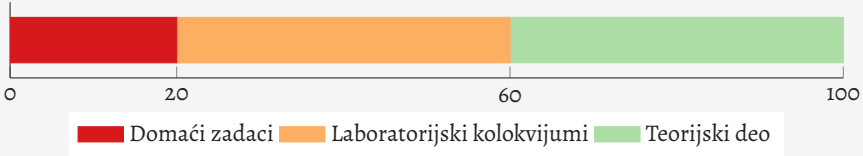
Uvod u programski jezik Pajton

Uvod u merno-informacione sisteme

Informacije o predmetu

- Tri domaća zadatka
- Dva laboratorijska kolokvijuma
- Teorijski deo



Informacije o predmetu

Domaći zadaci:

- Rade se samostalno kod kuće
- Predmetnom asistentu se prosleđuju (rok 7 dana)
- Nema popravnih
- Ne postoji minimalni broj bodova koji se mora osvojiti

Laboratorijski kolokvijumi:

- Rade se na računarima (Raspberry Pi) u laboratoriji
- Nakon dva kolokvijuma postoji opcija popravnog
- Na popravnom kolokvijumu može se popraviti samo jedan kolokvijum
- Beleži se poslednji osvojeni rezultat

Informacije o predmetu

Teorijski deo:

- Gradivo pređeno na predavanjima
- Dva dela (polažu se odjednom):
  - Python
  - Metrologija
- U svakom roku se može raditi popravni

Pismeni ispit:

- Ako nije osvojen dovoljan broj bodova
- Radi se na papiru u ispitnom roku
- Sastoji se od zadatka i teorije

Uvod u Pajton

Pajton programski jezik

- Tvorac Gvido Van Rosum

Pajton programski jezik

- Tvorac Gvido Van Rosum
- Prva verzija 1991. godine

Pajton programski jezik

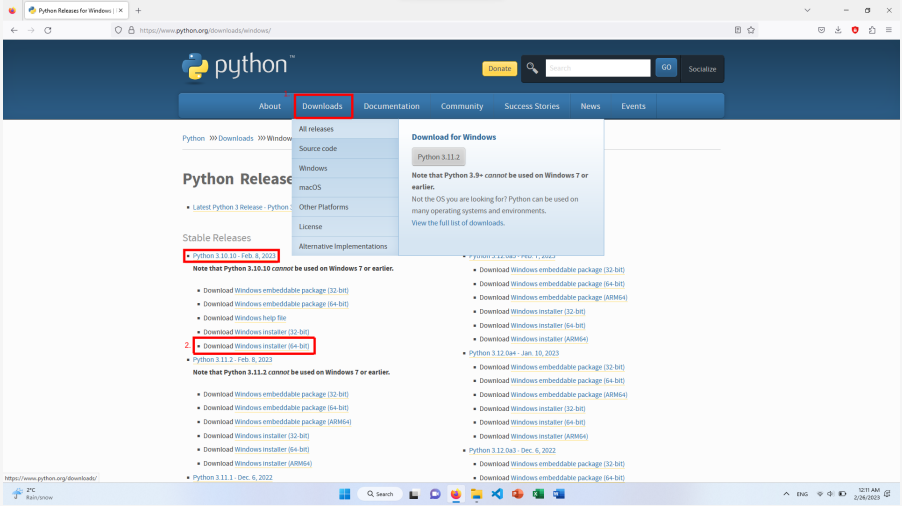
- Tvorac Gvido Van Rosum
- Prva verzija 1991. godine
- Ime dobio po Monti Pajtonu

Pajton programski jezik

- Tvorac Gvido Van Rosum
- Prva verzija 1991. godine
- Ime dobio po Monti Pajtonu
- Otvorenog koda
- Odlikuje ga jednostavnost

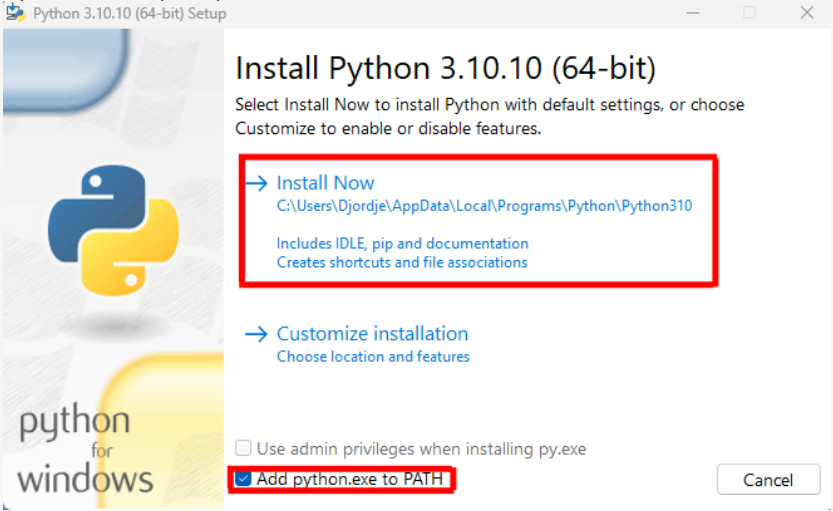
Instalacija Pajtona - Windows

Preuzimanje i instalacija Pajtona 3.10.



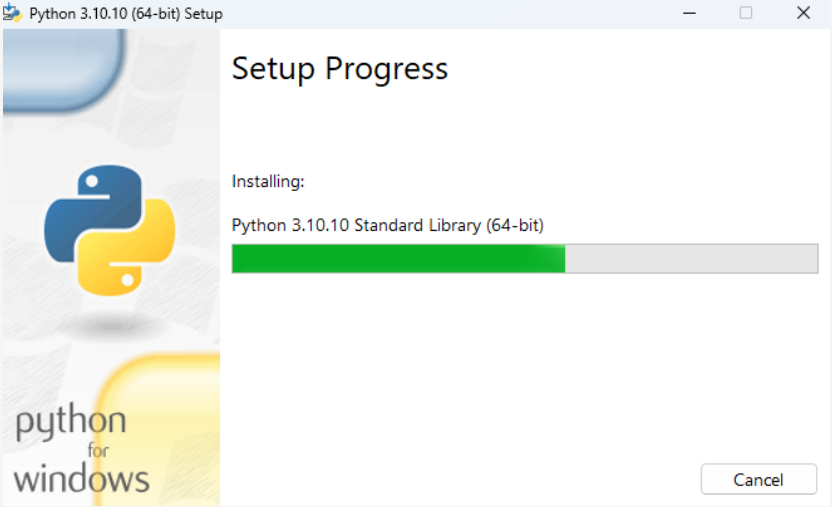
Instalacija Pajtona - Windows

Preuzimanje i instalacija Pajtona 3.10.



