YAPAY ZEKA VE PYTHON PROGRAMLAMA

Listeler
Emir Öztürk

Dr. Öğr. Üyesi Altan MESUT'un "Programlama Dillerine Giriş" ders slaytları kaynak alınmıştır.

LİSTE

- Birden fazla elemanın bir değişken ismi ile saklanması
- Diğer dillerdeki dizi tanımından farklı
- Elemanlar birbirlerinden farklı türde olabilir.

Birbiri ile ilişkili verilerin bir sıra halinde kaydedilmesi ve işlenmesi gerektiği durumlarda listeler kullanılır.
C dilinde Liste yerine Dizi (Array) kavramı vardır. Dizi içindeki elemanların hepsi aynı veri türündedir ve bellekte artarda saklanırlar. Python listelerinde aynı liste içinde farklı veri türlerinde elemanlar yer alabilir.
C dilindeki dizilerden diğer bir farkı ise, Python'da listeler aslında birer nesnedir (Python'da değişkenler ve sabitler bile nesnedir) ve bir elemanın eklenmesi, silinmesi, liste içinde kaç defa yer aldığının bulunması gibi yöntemler içerir.

LİSTE OLUŞTURMA

- liste = [1,"abc",3.2]
- liste = []
- liste = [i for i in range(10)]
- liste = list(range(10))

listeler farklı türlerde değişkenler içerebilirler.
Boş liste oluşturulabilir.
listelerin içerisinde range ile bir aralık verilerek eleman tanımlanabilir.
list() metodu ile de bir liste tanımlamak mümkündür.

LİSTELERDE İNDİS KULLANIMI

- Verilen indisteki elemana erişmek.
- liste[indis]
  - 0'dan başlar: liste[0]
  - Okunabilir: x = liste[1]
  - Değiştirilebilir liste[2] = 15

Listenin belirli bir elemanını görüntülemek için, elemanın listedeki sırası (indisini) liste isminden sonra köşeli parantez içinde verilebilir. Listelerin ilk elemanının indisi sıfırdır. Yani önceki slayttaki ilk tanım sonrası print(liste[2]) yazıldığında 5.67 görüntülenir.
Verilen sıra numarası listedeki eleman sayısına eşit yada daha fazla ise «list index out of range» hatası ortaya çıkar. Yani liste listesi 4 elemanlı olmasına rağmen son indis 3'tür. liste[4] elemanı yoktur
Listenin hangi elemanının değeri değiştirilecekse, yine köşeli parantez içinde elemanın indis değeri yazılmalıdır

---

ÖRNEK

- liste = [1,2,3,4,5,6,7,8,9,10]
- indis = 5
- print(liste[indis]) ?
- liste[indis] = liste[indis] + 1 ?
- liste[indis+1]+= 2 ?

SLICING

- Dilim
- Listenin belirli bir kısmını kesip almak
- Başlangıç:Bitiş:Artım
- Negatif indisler liste sonunu ifade eder.

Liste indisi olarak tek bir değer yerine bir dilim de verilebilir. Dilim belirli bir indis aralığıdır ve for döngüsünde kullanılan range fonksiyonunun argümanları ile benzer olarak [başlangıç indisi: bitiş indisi: artım değeri] şekilde verilir.
Artım değeri verilmezse 1 kabul edilir. Başlangıç indisi verilmezse listenin başından itibaren, bitiş indisi verilmez ise listenin sonuna kadar anlamına gelir.
Yine range fonksiyonuna benzer şekilde her 3 değer de negatif verilebilir.

ÖRNEK

- liste = [1,2,3,4,5,6,7,8,9,10]
- liste[1:3] ?
- liste[1:8:2] ?
- liste[4:0:-1] ?
- liste[:4] ?
- liste[7:] ?

LİSTEYE EKLEME

- append()
- extend()
- insert()

L.append(x): listenin sonuna x elemanını ekler.
L.extend(iterable): listenin sonuna bir iterable (eleman serisi) ekler.
+= ile de yukarıdaki 2 yöntemin yaptığı iş yapılabilmektedir:
liste.append(5) liste += [5]
liste.extend([5,6,7]) liste += [5,6,7]
L.insert(i, x): listenin i ile verilen indisine x ile verilen elemanı ekler. Sonraki elemanları birer kaydırır (indisleri artık 1 fazlası olur)
liste.insert(5, 'ali')

LİSTEDEN SİLME

- clear()
- remove()
- pop()

L.clear(): listedeki tüm elemanları siler.
L.remove(x): Değeri x olan elemanı listede arar ve ilk bulduğunu siler (diğerleri listede kalır). Eğer öyle bir eleman yoksa hata verir.
L.pop(i): i ile belirtilen indisteki elemanı listeden siler ve değerini döndürür. Eğer i verilmezse son elemanı siler ve değerini döndürür.

