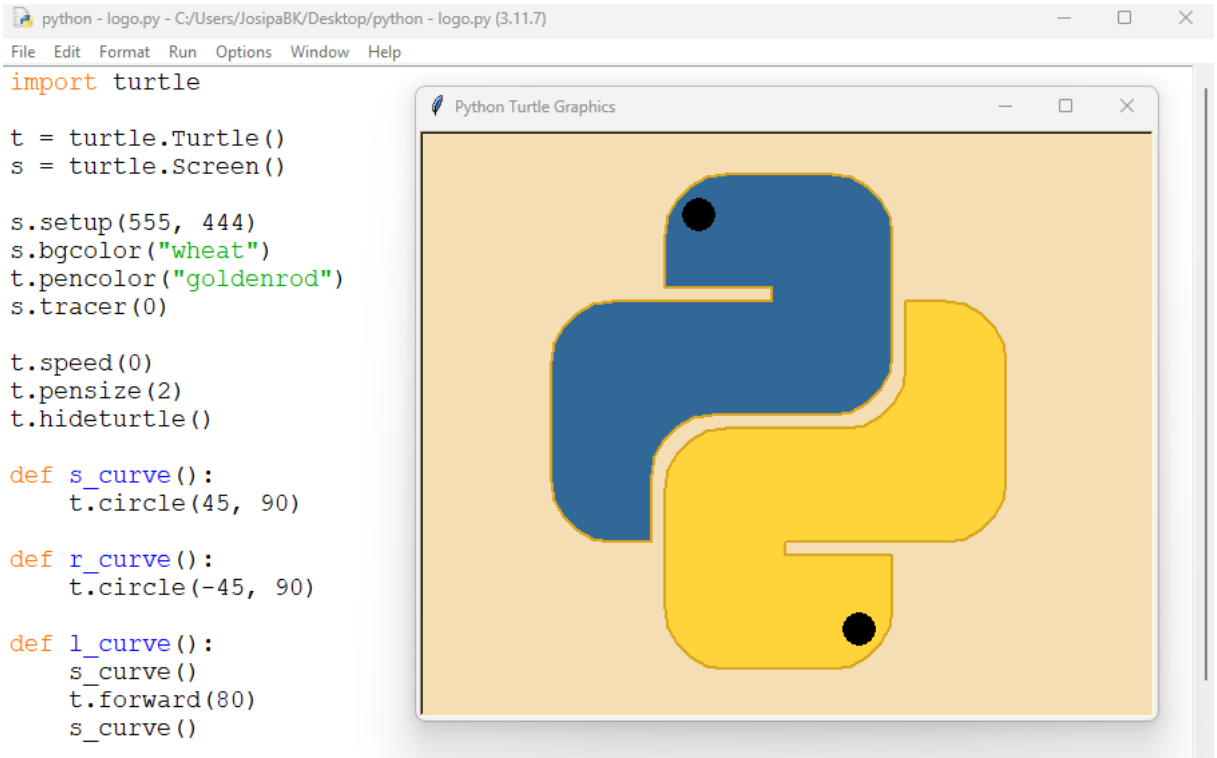
PRIRUČNIK ZA UČENIKE 5. RAZREDA

RAČUNALNO RAZMIŠLJANJE I PROGRAMIRANJE U PYTHONU

Josipa Baljkas Klisović, mag. educ. inf.



Informatika • 5. razred osnovne škole

Nastavna cjelina: Računalno razmišljanje i programiranje

Usklađeno s kurikulumom informatike za osnovnu školu

kolovoz, 2025.

IMPRESUM

Priručnik je izrađen za potrebe nastave informatike u 5. razredu osnovne škole.

Nastavna cjelina: Računalno razmišljanje i programiranje

Autor: Josipa Baljkas Klisović

E-mail: josipa.baljkas-klisovic@skole.hr

Ustanova: Katolička osnovna škola

Mjesto i datum: Šibenik, kolovoz 2025.

Verzija: 1.0

Pri izradi korišteni materijali:

- #mojportal5, udžbenik informatike u petom razredu osnovne škole, Školska knjiga 2019.
- mrežna stranica: https://kos.josipabk.com
- vlastiti materijali

Napomena: Slike u ovom priručniku predstavljaju kombinaciju sadržaja generiranog pomoću alata DALL·E i vlastitih snimki zaslona programskog sučelja.

Uvod u priručnik

Dragi učenici,

krećete u uzbudljivu avanturu učenja programiranja! Ovaj priručnik će vas korak po korak voditi kroz programski jezik Python.

Možda ste već čuli da je programiranje teško. Nemojte se brinuti – Python je jedan od najjednostavnijih programskih jezika na svijetu. Upravo zato se koristi za učenje programiranja u školama diljem svijeta.

Ako ste ranije radili u Scratchu, već znate razmišljati kao programeri. Tamo ste slagali blokove – sada ćete te iste ideje pisati riječima i znakovima. A ako niste radili Scratch, nema problema! Ovaj priručnik počinje od samog početka.

Što ćete naučiti?

U ovom priručniku naučit ćete:

- pokrenuti Python i koristiti ga kao kalkulator
- pisati naredbe koje računalo izvršava
- koristiti varijable za spremanje podataka
- pisati i pokretati vlastite programe
- tražiti podatke od korisnika
- crtati likove pomoću kornjačine grafike
- smisliti algoritam za rješavanje problema
- koristiti petlje za ponavljanje naredbi

Kako koristiti ovaj priručnik?

Svaka lekcija ima jasnu strukturu: najprije ćete pročitati objašnjenje, zatim pogledati primjere, a na kraju vježbati zadatke. Zadaci su označeni bojama prema težini:

🟢 **ZELENO** – laki zadaci za sve učenike

🟠 **NARANČASTO** – srednje teški zadaci

🔴 **CRVENO** – izazovni zadaci za one koji žele više

> 📌 **ZAPAMTI**
>
> Nitko ne mora znati sve odmah.
>
> Greške nisu neuspjeh – one su dio učenja!
>
> Svaki mali korak naprijed znači da ste bolji nego jučer.
>
> Vi to možete – samo trebate pokušati.

Pa, jeste li spremni? Hajdemo zajedno u svijet Pythona!

Sadržaj

Lekcija 1: Radno okružje Python

Uvod

U prvoj lekciji upoznat ćemo programski jezik Python. Naučit ćemo što je Python, kako ga pokrenuti i kako napisati prve naredbe.

Python je kao učenje nove abecede za pisanje priča. Umjesto slaganja blokova kao u Scratchu, sada ćemo naučiti "pisati" svoje ideje riječima i znakovima.

Računalo ne razumije hrvatski jezik kao mi. Ono razumije samo programski jezik – to je jezik kojim mu govorimo što treba napraviti. Python je jedan od tih jezika.

Ishodi učenja

Nakon ove lekcije moći ćeš:

- Objasniti što je Python i čemu služi
- Pokrenuti IDLE – razvojno okružje za Python
- Razlikovati interaktivno sučelje od uređivačkog sučelja
- Upisivati brojeve i matematičke izraze u interaktivno sučelje
- Koristiti matematičke operatore (+, –, *, /, //, %)
- Koristiti naredbu print() za ispis vrijednosti
- Ispisivati tekst (znakovne nizove) u Pythonu

Teorija

Što je Python?

Python je programski jezik. To znači da pomoću njega možemo razgovarati s računalom i reći mu što da radi.

U Scratchu smo to radili pomoću blokova. U Pythonu ćemo to raditi pisanjem riječi i znakova. Python je stvorio Guido van Rossum, programer iz Nizozemske. Python je besplatan i može se preuzeti s interneta (https://www.python.org/).

