

OSNOVE PROGRAMIRANJA PYTHON

Profesor
Dr Milan Paroški

Novi Sad, 2024-2025



Prošli čas: `a="a"`

Python obezbeđuje dva operatora:

is : jeste

i

is not : nije,

Operatori IS određuju da li dati operandi imaju isti identitet – to jest, da li upućuju na isti objekat.

Ovo nije isto što i jednakost (`==`)

Jednakost znači da se dva operanda odnose na objekte koji sadrže iste podatke, ali nisu nužno isti objekat.

Evo primera dva objekta koja su jednaka, ali nisu identična (Python editor):

Program:

```
x=1001  
y=1000+1  
print(x,y)
```

Rezultat:

1001 1001

`x == y`

Rezultat je?

True

JESU JEDNAKI

`x is y`

Rezultat je?

False `x` i `y` imaju iste podatke ali nisu isti objekat

**A DA NISU IDENTIČNI
POTVRDJUJE NAM I OVO:**

`id(x)`

3065611735952

`id(y)`

3065611736016



```
a="Python"
```

```
b=a
```

```
id(a)
```

```
3065611493104
```

```
id(b) ?
```

```
3065611493104
```

```
a is b
```

```
True
```

IDENTIČNI SU

```
a == b
```

```
True
```

JEDNAKI SU



Python IDLE editor:

```
a="a"  
print(a)  
print("aa")  
print(id(a))  
print(id("aa"))  
print("aa"==a+a)  
print("aa" is a+a)
```

Rezultat:

a

aa

140734188258368

2170642092272

True

False



Podsetnik: Python običan editor:

```
x=1001  
y=1000+1  
x==y
```

Rezultat:

True

JESU JEDNAKI

x is y

Rezultat je?

False

NISU IDENTIČNI

```
>>> id(x) :2330912370160
```

```
>>> id(y) :2330909580816
```

Python IDLE editor:

```
x=1001  
y=1000+1  
print(x==y)  
print(x is y)  
print(id(x))  
print(id(y))
```

Rezultati su?

True

JESU JEDNAKI

True

JESU IDENTIČNI

```
1860807134960
```

```
1860807134960
```

Razlika je u optimizaciji koja nastaje kada je u pitanju Python skripta (IDLE editor) u odnosu na naredbu po naredbu koju kucate u interpreteru.

U skripti mi imamo „kod“ koji će se potencijalno više puta izvršiti. Vrednosti u promenljivama, radi optimizacije, mogu imati isti ID jer im je jednaka vrednost, a nju samo čuvamo na jednoj memorijskoj lokaciji (ID-ju).

Kada pišemo naredbu po naredbu u interpreteru ta optimizacija nije zagarantovana, te se iz tog razloga vrednosti u promenljivama nalaze na različitim ID-jevima.

Opis ugrađene funkcije id to objašnjava: ("Two objects with non-overlapping lifetimes may have the same id() value."). Dva objekta sa životnim vekom koji se ne preklapaju mogu imati istu vrednost id().

id(object)

Return the "identity" of an object. This is an integer which is guaranteed to be unique and constant for this object during its lifetime. Two objects with non-overlapping lifetimes may have the same id() value.

