Скупови, торке, речници

Како смо већ говорили на неком од ранијих часова, Python нуди могућност да одређену колекцију података који су међу собом према неком смислу повезани сместиш по одређеном реду у листу. Сваком податку у листи се на основу позиције, тј. индекса може приступити. Поред листа, постоје још неке врсте такозваних структура у које се бележити колекције података. Тема овог часа су те додатне структуре и то: скупови, торке и речници.

За почетак одгледај следећу видео-лекцију

Python – скупови, торке, речници

Скупови

Са скуповима си се већ упознао у математици. Њих можеш направити и користи и у Python-у. Скуп се дефинише набрајањем елемената скупа између витичастих заграда, тј. { и }. На пример, скуп фудбалских стрелаца на неком турниру се може дефинисати овако:

```
strelci = {"Mesi", "Ibrahimović", "Nejmar", "Ronaldo"}
```

Ако би скуп био дефинисан овако

```
strelci = {"Mesi", "Ronaldo", "Mesi", "Ibrahimović",
"Ibrahimović", "Nejmar", "Nejmar"}
```

резулатат команде којом се исписују елементи скупа

```
print(strelci)
```

би у оба случаја био исти, јер се сваки елемент исписује једанпут. Дакле реултат би био,

```
{'Mesi', 'Ibrahimović', 'Nejmar', 'Ronaldo'}
```

Скуп је могуће формирати и од елемената већ дефинисане листе употребом функције **set**.

```
golovi = ["Mesi", "Ronaldo", "Mesi", "Ibrahimović",
"Ibrahimović", "Nejmar", "Nejmar"]
strelci = set(golovi)
```

Функцијом `set` можеш и од ниске добити скуп карактера. То може бити веома корисно у појединим задацима. На пример,

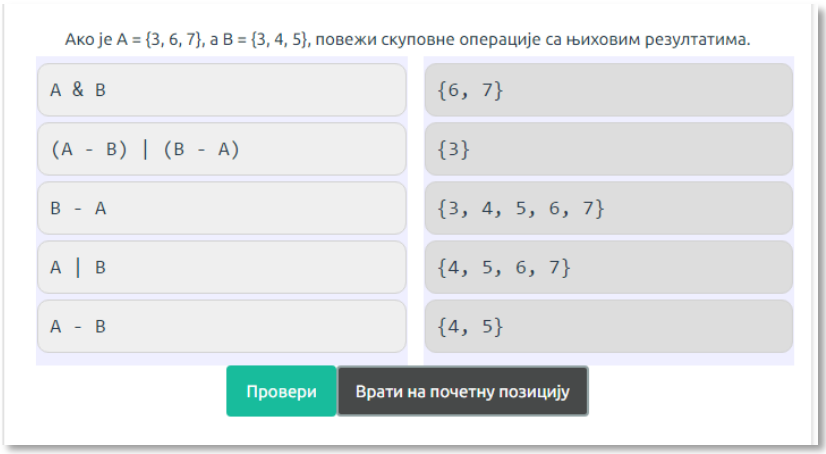
Резултат команде ће бити следећи:

```
{'l', 'e', 'P', 'p', 't', 'k', 'o', 'a'}
```

У језику Python можеш веома једноставно израчунати **унију**, **пресек** и **разлику скупова**. Ако су A и B скупови, онда је

- **A | B** њихова **унија**,
- **A & B** њихов **пресек**,
- **A - B** њихова **разлика**.

Пронађи у интерактивном уџбенику следећи задатак и реши га



Задатак 1.
Један скуп садржи девојчице из одељења које тренирају ритмичку гимнастику, а други оне које тренирају одбојку. Одреди скуп девојчица које тренирају оба спорта, скуп девојчица које тренирају бар један од њих и скуп девојчица које тренирају само одбојку.

Предлог решења

```
ritmicka = {"Ana", "Milica", "Jovana", "Gordana"}
odbojka = {"Tara", "Nađa", "Milica", "Jovana", "Aleksandra"}
dva_sporta = ritmicka & odbojka
bar_jedan_sport = ritmicka | odbojka
samo_odbojka = odbojka - ritmicka
print(dva_sporta)
print(bar_jedan_sport)
print(samo_odbojka)
```

Торке

Некада је за описивање неке појаве, догађаја, особе потребно навести неколико података. На пример, позиција фигуре на шаховској табли се може прецизно описати само ако имамо и ознаку врсте (слова од a до h) и ознаку колоне (броја од 1 до 8). Слично, позиције на географској карти се описују помоћу географске ширине и дужине тј. помоћу пара реалних бројева. Тако се град Париз налази на позицији која се може описати помоћу пара 48.8566 и 2.3522.

За записивање овакве колекције података, колекције у којој позиција елемента одређује њено значење (као што у ознаци позиције фигуре a7, a представља врсту, а 8 колону) у Python-у се користи **торка** структура. Торка се дефинише набрајањем елемената скупа између малих заграда, тј. (и). На пример, позиција Париза би била дефинисана овако

```
Pariz = (48.8566, 2.3522)
```

Када торка има само два елемента назива се **паром**.

