2. RADIONICA

LIGA PROGRAMIRANJA

python #5

Dubrovnik, 5. prosinca 2023.

zajednica tehničke kulture Dubrovnik

INFORMATIČKI KLUB FUTURA

SVEUČILIŠTE U DUBROVNIKU
UNIVERSITY OF DUBROVNIK

Creative Commons

- **slobodno smijete:**
  - **dijeliti** — umnožavati, distribuirati i javnosti priopćavati djelo
  - **remiksirati** — prerađivati djelo
- **pod slijedećim uvjetima:**
  - **imenovanje**. Morate priznati i označiti autorstvo djela na način kako je specificirao autor ili davatelj licence (ali ne način koji bi sugerirao da Vi ili Vaše korištenje njegova djela imate njegovu izravnu podršku).
  - **nekomercijalno**. Ovo djelo ne smijete koristiti u komercijalne svrhe.
  - **dijeli pod istim uvjetima**. Ako ovo djelo izmijenite, preoblikujete ili stvarate koristeći ga, preradu možete distribuirati samo pod licencom koja je ista ili slična ovoj.

U slučaju daljnjeg korištenja ili distribuiranja morate drugima jasno dati do znanja licencne uvjete ovog djela. Najbolji način da to učinite je linkom na ovu internetsku stranicu.

Od svakog od gornjih uvjeta moguće je odstupiti, ako dobijete dopuštenje nositelja autorskog prava.

Ništa u ovoj licenci ne narušava ili ograničava autorova moralna prava.

Tekst licence preuzet je s http://creativecommons.org/.

Raspored Lige programiranja

- 05.12.2023. – **3. radionica**
- 19.12.2023. – **2. kolo Lige programiranja**
- Ostali termini u 2024. godini bit će oglašeni na stranicama Lige programiranja
- Web stranica Lige programiranja:

https://www.futura.com.hr/liga-programiranja-u-pythonu-2023-2024/

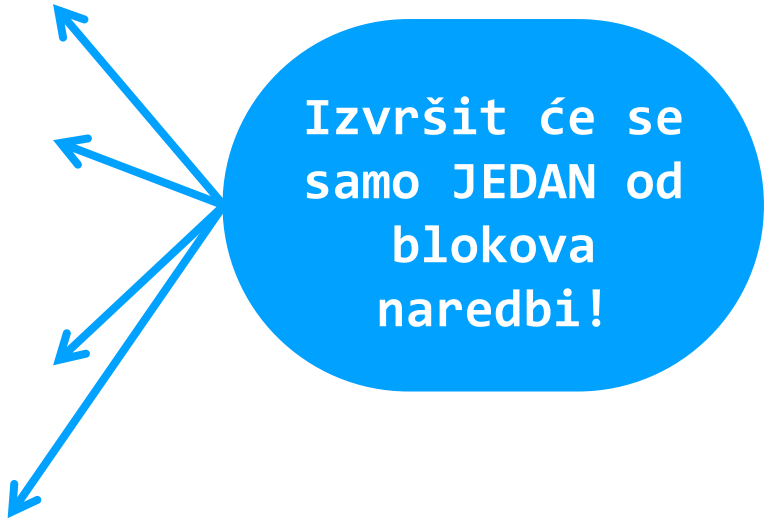
Pravila Lige programiranja

- Ekipno natjecanje škola **5./6. razredi**
- Ekipu škole **5./6. razredi** čine 2 ili 3 učenika iz 5. ili 6. razreda
- Za rezultate ekipa se uzimaju u obzir bodovi 2 najbolja učenika te ekipe za svako kolo lige
- Pojedinačno natjecanje **5./6.** i **7./8.**
- Nagrade najboljim ekipama i najboljim pojedincima u kategorijama 5./6. i 7./8. razredi

Ponavljanje gradiva s 1. radionice

- Donošenje odluka (grananje programa) - u slučaju višestrukog izbora:

```
...
if uvjet_1:
      blok naredbi_1
elif uvjet_2:
      blok  naredbi_2
...
elif uvjet_n:
      blok  naredbi_n
else:
      blok naredbi
 ...
```



Unos s tipkovnice

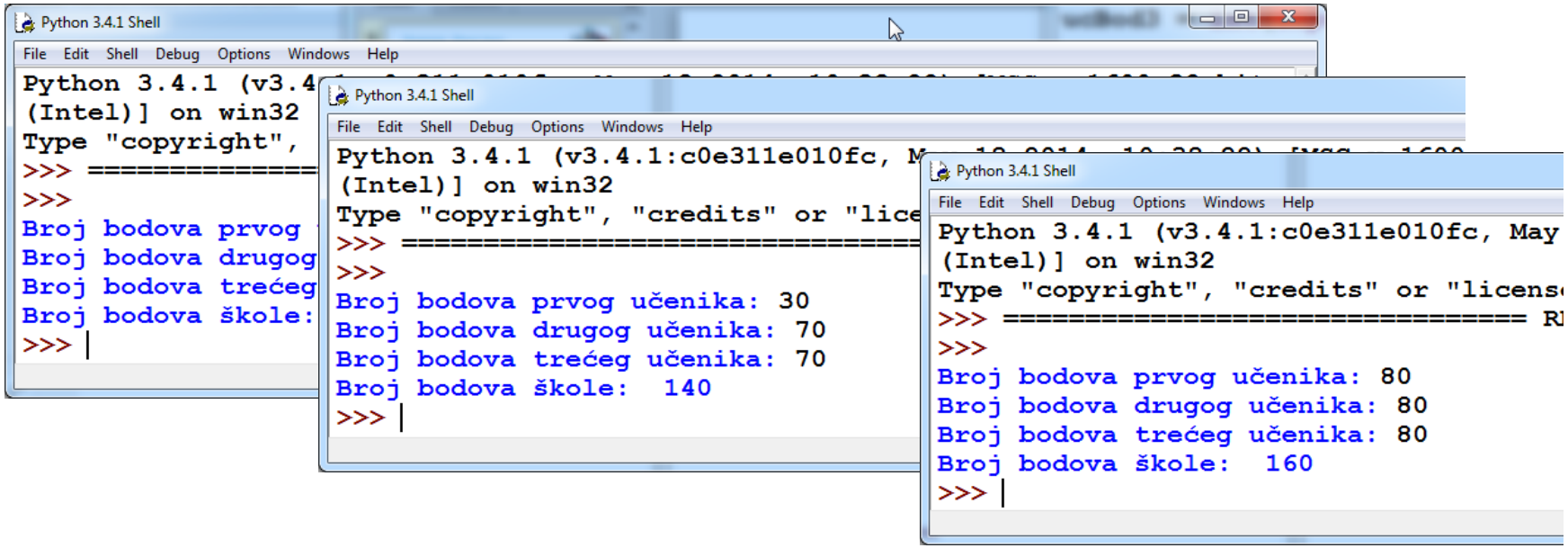
- Funkcija **input**
-
```
>>> ime = input('Unesi svoje ime: ')
Unesite vaše ime: Tomo
>>> print('Uneseno ime je:', ime)
Uneseno ime je: Tomo
```

