

Za radoznale: Razvojna okruženja i prvi program

Ovaj fajl nije obavezan. Na času smo sve radili najjednostavnijim putem, koji je za prvi program i najbolji. Ovde za nekoliko primera sa časa pokazujemo kako bi isto napisao neko ko programira duže. Ako vam nešto nije jasno, slobodno preskočite, jer većinu ovih stvari radimo na nekom od sledećih časova. Svaki primer je i u folderu `primeri/za_radoznale`, pa ga možete pokrenuti i menjati.

1. Ivica okvira bez brojanja znakova

Na času smo gornju i donju ivicu okvira kucali znak po znak i brojali da svi redovi budu iste dužine.

Na času:

```
print("+~::~::~::~::~::~::~::~::~::~::~::~::~::~::~+")
print("|   Mila Jovanović           |")
print("+~::~::~::~::~::~::~::~::~::~::~::~::~::~::~+")
```

Ispis:

```
+~::~::~::~::~::~::~::~::~::~::~::~::~::~::~+
|   Mila Jovanović           |
+~::~::~::~::~::~::~::~::~::~::~::~::~::~::~+
```

Isti okvir, ali ivicu pravi Python, jer string pomnožen brojem ponavlja se toliko puta:

```
print("+~" + "~" * 28 + "+")
print("|   Mila Jovanović           |")
print("+~" + "~" * 28 + "+")
```

Ispis:

```
+~::~::~::~::~::~::~::~::~::~::~::~::~::~::~+
|   Mila Jovanović           |
+~::~::~::~::~::~::~::~::~::~::~::~::~::~::~+
```

Zapis "~" * 28 znači „znak ~ dvadeset osam puta“, a znak + lepi tri stringa u jedan. Ako hoćemo širi okvir, promenimo samo broj, umesto da kucamo nove znakove i ponovo brojimo.

Zašto ovo nismo radili na času: spajanje i množenje stringova radimo u temi 06, zajedno sa ostalim radom sa stringovima. Na prvom času je važnije da vidimo da print ispisuje tačno ono što smo napisali.

Fajl: `primeri/za_radoznale/05_r1_ivica_ponavljanjem.py`

2. Ceo okvir u jednom stringu

U vežbi Level 3 ceo okvir smo ispisali jednim pozivom print, tako što je `sep="\n"` stavio prelazak u novi red između vrednosti.

Na času:

```
print("+~~~~~+",
      "| Mila Jovanović |",
      "| ilustracija |",
      "+~~~~~+", sep="\n")
```

Ispis:

```
+~~~~~+
| Mila Jovanović |
| ilustracija |
+~~~~~+
```

Isto može i sa trostrukim navodnicima. Tekst između njih može da se prostire kroz više redova, pa okvir u kodu izgleda tačno kao na ekranu:

```
print("""+~~~~~+
| Mila Jovanović |
| ilustracija |
+~~~~~+""")
```

Ispis:

```
+~~~~~+
| Mila Jovanović |
| ilustracija |
+~~~~~+
```

Ovo je višeredni string: prelasci u novi red koje otkucamo u kodu postaju deo teksta. Zato je ovakav zapis zgodan za crteže od znakova, jer ono što vidimo u kodu je ono što dobijamo u ispisu.

Zašto ovo nismo radili na času: trostruke navodnike smo na času samo pomenuli, da bismo ih razlikovali od komentara. Za prvi program je jasnije kad svaki red okvira ima svoj print, jer se lakše vidi gde je greška.

Fajl: `primeri/za_radoznaale/05_r2_okvir_u_jednom_stringu.py`

3. Lepši okvir od posebnih znakova

Na času smo okvir pravili od znakova sa tastature: +, ~ i |.

Na času:

```
print("+~~~~~+")
print("| Mila Jovanović |")
print("+~~~~~+")
```

Ispis:

```
+~~~~~+
| Mila Jovanović |
+~~~~~+
```

Python ume da ispiše i znakove kojih nema na tastaturi, na primer znakove za crtanje okvira sa duplom linijom:

```
print(" ")
print(" Mila Jovanović ")
print(" ")
```

Ispis:

```
Mila Jovanović
```

Ovi znakovi su deo Unikoda, spiska u kom svaki znak iz svih pisama na svetu ima svoj broj. Najlakše ih je kopirati sa neke stranice koja ih nabraja (potražite „box drawing characters“) i nalepiti u kod, između navodnika.

Zašto ovo nismo radili na času: na tastaturi ih nema, pa bismo na času trošili vreme na kopiranje, a neki stariji terminali ih ne prikazuju lepo. Za učenje funkcije print obični znakovi rade isti posao.

Fajl: `primeri/za_radoznale/05_r3_okvir_od_posebnih_znakova.py`

4. Okvir koji se sam prilagodi imenu

Na času je ime upisano u svaki red okvira. Kad promenimo ime, moramo da prepravimo i ivice.

Na času:

```
print("+~~~~~+")
print("| Mila Jovanović |")
print("+~~~~~+")
```

Ispis:

```
+~~~~~+
| Mila Jovanović |
+~~~~~+
```

Ako ime sačuvamo pod jednim imenom u programu (to je promenljiva), Python može sam da izmeri koliko je ime dugo i napravi ivicu tačne dužine:

```
ime = "Mila Jovanović"
sirina = len(ime) + 4
print("+~" + "~" * sirina + "+")
print("| " + ime + " |")
print("+~" + "~" * sirina + "+")
```

Ispis:

```
+~~~~~+
| Mila Jovanović |
+~~~~~+
```

Funkcija `len` broji znakove u stringu, zajedno sa razmacima. Dodajemo 4 jer oko imena stoje po dva razmaka sa svake strane. Promenite ime u prvom redu i pokrenite ponovo: okvir se sam skрати ili produži.

Zašto ovo nismo radili na času: promenljive i funkciju `len` radimo u temi 06. Ovo je program koji smo videli na slajdu „Najava“, samo bez unosa sa tastature.

Fajl: `primeri/za_radoznale/05_r4_okvir_prema_imenu.py`

5. Spisak ključnih reči iz samog Pythona

Na času smo videli tabelu sa svim ključnim rečima, rečima koje imaju posebno značenje i ne mogu biti imena. Ovaj program samo prebroji koliko ih je.

Na času:

```
import keyword
print(len(keyword.kwlist))
```

Ispis:

```
35
```

Isti modul ume i da odgovori da li je neka reč ključna, pa ne moramo da tražimo po tabeli:

```
import keyword
print(keyword.iskeyword("for"))
print(keyword.iskeyword("boja"))
print(keyword.iskeyword("True"))
print(keyword.iskeyword("true"))
```

Ispis:

```
True
False
True
False
```

Modul je gotov skup alata koji dolazi uz Python, a naredba `import` ga uključuje u naš program. Primetite poslednja dva reda: `True` je ključna reč, a `true` nije, jer Python razlikuje velika i mala slova.

Zašto ovo nismo radili na času: module i naredbu `import` radimo u temi 11. Na prvom času je dovoljno da znamo da ključne reči postoje i da ih okruženje boji posebnom bojom.

Fajl: `primeri/za_radoznale/05_r5_kljucne_reci.py`