Индексирање елемената у торкама врши се на исти начин као индексирање листа (на позицији 0 налази се први елемент торке, на позицји 1 други и тако даље). Ево и примера

```
Pariz = (48.8566, 2.3522)
sirina = Pariz[0]
duzina = Pariz[1]
print("Географска ширина:", sirina)
print("Географска дужина:", duzina)

```

Торке могу да буду елементи листе. Ево једне

```
meseci = [("јануар", 31), ("фебруар", 28), ("март", 31), ("април", 30),
      ("мај", 31), ("јун", 30), ("јул", 31), ("август", 31), ("септембар",
      30), ("октобар", 31), ("новембар", 30), ("децембар", 31)]
```

која представља списак месеци и боја дана у сваком од њих за годину која није преступна.

Задатак 2.
Напиши програм који за унети редни број месеца исписује његов назив и број дана у години која није преступна.

Овакву листу можеш искористити за решавање следећег задатка.

За почетак из листе мораш издвојити торку која се односи на одговарајући месец. Тек када имаш торку одговарајућег месеца, тј. један елемент из листе издвојен, тада одвајаш поједине податке из торке и то, као што си већ видео, преко индекса.

Предлог решења

share save run

```
meseci = [("јануар", 31), ("фебруар", 28), ("март", 31), ("април", 30), ("мај", 31), ("јун", 30), \
          ("јул", 31), ("август", 31), ("септембар", 30), ("октобар", 31), ("новембар", 30),
           ("децембар", 31)]
broj = int(input("Унеси редни број месеца:"))
mesec = meseci[broj - 1]
print("Назив:", mesec[0], "Број дана:", mesec[1])

```

да напишеш програм који за унети редни број месеца исписује његов назив и број дана.

Речници

Листе ти дају могућност да организујеш податке за које је познат неки редослед и у којима сваки податак има свој редни број (на пример, листу смо употребили да организујемо имена путника у авиону или податке о месецима у години) и приступ елементима листе врши се на основу њиховог индекса (редног броја, тј. позиције). Међутим, често се подаци организују тако да су прилагођени специјалној врсти претраге, као што је претрага у којој на основу датог кључа желимо да приступимо **вредности** која је придружена том **кључу**. На пример, аутомобили у каталогу имају придружене цене и ми желимо да у нашем програму можемо да одредимо цену аутомобила на основу његовог модела (то значи да модел представља кључ по којем вршимо претрагу). У тим ситуацијама је добро искористити другу врсту структуре коју нуди Python а то је **речник** (често се назива и **мапа** или **асоцијативни низ**). Ево примера једног речника

```
cene_automobila = {"fiat 500l": 11990,
                   "renault clio": 9650,
                   "toyota corolla": 13990}
```

Као што видиш, дефиниција речника личи на скуп чији су елементи парови кључева и вредности које су им придружене, облика

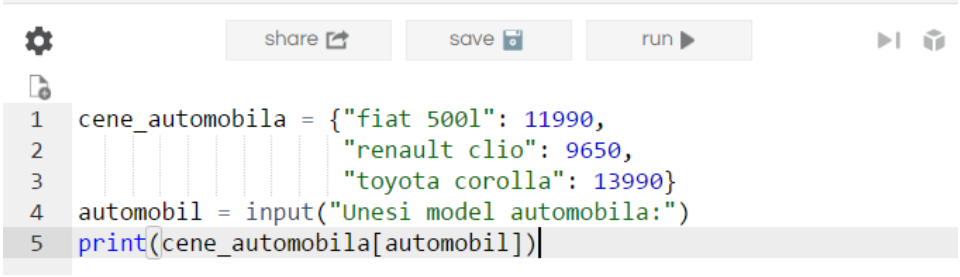
кључ : вредност

што су у претходном примеру марка и цена аутомобила. Приступ вредности са задатом вредношћу кључа, нпр. `"renault clio"`, на датом примеру изгледа овако:

```
cena = cene_automobila["renault clio"]
```

Задатак 3.
Напиши програм који за унети назив марке аутомобила исписује његову цену.

Предлог решења



Задатак 4.
Познате су географске координате неколико главних европских градова. За дато име града одреди њене географске координате. Одреди посебно географску дужину и посебно географску ширину.

Вредности у речнику могу бити и торке, као што ће бити у наредном задатку.

Предлог решења

share save run

```
1  gradovi = {"Beograd": (44.7866, 20.4489),
2             "Budimpešta": (47.4979, 19.0402),
3             "Beč": (48.2082, 16.3738),
4             "Bratislava": (48.1486, 17.1077)}
5  grad = input("Unesi ime grada: ")
6  koordinate = gradovi[grad]
7  print(koordinate)
8  print("Geografska širina: ", koordinate[0])
9  print("Geografska dužina: ", koordinate[1])
```