- Funkcija **input** sve što se unese s tipkovnice sprema kao niz znakova
- Primjer funkcije **input** s cijelim brojem
-
```
>>> broj = int(input('Unesi cijeli broj: ‘))
```

Zadatak: Bodovi

- Iz pravila: "*Za rezultate ekipa se uzimaju u obzir bodovi 2 najbolja učenika te ekipe za svako kolo lige.*"

  Napisati program kojim će se unijeti rezultati za tri učenika, a onda izračunati koliko ekipa ima bodova.



Zadatak: Bodovi

- Rješenje:

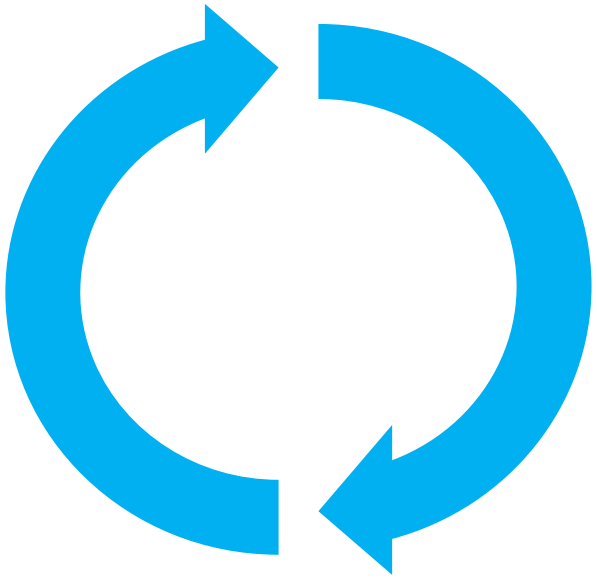
```
ucBod1 = int(input('Broj bodova prvog učenika: '))
ucBod2 = int(input('Broj bodova drugog učenika: '))
ucBod3 = int(input('Broj bodova trećeg učenika: '))
skolaBodovi=0

treci=ucBod1
if ucBod2 <= treci:
    treci = ucBod2
if ucBod3 <= treci:
    treci = ucBod3

skolaBodovi = ucBod1 + ucBod2 + ucBod3 - treci
print ('Broj bodova škole: ', skolaBodovi)
```

bodovi.py

Ponavljanje bloka naredbi

- Često je u programima potrebno određeni broj puta ponoviti blok istih naredbi:

```
...
za i := 1 do n činiti
      naredba_1
      ...
      naredba_z
...
```

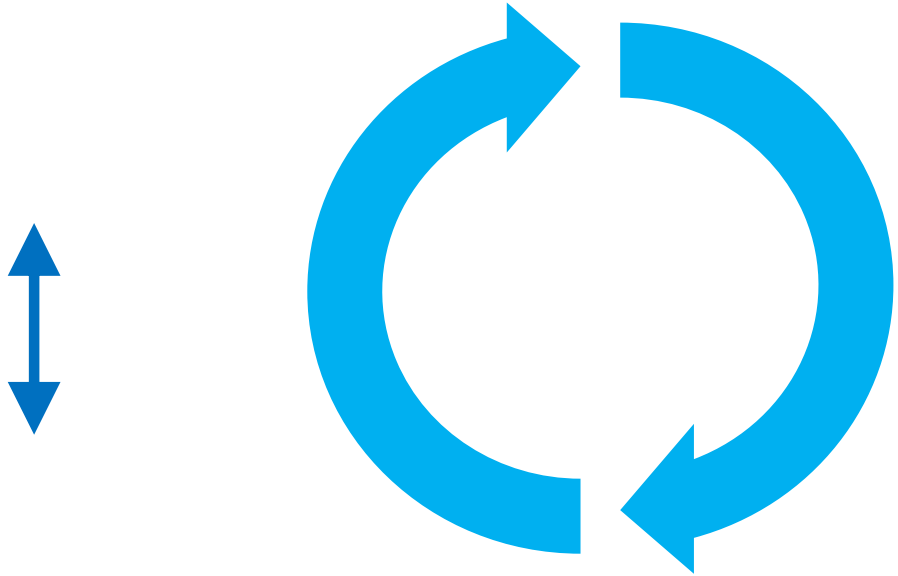


Programska petlja

Ponavljanje bloka naredbi

- U Pythonu:

```
...
for i in range(n):
        naredba_1
        ...
        naredba_z
...
```