POGLAVLJE 3

IZRAZI U JEZIKU PYTHON

Dr Milan Paroški
mparoski@singidunum.ac.rs

Univerzitet Singidunum

2024/2025

Sadržaj

1. Uvod

2. Izrazi

3. Operatori

4. Prioritet operatora

5. Konverzija tipova

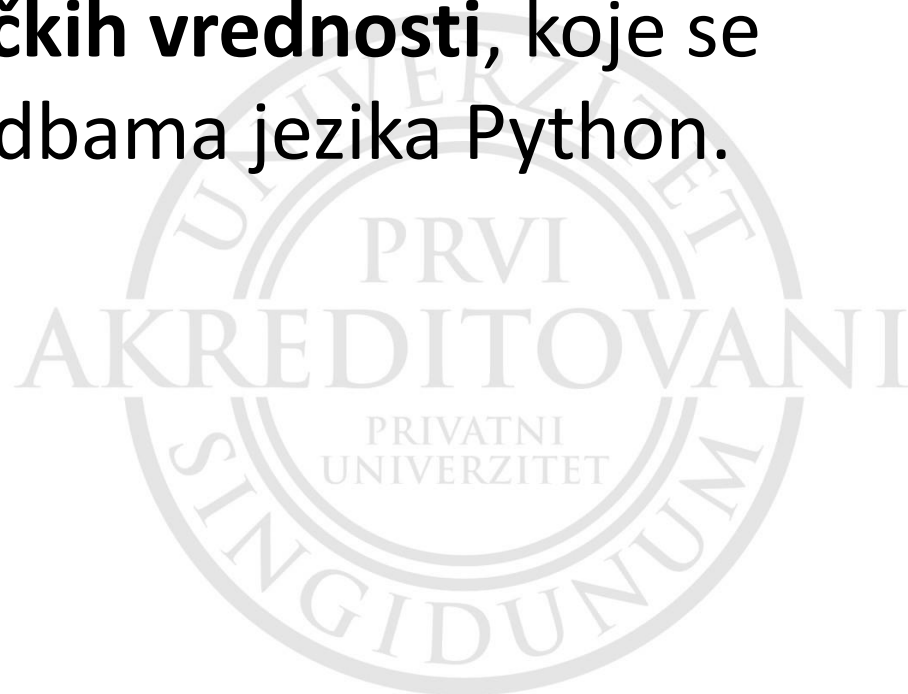
6. Zaokruživanje

7. Primer programa



1.UVOD

- **Izrazi omogućavaju računanje različitih numeričkih i nenumeričkih vrednosti, koje se koriste u različitim naredbama jezika Python.**



Izrazi u proceduralnim programskim jezicima opisuju računanja koja se mogu:

1. dodeliti promenljivoj :

$a = 2 + 5.26$

ili

2. koristiti kao vrednost argumenta neke funkcije:

`print(2 + 5.26)`



primeri vrednosti:

0, 1, -1234

- celobrojni (int) brojevi

1.25, .02, -146.75

- Decimalni (float) brojevi

-3j, -0.5 - 4.25j

- kompleksni (complex) brojevi

{ False, True }

- Logički (bool) simboli

'Pajton je lagan',

- tekstualni (str) niz znakova u Unicode-u



Unicode je univerzalni standard za kodiranje, prikaz i obradu teksta izrađenog u različitim jezicima i tehničkim sistemima.

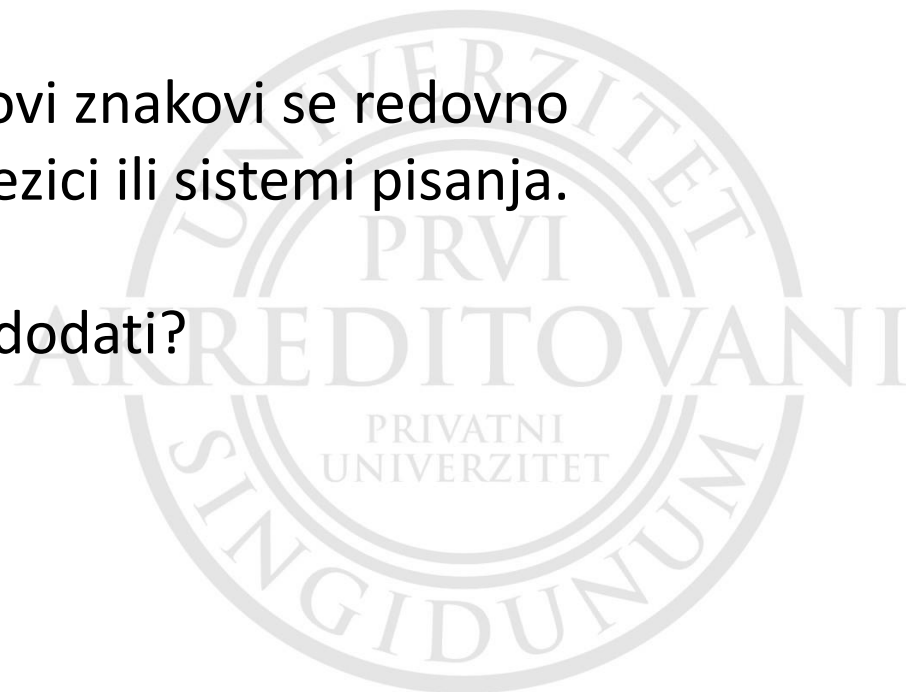
Njegova svrha je da omogući jedinstvenu reprezentaciju znakova za skoro sve pisane jezike, kao i simbole, brojeve i specijalne znakove, čime omogućava interoperabilnost ?
razmenu teksta između različitih uređaja i sistema.

Sadrži više od 143.000 znakova, a novi znakovi se redovno dodaju kako bi se podržali dodatni jezici ili sistemi pisanja.

Koji su recimo noviji znakovi koji su dodati?

Emotikoni

znak emotikona 😊 je U+1F604



Znaci su predstavljeni kao heksadecimalni brojevi i kreću se od U+0000 do U+10FFFF.

Koliko je to kombinacija?

10FFFF

$1 \times 16^5 + 0 \times 16^4 + 15 \times 16^3 + 15 \times 16^2 + 15 \times 16^1 + 15 \times 16^0$

1.114.111



`()`, `(1.25,True,'abc')`

- torka tuple-nepromenljiv niz objekata.