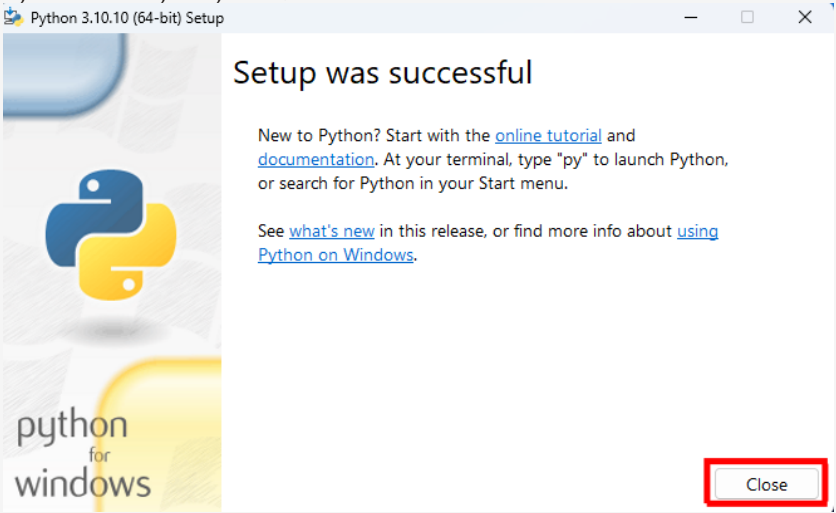
Instalacija Pajtona - Windows

Preuzimanje i instalacija Pajtona 3.10.



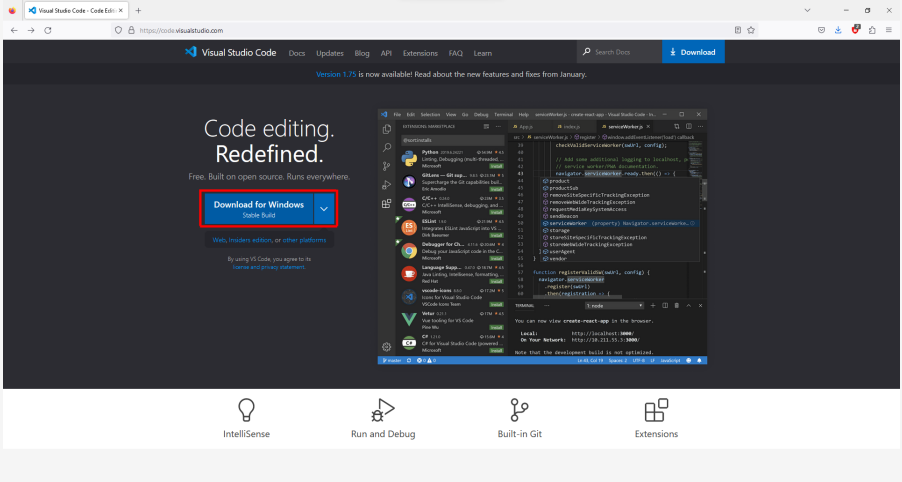
Instalacija Pajtona - Windows

Preuzimanje i instalacija Pajtona 3.10.