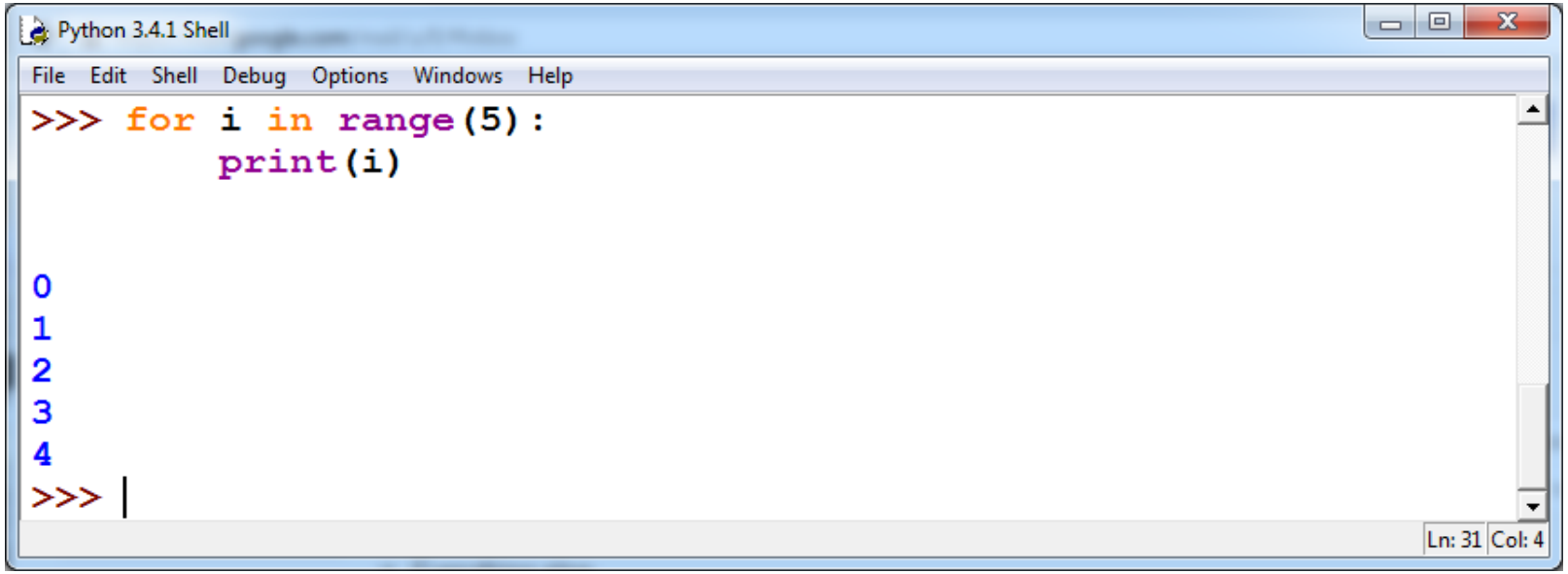


Programska petlja

Blok naredbi će se izvesti **n puta**, za vrijednosti varijable **i** od **0** do **n-1**.

Ponavljanje bloka naredbi

- Provjeriti u interaktivnom sučelju (*shell*):



Blok naredbi će se izvesti **5 puta**, za sve vrijednosti varijable **i** od **0** do **4**.

Ponavljanje bloka naredbi

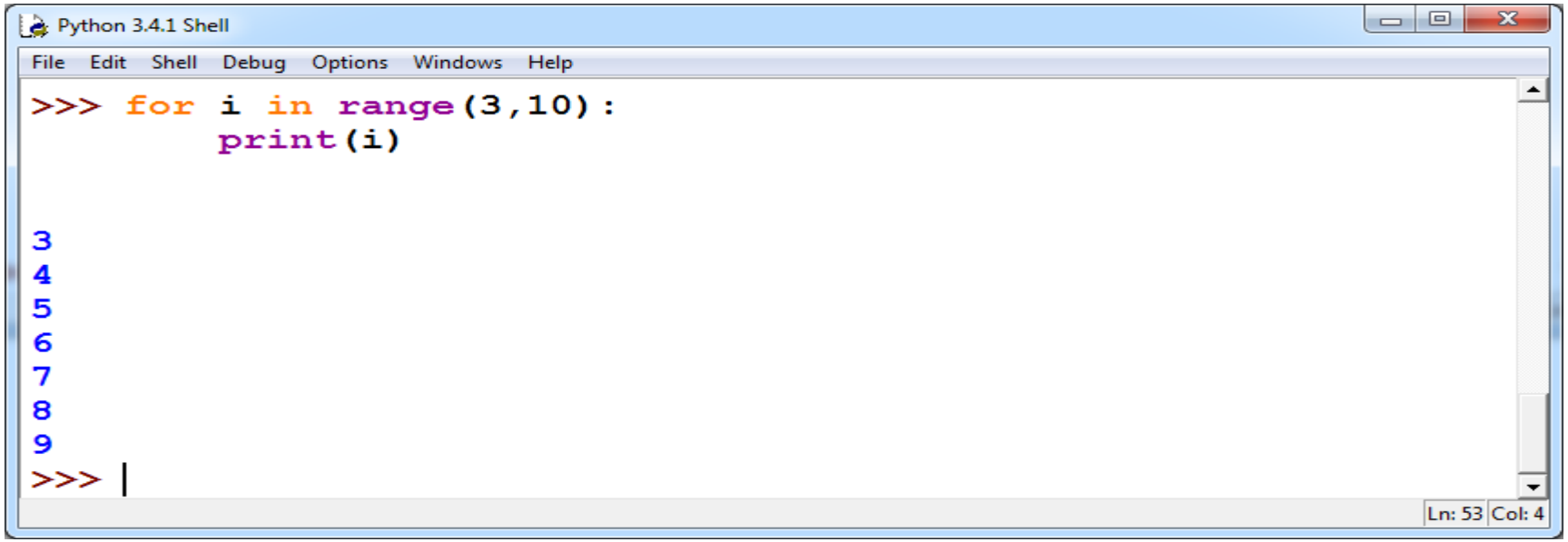
- Ako nam ne odgovara da vrijednost varijable **i** kreće od **0**:

```
...
for i in range(n,m):
      naredba_1
      ...
      naredba_z
...
```

Blok naredbi će se izvesti **m-n puta**, za sve vrijednosti varijable **i** od **n** do **m-1**.

Ponavljanje bloka naredbi

- Provjeriti u interaktivnom sučelju (*shell*):



- Blok naredbi će se izvesti **7 puta**, za sve vrijednosti varijable **i** od **3** do **9**