`[]`, `[1,2]`, `['A',1,1+2j]`

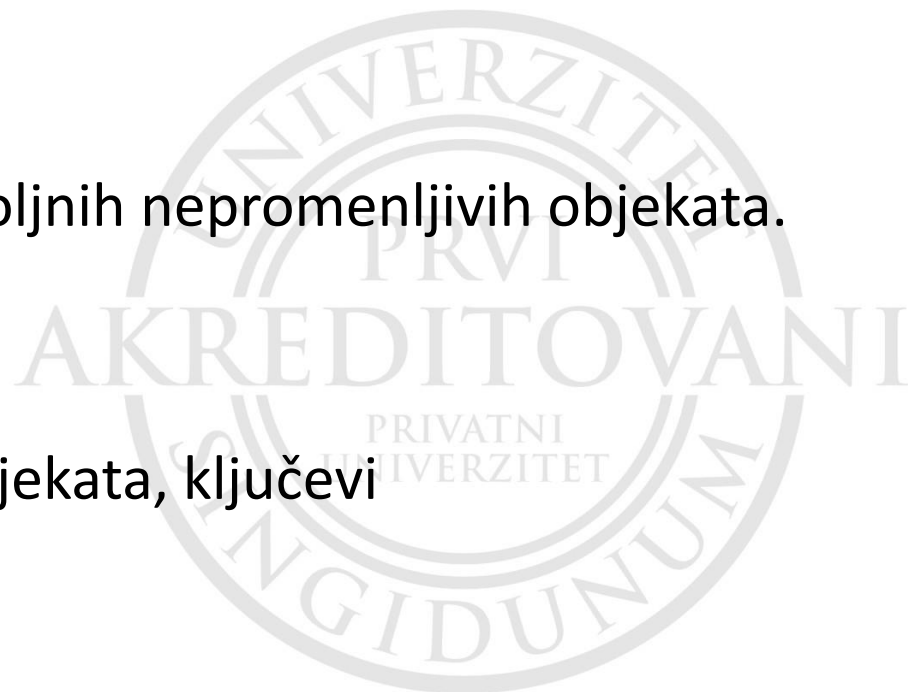
- lista list-promenljiv niz objekata.

`{}`, `{1,2}`, `['A',{1,2}]`

- skup set- neuredjen skup proizvoljnih nepromenljivih objekata.

`{11:'Bg', 21:'Ns', 22:'Ru'}`

- rečnik. Vrednosti proizvoljnih objekata, ključevi



2. IZRAZI

- Pojam
- Evaluacija izraza
- Operatori

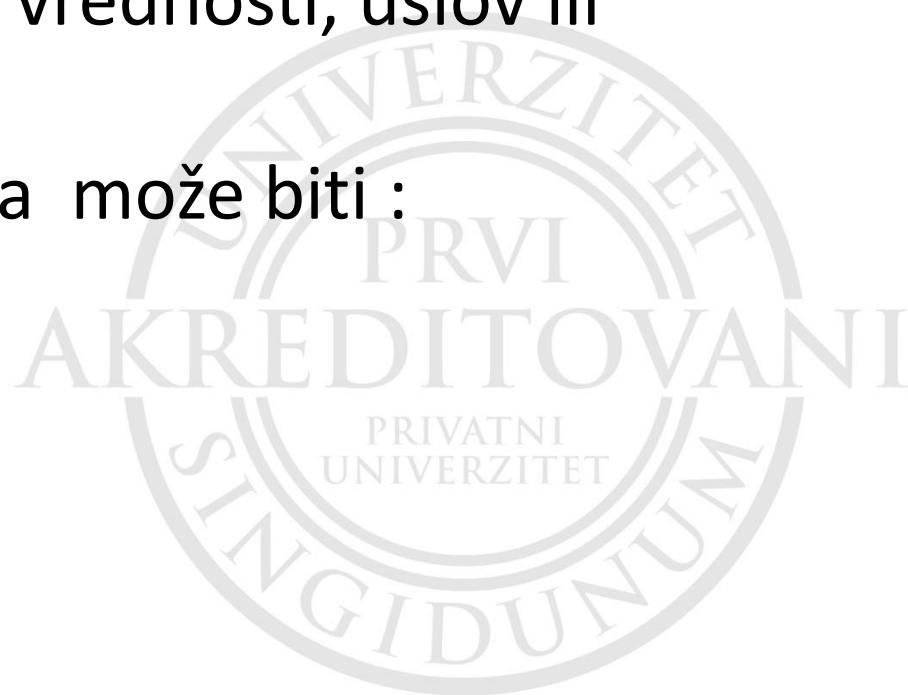


POJAM

- Izrazi su ispravne kombinacije *vrednosti* (konstanti), *promenljivih*, *funkcija* i *operatora*
- **Izrazi nakon izračunavanja (evaluacije) daju rezultat određenog tipa**, koji se može koristiti u drugim naredbama (dodela vrednosti, uslov ili argument funkcije)