Videozapis instalacije dostupan je na sljedećoj poveznici: YouTube – instalacija programa

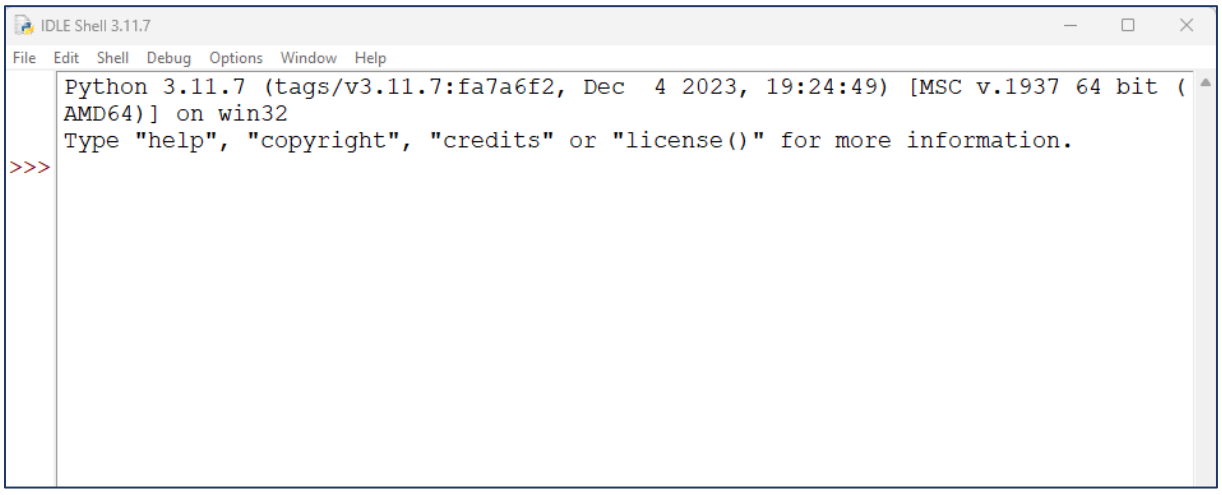
> **ZAPAMTI**
>
> Python je programski jezik kojim pišemo upute računalu. Te upute nazivamo programom ili kodom.

Kako pokrenuti Python?

Nakon što se Python instalira na računalo, pojavi se program koji se zove IDLE. Kada ga otvorimo, vidimo prozor s tri strelice: >>>

Te tri strelice znače: "Spreman sam! Napiši mi nešto!" Tu možemo odmah pisati naredbe i računalo ih odmah izvrši nakon što pritisnemo tipku Enter.

Taj prozor nazivamo interaktivno sučelje (Python Shell).



Interaktivno sučelje – kao kalkulator

U interaktivno sučelje možemo pisati:

- brojeve
- matematičke izraze (5+2, 10–3, 6*7, 15/4...)
- tekst (znakovni niz) – mora biti u polunavodnicima ili navodnicima
- naredbe, varijable, funkcije...

Sve što napišemo izvrši se odmah kad pritisnemo Enter.

Ako nešto pogriješimo, Python će ispisati poruku o pogrešci (BUG). To nije katastrofa – samo znači da nešto nismo dobro napisali.

Matematički operatori

Python može računati kao kalkulator. Koristi sljedeće operatore:

Operator	Značenje	Primjer	Rezultat
+	zbrajanje	5 + 2	7
–	oduzimanje	9 – 4	5
*	množenje	6 * 3	18
/	decimalno dijeljenje	5 / 2	2.5
//	cjelobrojno dijeljenje	5 // 2	2
%	ostatak dijeljenja (modulo)	5 % 2	1

PRIMJER

U interaktivno sučelje upišemo:

```
>>> 5 + 2
7
>>> 11 // 4
2
>>> 11 % 4
3
```

Python odmah izračuna i ispiše rezultat.

Redoslijed izvršavanja operatora

U Pythonu je, kao i u matematici, definiran redoslijed izvršavanja:

1. zagrade () – imaju najveći prioritet
2. množenje *, dijeljenje /, //, % – slijeva nadesno
3. zbrajanje + i oduzimanje – – slijeva nadesno

```
>>> 3 + 2 * 5
13
>>> (3 + 2) * 5
25
```

U prvom primjeru Python prvo pomnoži 2*5=10, pa zbroji 3+10=13.

U drugom primjeru zagrade mijenjaju redoslijed: najprije 3+2=5, pa 5*5=25.

Naredba print()

Naredba print() služi za ispisivanje sadržaja na ekran. To je jedna od najvažnijih naredbi u Pythonu.

U zagradu stavljamo ono što želimo da se ispiše. To može biti broj, tekst, račun ili više vrijednosti odjednom.

```
>>> print(10)
10
>>> print(10, 20, 30)
10 20 30
>>> print(9 + 4)
13
```

Ako u zagradu stavimo više stvari, odvajamo ih zarezom. Python ih ispiše razdvojene razmakom.

> **ZAPAMTI**
>
> Zarez u Pythonu odvaja više vrijednosti, a funkcija print() ih po zadanim postavkama ispisuje razdvojene razmakom.

Kako ispisati tekst?

Tekst u Pythonu nazivamo znakovni niz. Tekst uvijek mora biti u navodnicima (Shift+2) ili polunavodnicima (jedan klik na tipku gdje se nalazi i znak ?):

```
>>> print('Bok svima')
Bok svima
>>> print("Python je zabavan")
Python je zabavan
```

Spajanje teksta

Možemo spojiti dva teksta pomoću znaka + (to se zove konkatenacija):

```
>>> print('Fran' + 'Krsto')
FranKrsto
>>> print('Fran' + ' ' + 'Krsto')
Fran Krsto
```

Ako želimo razmak između riječi, moramo ga sami dodati kao ' '.

Ponavljanje teksta

Ako stavimo tekst i pomnožimo ga brojem, Python ga ponovi više puta:

```
>>> print('Fran' * 3)
FranFranFran
```

Miješanje teksta i brojeva u print()

```
>>> print('Zbroj je', 2 + 3 + 4)
Zbroj je 9
```

Python može istodobno ispisivati tekst i rezultate računanja.

Razdjelnik (sep)

Normalno Python između podataka stavlja razmak. Ali to možemo promijeniti pomoću sep (separator):

```
>>> print(2, 3, 4, sep=', ')
2, 3, 4,
```

```
>>> print(2, 3, 4, sep=':')
2:3:4:
```

Usporedba sa Scratchom

Ako ste radili Scratch, sjetite se bloka "reci". U Pythonu, naredba print() radi istu stvar – prikazuje tekst ili brojeve na ekranu. Samo umjesto da povučemo blok, napišemo naredbu riječima.

Boje teksta u IDLE-u

Python u IDLE-u koristi različite boje za različite vrste sadržaja:

- **Crna** – ono što mi upišemo (matematički izraz)
- **Plava** – rezultat koji Python izračuna
- **Crvena** – pogreška ili upozorenje
- **Zelena** – znakovni nizovi (tekst u navodnicima)
- **Ljubičasta** – naredbe (print, int, input...)
- **Narančasta** – funkcije (def...)

Zadaci za vježbu

Zadaci za razumijevanje

1. Što je Python?
2. Kako se zove program u kojem pišemo Python kod?
3. Što znači znak >>> u interaktivnom sučelju?
4. Što radi naredba print()?
5. Nabroji sve matematičke operatore koje Python koristi.

Što će ispisati ovaj kod?

6. Što će ispisati: print(5 + 3)?
7. Što će ispisati: print(10 * 2)?
8. Što će ispisati: print(3 + 2 * 5)?

🟠 9. Što će ispisati: print('Ana' + 'Marko')?

🟠 10. Što će ispisati: print('Hello' * 3)?

🟠 11. Što će ispisati: print(11 // 4)?

🟠 12. Što će ispisati: print(11 % 4)?

Praktični zadaci

🟢 13. Otvori IDLE i izračunaj koliko je 256 + 389.

🟢 14. Pomoću naredbe print() ispiši svoje ime.

🟢 15. Pomoću naredbe print() ispiši poruku 'Bok svima!'.

🟠 16. Napiši tri različita načina da ispišeš tekst 'Python je fora' u jednom retku.

🟠 17. Pomoću naredbe print() ispiši brojeve 1, 2, 3, 4, 5 razdvojene crticom (-). Koristi sep.

🔴 18. Izračunaj u Pythonu koliko je: (15 + 7) * 3 – 10 // 2. Najprije pokušaj izračunati na papiru, pa provjeri u Pythonu.