LİSTEDE ARAMA

- count()
- index()

L.count(x): x değerine eşit olan listede kaç eleman bulunduğunu döndürür.
L.index(x, start, end): x değerinin listede bulunduğu ilk indisi döndürür. Eleman listede yoksa hata döndürür. Eğer start verilirse o değerden sona kadar, end verilirse start ile end arasında arar.
start ve end verilse bile, start-end dilimindeki sırasını değil tüm listeye göre indis değerini döndürür

---

LİSTE YÖNTEMLERİ

- copy()
- reverse()
- sort()
  - Key
  - Reverse

L.copy(): listenin bir kopyasını döndürür.
L.reverse(): listenin elemanlarını ters sıraya çevirir.
L.sort(key=None, reverse=False): Listeyi nümerik veya alfabetik olarak sıralar. Seçimlik olan key= ile sıralama biçimi, reverse= ile ters sırada olup olmayacağı belirtilebilir.

LİSTE YÖNTEMLERİ

- len()
- max()
- min()
- sum()

len(a): a bir liste veya demet ise eleman sayısını, string ise karakter sayısını döndürür. Örnek: Aşağıdaki karşılaştırma True döndürür:
liste[len(liste)-1] == liste[-1]
max(a) ve min(a): a bir liste veya demet ise en büyük / küçük elemanı, string ise alfabetik sıraya göre en büyük / küçük olanı döndürür.
ASCII sırasına göre küçük harfler daha büyüktür
sum(a): a bir liste veya demet ise elemanlarının toplamını döndürür

LİSTE ÜZERİNDE DÖNGÜ KURMA

- for i in range(1,10):
  - liste[i]
- for i in range(0,len(liste)):
  - liste[i]
- for eleman in liste:
  - eleman
  - read-only

for döngüsünde range fonksiyonu ile çalışma aralığı vermek yerine; liste, demet veya string gibi bir iterable yapı verilerek her elemanı için döngü yaratılabilir:
a = "Bilgisayar Mühendisliği"
for i in a: print(i)
Bu kod a adındaki string'in bütün elemanlarını alt alta ekrana yazdırır.
Bu döngü şekli genellikle liste elemanlarının değerini değiştirmek için değil, toplamını almak veya hepsini ekrana yazdırmak gibi "read-only" işler için kullanılır.

LİSTE HİYERARŞİSİ

- Liste
  - [1,2,3]
- Liste listesi
  - [[1,2],[2,3],[3,4]]
- Matris
  - $\begin{bmatrix} 1 & 2 \\ 2 & 3 \\ 3 & 4 \end{bmatrix}$
- İç içe listeler
  - [[1,[2,3]],[2,5],[8],2]

Listeler iç içe tanımlanabilmektedirler. Bir liste kendi içerisinde birbirlerinden farklı elemanlar saklayabileceği için listenin bir alt elemanı bir sayıyken diğeri bir liste olabilir. Bu hiyerarşi istenildiği kadar aşağıya doğru ilerletilebilmektedir.

TUPLE

- Demet
- Sabit
- t = (1, 'Emir')
- t = (1, 'Emir', 2006)
- count()
- index()

Listeye benzer bir yapı olan demet, elemanlarının sabit yani değiştirilemez (immutable) olmasıyla ve tanımının köşeli parantezle değil normal parantezle yapılmasıyla farklıdır.
Demet = (34, 63, 'Hakan', 4.34)
A = 7,3 → 7 ve 3 elemanlarından oluşan demeti tanımlar
Demet tanımında tek bir eleman bile tanımlanacak olsa yanına virgül yazmak gereklidir (aksi halde bu bir değişken olur).
A = 7,
Demetlerde sadece count ve index yöntemleri vardır.

STRING

- Karakter katarı
- İndis kullanılabilir
- Okunabilir
- Eleman değiştirilemez
- + ve * işlemleri ile kullanılabilirler

Listeye benzer bir diğer yapı da string'tir. String aslında karakterlerden oluşan bir liste gibidir ve string'lerle de indis kullanımı mümkündür
String'ler de demetler gibi immutable bir yapıya sahiptir. Yani indis kullanımı ile bir elemanına değer atanamaz
Daha önce string'leri birleştirmek için + ve belirli sayıda tekrarlamak için * simgeleri kullanılmıştı. Bunlar listeler ve demetler ile de kullanılabilir.
String'ler listelere göre daha çok yönteme sahiptirler.

ÖRNEK

- liste = list('Emir')
- liste += "Ahmet"
- liste += [1,2]
- liste = liste[1:5]

UYGULAMALAR

- Kullanıcıdan girilen 10 değerin ortalamasını ve varyansını bulan uygulamayı yazınız.
- İki adet 3X3 kare matris toplamını bulan programı yazınız.
- Girilen bir string'in içerisinde verilen bir karakterden kaç tane olduğunu bulan uygulamayı yazınız.
- Üç adet 10 tamsayı içeren liste tanımlansın. 10 adet sayı girilen yeni bir listenin hangi listeye daha yakın olduğunu belirlenen bir yöntem ile tespit ediniz.