Rezultat izračunavanja izraza može biti :

- numerički,
- nenumerički ili
- logički



Dozvoljena je upotreba različitih tipova vrednosti u jednom izrazu, tako što se eksplicitno ili implicitno izvrši njihovo pretvaranje u neki zajednički (kompatibilni) tip vrednosti

Na primer u izrazu:

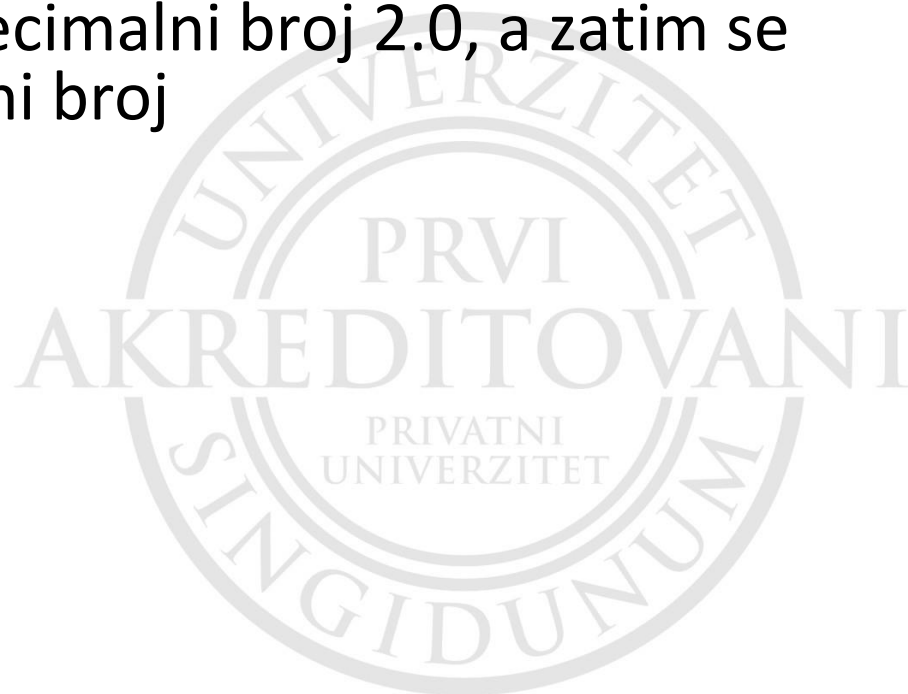
$a = 2 + 4.17$

zbir celog i decimalnog broja vrši se nakon implicitnog pretvaranja celog broja u decimalni broj 2.0, a zatim se kao rezultat dobija decimalni broj

$a = 2 + 4.17$

`print(a)?`

6.17



2.čas



- Operatori u jeziku Python definisani su za određeni tip vrednosti na koje se mogu primeniti

2 + "2" ŠTA DAJE?

Greška

Kako napisati ispravno?

2+int("2") ŠTA DAJE?

rezultat je 4

Kako ćemo ispravno sabrati 2 i "2.5" ?print(

print(2+float("2.5"))



Šta daje: `print("2"+2)` ?

Greška

Šta daje: `print("2"+str(2))` ?

22

Šta daje: `print("2"+"2")` ?

22



U odnosu na broj argumenata, operatori mogu biti :
jedan argument ?

- *unarni*

dva argumenta?

- *binarni*

Složeni izrazi se sastoje od više operatora različitih tipova



Primer programa

Većiti kalendar: broj dana između dva datuma

Koliko dana od nove ere je proteklo od neke zadate godine?



Deo formule koji se odnosi na godine:

$$n = 365 \cdot g + g // 4 - (g // 100 - g // 400) \quad ?$$

Godine koje nisu prestupne se zovu proste godine. Prestupna je svaka 4. godina, sem u slučaju da je deljiva sa 100 a nije sa 400.