🔴 19. Koristeći samo naredbu print(), ispiši sljedeće: Moje ime je ____ Imam ____ godina Živim u ____ (umjesto ____ stavi svoje podatke)

Ispravljanje pogrešaka

🟠 20. Pronađi pogrešku u sljedećem kodu i objasni što je krivo:

```
>>> print(Bok svima)
```

🟠 21. Pronađi pogrešku:

```
>>> print('Zbroj je' 5 + 3)
```

Ponavljanje

📌 ZAPAMTI

Python je programski jezik kojim pišemo upute računalu.

IDLE je program u kojem pišemo Python kod.

Interaktivno sučelje (>>>) služi za brzo isprobavanje naredbi.

Naredba print() ispisuje sadržaj na ekran.

Tekst (znakovni niz) uvijek pišemo u navodnicima ili polunavodnicima.

Operatori: + (zbrajanje), – (oduzimanje), * (množenje), / (dijeljenje), // (cjelobrojno dijeljenje), % (ostatak).

Pitanja za ponavljanje

1. Što je Python?
2. Što je algoritam?
3. Kako se zove program u kojem pišemo Python kod (radno okruženje)?
4. Gdje se pojavljuje znak >>> i što znači?
5. Što radi naredba print()?
6. Kako ispisujemo tekst pomoću naredbe print?
7. Zašto moramo koristiti navodnike kad pišemo tekst?
8. Objasni redoslijed izvršavanja matematičkih operatora.
9. Kako Python "zna" da je nešto tekst?
10. Koja je razlika između print('Ana','Marko') i print('Ana' + 'Marko')?

Lekcija 2: Varijable i naredba pridruživanja

Uvod

Kad pišemo računalne programe, često moramo zapamtiti neke podatke koje ćemo kasnije koristiti. Zamislite da radimo program koji računa rezultat utakmice, ili program koji pamti vaše ime. Da računalo ne bi svaki put ponovno pitalo korisnika, te podatke treba negdje spremiti. Za to nam služe varijable.

Ishodi učenja

Nakon ove lekcije moći ćeš:

- Objasniti što je varijabla
- Pridružiti vrijednost varijabli pomoću znaka =
- Ispravno imenovati varijable
- Koristiti varijable u matematičkim izrazima
- Spojiti znakovne nizove pomoću operatora **+** i *****
- Koristiti višestruko pridruživanje

Teorija

Što je varijabla?

Varijablu možemo zamisliti kao malu kutijicu u računalu. Svaka kutijica ima svoje ime i u nju možemo spremiti neku vrijednost. Ta vrijednost može biti broj, tekst ili bilo što drugo što računalo može zapamtiti.
Kada nam kasnije trebamo tu vrijednost, samo se pozovemo na ime te "kutijice" i računalo nam ju vrati.

> **ZAPAMTI**
>
> Varijabla je spremnik s imenom u koji spremamo vrijednost. Ta se vrijednost može koristiti i mijenjati tijekom izvođenja programa.

Kako pridružiti vrijednost varijabli?

Varijablama se dodjeljuje vrijednost pomoću znaka = (znak pridruživanja).

```
>>> a = 5
>>> b = 2
```

Varijabli a pridružili smo vrijednost 5. Varijabli b pridružili smo vrijednost 2.

> **⚠ VAŽNO**
>
> Znak = u programiranju NE znači "jednako" kao u matematici! U programiranju = znači "stavi u varijablu lijevo ono što piše desno". Zato kažemo da je to znak pridruživanja.

Nakon pridruživanja, vrijednost varijable možemo vidjeti tako da upišemo njezino ime:

```
>>> a
5
>>> b
2
```

Kako ispravno napisati naziv varijable?

Postoje pravila za imenovanje varijabli:

- U nazivu se mogu koristiti slova i znamenke (0–9)
- Naziv varijable NE može početi znamenkom
- NE mogu se koristiti simboli kao što su – / # @
- Razmak se NE može koristiti, ali za povezivanje riječi može se koristiti znak _ (donja crta). Npr.: p_znak, p_broj
- Ne smiju se koristiti riječi koje Python upotrebljava za naredbe, npr. print, int, min, max, True

PAZI

Python razlikuje velika i mala slova! Varijable a i A su DVIJE RAZLIČITE varijable.

```
>>> a = 5
>>> A = 10
>>> a
5
>>> A
10
```

Ne preporučuje se u nazivima varijabli upotrebljavati dijakritičke znakove hrvatske abecede (š, ž, č, ć, đ) jer mogu uzrokovati probleme na drugim računalima koji nemaju HR tipkovnicu.

Varijable s brojevima

Varijable koje sadrže brojčane vrijednosti mogu se upotrebljavati unutar matematičkih izraza:

```
>>> a = 5
>>> b = 2
>>> a + b
7
>>> a - b
3
>>> a * b
10
>>> a / b
2.5
```

Varijable s tekstom

Znakovni nizovi pridružuju se tako da se upisuju unutar polunavodnika ili navodnika:

```
>>> ime = 'August'
>>> prezime = 'Cesarec'
>>> ime
'August'
>>> prezime
'Cesarec'
```

Uz varijable sa znakovnim vrijednostima mogu se upotrebljavati operatori zbrajanja (+) i množenja (*):

```
>>> ime + prezime
'AugustCesarec'
>>> ime + ' ' + prezime
'August Cesarec'
>>> ime * 4
'AugustAugustAugustAugust'
```

Operator + spaja znakovne nizove. Operator * ponavlja znakovni niz zadani broj puta.

Višestruko pridruživanje

Ako želimo da više varijabli ima istu vrijednost, Python nam omogućuje višestruko pridruživanje:

```
>>> a = b = c = 2018
>>> a
2018
>>> b
2018
>>> c
2018
```

Sve tri varijable odjednom dobiju vrijednost 2018.

Posebni znakovi: \n i \t

Za dodatno oblikovanje ispisa koristimo posebne znakove:

\n – prelazak u novi redak

\t – tabulator (razmak od 8 znakova)

```
>>> print('Pozdrav', 'svima', sep='\n')
Pozdrav
svima
>>> print('Ime\tPrezime\tRazred')
Ime     Prezime     Razred
```

Usporedba sa Scratchom

U Scratchu ste možda koristili varijable - to su bile one male „kućice“ na pozornici koje su pokazivale brojeve. U Pythonu je isto, samo što varijablama dajemo imena pisanjem koda, a ne povlačenjem blokova.

Zadaci za vježbu

- 1. Što je varijabla? Objasni svojim riječima.
- 2. Navedi pravila za ispravne nazive varijabli.
- 3. Je li Python razlikuje mala i velika slova?
- 4. Stvori varijablu x i pridruži joj vrijednost 7. Ispiši ju.
- 5. Stvori varijablu ime i pridruži joj svoje ime. Ispiši ju.

- 6. Zadaj varijablama a=12 i b=5. Izračunaj i ispiši: opseg pravokutnika (2*a + 2*b).
- 7. Varijabli grad pridruži vrijednost 'Zagreb'. Ispiši poruku: 'Živim u Zagreb'.
- 8. Što će ispisati sljedeći kod?

```
ime = 'Luka'
print(ime * 2)
```

- 9. Koristi varijable i naredbu print() da ispišeš: Ime: Tvoje_ime Razred: 5 (koristi \t za tabulatore)

🔴 10. Izračunaj opseg pravokutnika ako mu je duljina 12, a širina 5. Koristi varijable duljina, sirina i opseg.

🔴 11. Pronađi pogrešku u sljedećem kodu:

```
2broj = 10
print(2broj)
```

🔴 12. Što znači naredba \n? Napiši primjer u kojem se koristi.

Ponavljanje

ZAPAMTI

Varijabla je spremnik s imenom u koji spremamo vrijednost.

Znak = je znak pridruživanja (ne jednakosti!).

Naziv varijable ne smije početi brojkom ni sadržavati razmake.

Python razlikuje velika i mala slova.

Operatorom + spajamo znakovne nizove.