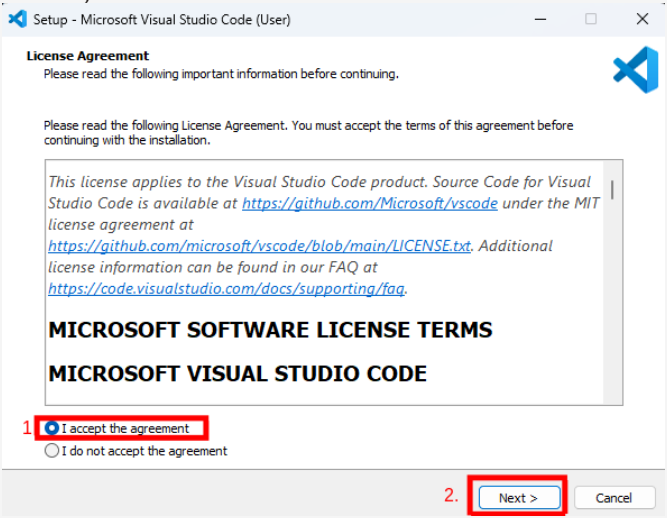
Instalacija VS Code - Windows

Preuzimanje i instalacija VS Code



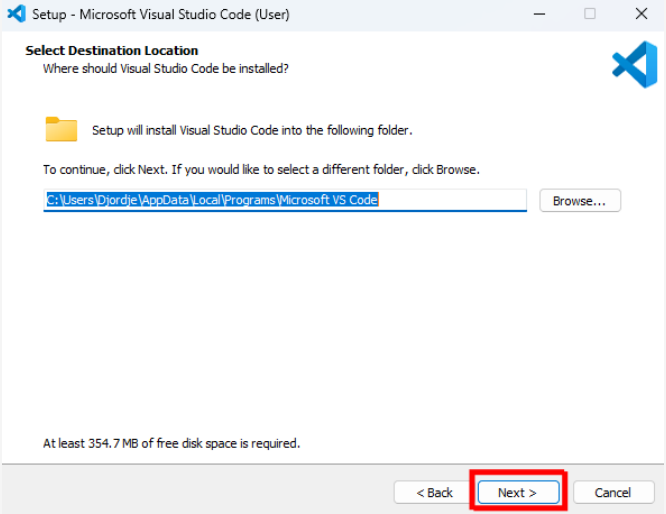
Instalacija VS Code - Windows

Preuzimanje i instalacija VS Code



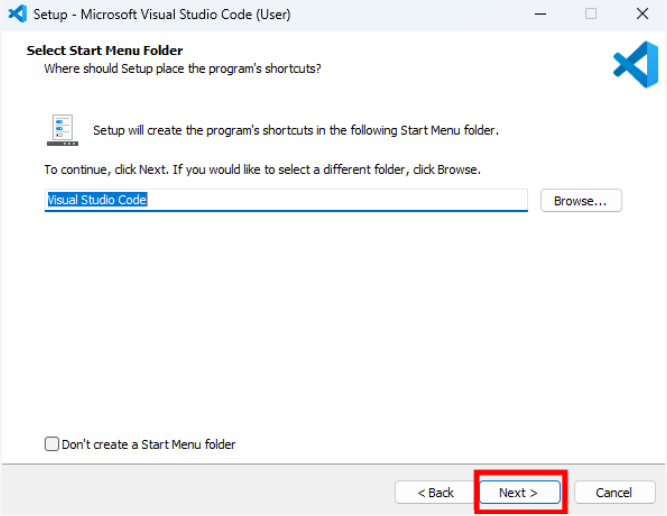
Instalacija VS Code - Windows

Preuzimanje i instalacija VS Code



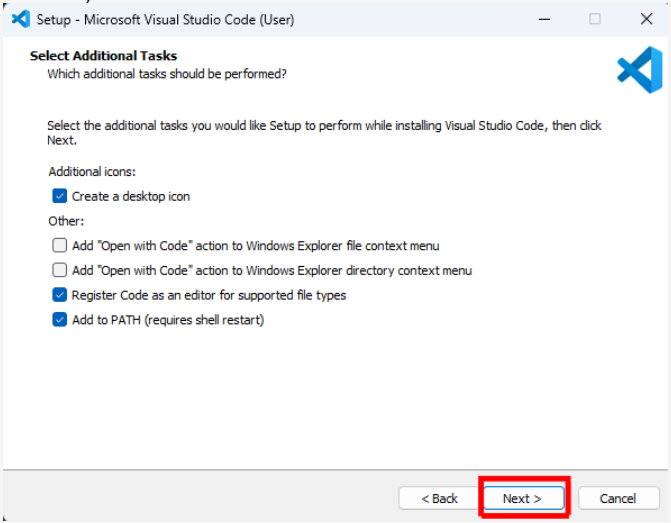
Instalacija VS Code - Windows

Preuzimanje i instalacija VS Code



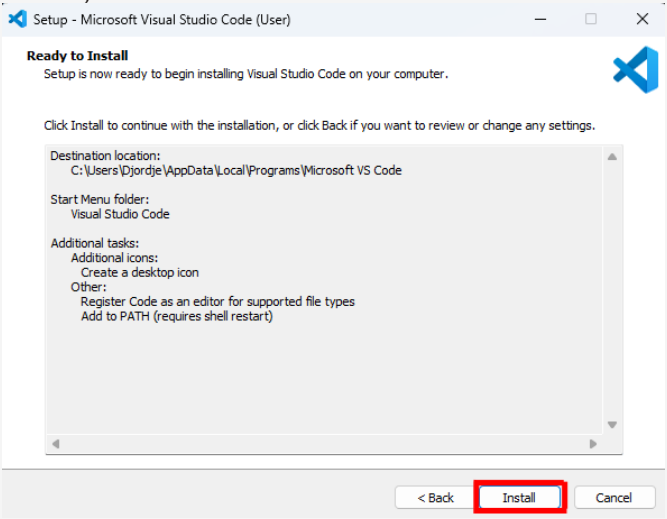
Instalacija VS Code - Windows

Preuzimanje i instalacija VS Code



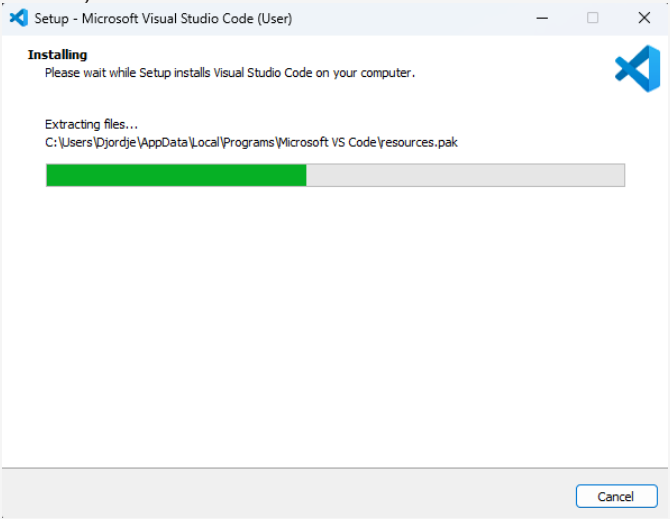
Instalacija VS Code - Windows

Preuzimanje i instalacija VS Code



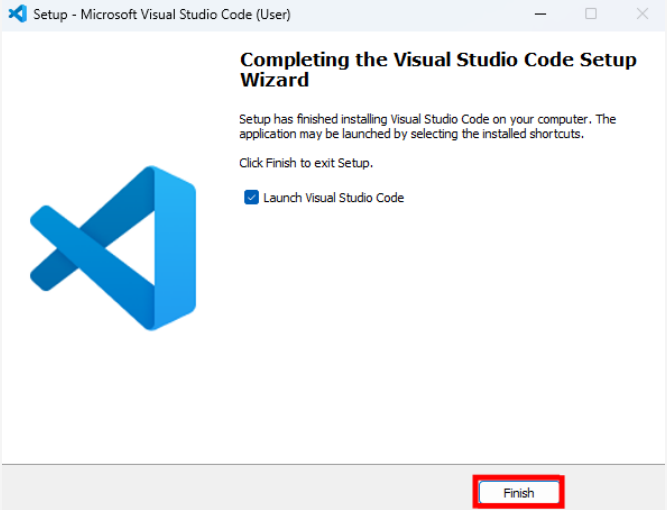
Instalacija VS Code - Windows

Preuzimanje i instalacija VS Code



Instalacija VS Code - Windows

Preuzimanje i instalacija VS Code



Instalacija Pajtona i VS Code - Ubuntu Linux

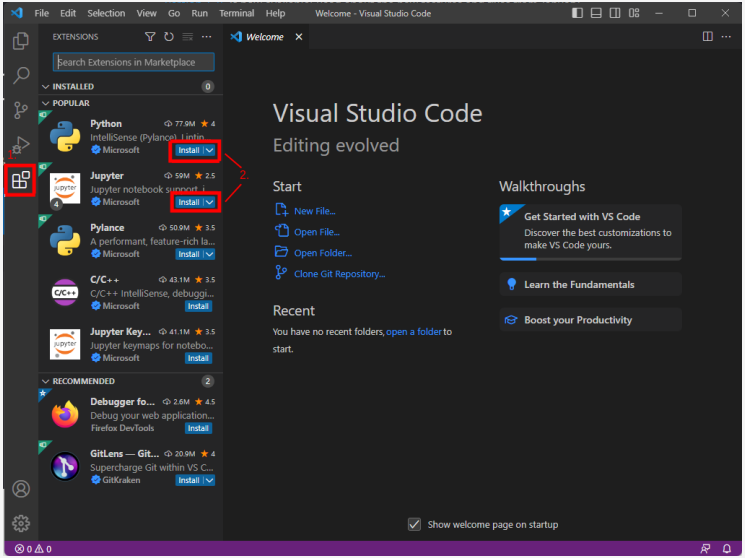
Instalacija Pajtona

```
sudo apt install python3.10
```

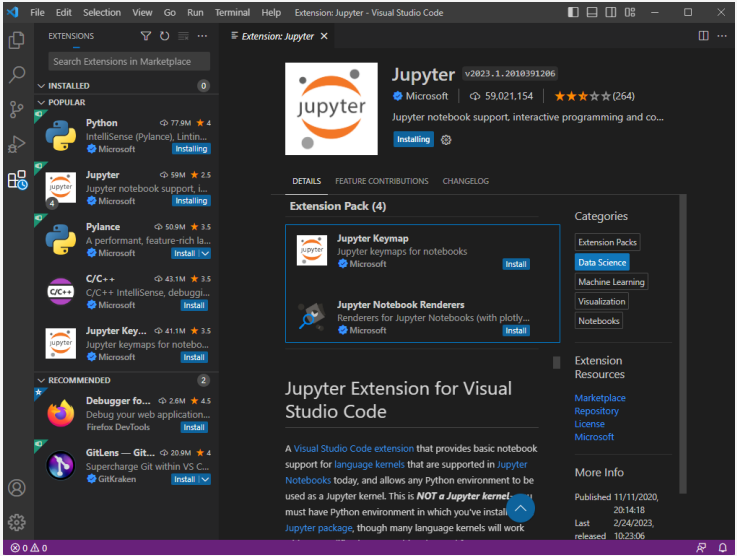
Instalacija VS Code korišćenjem snap-a

```
sudo snap install –classic code
```

