", "canterbury bells", "sweet pea", "english marigold", "tiger lily", "moon orchid", "bird of paradise", "monkshood", "globe thistle"]

print(type(flowers\_list))

print(flowers\_list)

<class 'list'>

['pink primrose', 'hard-leaved pocket orchid', 'canterbury bells', 'sweet pea', 'english marigold', 'tiger lily', 'moon orchid', 'bird of paradise', 'monkshood', 'globe thistle']

İlk bakışta, bilgileri bir Python dizesinde veya listesinde göstermeniz arasında çok fark yok gibi görünüyor. Ancak göreceğiniz gibi, bir listeyle daha kolay yapabileceğiniz birçok görev var. Örneğin, bir liste şunları yapmanızı kolaylaştırır:

Belirtilen bir konumdaki (birinci, ikinci, üçüncü vb.) bir öğeyi almak,öğe sayısını kontrol etmek ve

öğe ekleyip çıkarmak.

# Lists

## **Length**

Herhangi bir listedeki girdi sayısını, "length" ifadesinin kısaltması olan len() ile sayabiliriz. Parantez içinde sadece listenin adını belirtmeniz yeterlidir.

*# The list has ten entries*

print(len(flowers\_list))

10

## **Indexing**

Listedeki herhangi bir öğeye, listedeki konumuna göre (birinci, ikinci, üçüncü vb.) başvurabiliriz. Buna indeksleme denir.

Python'un sıfır tabanlı indeksleme kullandığını unutmayın; bu da şu anlama gelir:

Listedeki ilk girdiyi çekmek için 0,

Listedeki ikinci girdiyi çekmek için 1 ve

Listedeki son girdiyi çekmek için liste uzunluğundan bir eksik değer kullanırsınız.

print("First entry:", flowers\_list[0])

print("Second entry:", flowers\_list[1])

*# The list has length ten, so we refer to final entry with 9*

print("Last entry:", flowers\_list[9])

First entry: pink primrose

Second entry: hard-leaved pocket orchid

Last entry: globe thistle

Yan Not: Yukarıdaki kod hücresinde, birden fazla öğeyi (hem bir Python dizesi ("İlk giriş:" gibi) hem de listeden bir değeri (flowers\_list[0] gibi) yazdırmak için tek bir print() kullandığımızı fark etmiş olabilirsiniz. Python'da tek bir komutla birden fazla öğeyi yazdırmak için, bunları yalnızca virgülle ayırmamız gerekir.

## **Slicing**

Bir listenin bir bölümünü de çekebilirsiniz (örneğin, ilk üç girdi veya son iki girdi). Buna dilimleme denir. Örneğin:

İlk x girdiyi çekmek için [:x],

Son y girdiyi çekmek için [-y:] kullanırsınız.

print("First three entries:", flowers\_list[:3])

print("Final two entries:", flowers\_list[-2:])

First three entries: ['pink primrose', 'hard-leaved pocket orchid', 'canterbury bells']

Final two entries: ['monkshood', 'globe thistle']

Yukarıda gördüğünüz gibi bir listeyi dilimlediğimizde, kısaltılmış yeni bir liste döndürüyor.

## **Removing items**

.remove() ile listeden bir öğeyi kaldırın ve kaldırmak istediğiniz öğeyi parantez içine koyun.

flowers\_list.remove("globe thistle")

print(flowers\_list)

['pink primrose', 'hard-leaved pocket orchid', 'canterbury bells', 'sweet pea', 'english marigold', 'tiger lily', 'moon orchid', 'bird of paradise', 'monkshood']

## **Adding items**

.append() ile listeye bir öğe ekleyin ve eklemek istediğiniz öğeyi parantez içine alın.

flowers\_list.append("snapdragon")

print(flowers\_list)

['pink primrose', 'hard-leaved pocket orchid', 'canterbury bells', 'sweet pea', 'english marigold', 'tiger lily', 'moon orchid', 'bird of paradise', 'monkshood', 'snapdragon']

## **Lists are not just for strings**

Şimdiye kadar, listedeki her öğenin bir dize olduğu listelerle çalıştık. Ancak listeler, Boole, tamsayı ve ondalıklı sayılar dahil olmak üzere herhangi bir veri türünde öğeler içerebilir.

Örnek olarak, bir perakende mağazasında Nisan 2000'in ilk haftasında ciltli kitap satışlarını ele alalım.

hardcover\_sales = [139, 128, 172, 139, 191, 168, 170]

Burada, hardcover\_sales tam sayılardan oluşan bir listedir. Dizelerle çalışırken olduğu gibi, uzunluğu alma, tek tek girdileri çekme ve listeyi genişletme gibi işlemleri yapabilirsiniz.

print("Length of the list:", len(hardcover\_sales))

print("Entry at index 2:", hardcover\_sales[2])

Length of the list: 7

Entry at index 2: 172

Ayrıca min() ile minimumu, max() ile maksimumu da alabilirsiniz.

print("Minimum:", min(hardcover\_sales))

print("Maximum:", max(hardcover\_sales))

Minimum: 128

Maximum: 191

Listedeki her öğeyi eklemek için sum() fonksiyonunu kullanın.

print("Total books sold in one week:", sum(hardcover\_sales))

Total books sold in one week: 1107

Benzer hesaplamaları listenin dilimleriyle de yapabiliriz. Bir sonraki kod hücresinde, ilk beş günün toplamını (sum(hardcover\_sales[:5])) alıp beşe bölerek ilk beş günde satılan kitapların ortalama sayısını elde ederiz.

print("Average books sold in first five days:", sum(hardcover\_sales[:5])/5)

Average books sold in first five days: 153.8