Operatorom * ponavljamo znakovni niz.

\n = novi redak, \t = tabulator.

Lekcija 3: Moj prvi program

Uvod

Do sada smo u interaktivnom sučelju isprobavali naredbe jednu po jednu. Ali što ako želimo napraviti nešto veće? Zamislite da želite izračunati opseg trokuta, riješiti matematički zadatak ili nacrtati lik na ekranu. Za takve zadatke trebamo više naredbi, posloženih u pravom redu. Upravo zato nastaje ono što zovemo računalni program – mala datoteka koja čuva sve naše naredbe.

> **ZAPAMTI**
>
> Računalni program je datoteka u kojoj je zapisano više naredbi. Računalo ih izvršava točno onim redoslijedom kojim su napisane.

Ishodi učenja

Nakon ove lekcije moći ćeš:

- Objasniti što je računalni program
- Otvoriti uređivačko sučelje u IDLE-u (File → New File)
- Napisati jednostavan program s više naredbi
- Spremiti program kao .py datoteku
- Pokrenuti program (Run → Run Module ili F5)
- Provjeriti ispravnost programa i ispraviti pogreške

Teorija

Zašto trebamo programe?

U interaktivnom sučelju sve što napišemo izvrši se odmah. Ali kad zatvorimo prozor, naše naredbe nestaju. Ako želimo sačuvati naredbe, moramo ih zapisati u datoteku – to je program. Program je kao recept: da bismo dobili pravo jelo, moramo slijediti korake, jedan po jedan. Ako je neki korak pogrešan – rezultat neće biti dobar.

Uređivačko sučelje (New File)

Da ne moramo stalno pisati ispočetka, Python ima i drugi način rada. Kliknemo File → New File. Otvori se prazan bijeli prozor bez znaka >>>.

Tu pišemo program, red po red, i uvijek ga možemo uređivati. Program se izvršava tek kad ga mi pokrenemo.

Pisanje prvog programa

U uređivačkom sučelju napišimo sljedeći program:

```
a = 10
b = 20
c = 15
print('Opseg trokuta je', a + b + c)
```

Ovaj program:

1. Stvara varijablu a i sprema u nju broj 10
2. Stvara varijablu b i sprema u nju broj 20
3. Stvara varijablu c i sprema u nju broj 15
4. Izračuna zbroj a+b+c i ispiše poruku s rezultatom

Kako spremiti program?

Prije nego pokrenemo program, moramo ga spremiti:

1. Kliknemo File → Save As...
2. Odredimo mjesto gdje ćemo ga spremiti
3. Upišemo ime datoteke (npr. opseg)
4. Kliknemo Spremi

> **ZAPAMTI**
>
> Datoteka programa u Pythonu ima nastavak .py i ne treba ga upisivati pri spremanju – Python će ga automatski dodati.

Kako pokrenuti program?

Kliknemo Run → Run Module ili jednostavno pritisnemo tipku F5.

Rezultat se prikazuje u interaktivnom sučelju:

```
Opseg trokuta je 45
```

To je trenutak u kojem vidimo da naš program radi!

Što ako pogriješimo?

Ponekad program ne daje očekivani rezultat. To je normalno! Programeri stalno ispravljaju i mijenjaju svoje programe.

Koraci koje programer slijedi:

1. Napisati program
2. Spremiti program
3. Pokrenuti program
4. Provjeriti daje li program točno rješenje
5. Ako je potrebno – ispraviti program i ponovno ga pokrenuti

Usporedba sa Scratchom

U Scratchu ste slagali blokove jedan ispod drugoga. U Pythonu je isto – pišemo naredbe jednu ispod druge. Redoslijed je važan – računalo izvršava naredbe odozgo prema dolje.

Zadaci za vježbu

- 1. Objasnite što je računalni program.
- 2. Kako otvoriti uređivačko sučelje u IDLE-u?

🟢 3. Kojom tipkom pokrećemo program?

🟢 4. Napiši program koji ispisuje tvoje ime i prezime.

🟠 5. Napiši program koji stvara varijable a=7, b=3 i ispisuje njihov zbroj.

🟠 6. Napiši program koji varijabli ime pridružuje tvoje ime, a varijabli god tvoje godine. Program treba ispisati: 'Zovem se ___ i imam ___ godina.'

🟠 7. Napiši program za izračunavanje opsega pravokutnika sa stranicama duljina=12 i sirina=5.

🔴 8. Napiši program koji računa površinu i opseg kvadrata sa stranicom a=8. Ispiši oba rezultata s odgovarajućim porukama.

🔴 9. Pronađi pogrešku u programu:

```
a = 10
b = 20
c = 15
print('Opseg trokuta je' a + b + c)
```

Ponavljanje

📌 ZAPAMTI

Računalni program je datoteka s nizom naredbi koje računalo izvršava redom.

Program pišemo u uređivačkom sučelju (File → New File).

Program spremamo s File → Save As... (nastavak .py).

Program pokrećemo s Run → Run Module ili tipkom F5.

Ako rezultat nije točan, ispravljamo program i ponovno ga pokrećemo.

Lekcija 4: Rad s ulaznim vrijednostima

Uvod

Do sada smo u programima unaprijed pisali vrijednosti (npr. a=10). Ali što ako želimo da program radi s različitim podacima svaki put kad ga pokrenemo? Npr. da izračuna opseg bilo kojeg trokuta, ne samo onog sa stranicama 10, 20 i 15?

Za to koristimo ulazne vrijednosti. Program pita korisnika za podatke, a korisnik ih upisuje tipkovnicom.

Ishodi učenja

Nakon ove lekcije moći ćeš:

- Objasniti što su ulazne vrijednosti
- Koristiti naredbu input() za unos podataka s tipkovnice
- Koristiti naredbu int() za pretvaranje teksta u broj
- Pisati programe koji traže podatke od korisnika
- Prepoznati i objasniti česte pogreške pri unosu podataka

Teorija

Što su ulazne vrijednosti?

Ulazne vrijednosti su podaci koje korisnik unosi u program tijekom njegova izvođenja. Umjesto da vrijednosti unaprijed pišemo u program, možemo ih zatražiti od korisnika. Zamislite program kao dijalog: računalo pita, a vi odgovarate.

Naredba input()

U Pythonu se za unos podataka koristi naredba input(). U zagradu pišemo poruku koja će se prikazati korisniku:

```
ime = input('Kako se zoveš?: ')
print('Bok,', ime)
```

Kad pokrenemo program:

```
Kako se zoveš?: Vlaho
Bok, Vlaho
```

> ⚠ **VAŽNO**
>
> Naredba input() uvijek vraća tekst (znakovni niz)!
>
> Ako želimo da unos bude broj (da možemo računati), moramo ga pretvoriti pomoću naredbe int().

Pretvaranje teksta u broj – int()

Kada želimo da korisnik unese broj, moramo koristiti int() zajedno s input():

```
a = int(input('Unesi duljinu stranice a: '))
```

Ovdje se događaju tri stvari:

1. Računalo ispisuje poruku korisniku
2. Korisnik upisuje vrijednost
3. int() tu vrijednost pretvara u cijeli broj, koji se sprema u varijablu a

> **PRIMJER**
>
> Program koji zbraja dva broja:
>
> ```
> a = int(input('Prvi broj: '))
> b = int(input('Drugi broj: '))
> c = a + b
> print('Njihov zbroj je', c)
> ```

Rezultat:

```
Prvi broj: 7
Drugi broj: 9
Njihov zbroj je 16
```

Unos znakovnih nizova

Ako želimo unijeti tekst (ime, grad...), koristimo samo input() bez int():

```
ime = input('Upiši svoje ime: ')
print('Zovem se', ime)
```

Česte pogreške pri unosu podataka

1. Upisivanje teksta umjesto broja – ako program očekuje broj, a korisnik upiše riječ (npr. "petnaest"), Python javlja grešku (ValueError).
2. Zaboravljanje int() – ako koristimo samo input() bez int(), Python će vrijednost tretirati kao tekst, a ne kao broj.

PAZI

Za unos znakovnog niza koristi se samo naredba input().