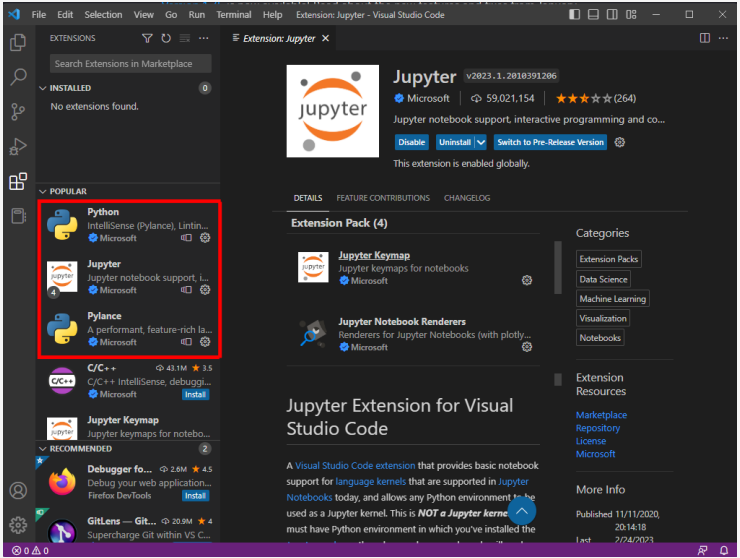
Podešavanje VS Code aplikacije



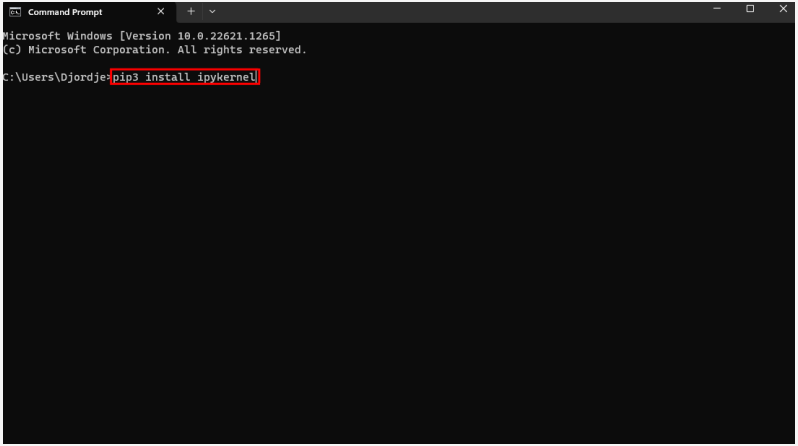
Podešavanje VS Code aplikacije



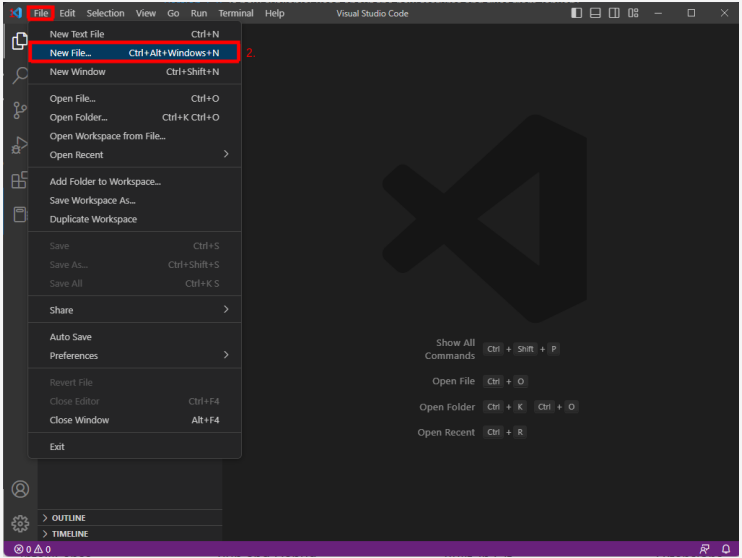
Podešavanje VS Code aplikacije



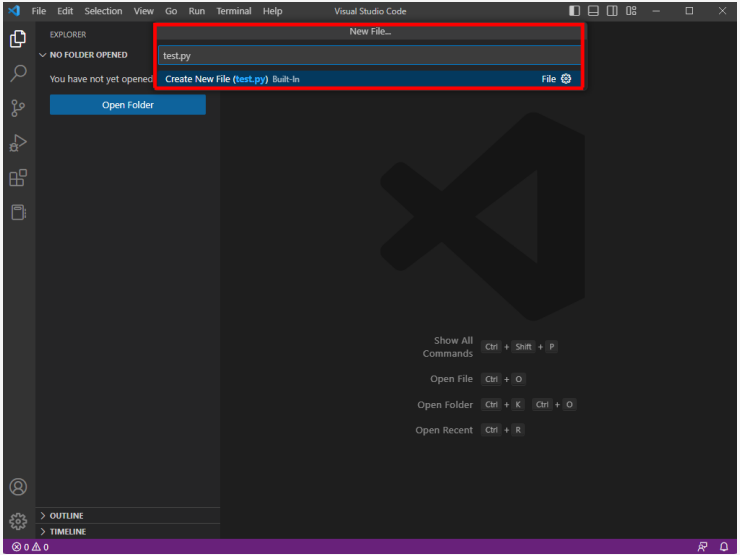
Podešavanje VS Code aplikacije



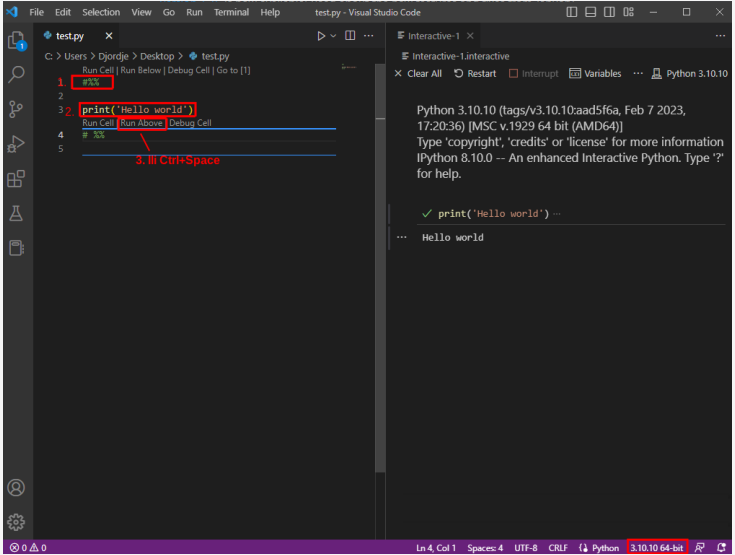
Realizacija prve aplikacije



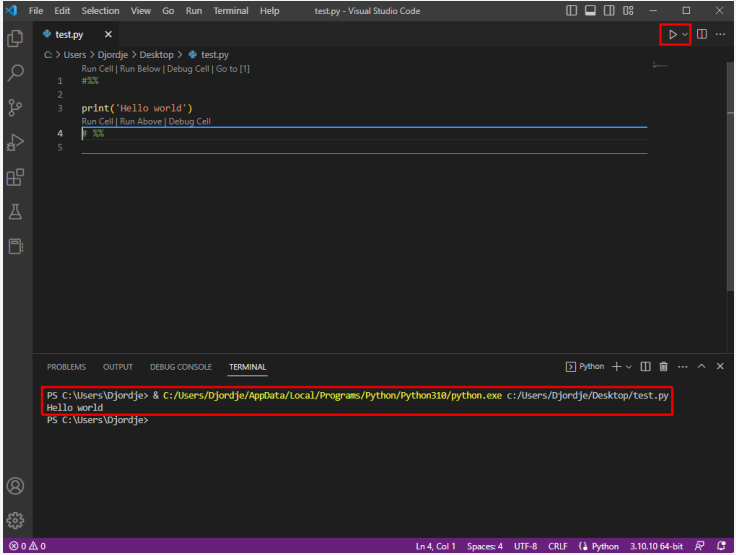
Realizacija prve aplikacije



Realizacija prve aplikacije



Realizacija prve aplikacije



Osnovne konvencije

- Ne postoji main funkcija, pa program izvršava prvu instrukciju u fajlu

Osnovne konvencije

- Ne postoji main funkcija, pa program izvršava prvu instrukciju u fajlu
- Kraj linije predstavlja i kraj naredbe (ne piše se ; na kraju)

```
1  print('Hello world')
```

Osnovne konvencije

- Ne postoji main funkcija, pa program izvršava prvu instrukciju u fajlu
- Kraj linije predstavlja i kraj naredbe (ne piše se ; na kraju)
- Komentar započinje hash simbolom # i završava se krajem linije

```
1 # Ovo je komentar
2 print('Hello world')
```

Osnovne konvencije

- Ne postoji main funkcija, pa program izvršava prvu instrukciju u fajlu
- Kraj linije predstavlja i kraj naredbe (ne piše se ; na kraju)
- Komentar započinje hash simbolom # i završava se krajem linije

```
1  # Ovo je komentar
2  print('Hello world')
```

- Imena promenljivih mogu sadržati slova, cifre i donju crtu (underscore _)

Osnovne konvencije

- Ne postoji main funkcija, pa program izvršava prvu instrukciju u fajlu
- Kraj linije predstavlja i kraj naredbe (ne piše se ; na kraju)
- Komentar započinje hash simbolom # i završava se krajem linije

```
1 # Ovo je komentar
2 print('Hello world')
```

- Imena promenljivih mogu sadržati slova, cifre i donju crtu (underscore _)
- Pajton je case sensitive (razlikuje velika i mala slova)

Osnovne konvencije

- Ne postoji main funkcija, pa program izvršava prvu instrukciju u fajlu
- Kraj linije predstavlja i kraj naredbe (ne piše se ; na kraju)
- Komentar započinje hash simbolom # i završava se krajem linije

```
1 # Ovo je komentar
2 print('Hello world')
```

- Imena promenljivih mogu sadržati slova, cifre i donju crtu (underscore _)
- Pajton je case sensitive (razlikuje velika i mala slova)
- Postoje rezervisane reči koje se ne smeju koristiti

Rezervisane reči

False	def	if	raise	None	del
import	return	True	elif	in	try