Godine 1800, 1900. i 2100. nisu prestupne, dok 1600, 2000. i 2400 jesu.

$$g=2022$$

$$n=365 \cdot 2022 + 2022 // 4 - (2022 // 100 - 2022 // 400) = 738520$$


```
print("Unesite godinu:")  
g=int(input())  
n = 365*g + g//4 - (g//100 - g//400)  
print(n)
```

Unesite godinu:

2022

738520



Razlika u danima izmedju 2 godine

```
print("Unesite godinu1:")
```

```
g=int(input())
```

```
n = 365*g + g//4 - g//100 + g//400
```

```
print("Unesite godinu2:")
```

```
g=int(input())
```

```
m = 365*g + g//4 - g//100 + g//400
```

```
print(n-m)
```

Unesite godinu1:

2024

Unesite godinu2:

1962

22646

Broj dana između dva datuma računa se jednostavno kao razlika broja dana od početka računanja za prvi (n_1) i drugi datum (n_2):

$$n = n_2 - n_1$$



Algoritam razlike 2 datuma

- Broj dana n između zadanog datuma $dan.mesec.godina$ i nekog fiksnog početnog datuma, npr. Početka naše ere, može se dobiti izrazom:

$$n = 365 \cdot g + g // 4 - g // 100 + g // 400 + (m \cdot 306 + 5) // 10 + dan - 1$$

gde su:

$$m = (mesec + 9) \% 12$$

$$g = godina - m // 10$$



Objasnjenje formule

Pojasnili smo prvi deo formule

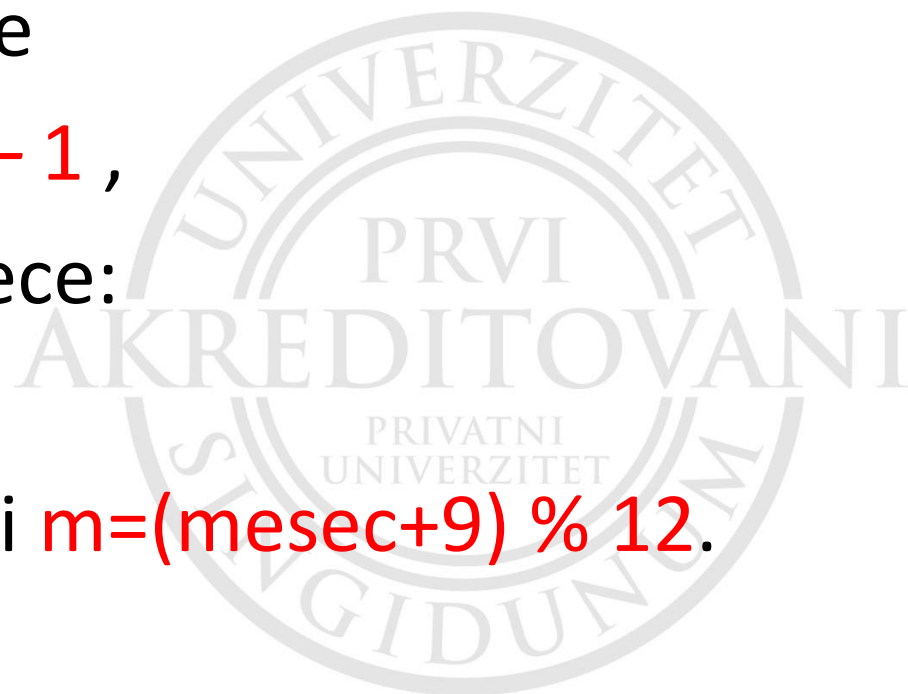
$$(365 \cdot g + g // 4 - (g // 100 - g // 400))$$

Što se tice ostatka formule

$$+ (m \cdot 306 + 5) // 10 + dan - 1 ,$$

njeno objasnjenje je sledece:

Za početak cemo objasniti $m = (mesec + 9) \% 12$.



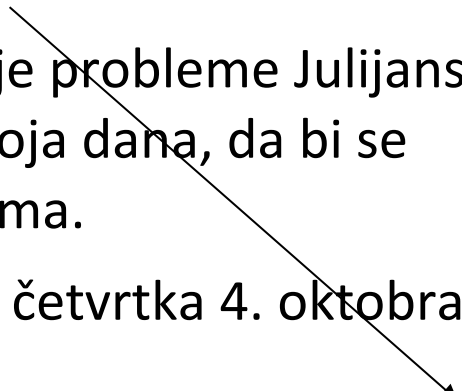
$$m = (\text{mesec} + 9) \% 12$$

Gregorijanski kalendar je donet 1582 godine, i rešio je probleme Julijanskog (godina se produžavala) izbacivanjem izvesnog broja dana, da bi se kalendar vratio u sinhronizaciju sa godišnjim dobima.

Iz kalendara je izostavljeno 10 dana, tako da je posle četvrtka 4. oktobra 1582. sledio petak 15. oktobar.

Gledamo po Gregorijanskom kalendaru
od **oktobra** 1582 godine.

Najmanji mesec je januar, te samim tim dobijamo
da je 10, što prestavlja mesec oktobar, ako
unesemo broj 1 (januar) u python dobijamo 10.



mesec	RB	formula
januar	1	10
februar	2	11
mart	3	0
april	4	1
maj	5	2
jun	6	3
jul	7	4
avgust	8	5
septembar	9	6
oktobar	10	7
novembar	11	8
decembar	12	9

$$g = godina - m // 10$$

g označava samu godinu u formuli.

