

1. Uvod u programski jezik Python

1.1 Cilj Vježbe

Upoznati se s programskim jezikom Python i Visual Studio Code IDE.

1.2 Teorijska pozadina

1.2.1 Python programski jezik

Programski jezik Python je popularni jezik visoke razine opće namjene. Python interpreter dostupan je za instalaciju na različitim operativnim sustavima. Trenutno aktualna verzija Python-a je 3.9. U ovom predlošku primjeri su napisani za verziju 3.9.15. Python je objektno orijentirani programski jezik. Dani teorijski opis daje samo kratki pregled Python programskog jezika.

Python skripte

Programiranje u programskom jeziku Python svodi se na pisanje tekstualnih datoteka koje sadrže odgovarajuće programske naredbe, a datoteka ima ekstenziju .py. Izvršavanje skripte moguće je pokrenuti iz komandnog prozora na način:

```
python ime_skripte.py
```

Rijetko se radi izravno pokretanje skripti iz komandnog prozora već se koriste naredbe iz IDE-a.

Python varijable, tipovi podataka i operatori

Varijabla je spremnik za određene podatke. Tip podataka specificira kakav se podatak pohranjuje u varijablu. Ime varijable je proizvoljno pri čemu mogu sadržavati, brojke, slova i underscore (_), ali ne mogu započeti sa brojkom. Osim toga Pythonove ključne riječi (kao npr. class) ne mogu se koristiti kao imena varijabli. Budući da je Python objektno orijentirani programski jezik, tipovi podataka su odgovarajuće klase, a varijable su instance ovih klasa. Dostupni tipovi podataka u Pythonu dani su u tablici 1.1.

Tip podatka	Klase	Primjer
Numerički	int, float, complex	var = 5 var = 5.2 var = 5+3j
String	str	var = "Hello world"
Sekvence	list, tuple, range	var = [2, 3, 5] var = (2, 3, 5) var = range(6)
Mapiranje	dict	var = { 'a':2,'b':3,'c':10}
Logički	bool	var = True
Set	set, frozenset	var = {"app", "bcw", "cfe"} var = frozenset({"ja", "ti", "on"})
None	NoneType	var = None

Tablica 1.1: Tipovi podataka u Pythonu

Tip operatora	Primjer
Aritmetički operatori	+, -, /, *
Operatori pridruživanja	=, +=, -=
Operatori usporedbe	==, !=, >, <
Logički operatori	and, or, not
Logički	var = True
Bitovni operatori	&, , ~, ^
Specijalni operatori	is, is not, in, not in

Tablica 1.2: Tipovi operatora u Pythonu

Glavni tipovi operatora u Pythonu dani su u tablici 1.2. Za svaki tip operatora dano je i nekoliko primjera. Jedan od osnovnih operatora je operator pridruživanja = koji dodijeljuje vrijednosti varijabli. Na raspolaganju su i drugi operatori (za zbrajanje, množenje, i sl.). Python podržava i standardne operatore usporedbe (jednako, različito, veće, manje, ...) čije izvršavanje daje vrijednosti True ili False. U primjeru 1.1 prikazano je instanciranje varijable koja pohranjuje cijelobrojnu vrijednost te se ona ispisuje na ekran. U primjeru 1.2 dan je primjer dodjeljivanje logičke vrijednosti varijabli. Primijetite kako se pišu komentari u Python – koristi se znak #.

■ Primjer 1.1 Upotreba aritmetičkog operatora

```
x = 23
print(x)
x = x + 7
print(x) # komentar: ispis varijable na ekranu
```

■ Primjer 1.2 Upotreba operatora usporedbe

```
x = 23
y = x > 10
print(y)
```

Kontrola toka programa

U Pythonu je na raspolaganju uvjetni izraz `if...else` koji služi za odlučivanje odnosno grananje programa ovisno o definiranom logičkom uvjetu. U primjeru 1.3. prikazano je grananje programa na temelju logičkog uvjeta $x < 10$. Primijetite kako se u Python blok koda definira uvlakom (tipično znak **TAB**). U Pythonu su na raspolaganju i iterativne naredbe `for` i `while` koje omogućuju višestruko uzastopno izvršavanje programskog koda odnosno petlje kao što prikazuje primjer 1.4. Ključna riječ `break` se koristi za prekidanje petlje (tipično kada je ispunjen neki logički uvjet). Ključna riječ `continue` se koristi za preskakanje trenutne iteracije petlje.