and	else	is	while	as	except
lambda	with	assert	finally	nonlocal	yield
break	for	not	class	form	or
continue	global	pass			

Tipovi podataka

- Svi tipovi podataka su objekti

Tipovi podataka

- Svi tipovi podataka su objekti
- Ne specificira se tip prilikom definisanja promenljive

Tipovi podataka

- Svi tipovi podataka su objekti
- Ne specificira se tip prilikom definisanja promenljive
- Pajton ima nekoliko ugrađenih tipova podataka

Tipovi podataka

- Svi tipovi podataka su objekti
- Ne specificira se tip prilikom definisanja promenljive
- Pajton ima nekoliko ugrađenih tipova podataka
  - Tekstualni tip (*str*)
- Tip se može proveriti pozivom funkcije *type*

Tipovi podataka

- Svi tipovi podataka su objekti
- Ne specificira se tip prilikom definisanja promenljive
- Pajton ima nekoliko ugrađenih tipova podataka
  - Tekstualni tip (*str*)
  - Numerički tip (*int*, *float*, *complex*)
- Tip se može proveriti pozivom funkcije *type*

Tipovi podataka

- Svi tipovi podataka su objekti
- Ne specificira se tip prilikom definisanja promenljive
- Pajton ima nekoliko ugrađenih tipova podataka
  - Tekstualni tip (*str*)
  - Numerički tip (*int*, *float*, *complex*)
  - Istinitosni/logički tip (*bool*)
- Tip se može proveriti pozivom funkcije *type*

Tipovi podataka

- Svi tipovi podataka su objekti
- Ne specificira se tip prilikom definisanja promenljive
- Pajton ima nekoliko ugrađenih tipova podataka
  - Tekstualni tip (*str*)
  - Numerički tip (*int*, *float*, *complex*)
  - Istinitosni/logički tip (*bool*)
  - Kolekcije (*list*, *tuple*)
- Tip se može proveriti pozivom funkcije *type*

Tipovi podataka

- Svi tipovi podataka su objekti
- Ne specificira se tip prilikom definisanja promenljive
- Pajton ima nekoliko ugrađenih tipova podataka
  - Tekstualni tip (*str*)
  - Numerički tip (*int*, *float*, *complex*)
  - Istinitosni/logički tip (*bool*)
  - Kolekcije (*list*, *tuple*)
  - Rečnici (*dict*)
- Tip se može proveriti pozivom funkcije *type*

Tipovi podataka

- Svi tipovi podataka su objekti
- Ne specificira se tip prilikom definisanja promenljive
- Pajton ima nekoliko ugrađenih tipova podataka
  - Tekstualni tip (*str*)
  - Numerički tip (*int*, *float*, *complex*)
  - Istinitosni/logički tip (*bool*)
  - Kolekcije (*list*, *tuple*)
  - Rečnici (*dict*)
  - Skupovi (*set*)
- Tip se može proveriti pozivom funkcije *type*