Za unos cijelog broja ispred naredbe input() mora se pozvati naredba int().

Primjer: Opseg i površina kvadrata

```
a = int(input('Upiši stranicu kvadrata: '))
o = 4 * a
p = a * a
print('Opseg kvadrata je', o, ', a površina', p)
```

Unos više vrijednosti odjednom

Ponekad nam je praktično unijeti više podataka u jednom retku. Python to omogućuje uz pomoć split():

```
ime, prezime, mjesto = input('Upiši ime, prezime i mjesto: ').split()
print('Zovem se', ime, prezime)
print('Mjesto rođenja je', mjesto)
```

Vrijednosti se odvajaju razmakom.

Zadaci za vježbu

🟢 1. Što su ulazne vrijednosti u računalnom programu?

🟢 2. Koja se Python naredba koristi za unos vrijednosti s tipkovnice?

🟢 3. Kada moramo koristiti int() zajedno s input()?

🟢 4. Napiši program koji pita korisnika za ime i ispisuje: 'Bok, ___!'

🟠 5. Napiši program koji od korisnika traži duljinu stranice kvadrata, a zatim izračuna i ispiše opseg.

🟠 6. Napiši program koji traži dva broja i ispisuje njihov umnožak (rezultat množenja).

🟠 7. Napiši program u koji će korisnik upisati godinu rođenja svog roditelja, a program će izračunati koliko roditelj ima godina (koristite 2025. godinu).

🔴 8. Napiši program koji računa opseg i površinu pravokutnika. Program traži od korisnika duljinu i širinu.

🔴 9. Petar i Luka brali su jabuke. Petar je nabrao k kilograma. Luka je nabrao dvostruko više od Petra. Napiši program koji traži koliko je Petar nabrao i ispisuje koliko je Luka nabrao.

🔴 10. Tena i Ema kupuju bombone. Za 5 bombona Tena je platila nekoliko eura. Napiši program koji traži koliko je Tena platila i ispisuje koliko košta jedan bombon.

Ponavljanje

ZAPAMTI

Ulazne vrijednosti su podaci koje korisnik unosi u program.

Naredba input() služi za unos podataka s tipkovnice.

input() uvijek vraća tekst – za broj koristimo int(input(...)).

Za unos teksta koristimo samo input() bez int().

Ako program očekuje broj, a dobije tekst – javlja se pogreška.

Lekcija 5: Kako radi moj program?

Uvod

Kada prvi put krenemo učiti programiranje, može nam se činiti kao da računalo "magično" radi stvari koje napišemo. Ali iza toga nema nikakve magije – program radi točno onako kako smo mu rekli, korak po korak.
Da bismo bolje razumjeli programiranje, važno je znati što se događa kada računalo izvršava naš program.

Ishodi učenja

Nakon ove lekcije moći ćeš:

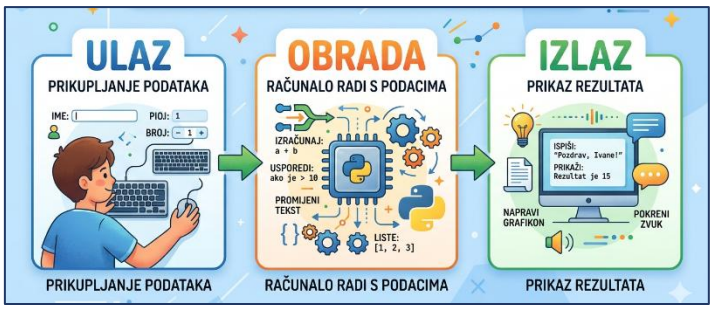
- Prepoznati tri osnovna dijela programa: ulaz, obrada, izlaz
- Pratiti izvršavanje programa korak po korak
- Razlikovati sintaktičke i logičke pogreške
- Pronaći i ispraviti pogreške u programu

Teorija

Tri dijela svakog programa

Svaki program možemo zamisliti kao putovanje kroz tri koraka:

1. **ULAZ (unos podataka)** – unosimo informacije koje su programu potrebne
2. **OBRADA (računanje, uspoređivanje)** – program koristi podatke i obavlja neki posao
3. **IZLAZ (prikaz rezultata)** – program ispisuje odgovor



PRIMJER

Program koji računa opseg pravokutnika:

```
a = int(input('Upiši stranicu a: '))
b = int(input('Upiši stranicu b: '))
o = 2 * a + 2 * b
p = a * b
print('Opseg je', o)
print('Površina je', p)
```

ULAZ: unos stranica a i b

OBRADA: izračunavanje opsega i površine

IZLAZ: ispis rezultata

Kako pratimo izvršavanje programa?

Zamislimo program koji pretvara kilometre u metre:

```
km = int(input('Unesite udaljenost u kilometrima: '))
m = km * 1000
print('Udaljenost od', km, 'km iznosi', m, 'm.')
```

Pratimo korak po korak:

1. Program traži unos – korisnik upisuje 23 → varijabla km = 23
2. Program računa m = 23 * 1000 = 23000 → varijabla m = 23000
3. Program ispisuje: Udaljenost od 23 km iznosi 23000 m.

Vrste pogrešaka

1. Sintaktičke pogreške

To su pogreške u pisanju samog programa – kao kada u hrvatskom pogrešno napišemo riječ. Primjeri: zaboravljen zarez, krivo napisan naziv varijable, nedostaje zagrada. Kada se takva pogreška dogodi, program se neće ni pokrenuti. Python će ispisati poruku o pogrešci.

2. Logičke pogreške

Ovo su posebne pogreške – program se pokrene, ali rezultat nije točan. To se događa kad je formula pogrešna ili kad smo krivim redoslijedom napisali naredbe. Takve pogreške najčešće otkrivamo usporedbom očekivanih i dobivenih rezultata.

> ⚠ **VAŽNO**
>
> Ako želimo otkriti pogrešku, moramo pratiti što program radi – od unosa, preko obrade, do izlaza.

Zadaci za vježbu

🟢 1. Koji su tri osnovna dijela svakog programa?

🟢 2. Što je sintaktička pogreška? Daj primjer.

🟢 3. Što je logička pogreška? Daj primjer.

🟠 4. Za sljedeći program odredi koji je dio ULAZ, koji OBRADA, a koji IZLAZ:

```
a = int(input('Upiši broj: '))
b = a * 5
print('Rezultat je', b)
```

🟠 5. Pronađi pogrešku u programu (logička pogreška):

```
a = int(input('Upiši broj: '))
print('Višekratnici broja', a, 'su:', a+1, a+2, a+3, a+4, a+5)
```

Pomoć: Višekratnici broja 3 su: 3, 6, 9, 12, 15 (ne 4, 5, 6, 7, 8).

🔴 6. Napiši program koji pretvara temperature iz Celzija u Fahrenheite. Formula: F = C * 9/5 + 32. Prati izvršavanje za ulaz C = 20.

🔴 7. Marija želi ispeći tortu. Mase namirnica su u gramima, a u receptu su u dekagramima. Napiši program prema dijagramu tijeka: unesi grame → podijeli s 10 → ispiši dekagrame.

Ponavljanje

ZAPAMTI

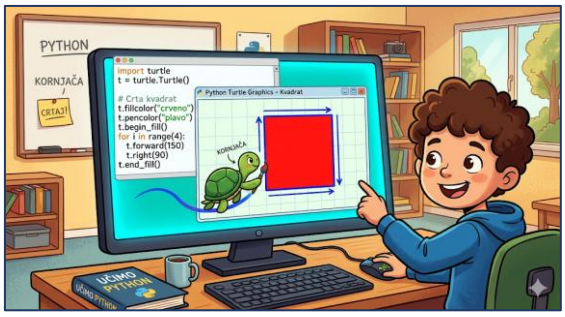
Svaki program ima tri dijela: ULAZ → OBRADA → IZLAZ.

Sintaktičke pogreške su pogreške u pisanju koda – program se ne pokreće.

Logičke pogreške – program radi, ali daje krivi rezultat.

Pogreške otkrivamo praćenjem programa korak po korak.