Ako uzmemo 2010 godinu i mesec maj (5), dobijamo da je $m = ?$

$$m = (5+9) \% 12 = 2,$$

a g ostaje 2010?

$$g = 2010 - 2 // 10 = 2010 - 0 = 2010$$

Znači peti mesec 2010 postaje drugi mesec 2010

Na primer: mesec januar (1) i godina (2010), dobijamo da je $m, g = ?$

$$m = 10,$$

$$g = 2009$$

$m // 10$ označava da ako je m veći od 10, da će se on deliti sa 10, te će se ostatak pisati.

Znači prvi mesec 2010 postaje deseti mesec 2009

A treći mesec 2010 postaje prvi mesec 2010

$$(m * 306 + 5) // 10$$

U ovom delu formule dobijamo broj dana koji je ostao u jednoj godini.

m oznacava mesec

365 – broj dana u godini ?

imamo 7 meseci po 31 dan (217 dana), i 4 meseca po 30 dana (120 dana), a februar ima 28 dana = 365

306 oznacava ostatak dana?

treći mesec 2010 postaje prvi mesec 2010

365-59: Od ukupnih dana kroz jednu godinu moramo da oduzmemo 59 dana (31+28), iz razloga sto smo uzeli **oktobar** kao pocetak merenja.

5 oznacava da se dodaju dani zbog prestupnih godina (5 prestupnih godina na 400 je razliak u dva kalendara)

Poslednji deo formule oznacava da se deli sa 10 i uzima ceo broj, jer smo poceli da gledamo od oktobra.

dan – 1

dan predstavlja broj dana koji smo uneli

Zašto **dan -1**?

1 označava broj dana koji se oduzima od
unesenog dana iz razloga što se ne računa

"**danasnji dan**,"

ako smo uneli 10.10.2010. godine, oduzima se
jedan dan jer ne se taj dan ne ubraja kao
razliku dana.

Unos koda u python

Na pocetku unosimo podatke za **prvi datum**, pa za **drugi datum**.

Posle toga prilagodjavamo brojanje **meseci i godina**

Nakon dobijenog **m1,m2 i g1,g2** mozemo da predjemo na glavnu formulu, gde racunamo redne brojeve dana u godini.

Nakon sto dobijemo **n1 i n2**, od **n2 oduzmemo n1** i dobijamo **n** (sto predstavlja razliku dana izmedju jednog datuma i drugog)

- Osnovnu algoritam računarskog rešenja je:

1. učitati dan1, mesec1, godina1

2. učitati dan2, mesec2, godina2

3. prilagoditi brojanje meseci $m1 = (\text{mesec1} + 9) \% 12$ i $m2 = (\text{mesec2} + 9) \% 12$

4. prilagoditi brojanje godina $g1 = \text{godina1} - m1 // 10$ i $g2 = \text{godina2} - m2 // 10$

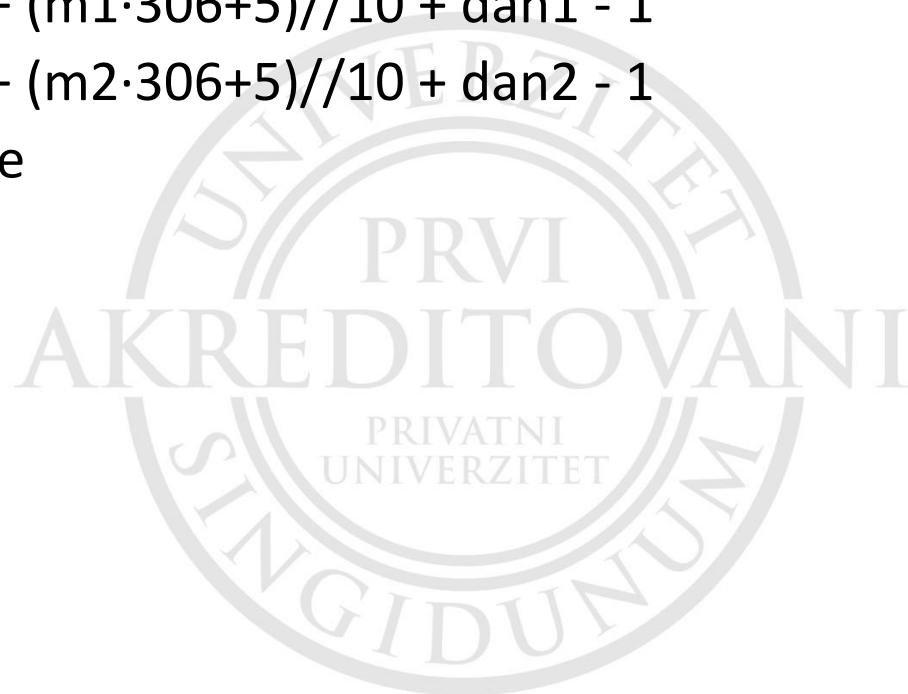
5. izračunati redne brojeve dana:

$$n1 = 365 \cdot g1 + g1 // 4 - g1 // 100 + g1 // 400 + (m1 \cdot 306 + 5) // 10 + \text{dan1} - 1$$

$$n2 = 365 \cdot g2 + g2 // 4 - g2 // 100 + g2 // 400 + (m2 \cdot 306 + 5) // 10 + \text{dan2} - 1$$

