

Višedimenzionalne liste podataka

Lista je linearni tip podatka koji sadrži kolekciju elemenata istih ili različitih tipova koji su jedan do drugog smešteni u memoriji. U svakom programskom jeziku, liste ili nizovi su prikazani kao `ime_niza[indeks]`, gde je indeks broj koji nam govori koji element se nalazi na tom indeksu u listi. Kako su elementi smešteni jedan do drugoga u memoriji, tako su i brojevi indeksa isto kontinualni. Indeksiranje počinje od nule i završava se na $n-1$, gde je n veličina niza.

Koncept višedimenzionalnih lista se može objasniti kao tehnika definisanja i skladištenja podataka u višedimenzionalnom formatu. Višedimenzionalna lista se može implementirati pomoću dve ugneždene funkcije.

Liste u Pythonu se definišu na sledeći način: `lst = []`. Ako želimo da ih inicijalizujemo sa vrednostima, uradićemo to ovako: `lst = [value1, value2, value3]`. Kada želimo da dobavimo određeni element iz liste, koristimo indekse: `lista(indeks željene vrednosti počevši od nule)`. Najčešće korišćen tip višedimenzionalnih lista i tip lista na koje ćemo se fokusirati u ovoj nastavnoj jedinici jeste dvodimenzionalna lista, koji se definiše kao lista lista, ali se ovi principi mogu primeti i na liste sa „višim nivoima dubine“, ili sa više dimenzija. Dvodimenzionalne liste su idealne za predstavljanje matrica i tabela.

Dvodimenzionalne liste

Ilustracija jedne dvodimenzionalne liste se vidi na slici 12.1.

| | Column 1 | Column 2 | Column 3 | Column 4 | |
|-------------|----------------------------|----------------------------|----------------------------|----------------------------|---------------------------|
| Row indexes | 0 | 1 | 2 | 3 | Column indexes |
| Row 1 → 0 | <code>a[0][0]</code> 15 | <code>a[0][1]</code> 20 | <code>a[0][2]</code> 25 | <code>a[0][3]</code> 30 | ← Array <code>a[0]</code> |
| Row 2 → 1 | <code>a[1][0]</code> 20 | <code>a[1][1]</code> 30 | <code>a[1][2]</code> 40 | <code>a[1][3]</code> 50 | ← Array <code>a[1]</code> |
| Row 3 → 2 | <code>a[2][0]</code> 60 | <code>a[2][1]</code> 65 | <code>a[2][2]</code> 70 | <code>a[2][3]</code> 80 | ← Array <code>a[2]</code> |

Slika 12.1. Prikaz dvodimenzionalne liste¹

Na ovoj slici nam je grafički prikazan sastav jedne dvodimenzionalne liste. One najviše podsećaju na matrice u matematici. Dakle, imamo redove i kolone. Jedan red (prvi) je sastavljen od jedne liste čiji su elementi 15, 20, 25 i 30. Dvodimenzionalna lista sa ove ilustracije je u Pythonu definisana kao što je prikazano ispod:

¹ <https://refreshjava.com/images/java/multiDimensionalArray.png>

Definisanje dvodimenzionalne liste sa slike

Time naša lista izgleda ovako:

```
a = [[15, 20, 25, 30],  
      [20, 30, 40, 50],  
      [60, 65, 70, 80] ]
```

Objašnjenje:

Prvi red odn. lista [15, 20, 25, 30] će imati indeks 0, drugi red 1, a treći 2. Ako bismo želeli da pristupimo prvom redu odn. listi [15, 20, 25, 30] pisali bismo a[0].

Definišimo prvo jednu jednostavniju listu koja će sadržati studente iz jedne grupe: `group1 = ['student1', 'student2', 'student3']`.

Ako bismo imali više grupa studenata, mogli bismo ovako da ih definišemo:

Primer više lista

```
group1 = ['student1', 'student2', 'student3']  
group2 = ['student4', 'student5', 'student6']  
group3 = ['student7', 'student8', 'student9']  
group4 = ['student10', 'student11', 'student12']
```

Sada imamo četiri različite liste u kojima skladištimo podatke o studentima po datim grupama. Možda ovo deluje kao ispravan način, ali postavlja se pitanje – šta ako imamo više od četiri grupe, sto grupa, hiljadu grupa? Zato ovo nije dovoljno efikasan način za rešavanje ovog problema. U ovakvim situacijama nam pomažu višedimenzionalne liste. Na taj način, sve grupe studenata, nevezano za broj grupa, možemo smestiti u jednu listu.