Lekcija 6: Crtanje u Pythonu



Uvod

Do sada smo u programiranju pisali, računali i ispisivali rezultate. Krajnje je vrijeme da naučimo i nešto zabavno – crtanje u Pythonu!

U Pythonu za crtanje koristimo kornjaču. Kornjača je zamišljeni lik koji se pojavljuje na ekranu i u ruci drži olovku. Ona sluša naše naredbe i radi točno ono što joj kažemo.

Ako ste radili u Scratchu, već znate kako to funkcionira – tamo ste pomicali likove po pozornici. Ovdje je isto, samo što umjesto blokova pišemo naredbe tekstom.

Ishodi učenja

Nakon ove lekcije moći ćeš:

- Aktivirati modul kornjačine grafike (turtle)
- Koristiti naredbe za pomicanje i okretanje olovke
- Crtati jednostavne geometrijske likove (kvadrat, pravokutnik, trokut)
- Koristiti naredbe za podizanje i spuštanje olovke
- Obojiti crteže pomoću naredbi color() i begin_fill()/end_fill()
- Izračunati vanjski kut pravilnog mnogokuta

Teorija

Pokretanje kornjačine grafike

Da bi kornjača mogla crtati, moramo na početku programa napisati jednu važnu naredbu:

```
from turtle import*
```

Ova naredba govori Pythonu da želimo koristiti modul za crtanje koji se zove turtle. Bez ove naredbe nema crtanja!

Osnovne naredbe za crtanje

Naredba	Skraćeno	Opis
forward(a)	fd(a)	Pomiče olovku naprijed za a koraka
backward(a)	bk(a)	Pomiče olovku unatrag za a koraka
left(kut)	lt(kut)	Okreće olovku ulijevo za zadani kut
right(kut)	rt(kut)	Okreće olovku udesno za zadani kut
penup()	pu()	Podiže olovku (ne crta pri pomicanju)
pendown()	pd()	Spušta olovku (crta pri pomicanju)
showturtle()	st()	Prikazuje olovku na ekranu
hideturtle()	ht()	Skriva olovku na ekranu
home()		Vraća olovku na početni položaj
reset()		Vraća olovku na početak i briše crteže

Crtanje kvadrata

Kvadrat ima 4 jednake stranice i 4 prava kuta (90°). Za crtanje ponavljamo: idi naprijed, okreni se ulijevo za 90°.

```
from turtle import*
fd(100); lt(90)
fd(100); lt(90)
fd(100); lt(90)
fd(100); lt(90)
```

Crtanje pravilnih oblika i kutovi

Kod pravilnih oblika vrijedi važno pravilo: puni krug iznosi 360 stupnjeva.

Da bismo znali za koliko se kornjača mora okrenuti, puni krug dijelimo s brojem stranica oblika:

• Kvadrat: 360° / 4 = 90°

- Jednakostraničan trokut: 360° / 3 = 120°
- Šesterokut: 360° / 6 = 60°
- Osmerokut: 360° / 8 = 45°

Crtanje jednakostraničnog trokuta

```
from turtle import*
title('Crtanje')
fd(100); lt(120)
fd(100); lt(120)
fd(100); lt(120)
```

Bojanje oblika

Za bojanje koristimo naredbe color(), begin_fill() i end_fill():

```
from turtle import*
color('red', 'blue')
begin_fill()
fd(150); lt(90)
fd(150); lt(90)
fd(150); lt(90)
fd(150); lt(90)
end_fill()
```

Naredba color('red', 'blue') postavlja crvenu boju crte i plavu boju ispune.

Podizanje i spuštanje olovke

Ponekad želimo pomaknuti kornjaču bez crtanja. Tada koristimo pu() (penup) i pd() (pendown):

```
fd(100)
pu()      # podigni olovku
fd(100)   # pomakni se bez crtanja
pd()      # spusti olovku
fd(100)   # opet crta
```

Zadaci za vježbu

- 1. Koja naredba mora biti na početku svakog programa za crtanje?
- 2. Što radi kornjača kada dobije naredbu fd(100)?
- 3. Što radi naredba lt(90)?
- 4. Nacrtaj crtu duljine 200 piksela.

- 5. Nacrtaj kvadrat sa stranicom 120.
- 6. Nacrtaj pravokutnik sa stranicama 150 i 80.
- 7. Nacrtaj jednakostraničan trokut sa stranicom 100.
- 8. Nacrtaj dvije crte s razmakom (koristi pu() i pd()).

- 9. Nacrtaj pravilni šesterokut sa stranicom 80.
- 10. Nacrtaj kvadrat ispunjen plavom bojom s crvenim rubom.
- 11. Nacrtaj dva kvadrata jedan do drugoga (koristi pu/pd za pomicanje).

Ponavljanje

ZAPAMTI

Za crtanje koristimo modul turtle – pokrenemo ga s: from turtle import*

fd() – naprijed, bk() – natrag, lt() – lijevo, rt() – desno

pu() – podigni olovku, pd() – spusti olovku

Kut okretanja za pravilni mnogokut: 360° / broj stranica

Za bojanje: color(), begin_fill(), end_fill()

Lekcija 7: Korak po korak do rješenja

stvori Ilustraciju za python: Dijagram tijeka s početkom, ulazom, obradom i krajem

Uvod

Do sada smo u nastavi informatike već puno toga naučili. Pisali smo programe, računali, ispisivali rezultate, čak i crtali. Pri stvaranju rješenja najteži dio bio nam je odrediti kojim redoslijedom složiti naredbe.

Kada rješavamo neki zadatak, važno je ne krenuti odmah pisati program, nego prvo razmisliti što zapravo želimo napraviti. Moramo odlučiti koje ćemo korake napraviti i kojim redom.

Upravo taj redoslijed koraka naziva se algoritam.

Ishodi učenja

Nakon ove lekcije moći ćeš:

- Objasniti što je algoritam
- Navesti primjere algoritama iz svakodnevnog života
- Razlikovati tri algoritamske strukture: slijed, grananje i ponavljanje
- Prikazati algoritam dijagramom tijeka
- Prepoznati simbole dijagrama tijeka
- Napisati jednostavan program prema algoritmu

Teorija

Što je algoritam?

Algoritam je točan i jasan postupak kojim rješavamo neki problem ili dolazimo do željenog cilja. Algoritme ne koristimo samo u računalima – koristimo ih i u svakodnevnom životu. Npr., kada kuhamo čaj, ne radimo to nasumično. Znamo da prvo moramo uliti vodu u posudu, zatim je zagrijati, staviti čaj u šalicu i na kraju pričekati da čaj odstoji.

PRIMJER

Algoritam za kuhanje čaja:

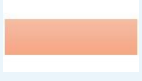
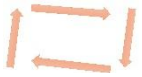
1. U posudu ulij vodu
2. Uključi kuhalo
3. Pričekaj da voda prokuha
4. Stavi vrećicu s čajem u šalicu
5. Kad voda prokuha, isključi kuhalo
6. Vruću vodu ulij u šalicu s čajem
7. Poklopi šalicu i ostavi čaj da odstoji 3 minute

Tri algoritamske strukture

1. **Algoritam slijeda** – koraci se izvršavaju red po red, slijedno, bez preskakanja. Većina naših dosadašnjih programa koristi ovu strukturu.
2. **Algoritam ponavljanja** – neki korak se ponavlja više puta. U programiranju za to koristimo petlje.
3. **Algoritam grananja** – odabir koraka za rješavanje ovisi o nekom uvjetu. Ako je uvjet ispunjen, radi se jedno; ako nije, radi se drugo.