Numerički tip podataka

- Može biti *int*, *float* ili *complex*

Numerički tip podataka

- Može biti *int*, *float* ili *complex*
- Celobrojne vrednosti se čuvaju kao *int* i nemaju ograničenje

Numerički tip podataka

- Može biti *int*, *float* ili *complex*
- Celobrojne vrednosti se čuvaju kao *int* i nemaju ograničenje

```
# Celobrojna vrednost
x = 1234
y = 112412415125211512251
# _ za preglednost interpreter ga zanemaruje
z = 1_000_000_000_000_000
```

Numerički tip podataka

- Može biti *int*, *float* ili *complex*
- Celobrojne vrednosti se čuvaju kao *int* i nemaju ograničenje
- Realni brojevi se čuvaju kao *float* u IEEE 754 64-bitnom formatu

Numerički tip podataka

- Može biti *int*, *float* ili *complex*
- Celobrojne vrednosti se čuvaju kao *int* i nemaju ograničenje
- Realni brojevi se čuvaju kao *float* u IEEE 754 64-bitnom formatu

```
# Realna vrednost
x = 1.23241
PI = 3.141592653589793
# Moguce je zapisivati brojeve u naucnoj notaciji
a = 8E124
b = -32.1e5
# Beskonacno
c = float('inf')
```

Numerički tip podataka

- Može biti *int*, *float* ili *complex*
- Celobrojne vrednosti se čuvaju kao *int* i nemaju ograničenje
- Realni brojevi se čuvaju kao *float* u IEEE 754 64-bitnom formatu
- Kompleksni brojevi se predstavljaju u formi a+bj

Numerički tip podataka

- Može biti *int*, *float* ili *complex*
- Celobrojne vrednosti se čuvaju kao *int* i nemaju ograničenje
- Realni brojevi se čuvaju kao *float* u IEEE 754 64-bitnom formatu
- Kompleksni brojevi se predstavljaju u formi a+bj

```
# Kompleksna vrednost
x = -1j
a = 4-2j
```

Stringovi

- Predstavljaju objekte klase *str*

Stringovi

- Predstavljaju objekte klase *str*
- Predstavljaju nepromenljive tipove podataka

Stringovi

- Predstavljaju objekte klase *str*
- Predstavljaju nepromenljive tipove podataka
- Za definiciju stringova moguće je koristiti jednostuke ili trostruke apostrofe i navodnike

Stringovi

- Predstavljaju objekte klase *str*
- Predstavljaju nepromenljive tipove podataka
- Za definiciju stringova moguće je koristiti jednostuke ili trostruke apostrofe i navodnike
- Jednostruki se korsite za linijske stringove, trostuki se korsite za višelinijske stringove

Stringovi

- Predstavljaju objekte klase *str*
- Predstavljaju nepromenljive tipove podataka
- Za definiciju stringova moguće je koristiti jednostuke ili trostruke apostrofe i navodnike
- Jednostruki se korsite za linijske stringove, trostuki se korsite za višelinijske stringove

```
# Jednolinijski string
a = 'Jednolinijski string 1'
b = "Jednolinijski string 2"
c = ''' Viselinijski
string 1'''
d = """ Viselinijski
string 2"""
```

Escape sekvenca

sekvenca	rezultat
\\	backslash (\)
\'	apostrof
\"	navodnici
\n	novi red

Manipulacija stringovima

- Funckija *len* vraća dužinu stringa

Manipulacija stringovima

- Funckija *len* vraća dužinu stringa

```
a = 'Ovo je test string'
print(len(a)) # 18
```

Manipulacija stringovima

- Funckija *len* vraća dužinu stringa
- Pristup pojedinačnim elementima stringa

Manipulacija stringovima

- Funckija *len* vraća dužinu stringa
- Pristup pojedinačnim elementima stringa

```
1 a = 'Ovo je test string'
2 print(a[4]) # j
3 print(a[16]) # n
4 print(a[-2]) # n
```

Manipulacija stringovima

- Funckija *len* vraća dužinu stringa
- Pristup pojedinačnim elementima stringa
- Prsitup delovima stringa string[start, stop, korak]

Manipulacija stringovima

- Funckija *len* vraća dužinu stringa
- Pristup pojedinačnim elementima stringa
- Prsitup delovima stringa string[start, stop, korak]

```
a = 'Ovo je test string'
print(a[7:11]) # test
print(a[-6:]) # string
print(a[::2]) # Ooj etsrn
```

Manipulacija stringovima

- Funckija *len* vraća dužinu stringa
- Pristup pojedinačnim elementima stringa
- Prsitup delovima stringa string[start, stop, korak]

```
1 a = 'Ovo je test string'
2 print(a[7:11]) # test
3 print(a[-6:]) # string
4 print(a[::2]) # Ooj etsrn
```

- Ako se ne navede start podrazumevana vrednost je start stringa, za stop je kraj stringa, a za korak je jedan

Indeksi stringa

0	1	2	3	4	5	6	7	8	9	10	11	12
O	V	O		J	E		S	T	R	I	N	G
-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

Logički tip

- Predstavlja objekat klase *bool*

Logički tip

- Predstavlja objekat klase *bool*
- Može imati stanja tačno (*True*) i netačno (*False*)

Logički tip

- Predstavlja objekat klase *bool*
- Može imati stanja tačno (*True*) i netačno (*False*)
- Kod numeričkih tipova sve osim nule je tačno, a nula je netačno

Logički tip

- Predstavlja objekat klase *bool*
- Može imati stanja tačno (*True*) i netačno (*False*)
- Kod numeričkih tipova sve osim nule je tačno, a nula je netačno
- Kod drugih tipova sve što je neprazno je tačno, a prazno je netačno

Logički tip

- Predstavlja objekat klase *bool*
- Može imati stanja tačno (*True*) i netačno (*False*)
- Kod numeričkih tipova sve osim nule je tačno, a nula je netačno
- Kod drugih tipova sve što je neprazno je tačno, a prazno je netačno

```
a = bool('') # False
b = bool(0) # False
c = bool(15.2) # True
d = bool('bla') # True
```

Aritmetički operatori

operator	značenje
+	sabiranje
-	oduzimanje
*	množenje
/	deljenje
%	moduo
**	stepenovanje
//	celobrojno deljenje

Dodavanjem = uz operator dobija se bočni efekat +=, -= ...