6. Broj dana između dva zadana datuma je

$$n = n2 - n1$$



1. Unos prvog datuma

```
print("Unesite prvi datum")  
dan1= int(input(" dan: "))  
mesec1= int(input(" mesec: "))  
godina1 = int(input("godina: "))
```

2.Unos drugog datuma

```
print("Unesite drugi datum")  
dan2= int(input(" dan: "))  
mesec2= int(input(" mesec: "))  
godina2 = int(input("godina: "))
```

3. prilagoditi brojanje meseci m1 i m2

```
m1= (mesec1+9) % 12  
m2= (mesec2+9) % 12
```

4. prilagoditi brojanje godina g1 i g2

```
g1 = godina1 - m1 // 10  
g2 = godina2 - m2 // 10
```

P

r

O

g

r

a

m



Nastavak programa

5. izračunati redne brojeve dana n1 i n2

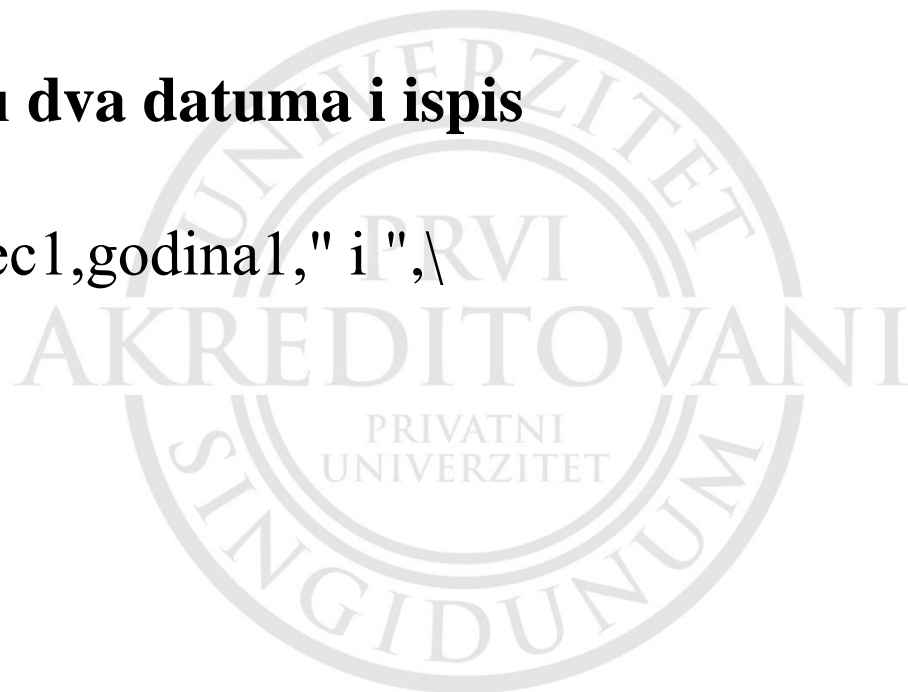
```
n1 = 365*g1 + g1 // 4 - g1//100 + g1 // 400 + \  
(m1*306+5) // 10 + dan1 - 1
```

```
n2 = 365*g2 + g2 // 4 - g2//100 + g2 // 400 + \  
(m2*306+5) // 10 + dan2 - 1
```

6. Računanje broja dana između dva datuma i ispis

```
n = n2 - n1
```

```
print("Broj dana između",dan1,mesec1,godina1," i ",\  
dan2, mesec2, godina2, " je ", n)
```



Izvršavanje

Unesite prvi datum

dan: 11

mesec: 11

godina: 1918

Unesite drugi datum

dan: 11

mesec: 11

godina: 2017

Broj dana između 11 11 1918 i 11 11 2017 je 36160



Kalkulator datuma - Dani između dana (onlinealarmkur.com)



Manipulacije sa podacima vezanim za datum i vreme

- Python nema ugrađen tip podataka vezan za datum i vreme pa su ovi tipovi, kao i mogućnost manipulacije nad njima definisani preko spoljnog modula `datetime`.
- Zbog toga je potrebno, ukoliko planiramo da radimo sa bilo kakvim podacima vezanim za datum i vreme, izvršiti uvoz ovog modula na početku programskog koda

- **`import datetime`**

uz opciono dodavanje aliasa zbog lakšeg pozivanja:

- **`import datetime as dv`**



Biblioteka datetime

```
import datetime
```

```
print(datetime.today())
```

Šta je rezultat?