Ponavljanje bloka naredbi

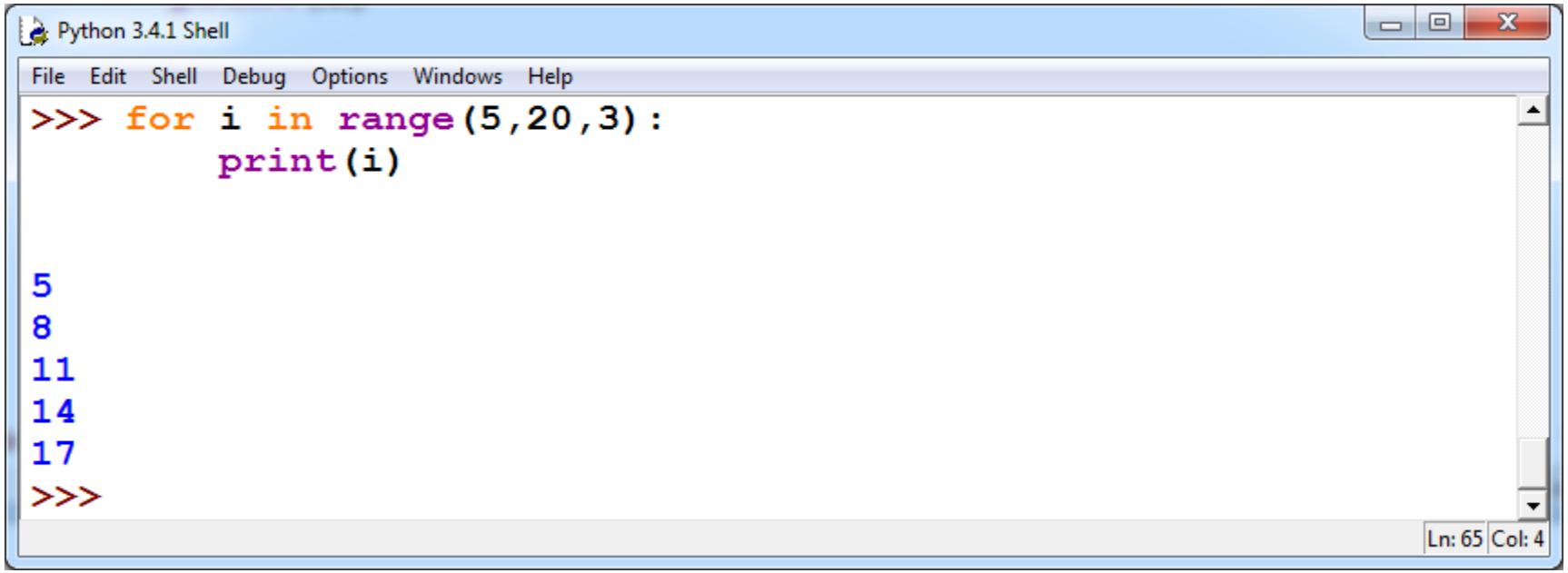
- Isto tako možda nam ne odgovara da se vrijednost varijable **i** mijenja u koracima po **1**:

```
...
for i in range(n,m,k):
        naredba_1
        ...
        naredba_z
...
```

Blok naredbi će se izvesti **m-n puta**, za vrijednosti varijable **i** od **n** do **m-1**, ali će se **i** svaki put uvećati za **k**.

Ponavljanje bloka naredbi

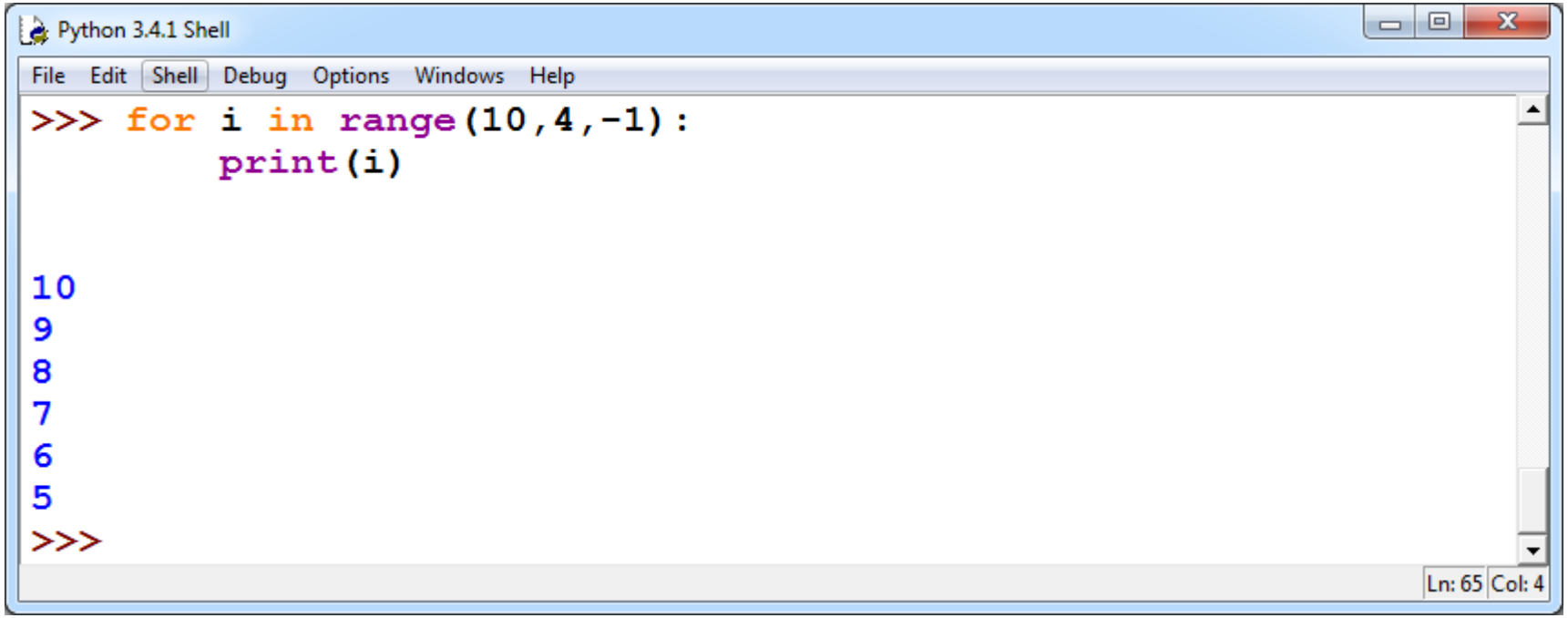
- Provjeriti u interaktivnom sučelju (*shell*):



Varijabla **i** mijenja vrijednost od **5** do **19** (tj. 20-1), a korak promjene je **3**.