■ Primjer 1.3 Grananje if-else

```
x = 23
if x < 10:
    print("x je manji od 10")
else:
    print("x je veci ili jednak od 10")
```

■ Primjer 1.4 Petlje while i for

```
i = 5
while i > 0:
    print(i)
    i = i - 1
print("Petlja gotova")

for i in range(0,5):
    print(i)
```

Python liste

Liste (engl. *list*) u Pythonu je kolekcija ili niz podataka koji mogu biti različitog tipa. Definira se pomoću uglatih zagrada. Pristupanje pojedinom elementu liste provodi se pomoću indeksa, pri čem i indeksiranje elemenata liste kreće od 0, npr. `lst[5]` izdvojiti će šesti element liste `lst`. Izdvajanje više elemenata liste moguće je provesti na način da se definira početni indeks (`start`), krajnji indeks (`end`) i korak (`step`) na sljedeći način `lst[start:end:step]`. Ako se ne definira `start`, tada izdvajanje kreće od prvog elementa liste. Ako se ne definira `end`, tada izdvajanje ide do zadnjeg elementa lista (ali se on ne uključuje!). Ako se korak `step` ne definira, tada se podrazumijeva izdvajanje svakog elementa između `start` i `end`. U primjeru 1.5 moguće je vidjeti primjer definiranje liste te izdvajanje elemenata liste. Dodavanje elemenata na listu se izvodi pomoću metode `.append()`, dok se uklanjanje zadnjeg elemenata liste izvodi pomoću `.pop()`. U drugom primjeru 1.5 moguće je vidjeti neke od dostupnih operacija na listama te princip iteriranja kroz listu.

■ Primjer 1.5 Liste

```
lstEmpty = []
```

```
lstFriend = ['Marko', 'Luka', 'Pero']

lstFriend.append('Ivan')

print(lstFriend[0])

print(lstFriend[0:1:2])
print(lstFriend[:2])
print(lstFriend[1:])
print(lstFriend[1:3])
```

```
a = [1, 2, 3]
b = [4, 5, 6]
c = a + b
print(c)
print(max(c))

c[0] = 7
c.pop()
for number in c:
    print('List number ', number)
print('Done!')
```

Stringovi

String je niz znakova i u Pythonu se definira pomoću jednostrukih ili dvostrukih navodnika, npr. "Hello world!". Za razliku od listi, stringovi se ne mogu mijenjati (engl. *immutable*). Stringovi se indeksiraju i njegovi dijelovi izdvajaju na isti način kao i liste, pomoću dvotočke kao što je demonstrirano u primjeru 1.6. Postoje gotove funkcije za rad sa stringovima kao što prikazuje drugi primjer 1.6.

■ Primjer 1.6 Stringovi

```
fruit = 'banana'
index = 0
count = 0

while index < len(fruit):
    letter = fruit[index]
    if letter == 'a':
        count = count + 1

    print(letter)
    index = index + 1

print(count)

print(fruit[0:3])
print(fruit[0:])
print(fruit[2:6:1])
print(fruit[0:-1])
```

```

line = 'Dobrodosli u nas grad'

if(line.startswith('Dobrodosli')):
    print('Prva rijec je Dobrodosli')
elif(line.startswith('dobrodosli')):
    print('Prva rijec je dobrodosli')

line.lower()
print(line)

data = 'From: pero@yahoo.com'
atpos = data.find('@')
print(atpos)

```

Tuple

Tuple tip podatka vrlo je sličan listi. Razlika je u tome što se elementi tuple-a ne mogu mijenjati. Tuple se definira kao niz podataka koji se odvajaju zarezom unutar oble zagrade. Međutim, zagrada nije nužna. Elementi tuple-a se indeksiraju i izdvajaju na isti način kao i elementi liste kao što prikazuje primjer 1.7.

■ Primjer 1.7 Tuple

```

letters = ('a', 'b', 'c', 'd', 'e')
numbers = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11)
mixed = (1, 'Hello', 3.14)

print(letters[0])
print(letters[1:4])

for letter in letters:
    print(letter)

```

Python rječnici

Rječnik je tip podataka u Pythonu gdje su podaci pohranjeni u obliku parova ključ-vrijednost (engl. *key-value*). Definira se pomoću vitičastih zagrada. Pristupanje vrijednostima izvršava se specificiranjem ključa kako je prikazano u primjeru 1.8.

■ Primjer 1.8 Rječnik

```

hr_num = {'jedan':1, 'dva':2, 'tri':3}

print(hr_num)
print(hr_num['dva'])

hr_num['cetiri'] = 4
print(hr_num)

```

Ostalo

Uz Python dolazi standardna biblioteka koja sadrži mnogo ugrađenih modula. Tipičan primjer su matematički i numerički moduli kao koji sadrže matematičke funkcije, generatore nasumičnih brojeva i sl. Moduli se uključuju pomoću ključne riječi `import` kao što je prikazano u primjeru 1.9. Nadalje, uobičajeno je blok koda koji se često izvršava staviti u zasebnu cjelinu – funkciju. Funkcije u Python definiraju se pomoću ključne riječi `def` kao što prikazuje primjer 1.10.

