24. час: **Ниске (обрада)**

Поред бројева, рачунари су веома добри и у раду са текстом. Текст се састоји од слова (малих и великих), цифара, размака, интерпункцијских знакова (на пример тачака, зареза, упитника, узвичника) и слично. Све те знакове једним именом називамо **карактери**.

Неки програмски језици подржавају само веома узак скуп карактера (од слова је могуће користити само слова енглеске абецеде), међутим, програмски језик Python3 користи широк скуп карактера који обухвата и све карактере потребне за писање на већини језика света, укључујући и слова ћириличког и латиничког писма која се користе у српском језику.

Поменути основни скуп карактера довољан само за запис текста на енглеском језику назива се **ASCII**, док се овај шири скуп карактера назива **Unicode**.

Низ карактера чини ниску или стринг (од енглеске речи string која значи ниска). Ниске се у програму записују између наводника. На пример, ниске су "Zdravo" или "Programski jezik Python.". Уместо двоструких равноправно се могу користити и једноструки наводници (на пример, 'Zdravo'), међутим, да се не би збуњивао користићемо двоструке наводнике.

Тема овог часа су управо ниске и рад са нискама у програмском језику Python. За почетак одгледај следећу видео-лекцију

Python – ниске

У програмском окружењу Python испиши и покрени команду којом се исписује поздравна реченица Здраво, свете!. Команда би могла да изгледа овако:

```
print("Zdravo, svete!")
```

Има ли цифара у ниски?

Ниске у себи могу да садрже цифре, па чак могу да буду састављене искључиво од цифара. Међутим, када се на две ниске примени оператор + ниске се надовезују. Тако је резултат операције "12" + "34" једнак "1234", а не 46 како би неки очекивали. Помоћу input уноси се увек текст тј. резултат ове операције је увек ниска, чак иако тај текст садржи само цифре. Имајући ово у виду покушај да предвидиш шта ће израчунати наредни програм, када након његовог покретања корисник унесе 3, а затим и 5.

```
1 prvi_sabirak  = input("Unesi prvi sabirak: ")
2 drugi_sabirak = input("Unesi drugi sabirak: ")
3 zbir = prvi_sabirak + drugi_sabirak
4 print(zbir)
5
```

Ако текст садржи само цифре, онда се број представљен тим цифрама може добити помоћу `int`. На пример, `int("123")` је број 123. Тако је `int("12") + int("34")` једнако 12 + 34 тј. 46. Стога се учитавање броја може постићи помоћу `int(input("Unesi broj: "))`

Имајући наведену функцију у виду програм који сабира два учитана броја може поправити на следећи начин:

```
1 prvi_sabirak  = int(input("Unesi prvi sabirak: "))
2 drugi_sabirak = int(input("Unesi drugi sabirak: "))
3 zbir = prvi_sabirak + drugi_sabirak
4 print(zbir)
5
```

Операције и уграђене функције за рад са нискама

Слично као и над бројевима и над нискама се могу вршити одређене операције. Једна од основних операција је **спајање две ниске**. Ова операција донекле подсећа на сабирање и обележава се знаком **+**. На пример,

Задатак 1.
Напиши програм у којем се од ниске која представља име и ниске која представља презиме једне особе формира и исписује ниска која саржи име и презиме те особе.

вредност израза `"abraka" + "dabra"` је `"abrakadabra"`.

```
ime = "Petar"
prezime = "Kralj"
ime_i_prezime = ime + prezime
print(ime_i_prezime)
```

Ако желиш да име и презиме имају размак између додај размак испред презимена.

```
prezime = " Kralj"
```

Ако желиш да име и презиме имају размак између додај размак испред презимена.

```
prezime = " Kralj"
```

Размак у тексту се, такође, сматра карактером, па се ниска која садржи један размак записује овако “ ”. Тако је претходно решење кориговати тако да се не прави промена у ниски која бележи презиме већ да се размак уметне као додатне ниска између имена и презимена.

```
ime = "Petar"
prezime = "Kralj"
ime_i_prezime = ime + " " + prezime
print(ime_i_prezime)
```

Још једна интересантна операција је да се ниска **помножи природним бројем**. Слично као што код бројева множење представља узастопно сабирање, исти је случај и овде и може се наслутити да множење ниске бројем заправо представља њено понављање одређени број пута. На пример,

вредност израза `"ba"*2` је `"baba"`.

Дужина ниске, издвајање делова ниске

За рад са нискама Python нуди више уграђених функција. Једна од њих је `len`, функција којом се одређује број карактера ниске, тј. дужина ниске. Дужину ниске тј. број њених карактера можемо добити помоћу функције len. Тако је

`len("Zdravo")` једнако 6,

јер ниска `"Zdravo"` има тачно 6 карактера. Своје разумевање ове функције провери тако што ћеш одговорити на питања у интерактивном уџбенику која се налазе у секцији Дужина ниске, издвајање делова ниске.

Карактери у ниски имају своје редне бројеве тј. **позиције**. Први карактер се налази на позицији 0, други на позицији 1 и тако даље. На пример, карактери у ниски `"Zdravo svima!"` се броје на следећи начин.

0	1	2	3	4	5	6	7	8	9	10	11	12
Z	d	r	a	v	o		s	v	i	m	a	!

Могуће је **издвојити појединачни карактер из ниске**. На пример, ако је

`ime = "Zorana"`

тада се

карактер `Z` може добити изразом `ime[0]`,

а карактер `r` изразом `ime[2]`.

Подржани су и негативни индекси тако што -1 означава последњи карактер, -2 претпоследњи и тако даље.

-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1
Z	d	r	a	v	o		s	v	i	m	a	!