Primer smeštanja više grupa u jednu listu

```
groups = [['student1', 'student2', 'student3'],  
           ['student4', 'student5', 'student6'],  
           ['student7', 'student8', 'student9'],  
           ['student10', 'student11', 'student12']]
```

Ovako smo definisali dvodimenzionalnu listu elemenata. Dakle, reč je o listi čiji su elementi pojedinačne liste koje nazivamo i ugneždene liste. Elementi u dvodimenzionalnoj listi se takođe odvajaju zarezom kao i u jednodimenzionalnoj listi. Za pristup elementu (jednoj od grupa studenata u ovom slučaju) koristimo indeksiranje kao i kod jednodimenzionalne liste i to naredbom `groups[1]`, čime dobijamo drugi element u listi: `['student4', 'student5', 'student6']`.

Generalno se za pristup ugnežđenim listama koriste dva indeksa. Prvi indeks je za definisanje elementa (unutrašnje liste), a drugi indeks je tu radi dobavljanja samog elementa unutar te unutrašnje liste. Na primer, ako želimo da dobavimo devetog studenta iz dvodimenzionalne liste „grupe”, učinimo to sledećom naredbom:

```
groups[2][2]
```

Prvi indeks koji smo zadali nam dobavlja element `['student7', 'student8', 'student9']`, a drugi indeks nam dobavlja upravo element iz te unutrašnje liste `'student9'`.

Dodavanje i menjanje elemenata u dvodimenzionalnoj listi

Nakon što smo naučili kako da kreiramo dvodimenzionalne liste, bitno nam je da znamo i kako da ih menjamo. Ove metode ćemo izvršiti nad listom iz prethodnog primera.

Da bismo dodali element, moramo prvo razmisliti u kom nivou dubine dodajemo, da li želimo da dodamo grupu studenta ili da dodamo jednog studenta grupi. U oba slučaja koristimo metodu `append()`.

Primer

```
groups.append(['student13', 'student14', 'student15'])
```

Time naša lista izgleda ovako:

```
[['student1', 'student2', 'student3'],  
 ['student4', 'student5', 'student6'],  
 ['student7', 'student8', 'student9'],  
 ['student10', 'student11', 'student12'],  
 ['student13', 'student14', 'student15']]
```

Ovim smo naknadnu podlistu grupe studenata dodali na kraj liste. Kako smo u prethodnim lekcijama naučili da liste podržavaju i metodu `insert(<[index]>,element)` koja nam omogućava da preciziramo indeks gde želimo da ubacimo element, u sledećem primeru smo ubacili podlistu na treće mesto (indeks 2):

Primer

```
groups.insert(2,['student13', 'student14', 'student15'])
```

Sada naša lista izgleda ovako:

```
[['student1', 'student2', 'student3'],  
 ['student4', 'student5', 'student6'],  
 ['student13', 'student14', 'student15'],  
 ['student7', 'student8', 'student9'],  
 ['student10', 'student11', 'student12']]
```

Ako želimo da izmenimo postojeći element (ili pod-element), moramo proslediti njegov indeks, pa tako, ako želimo da student12 pretvorimo u student13, koristićemo sledeću notaciju (koristeći prvobitni primer):

Primer

```
groups[3][2] = 'student13'
```

I tako dobijamo sledeću dvodimenzionalnu listu:

```
[['student1', 'student2', 'student3'],  
 ['student4', 'student5', 'student6'],  
 ['student7', 'student8', 'student9'],  
 ['student10', 'student11', 'student13']]
```

Nakon analize ispisa vidimo da se poslednji element poslednje unežđene liste promenio. Na sličan način možemo promeniti i čitavu ugneždenu listu, prosleđujući samo jedan indeks, i to:

Primer

```
groups[3] = ['student13', 'student14', 'student15']
```

Sada naša dvodimenzionalna lista izgleda ovako:

```
[['student1', 'student2', 'student3'],  
 ['student4', 'student5', 'student6'],  
 ['student7', 'student8', 'student9'],  
 ['student13', 'student14', 'student15']]
```

Brisanje elementa iz dvodimenzionalne liste

Nakon dodavanja novih i menjanja postojećih elemenata, potrebno nam je i da znamo kako da obrišemo neželjeni element. To radimo na dva načina, ključnom rečju 'del' i primenom metode `pop()`. Na oba načina možemo brisati i ugneždene liste, kao i pojedinačne elemente ugnežđenih lista.