Ponavljanje bloka naredbi

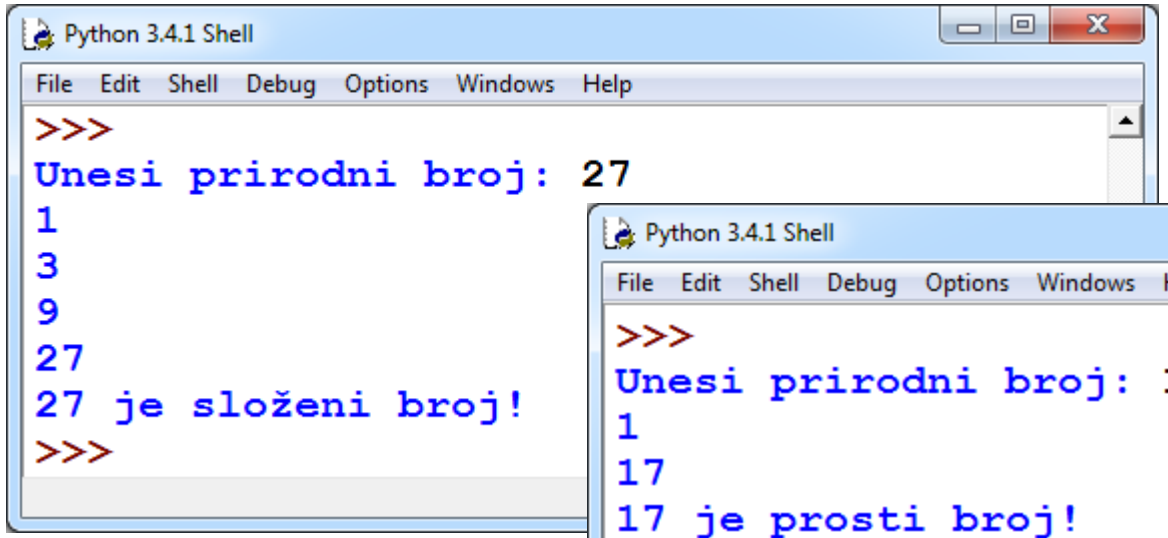
- Može li korak biti negativan broj?

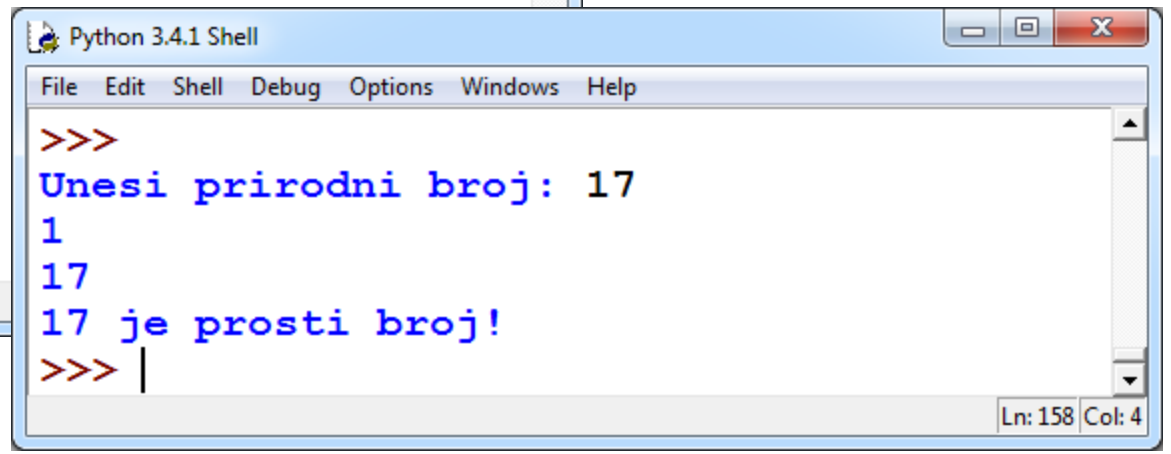


Varijabla **i** mijenja vrijednost od **10** do **5** (tj. 4+1), a korak promjene je **-1**.

Zadatak: pronađi djelitelje!

- Unijeti prirodni broj N i ispisati sve njegove djelitelje.
- Ako broj N nema djelitelja osim 1 i N, ispisati: "N je prosti broj". Inače ispisati: "N je složeni broj"





Zadatak: pronađi djelitelje!

- Rješenje:

```
N = int(input('Unesi prirodni broj: '))

brDjel = 0

for i in range(1,N+1):
    if N%i == 0:
        print(i)
        brDjel = brDjel + 1

if brDjel == 2:
    print(N,'je prosti broj!')
else:
    print(N,'je složeni broj!')
```

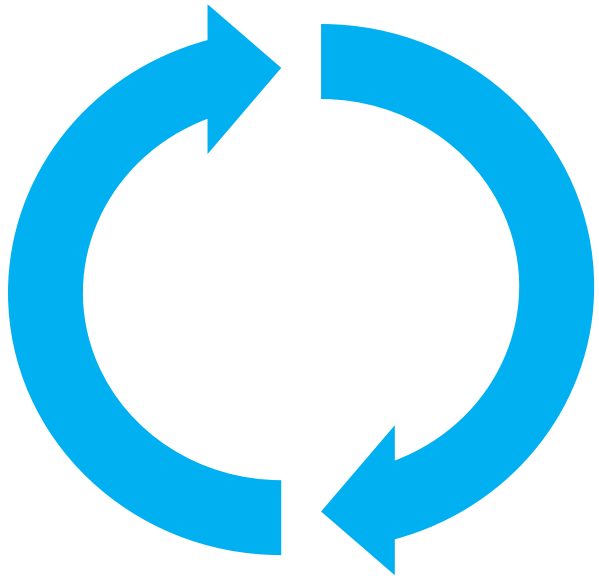
Može li se program napisati tako da ima manje od *N prolaza* kroz petlju?

djelitelji.py

Uvjetno ponavljanje bloka naredbi

- Ponekad je u programima potrebno određeni broj puta ponoviti blok istih naredbi, ali samo dok je neki uvjet ispunjen:

```
...
dok je uvjet činiti
        naredba_1
        ...
        naredba_z
...
```

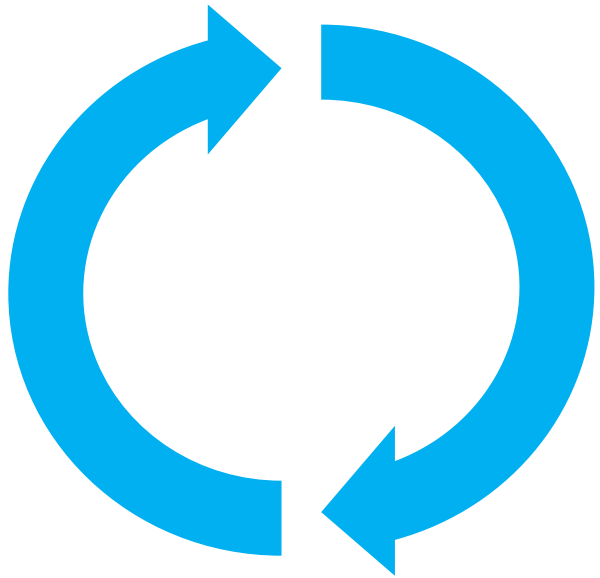


Programska petlja

Uvjetno ponavljanje bloka naredbi

- U Pythonu:

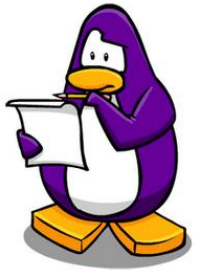
```
...
while uvjet:
      naredba_1
      ...
      naredba_i
...
```



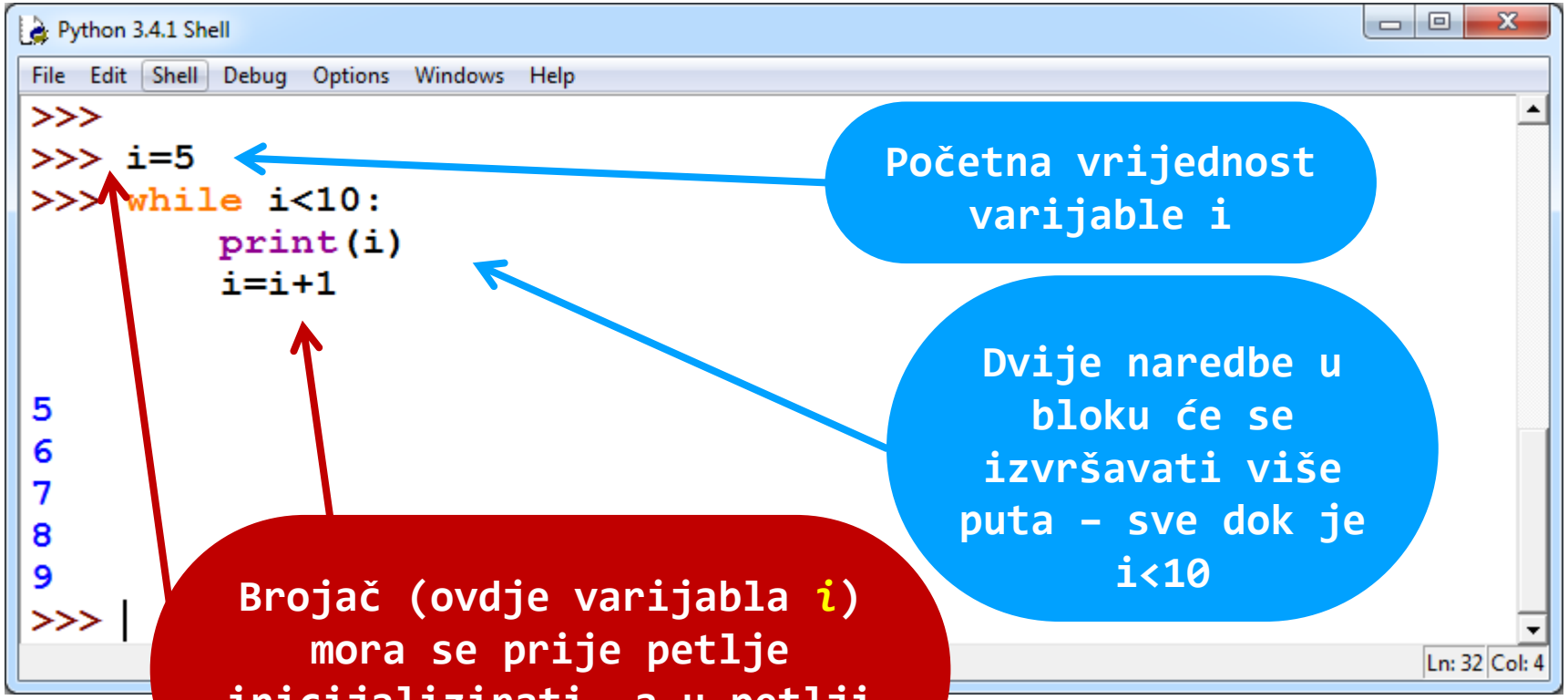
Programska petlja

Blok naredbi će se izvoditi dok je ***uvjet*** ispunjen (daje vrijednost True).

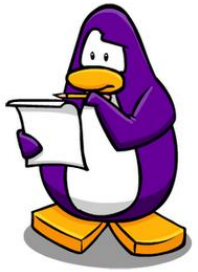
Uvjetno ponavljanje bloka naredbi



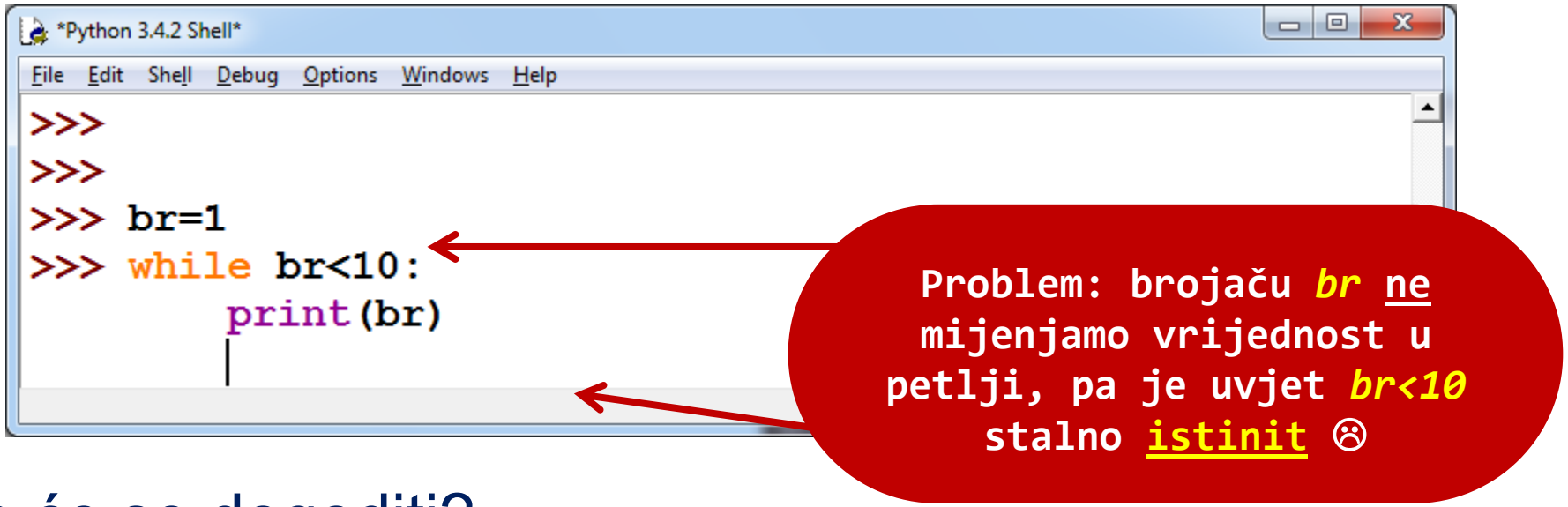
- Provjeriti u interaktivnom sučelju (*shell*):