На пример, ако је

`ime = "Zorana"`

тада је

`ime[-1]` карактер `a`

док је `ime[-3]` карактер `r`.

Још једна операција која је често корисна је **издвајање дела ниске**. Приликом издвајања може се навести распон позиција, при чему се издвајају карактери из тог распона рачунајући прву, а не рачунајући другу наведену позицију. На пример, ime[2:5] издваја карактере имена на позицијама 2, 3 и 4 (распон [2:5] је полуотворен тј. позиција 2 је урачуната, а позиција 5 није). Ако је

```
ime = "Predrag"
```

тада је

`ime[2:5]` једнако `"edr"`.

Ако се горња граница не наведе, тада се издваја део ниске до њеног краја. Тако је

`ime[3:]` једнако `"drag"`.

Претрага ниске

Често је потребно да проверимо да ли једна ниска садржи неки карактер или садржи неку другу ниску. То можеш урадити употребом функције **`find`**. На пример, нека је

```
Rec = “ponedeljak”
```

Тада је

`Rec.find(“d”)` једнако 4

`Rec.find(“c”)` једнако -1, јер траженог карактера нема у стрингу

`Rec.find(“e”)` једнако 3, јер функција враћа позицију прве појаве `траженог карактера или тражене ниске`

`Rec.find(“P”)` једнако -1, јер се велика мала слова сматрају различитим карактерима

Задатак 2.
Напиши програм у којем се на основу дате ниске која садржи име и презиме добијају ниска са именом и ниска с презименом те особе.

Да би решио проблем, за почетак одреди позицију размака, који у датој нисци раздваја име од презимена. Затим на осбноу те позиције издвој делове полазног стринга и то део до дате позиције за име и део који садржи карактере од првог наког размака па до краја стринга.

share save run

```
1  ime_i_prezime = "Љубица Љубичић"
2  razmak = ime_i_prezime.find(" ")
3  ime = ime_i_prezime[0:razmak]
4  prezime = ime_i_prezime[razmak+1:]
5  print("Име: ", ime)
6  print("Презиме: ", prezime)
7
```

ЗАДАТАК ЗА ДРУГУ НЕДЕЉУ УЧЕЊА НА ДАЉИНУ

До тачних одговора на ПИТАЊА лакше ћеш доћи ако користиш:

- **Дигитални уџбеник - Инфораматика и рачунарство за 6. разред Нови Логос**
- **Материјал са: Petlja.org**
  https://petlja.org/biblioteka/r/lekcije/prirucnik-python/strukturepodataka-cas13#id49

Одговоре пошаљи предметном наставнику на проверу електронском поштом на (E- mail) адресу:

dragisa.bojanic@gmail.com или путем Viber –а на 060/332 58 72

поштујући правила комуникације преко интернета, и обавезно се потпиши пуним именом и презименом наводећи и одељење, како би наставник знао о ком ученику је реч.

Желим ти пуно успеха у раду !!!

ПИТАЊА

1. Који од наредних израза крију у себи исправно записан назив воћа?

- [ ] A. "ba" + "na" * 2
- [ ] B. "a" + "na" * 2 + "s"
- [ ] C. "ba" * 2 + "na"
- [ ] D. "an" * 2 + "nas"

2. Који је резултат извршавања претходног програма ако корисник унесе прво 3, а затим 5.

- ○ A. 8
- ○ B. 15
- ○ C. "35"
- ○ D. "8"

3.

Ниска може да садржи и децимални запис неког броја и тада се број представљен том ниском може добити помоћу `float`. На пример, `float("123.45")` је број `123.45`. Претварање ниске у број је и у овом случају веома важно урадити, јер се у супротном оператор `+` односи на надовезивање ниски, а не на сабирање бројева. Провери да ли ово добро разумеш.

ПОВЕЖИ

`3.5 + "3.5"`	`7.0`
`float("2.5") + 1.5`	`4.0`
`float("3.5") + float("3.5")`	`greška`
`"3.5" + "0.5"`	`"3.50.5"`

4.

Вредност `len("Popokatepetl")` је

5.

Вредност `len("Супер Марио 3!")` је

6.

Као и у другим торкама и листама и карактери у ниски имају своје редне бројеве тј. позиције. Први карактер се налази на позицији 0, други на позицији 1 и тако даље. Могуће је издвојити појединачни карактер из ниске. На пример, ако је `ime = "Zorana"` тада се карактер `Z` може добити изразом `ime[0]`, а карактер `r` изразом `ime[2]`.
Подржани су и негативни индекси тако што -1 означава последњи карактер, -2 претпоследњи и тако даље. На пример, ако је `ime = "Zorana"` тада је `ime[-1]` карактер `a` док је `ime[-4]` карактер `r`.
И издвајање дела ниске (подниске) функционише на исти начин као и код других торки тј. листа. Подсети се овога.

Дата је ниска `s = "Programiranje je mnogo zabavno"`. Вредност израза `s[0:4]` је

Израз којим је из ниске `s` могуће издвојити реч `mnogo` је

7. Шта се добија као резултат извршења следећег програма

```
ime_i_prezime = "Љубица Љубичић"
razmak = ime_i_prezime.find(" ")
ime = ime_i_prezime[0:razmak]
prezime = ime_i_prezime[razmak+1:]
print("Име: ", ime)
print("Презиме: ", prezime)

```

8. Заврши наредни програм тако да резултат буде исти као у задатку 7.

Овај задатак има и лепше решење које користи функцију `split`, која формира листу речи дате реченице (врши поделу ниске на подниске на основу размака који се јављају унутар ниске).

```
ime_i_prezime = "Љубица Љубичић"
(ime, prezime) = ime_i_prezime.split()
print(
print(
```