Primer

Naredbom `del groups[2]` brišemo treći element liste grupa, pa tako naša lista ostaje sa tri elementa (tri ugneždene liste umesto prvobitnih četiri). Kako izgleda možete pogledati u radnom okruženju.

Radno okruženje

```
groups = [['student1', 'student2', 'student3'],  
          ['student4', 'student5', 'student6'],  
          ['student7', 'student8', 'student9'],
```

```
['student10', 'student11', 'student12']

del groups[2]

print(groups)
```

Dobićemo listu `groups` bez elementa na poziciji 2 odn. podliste `['student7', 'student8', 'student9']`.

Ako tačno znamo koji element ugneždene liste želimo da obrišemo, uradićemo to prosleđivanjem i drugog indeksa: `del groups[2][0]`. U ovom slučaju izbacujemo element na poziciji 0 podliste koja se nalazi na poziciji 2. odn. u listi `['student7', 'student8', 'student9']`, izbacujemo element `'student7'`.

Istu funkcionalnost dobijamo pri korišćenju metode `.pop()` i to:

- `groups.pop(2)` za prvi slučaj, gde brišemo čitavu treću ugneždenu listu i
- `groups[2].pop(0)` za drugi slučaj, gde brišemo prvi element treće ugneždene liste.

Pitanje

Naredba `groups.sort()` nad našom dvodimenzionalnom listom će imati za rezultat:

- `SyntaxError`
- **sortiranje**
- `AttributeError`

Objašnjenje:

Tačan odgovor je da će se sortiranje ipak izvršiti. Metoda `.sort()` kao i funkcija `sorted()` mogu sortirati i liste sa više dimenzija, ali se za `'key'` parametar podrazumeva prvi element. Dakle, sortiranje će se izvršiti po prvom elementu svake ugneždene liste, pa će tako naša dvodimenzionalna lista izgledati ovako:

```
['student1', 'student2', 'student3'],
['student10', 'student11', 'student13'],
['student4', 'student5', 'student6'],
['student7', 'student8', 'student9']]
```

Ako želimo drugačiji ključ (kriterijum) sortiranja, moramo definisati sopstvenu funkciju i zadati je kao `'key'` parametar.

Iteracija kroz dvodimenzionalnu listu

Kako smo za iteraciju kroz jednodimenzionalnu listu koristili jednu `for` petlju, za iteraciju kroz dvodimenzionalne liste, ako želimo da prođemo i kroz svaki element ugneždene liste, moramo koristiti dve. Pravilo je da nam je potrebno onoliko `for` petlji koliko dimenzija lista ima. Iteraciju ćemo pokazati koristeći prvobitni primer grupa studenata.

| Primer |
|---|
| Kod |
| <pre>groups = [['student1', 'student2', 'student3'], ['student4', 'student5', 'student6'], ['student7', 'student8', 'student9'], ['student10', 'student11', 'student12']] for group in groups: for student in group: print("Group: {}, {}".format(groups.index(group), student), end = ' ', ' ') print('')</pre> |

Tabela 12.1. Iteracija kroz dvodimenzionalnu listu

Upišite kod u radno okruženje i proverite rezultat:

Radno okruženje

Takođe je moguće i koristiti jednu `for` petlju, ali tako nećemo dobiti svakog studenta posebno, nego ćemo iterirati po grupama:

| Primer |
|---|
| Kod |
| <pre>groups = [['student1', 'student2', 'student3'], ['student4', 'student5', 'student6'], ['student7', 'student8', 'student9'], ['student10', 'student11', 'student12']] for group in groups: print('Nested list: {}, group {}'.format(groups.index(group), group))</pre> |
| Rezultat |
| <pre>Nested list: 0, group ['student1', 'student2', 'student3'] Nested list: 1, group ['student4', 'student5', 'student6'] Nested list: 2, group ['student7', 'student8', 'student9'] Nested list: 3, group ['student10', 'student11', 'student12']</pre> |

Tabela 12.2. Primer for petlje za iteraciju po grupama

Rezime

- Ako je element liste sama lista, onda je reč o višedimenzionalnoj listi.
- Sve metode koje podržavaju obične liste u Pythonu koje smo obrađivali do sada mogu se primeniti na dvodimenzionalne liste, kao i na liste sa više dimenzija.
- Pored višedimenzionalnih lista, možemo koristiti i višedimenzionalne n-torke.



linkgroup