Dijagram tijeka

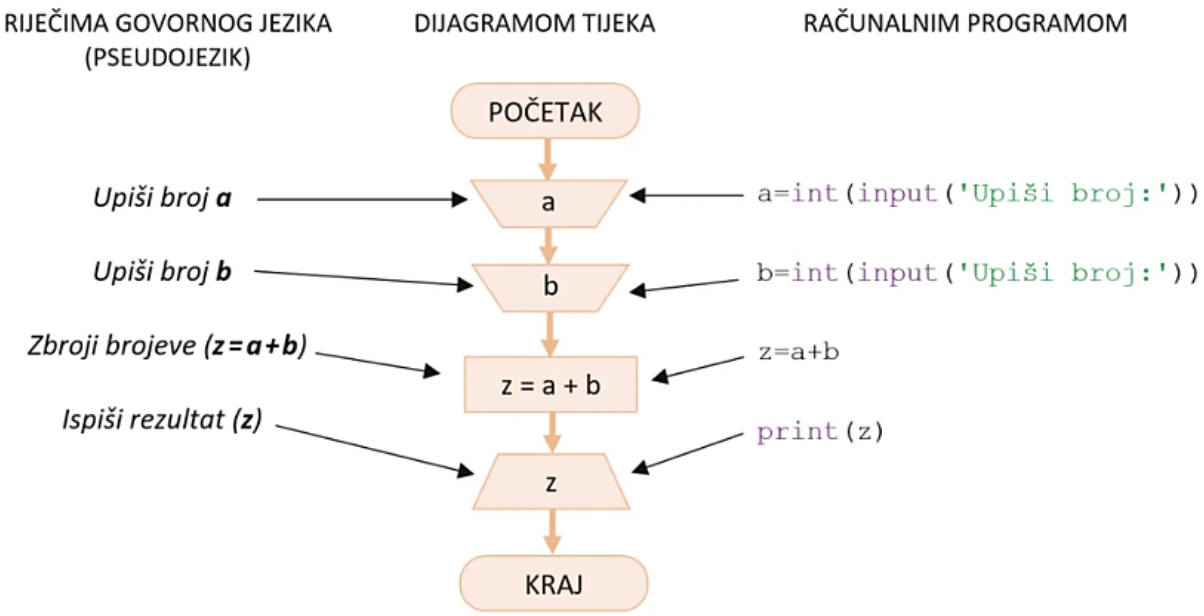
Dijagram tijeka je grafički prikaz algoritma. Za izradu dijagrama koriste se dogovoreni grafički simboli:

Simbol	Značenje
	Početak i kraj programa
	Unošenje podataka (ULAZ)
	Obrada podataka (formula)
	Simbol odluke ili petlje
	Ispis rezultata (IZLAZ)
	Pokazuju smjer kojim radnja teče

[Ilustracija: Dijagram tijeka za zbrajanje dva broja: POČETAK → unesi a → unesi b → z=a+b → ispiši z → KRAJ]

Od algoritma do programa

Prikažimo algoritam za zbrajanje dvaju brojeva na tri načina:



Zadaci za vježbu

1. Što je algoritam? Objasni svojim riječima.
2. Navedi primjer algoritma iz svakodnevnog života.
3. Koje su tri algoritamske strukture? Objasni svaku ukratko.
4. Što je dijagram tijeka?

5. Osmisli algoritam za spremanje za školu. Zapiši korake.
6. Napiši računalni program prema sljedećem algoritmu: unesi masu u gramima → podijeli s 10 → ispiši vrijednost u dekagramima.
7. Za program iz zadatka 6, nacrtaj dijagram tijeka.

🔴 8. Napiši program prema dijagramu tijeka koji računa prosjek triju upisanih ocjena iz informatike.

🔴 9. Osmisli algoritam za neki problem iz matematike i napiši program prema njemu.

Ponavljanje

ZAPAMTI

Algoritam je točan i jasan postupak rješavanja nekog problema.

Tri strukture: slijed, ponavljanje, grananje.

Algoritam možemo prikazati riječima (pseudojezik), dijagramom tijeka ili programom.

Dijagram tijeka koristi dogovorene simbole (oval, pravokutnik, romb, paralelogram, strelice).

Lekcija 8: Petljamo petlju

[Ilustracija: Kornjača crta kvadrat pomoću petlje for]

Uvod

Do sada smo u programiranju naučili kako se naredbe izvršavaju redom, jedna za drugom. To je algoritamska struktura slijeda.

Sada idemo korak dalje. Upoznat ćemo se s naredbom koja nam omogućuje da neke naredbe ponovimo više puta, bez da ih moramo stalno iznova pisati. Ta naredba naziva se petlja.

Ishodi učenja

Nakon ove lekcije moći ćeš:

- Objasniti što je petlja i zašto je koristimo
- Napisati petlju for s naredbom range()
- Objasniti kako radi brojač u petlji
- Koristiti petlju za crtanje geometrijskih likova
- Koristiti petlju za ispis brojeva
- Prepoznati česte pogreške pri radu s petljom for

Teorija

Zašto nam trebaju petlje?

Prisjetimo se kako smo crtali kvadrat:

```
from turtle import*
fd(100); lt(90)
fd(100); lt(90)
fd(100); lt(90)
fd(100); lt(90)
```

Iste naredbe ponovili smo četiri puta! Program je dugačak i nepregledniji. Što ako trebamo nacrtati oblik s 20 stranica? Za to koristimo petlje – naredbe koje nam omogućuju da isto ponavljanje napišemo samo jednom.

Petlja for

Petlja for koristi se kada točno znamo koliko puta želimo ponoviti neke naredbe.

```
from turtle import*
for i in range(4):
   fd(100); lt(90)
```

Ovdje kornjača četiri puta ponavlja iste naredbe i crta kvadrat.

> ⚠ **VAŽNO**
>
> Naredbe koje se ponavljaju uvijek moraju započeti uvlakom (tipka TAB ili 4 razmaka)!
>
> Bez uvlake Python neće znati koje naredbe pripadaju petlji.

Kako radi petlja for?

Petlja for koristi varijablu brojač. U našem primjeru to je varijabla i.

Petlja radi ovako:

1. Brojač dobije početnu vrijednost
2. Izvrše se naredbe unutar petlje
3. Brojač se poveća
4. Provjerava se treba li petlju još izvršavati
5. Kada se dosegne granica, petlja završava

Naredba range()

U petlji for koristimo naredbu range(). Ona određuje:

- od koje vrijednosti brojač počinje
- do koje vrijednosti ide
- za koliko se povećava

range()	Objašnjenje
range(4)	Brojač: 0, 1, 2, 3 (4 ponavljanja). Počinje od 0.
range(1, 4)	Brojač: 1, 2, 3 (počinje od 1, ide do 3)
range(1, 10, 2)	Brojač: 1, 3, 5, 7, 9 (od 1 do 9, korak 2)
range(2, 12, 2)	Brojač: 2, 4, 6, 8, 10 (parni brojevi od 2 do 10)

ZAPAMTI

Ako nije navedena početna vrijednost, Python počinje od 0.

Ako nije navedena vrijednost uvećavanja brojača, Python povećava broj za 1.

Granična vrijednost NIJE uključena! range(4) daje: 0, 1, 2, 3 (ne i 4!).

Petlja for i crtanje likova

Petlju for možemo koristiti za crtanje pravilnih geometrijskih likova:

- Kvadrat → range(4), kut 90°
- Trokut → range(3), kut 120°
- Šesterokut → range(6), kut 60°

PRIMJER

Crtanje trokuta s ulaznom duljinom stranice:

```
from turtle import*
a = int(input('Unesite duljinu: '))
for i in range(3):
    fd(a); lt(120)
```

Petlja for i ispis brojeva

Petlju for ne koristimo samo za crtanje, nego i za ispis brojeva:

```
n = int(input('Unesite prirodni broj: '))
for i in range(1, n):
    print(i, end=' ')
```

Za n=7 program ispisuje: 1 2 3 4 5 6

Parametar end=' ' govori Pythonu da nakon ispisa ne ide u novi red, nego stavi razmak.

Česte pogreške

Najčešća pogreška pri radu s petljom for je pogrešno zadana početna ili granična vrijednost brojača. Ako je početna vrijednost veća od granične, petlja se neće izvršiti nijednom!

```
for i in range(4, 1):
    print('Bravo')    # Ovo se neće ispisati!
```

Zadaci za vježbu

🟢 1. Što je petlja? Objasni svojim riječima.

🟢 2. Koja je uloga brojača u petlji for?