Uvjetno ponavljanje bloka naredbi



- Treba paziti da se ne napiše "beskonačna" petlja ☹ :

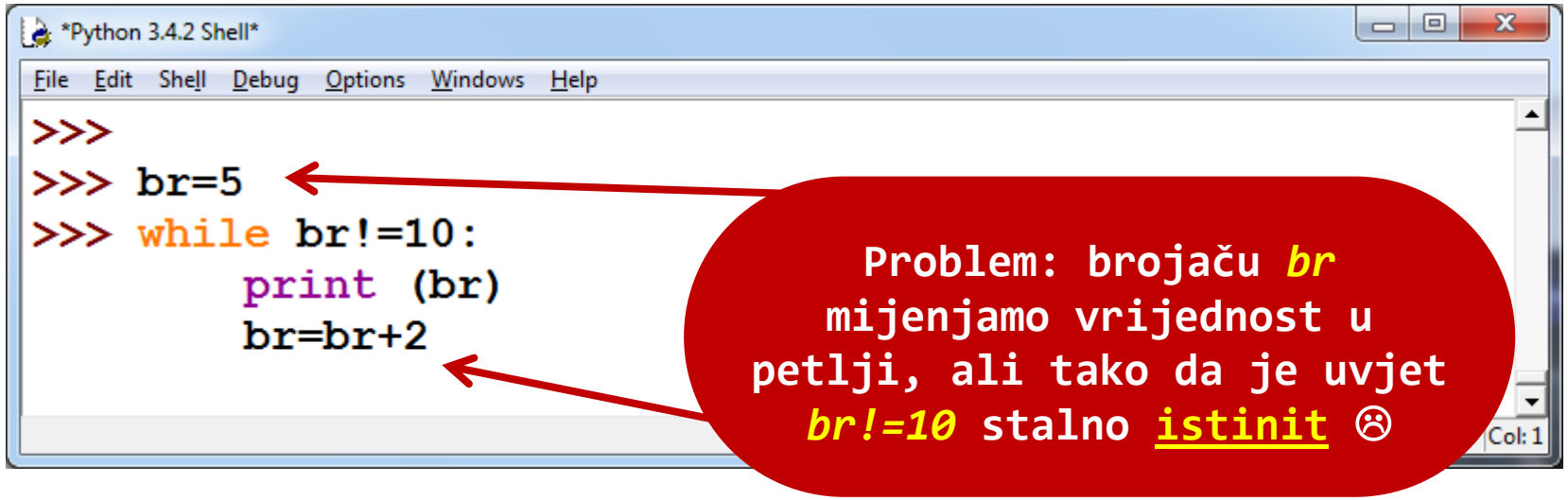


Što će se dogoditi?

Napisali smo "beskonačnu" petlju, pa je moramo prekinuti istovremenim pritiskom na **Ctrl C**

Uvjetno ponavljanje bloka naredbi

- Treba paziti da se ne napiše "beskonačna" petlja ☹ :



Što će se dogoditi?

Napisali smo "beskonačnu" petlju, pa je moramo prekinuti istovremenim pritiskom na **Ctrl C**

Programske petlje - zadatak

- ***for*** i ***while*** programske petlje često se koriste kada je potrebno unijeti više podataka, ali u vrijeme pisanja programa nije poznat njihov broj.
- Primjer: potrebno je napisati program za izračun prosječne ocjene iz nekog predmeta, ali broj ocjena koje se uzimaju u obzir može biti različit.

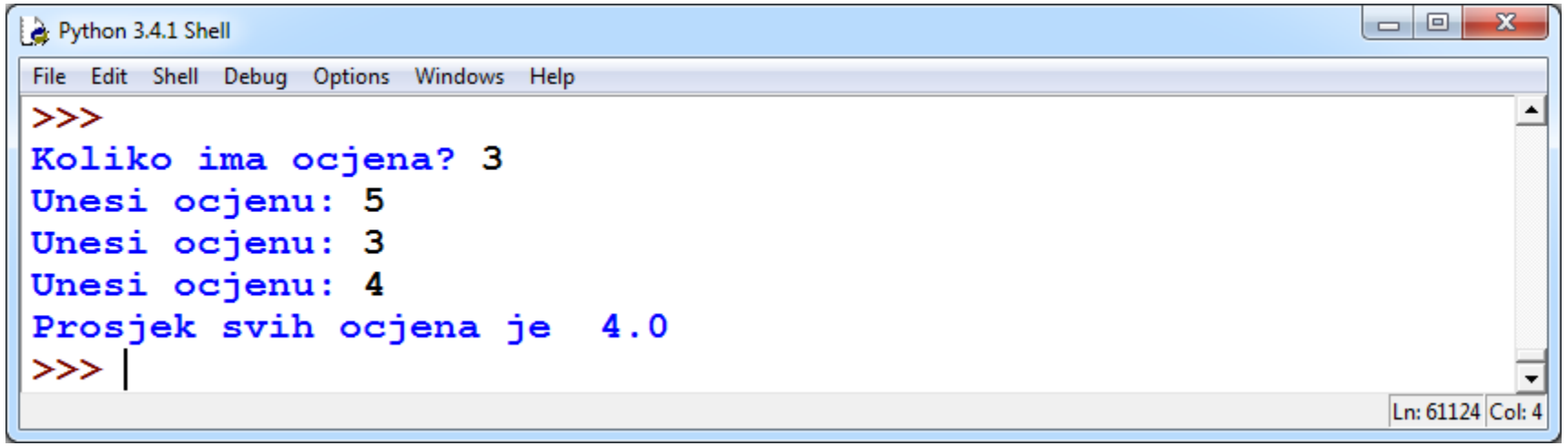
  U ovom programu moramo koristiti ***for*** ili ***while*** programsku petlju!