■ Primjer 1.9 Korištenje modula

```
import random
import math

for i in range(10):
    x = random.random()
    y = math.sin(x)
    print('Broj:', x, ' Sin(broj):', y)
```

■ Primjer 1.10 Funkcije

```
def print_hello():
    print("Hello world")

print_hello()
```

Otvaranje tekstualnih datoteka se može učiniti putem ugrađene funkcije `open` koja vraća rukovatelj datotekom. Ovaj objekt ima metodu `.readLine()` koja služi za čitanje sadržaja datoteke red po red. U primjeru 1.11 otvara se datoteka `example.txt` te se čita pred po red i ispisuje na ekran.

■ Primjer 1.11 Otvaranje tekstualne datoteke i čitanje red po red

```
fhand = open('example.txt')
for line in fhand:
    line = line.rstrip()
    print(line)
    words = line.split()

fhand.close()
```

1.3 Priprema za vježbu

Upoznajte se s programskim jezikom Python prema poglavlju 1.2. Po potrebi koristite i dodatnu literaturu. Analizirajte primjere 1.1 do 1.11

1.4 Rad na vježbi

1. Isprobajte Python primjere iz poglavlja 1.2. Python u Visual Studio Code IDE. Razmislite o svakoj liniji programskega koda i što je njen rezultat.

2. Riješite dane zadatke.

Zadatak 1.4.1 Napišite program koji od korisnika zahtijeva unos radnih sati te koliko je plaćen po radnom satu. Koristite ugrađenu Python metodu `input()`. Nakon toga izračunajte koliko je korisnik zaradio i ispišite na ekran. Na kraju prepravite rješenje na način da ukupni iznos izračunavate u zasebnoj funkciji naziva `total_euro`.

Primjer:

Radni sati: 35 h
eura/h: 8.5
Ukupno: 297.5 eura

Zadatak 1.4.2 Napišite program koji od korisnika zahtijeva upis jednog broja koji predstavlja nekakvu ocjenu i nalazi se između 0.0 i 1.0. Ispišite kojoj kategoriji pripada ocjena na temelju sljedećih uvjeta:

= 0.9 A
= 0.8 B
= 0.7 C
= 0.6 D
< 0.6 F

Ako korisnik nije utipkao broj, ispišite na ekran poruku o grešci (koristite `try` i `except` naredbe). Također, ako je broj izvan intervala [0.0 i 1.0] potrebno je ispisati odgovarajuću poruku.

Zadatak 1.4.3 Napišite program koji od korisnika zahtijeva unos brojeva u beskonačnoj petlji sve dok korisnik ne upiše „Done“ (bez navodnika). Pri tome brojeve spremajte u listu. Nakon toga potrebno je ispisati koliko brojeva je korisnik unio, njihovu srednju, minimalnu i maksimalnu vrijednost. Sortirajte listu i ispišite je na ekran. Dodatno: osigurajte program od pogrešnog unosa (npr. slovo umjesto brojke) na način da program zanemari taj unos i ispiše odgovarajuću poruku.

Zadatak 1.4.4 Napišite Python skriptu koja će učitati tekstualnu datoteku naziva `song.txt`. Potrebno je napraviti rječnik koji kao ključeve koristi sve različite riječi koje se pojavljuju u datoteci, dok su vrijednosti jednake broju puta koliko se svaka riječ (ključ) pojavljuje u datoteci. Koliko je riječi koje se pojavljuju samo jednom u datoteci? Ispišite ih.

Zadatak 1.4.5 Napišite Python skriptu koja će učitati tekstualnu datoteku naziva `SMSSpamCollection.txt` [1]. Ova datoteka sadrži 5574 SMS poruka pri čemu su neke označene kao *spam*, a neke kao *ham*. Primjer dijela datoteke:

```
ham    Yup next stop.  
ham    Ok lar... Joking wif u oni...  
spam   Did you hear about the new "Divorce Barbie"? It comes with all of Ken's stuff!
```

- Izračunajte koliki je prosječan broj riječi u SMS porukama koje su tipa ham, a koliko je prosječan broj riječi u porukama koje su tipa spam.
- Koliko SMS poruka koje su tipa spam završava uskličnikom ?

1.5 Izvještaj s vježbe

Kao izvještaj s vježbe prihvaća se web link na repozitorij pod nazivom OSU_LV.

Literatura

[1]<https://archive.ics.uci.edu/ml/datasets/SMS+Spam+Collection#>