🟢 3. Što će ispisati: for i in range(3): print('Bok')?

🟠 4. Nacrtaj kvadrat sa stranicom 80 koristeći petlju for.

🟠 5. Nacrtaj jednakostraničan trokut sa stranicom 100 koristeći petlju for.

🟠 6. Ispiši sve brojeve od 1 do 10 koristeći petlju for.

🟠 7. Ispiši sve parne brojeve od 2 do 20.

🔴 8. Napiši program koji od korisnika traži duljinu stranice i crta pravilni šesterokut.

🔴 9. Napiši program koji ispisuje prvih 5 višekratnika broja koji unese korisnik.

🔴 10. Hoće li se petlja for i in range(2, 10, 2) izvršiti? Ako da, koliko puta i koje će vrijednosti imati brojač i?

🔴 11. Napiši program koji crta spiralu – kvadrat koji se smanjuje (npr. stranice: 200, 180, 160, 140...).

Ponavljanje

ZAPAMTI

Petlja for služi za ponavljanje naredbi određeni broj puta.

Naredba range() određuje koliko puta se petlja ponavlja.

Naredbe unutar petlje MORAJU imati uvlaku!

range(4) = 0,1,2,3 (4 ponavljanja); range(1,4) = 1,2,3; range(1,10,2) = 1,3,5,7,9

Kut okretanja za pravilni mnogokut = 360° / broj stranica

Završno ponavljanje

Čestitam! Prošli ste kroz svih 8 lekcija o programiranju u Pythonu.

Ponovimo najvažnije pojmove.

Kratki sažetak

Python je programski jezik kojim pišemo upute računalu.

IDLE je razvojno okruženje u kojem pišemo Python kod.

Interaktivno sučelje (>>>) služi za brzo isprobavanje naredbi.

Uređivačko sučelje služi za pisanje programa koje možemo spremiti.

print() ispisuje sadržaj na ekran.

Varijabla je spremnik s imenom za pohranu vrijednosti.

input() traži unos podataka od korisnika.

int() pretvara tekst u cijeli broj.

Algoritam je jasan postupak rješavanja problema.

Dijagram tijeka je grafički prikaz algoritma.

Petlja for ponavlja naredbe zadani broj puta.

Turtle je modul za crtanje kornjačinom grafikom.

Završna pitanja za ponavljanje

1. Što je Python i čemu služi?
2. Koja je razlika između interaktivnog i uređivačkog sučelja?
3. Što radi naredba print()?
4. Što je varijabla i kako joj pridružujemo vrijednost?
5. Kako tražimo podatke od korisnika?
6. Koji su tri dijela svakog programa?
7. Koja je razlika između sintaktičke i logičke pogreške?
8. Što je algoritam? Navedi primjer.
9. Koja naredba pokreće modul za crtanje kornjačinom grafikom?

10. Kako radi petlja for? Objasni na primjeru.
11. Što znači range(5)? Koje vrijednosti poprima brojač?
12. Kako izračunavamo kut za crtanje pravilnog mnogokuta?

Završni zadaci

🟢 1. Napiši program koji ispisuje tvoje ime 5 puta koristeći petlju.

🟠 2. Napiši program koji od korisnika traži dva broja i ispisuje zbroj, razliku, umnožak i količnik.

🟠 3. Nacrtaj pravilni peterokut (5 stranica) koristeći petlju for.

🔴 4. Napiši program koji od korisnika traži broj i ispisuje tablicu množenja za taj broj (od 1 do 10).

🔴 5. Nacrtaj tri kvadrata, jedan do drugoga, koristeći petlju for i naredbe pu()/pd().

Mali rječnik pojmova

Algoritam – Točan i jasan postupak rješavanja nekog problema.

Bug – Pogreška u programu. Naziv dolazi od engleske riječi za kukca.

Brojač – Varijabla u petlji for koja prati koliko se puta petlja izvršila.

Dijagram tijeka – Grafički prikaz algoritma pomoću dogovorenih simbola.

IDLE – Program u kojem pišemo i pokrećemo Python kod.

input() – Naredba koja traži unos podataka od korisnika putem tipkovnice.

int() – Naredba koja pretvara tekst (znakovni niz) u cijeli broj.

Interaktivno sučelje – Prozor u IDLE-u sa znakom >>> za brzo isprobavanje naredbi.

Logička pogreška – Program radi, ali daje krivi rezultat.

Naredba – Uputa koju dajemo računalu da nešto napravi.

Operator – Znak za matematičku operaciju (+, –, *, /, //, %).

Petlja for – Naredba za ponavljanje bloka naredbi određeni broj puta.

Pridruživanje – Zapisivanje vrijednosti u varijablu pomoću znaka =.

print() – Naredba za ispis sadržaja na ekran.

Program – Datoteka s nizom naredbi koje računalo izvršava redom.

Python – Programski jezik kojim pišemo upute računalu.

range() – Naredba koja definira raspon brojeva za petlju for.

sep – Parametar u print() koji određuje znak između ispisanih vrijednosti.

Sintaktička pogreška – Pogreška u pisanju koda. Program se ne može pokrenuti.

Turtle – Modul za crtanje kornjačinom grafikom u Pythonu.

Uređivačko sučelje – Prazan prozor u IDLE-u za pisanje programa (File → New File).

Varijabla – Spremnik s imenom u koji se pohranjuje vrijednost.

Znakovni niz – Tekst u Pythonu – piše se unutar navodnika ili polunavodnika.

Rješenja odabranih zadataka

Lekcija 1

Zadatak 6: print(5 + 3) ispisuje 8

Zadatak 7: print(10 * 2) ispisuje 20

Zadatak 8: print(3 + 2 * 5) ispisuje 13 (prvo množenje, pa zbrajanje)

Zadatak 9: print('Ana' + 'Marko') ispisuje AnaMarko

Zadatak 10: print('Hello' * 3) ispisuje HelloHelloHello

Zadatak 11: print(11 // 4) ispisuje 2 (cjelobrojno dijeljenje)

Zadatak 12: print(11 % 4) ispisuje 3 (ostatak dijeljenja)

Zadatak 20: Nedostaju navodnici: print('Bok svima')

Zadatak 21: Nedostaje zarez: print('Zbroj je', 5 + 3)

Lekcija 2

Zadatak 6:

```
a = 12
b = 5
opseg = 2 * a + 2 * b
print('Opseg pravokutnika je', opseg)
```

Zadatak 10:

```
duljina = 12
sirina = 5
opseg = 2 * (duljina + sirina)
print('Opseg pravokutnika je', opseg)
```

Zadatak 11: Naziv varijable ne smije početi brojkom. Treba: broj2 = 10

Lekcija 3

Zadatak 5:

```
a = 7
b = 3
print('Zbroj je', a + b)
```

Zadatak 8:

```
a = 8
opseg = 4 * a
povrsina = a * a
print('Opseg kvadrata je', opseg)
print('Površina kvadrata je', povrsina)
```

Zadatak 9: Nedostaje zarez između teksta i izraza: print('Opseg trokuta je', a + b + c)

Lekcija 4

Zadatak 5:

```
a = int(input('Upiši stranicu kvadrata: '))
o = 4 * a
print('Opseg kvadrata je', o)
```

Zadatak 8:

```
a = int(input('Upiši duljinu: '))
b = int(input('Upiši širinu: '))
o = 2 * a + 2 * b
p = a * b
print('Opseg je', o)
print('Površina je', p)
```

Lekcija 5

Zadatak 5 – ispravljeni program: Trebalo bi pisati a*1, a*2, a*3, a*4, a*5 umjesto a+1, a+2...

```
a = int(input('Upiši broj: '))
print('Višekratnici broja', a, 'su:', a*1, a*2, a*3, a*4, a*5)
```

Lekcija 8

Zadatak 4:

```
from turtle import*
for i in range(4):
    fd(80); lt(90)
```

Zadatak 7:

```
for i in range(2, 22, 2):
    print(i, end=' ')
```

Zadatak 10: Da, petlja se izvršava 4 puta. Brojač poprima vrijednosti: 2, 4, 6, 8.