Korišćenje aritmetičkih operatora

```
a = 4
b = 5

print(a+b)# 9
print(a-b)# -1
print(a*b)# 20
print(a/b)# 0.8
print(b//a)# 1
print(b**a)# 625
print(b%a)# 1

print(a)# 4
a+=b
print(a)# 9
```

Dodatne napomene

- Ako se kombinuju različiti numerički tipovi uvek će se konvertovati u viši rang

Dodatne napomene

- Ako se kombinuju različiti numerički tipovi uvek će se konvertovati u viši rang
- Najviši rang ima *complex*, pa *float*, pa *int*

Dodatne napomene

- Ako se kombinuju različiti numerički tipovi uvek će se konvertovati u viši rang
- Najviši rang ima *complex*, pa *float*, pa *int*

```
a = 4
b = -3.25
c = 1+2j

d = a*a
print(d)# 16
print(type(d))# <class 'int'>
d = a*b
print(d)# -13.0
print(type(d))# <class 'float'>
d *= c
print(d)# (-13-26j)
print(type(d))# <class 'complex'>
```

Dodatne napomene

- Ako se kombinuju različiti numerički tipovi uvek će se konvertovati u viši rang
- Najviši rang ima *complex*, pa *float*, pa *int*
- Operatorima je moguće vršiti manipulaciju nad stringovima

Dodatne napomene

- Ako se kombinuju različiti numerički tipovi uvek će se konvertovati u viši rang
- Najviši rang ima *complex*, pa *float*, pa *int*
- Operatorima je moguće vršiti manipulaciju nad stringovima

```
a = 'Na'
b = 'Bat'
c = 'man'

print(a*10)# NaNaNaNaNaNaNaNaNaNa
print(b+c)# Batman
print(a*10+' '+b+c)# NaNaNaNaNaNaNaNaNaNa Batman
```

Promenljive i memorija

```
Ime = 'Pera'
ID = 15
```

Promenljive i memorija

```
1  Ime = 'Pera'
2  ID = 15
```

Relacioni operatori

operator	značenje
<	manje
>	veće
=<	manje ili jednako
=>	veće ili jednako
==	provera jednakosti
!=	provera nejednakosti

Logički operatori

operator	značenje
and	i
or	ili
not	negacija

Relacioni operatori primeri

```
a = 12.4
b = -1
c = 12.7

print(a < b)# False
print(c > b)# True
print(c == b)# False
print(a != c)# True
print(c >= b)# True
print(a <= c)# True
```

Logički operatori primeri

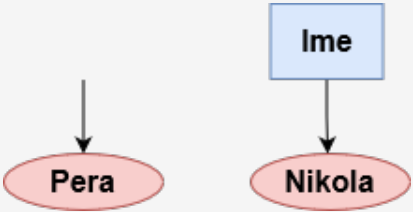
```
a = False
b = True
c = bool(0)

print(a and b)# False
print(c or b)# True
print(not a)# True
```

Promenljive i memorija

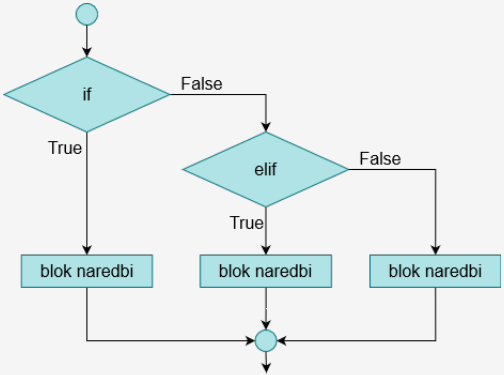
```
1  Ime = 'Pera'
2  Ime = 'Nikola'
```

Promenljive i memorija

```
1  Ime = 'Pera'
2  Ime = 'Nikola'
```

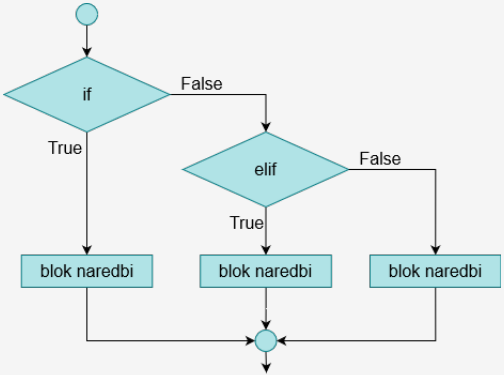


Naredba *if*



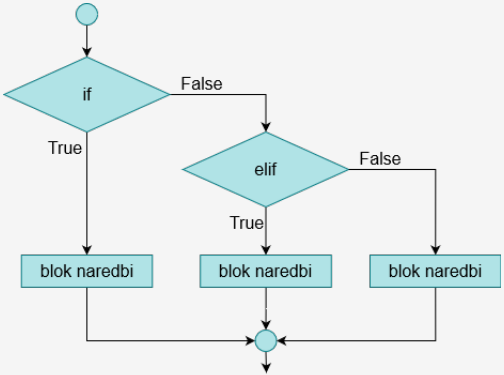
- Ne pišu se zagrade oko uslova

Naredba *if*



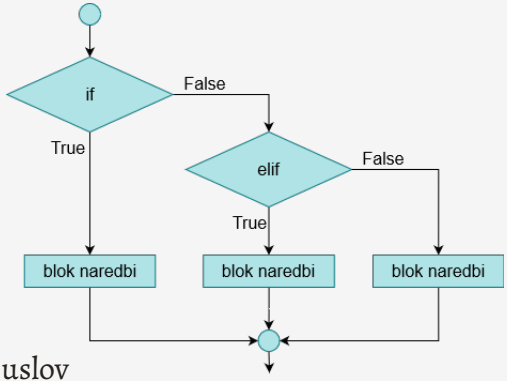
- Ne pišu se zagrade oko uslova
- Dvotačka pre bloka naredbi

Naredba *if*



- Ne pišu se zagrade oko uslova
- Dvotačka pre bloka naredbi
- Blok naredbi se uvlači

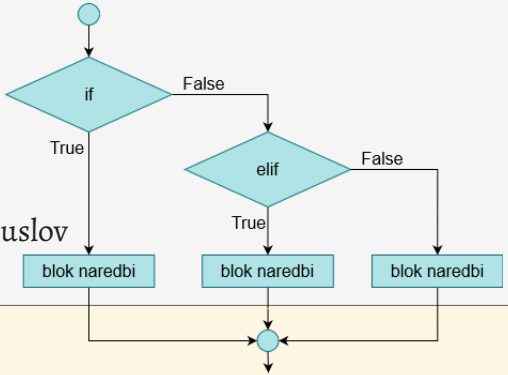
Naredba *if*



- Ne pišu se zagrade oko uslova
- Dvotačka pre bloka naredbi
- Blok naredbi se uvlači
- Uz *if* moguće je dodati *else* koji nema uslov

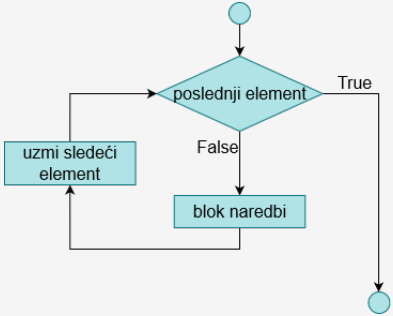
Naredba *if*

- Ne pišu se zagrade oko uslova
- Dvotačka pre bloka naredbi
- Blok naredbi se uvlači
- Uz *if* moguće je dodati *else* koji nema uslov
- Moguće je dodati i međuuslove *elif*