Greška

```
import datetime
```

```
print(datetime.date.today())
```

2024-10-09

```
import datetime
```

```
print(datetime.time(11,33,44))
```

11:33:44

```
import datetime
```

```
print(datetime.datetime.now())
```

2024-10-08 13:51:46.585083



Pregled najvažnijih datumskih opcija

%a

značenje i primer

Skraćeni naziv dana u nedelji (Mon, Tue, Wed, Thu, Fri, Sat, Sun).

Kako ispisati skraćeni naziv današnjeg dana?

Import datetime

```
datum=datetime.date.today()
```

```
print(f"{datum:%a}")
```

Tue



Pregled otalih najvažnijih datumskih opcija

značenje i primer

<u>a</u>	Skraćeni naziv dana u nedelji (Mon, Tue, Wed, Thu, Fri, Sat, Sun).
<u>A</u>	Pun naziv dana (Monday...
<u>d</u>	Dan – numerička vrednost
<u>m</u>	Mesec – numerička vrednost
<u>Y</u>	Godina – numerička vrednost

```
import datetime
datum=datetime.date.today()
print(f" { datum:% a }")
print(f" { datum:% A }")
print(f" { datum:% d }")
print(f" { datum:% m }")
print(f" { datum:% Y }")
```

Tue
Tuesday
08
10
2024



%w

Broj koji odgovara danu u sedmici – počev od 0

%b

Skraćeni naziv meseca

%j

Redni broj dana u godini

%B

Pun naziv meseca

```
import datetime
datum=datetime.date.today()
print(f" { datum:% w }")
print(f" { datum:% b }")
print(f" { datum:% j }")
print(f" { datum:% B }")
```

3

Oct

283

October



<u>%w</u>	Prikaz sati u formatu 0:24
<u>%b</u>	Prikaz sati u formatu 0:12
<u>%j</u>	Prikaz minuta
<u>%B</u>	Prikaz sekundi

```
import datetime
datum=datetime.datetime.now()
print(f" { datum:%H} ")
print(f" { datum:%I} ")
print(f" { datum:%M} ")
print(f" { datum:%S} ")
```

Rezultat sinoć/ rezultat u toku predavanja

23	11
11	11
04	47
23	15

```
import datetime as dv
danasnji = dv.date.today();
print("Datum je: ", danasnji)
danasnji = f"{danasnji:%d.%m.%Y}"
print("Datum po naški je: ", danasnji)
dani = dv.date.today();
dani = f"{dani: %A %a %B %b}"
print("Nazivi dana i meseca su :",dani)
broj = dv.date.today();
broj = f"{broj: %w %j}"
print("redni broj dana u sedmici i godini je:",broj)
```

Datum je: 2024-10-09

Datum po naški je: 09.10.2024

Nazivi dana i meseca su : Wednesday Wed October Oct

redni broj dana u sedmici i godini je: 3 283

#Primer: Prikaz trenutnog vremena

```
import datetime as dv
```

```
trenutno = dv.datetime.now().time()
```

```
# vreme se ispisuje u formatu 00:00:00 (sati:minuti:sekundi)
```

```
print("Trenutno vreme je:", trenutno)
```

```
trenutno = f"{trenutno: %H %M %S} "
```

```
print("Prikaz vremena u formatu 0-24 ", trenutno)
```

```
trenutno = dv.datetime.now().time()
```

```
trenutno = f"{trenutno: %I %M %S %p} "
```

```
print("Prikaz vremena u formatu 0-12 pa u AM/PM:", trenutno)
```

Trenutno vreme je: 15:29:30.774229

Prikaz vremena u formatu 0-24 15 29 30

Prikaz vremena u formatu 0-12 pa u AM/PM: 03 29 30 PM

3.čas



#Određivanje starosti osobe

```
import datetime as dv
```

```
danasnji_dan = dv.date.today()
```

```
datum_rodjenja = dv.date(1962, 11, 11)
```

```
starost = danasnji_dan - datum_rodjenja
```

```
print(starost)
```

21612 days, 0:00:00



#Primer: Dodavanje i oduzimanje broja dana

import datetime as dv

danasnji = dv.date.today();

trajanje = dv.timedelta(days=100)

print("Pre 100 dana:", danasnji - trajanje)

print("Za 100 dana:", danasnji + trajanje)

Pre 100 dana: 2024-06-30

Za 100 dana: 2025-01-16