Programske petlje - zadatak

- Rješenje – pomoću ***for*** petlje:

prosjek1.py

```
brOcj = int(input('Koliko ima ocjena? '))
zbroj = 0
for i in range (1,brOcj+1):
    ocjena = int(input('Unesi ocjenu: '))
    zbroj = zbroj+ocjena

print ('Prosjek svih ocjena je ', zbroj/brOcj)
```



Programske petlje - zadatak

- Rješenje – pomoću ***while*** petlje. Moramo se dogovoriti koji podatak prekida petlju (npr.**0**)

```
zbroj = 0
brOcj = 0
ocjena = int(input('Unesi ocjenu (0 za kraj): '))
while ocjena != 0:
    brOcj = brOcj + 1
    zbroj = zbroj + ocjena
    ocjena = int(input('Unesi ocjenu (0 za kraj): '))
print ('Prosjek svih ocjena je ',zbroj/brOcj)
```

prosjek2.py

Programske petlje - zadatak

- Kako spriječiti unos pogrešnih podataka?

```
brOcj = 0
while brOcj < 1:
    brOcj=int(input('Koliko ima ocjena? '))
zbroj = 0
ocjena = 0
for i in range (1,brOcj+1):
    while ocjena < 1 or ocjena > 5:
        ocjena = int(input('Unesi ocjenu: '))
        if ocjena < 1 or ocjena > 5:
            print('Neispravna ocjena')
    zbroj = zbroj + ocjena
    ocjena = 0
print ('Prosjek svih ocjena je ', zbroj/brOcj)
```

Ne može se unijeti 0 ili negativan broj

prosjek1provjera.py

Ne može se unijeti neispravna ocjena

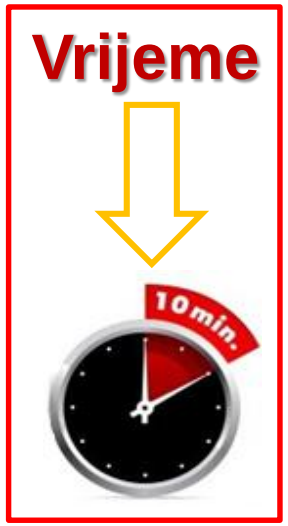
Zadatak: Trening

- Andro trenira nogomet, pa da bi bio u dobroj formi svako jutro prije škole trči oko svoje zgrade. Kako ne bi zakasnio u školu odlučio je stati s trčanjem nakon što mu za neki krug treba više od 60 sekundi.
- Napisati program u kojem se unosi vrijeme za svaki krug redom, a nakon posljednjeg kruga se ispisuje broj pretrčanih krugova, i ukupno vrijeme u minutama i sekundama.

Zadatak: Trening – testni podaci

- Andro prestaje trčati nakon što mu za neki krug treba više od 60 sekundi

Ulaz 50 50 80	**Ulaz** 30 40 50 60 70	**Ulaz** 40 50 50 60 60 65	**Ulaz** 65
Izlaz 3 3 min 0 s	**Izlaz** 5 4 min 10 s	**Izlaz** 6 5 min 25 s	**Izlaz** 1 1 min 5 s



Zadatak: Trening

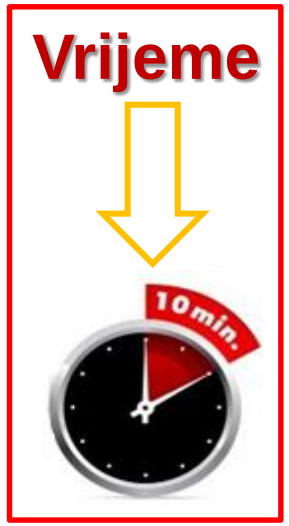
- Rješenje:

```
ukSek = 0
brKrug = 0
sekKrug = 0

while sekKrug <= 60:
        sekKrug=int(input('Unesi vrijeme za krug: '))
        ukSek = ukSek + sekKrug
        brKrug = brKrug + 1

print('Broj pretrčanih krugova: ',brKrug)
print('Adro je trčao',ukSek//60,'min i',ukSek%60,'s')
```

trening.py

Zadatak: Lift

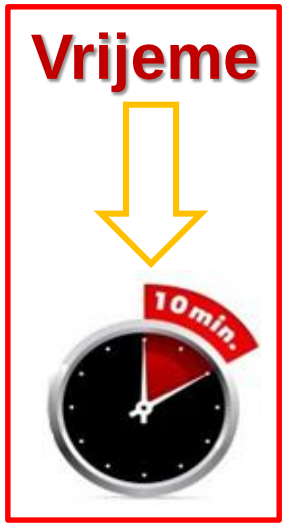
- Futura je počela radionice održavati na 10. katu svog nebodera. Zato učenici često koriste lift. Nije ograničen broj učenika u liftu, ali je težina ograničena na najviše 200 kg.
- Potrebno je upisati broj učenika koji čekaju ispred lifta. Zatim je potrebno za svakog od učenika upisati njegovu težinu koja mora biti barem 30 kg, odnosno ne preko 100kg. Učenici ulaze redom u lift, ako po svojoj težini smiju ući.
- Ispisati koliko je učenika na kraju ušlo u lift, kao i kolika je njihova ukupna težina.



Zadatak: Lift – testni podaci

- ukup. težina učenika u liftu ne može biti preko 200kg
- težina učenika mora biti barem 30 kg, odnosno ne veća od 100kg

Ulaz	**Ulaz**	**Ulaz**	**Ulaz**
4 80 70 60 40	5 60 100 50 90 20 110 40	5 25 60 30 40 80 60	6 40 50 50 70 35 110 50
Izlaz	**Izlaz**	**Izlaz**	**Izlaz**
3 190	3 200	4 190	4 175



Zadatak: Lift

- Rješenje:

lift.py

```
brUcRed = int(input('Koliko ima učenika? '))
tezUc = 0
zbrTez = 0
brUcLift = 0
for i in range (1,brUcRed+1):
    while tezUc < 30 or tezUc > 100:
        tezUc = int(input('Unesi težinu učenika: '))
    if zbrTez+tezUc <= 200:
        zbrTez = zbrTez+tezUc
        brUcLift = brUcLift+1
    tezUc = 0
print ('Broj učenika u liftu: ',brUcLift)
print ('Ukupna težina učenika u liftu: ',zbrTez)
```

Ne zaboravite!

- Za 15 dana – **u utorak 19.12.2023. u 17:00**–
  2. kolo Lige programiranja