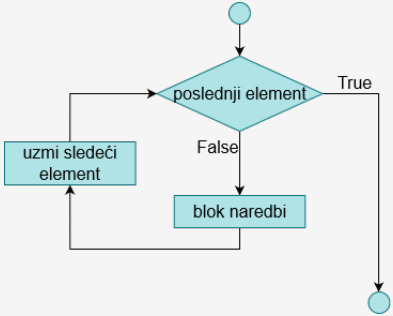

```
a = 5
b = 3
if a<b:
    printf('a je manje od b')
elif a>b:
    printf('a je vece od b')
else:
    printf('a i b su jednaki')
```

for petlja



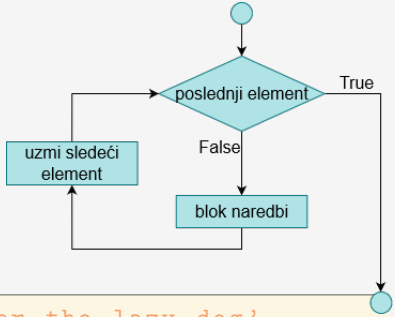
- Iterira se kroz bilo koju sekvencu

for petlja



- Iterira se kroz bilo koju sekvencu
- Sekvence mogu biti stringovi, liste, torke…

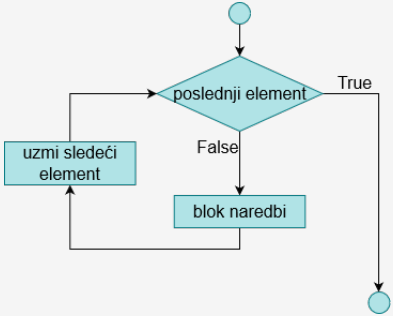
for petlja



- Iterira se kroz bilo koju sekvencu
- Sekvence mogu biti stringovi, liste, torke…

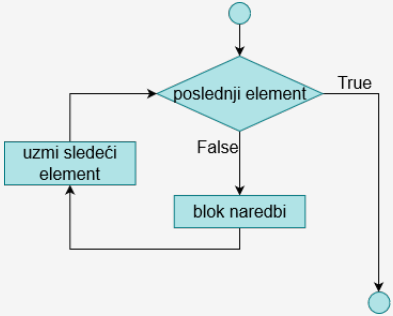
```
1  a = 'The quick brown fox jumps over the lazy dog'
2  for i in a:
3      print(i)
```

for petlja



- Iterira se kroz bilo koju sekvencu
- Sekvence mogu biti stringovi, liste, torke…
- Moguće je generisati sekvencu funkcijom *range*

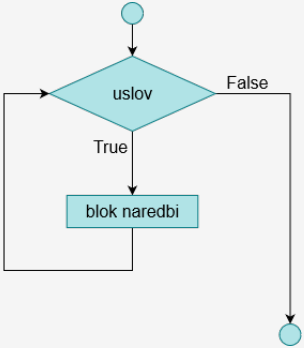
for petlja



- Iterira se kroz bilo koju sekvencu
- Sekvence mogu biti stringovi, liste, torke...
- Moguće je generisati sekvencu funkcijom *range*
- *range*(stop) ili *range*(start, stop, korak)

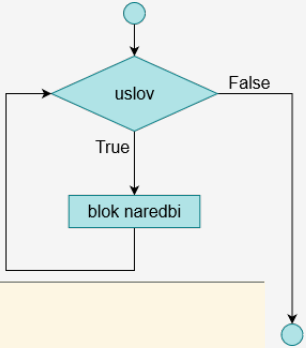
```
1  for i in range(0,11,2):
2      print(i)
3
4  for i in range(11):
5      print(i)
```

while petlja



- Izvršava se blok naredbi dok je uslov tačan

while petlja



- Izvršava se blok naredbi dok je uslov tačan
- Kada uslov postane netačan petlja završava rad

```
1  i=0
2  while i<10:
3      print(i)
4      i+=1
```

Upravljanje tokom petlje

- Pomoću *break* naredbe moguće je prekinuti izvršavanje petlje

Upravljanje tokom petlje

- Pomoću *break* naredbe moguće je prekinuti izvršavanje petlje

```
1  a = 'Lorem ipsum'
2  for i in a:
3    if i == 's':
4      break
5    print(i)
```

Upravljanje tokom petlje

- Pomoću *break* naredbe moguće je prekinuti izvršavanje petlje
- Pomoću *continue* naredbe moguće je preskočiti jednu iteraciju

```
1 i = 0
2 while i < 5:
3   i += 1
4   if i == 2:
5     continue
6   print(i)
```

Unos sa tastature

- Koristi se funkcija *input*

Unos sa tastature

- Koristi se funkcija *input*
- Kao parametar prenosi se poruka korisniku

Unos sa tastature

- Koristi se funkcija *input*
- Kao parametar prenosi se poruka korisniku
- Povratna vrednost funkcije je string unet sa tastature (pritiskom na enter završava se unos)

Unos sa tastature

- Koristi se funkcija *input*
- Kao parametar prenosi se poruka korisniku
- Povratna vrednost funkcije je string unet sa tastature (pritiskom na enter završava se unos)
- Neophodno je konvertovati string u odgovarajući tip

Unos sa tastature

- Koristi se funkcija *input*
- Kao parametar prenosi se poruka korisniku
- Povratna vrednost funkcije je string unet sa tastature (pritiskom na enter završava se unos)
- Neophodno je konvertovati string u odgovarajući tip

```
1 # I nacin
2 string = input('Koliko imas godina?')
3 godine = int(string)
4 # II nacin
5 godine = int(input('Koliko imas godina?'))
```

Ispis u konzolu

- Koristi se funkcija *print*

Ispis u konzolu

- Koristi se funkcija *print*
- Funkcija može primiti više parametara

Ispis u konzolu

- Koristi se funkcija *print*
- Funkcija može primiti više parametara

```
string = 'Ovo je'
print(string, 'test')
```

Ispis u konzolu

- Koristi se funkcija *print*
- Funkcija može primiti više parametara
- Podrazumevano se sa svakim pozivom izvrši ispis u zaseban red)

Ispis u konzolu

- Koristi se funkcija *print*
- Funkcija može primiti više parametara
- Podrazumevano se sa svakim pozivom izvrši ispis u zaseban red)
- Moguće je podesiti parametar *end* na vrednost poslednjeg karaktera umesto podrazumevanog novog reda (\n)

Ispis u konzolu

- Koristi se funkcija *print*
- Funkcija može primiti više parametara
- Podrazumevano se sa svakim pozivom izvrši ispis u zaseban red)
- Moguće je podesiti parametar *end* na vrednost poslednjeg karaktera umesto podrazumevanog novog reda (\n)

```
1 # Ispis u dva reda
2 print('Ovo je')
3 print('test')
4 # Ispis u jednom redu
5 print('Ovo je', end=' ')
6 print('test')
```

Hvala na pažnji!