Funkcije u Python-u

Funkcije

- Funkcija kao crna kutija:
  - Na jednoj strani ima ulaz u koji ubacimo ono što želimo „obraditi”. Mi ne znamo kako funkcija obrađuje ono što smo joj predali (čak nam nije bitno)
  - Na drugoj strani ima izlaz gdje se pojavi rezultat obrade. Kada funkcija završi obradu onoga što smo joj predali na obradu, funkcija nam vrati obrađeni rezultat.
- Ubacite novac u aparat i dobijt ćete kavu, čaj, slatkiš …

Illustration by Freepik Storyset

Funkcije

- Skup instrukcija koje se izdvojene u zasebnu cjelinu.
  - Instrukcije u funkciji izdvojene su kao i u FOR ili WHILE petlji
  - Samo za pokretanje tih instrukcija vrijede drugačija pravila.
- Funkcije pokrećemo pomoću „poziva” funkcije
- Namjena funkcija je izdvojiti kôd koji se ponavlja u zasebnu cjelinu. Tako dobivamo:
  - Jednom napisani kôd možemo koristiti koliko god nam puta treba
  - Održavanje kôda je jednostavnije
  - Debugging je jednostavniji jer nema redundancije (ponavljanja)
- Funkcija bi trebala raditi samo jednu aktivnost. Funkcija koja pohranjuje podatke u bazu, ne treba računati s tim podacima, ne treba pitati korisnika da unese podatke … funkcija iz primjera ima samo jednu funkciju, a to je pohrana podataka u bazu

- Metode – pojednostavljeno, metode su funkcije unutar klasa.

Funkcije imaju

- Naziv
  - Koristi se pokretanje funkcije (ovo još nazivamo ”poziv” funkcije)
  - Vrijede ista pravila kao i za varijable
- Argumente ili parametre
  - To su podaci koje će funkcija prihvatiti na ulazu i obratiti
  - Funkcije mogu, ali i ne moraju imati parametre. To znači da ima funkcija koje ne zahtijevaju ništa na ulazu da bi obavile posao (funkcija koja ispisuje glavni izbornik). Dovoljno je samo pozvati takve funkcije
- Tijelo ili blok instrukcija
  - Predstavlja skup instrukcija koje će izvršiti potrebnu transformaciju ulaznih podataka ili će samo izvršiti određenu akciju.
  - Ovo predstavlja način kako funkcija obavlja svoju namjenu.
- Rezultat ili return parametar
  - Funkcije nakon obrade argumenata „vrate” nekakav rezultat. Neke funkcije ne vrate „ništa”, nego samo ispišu poruku o uspjehu ili neuspjehu aktivnosti.

Funkcije – osnovna podjela

Ugrađene funkcije

- Svaki programski jezik ima ugrađene funkcije
  - help()
  - print()
  - input()
  - int(), float(), str()
  - len()
  - …

Korisnički definirane funkcije

- Funkcije koje kreira programer ovisno o potrebama programa

Ugrađene funkcije

- Do sada smo upoznali neke ugrađene funkcije
- Uz Python se često dodaje „Batteries included”
  - Dakle, većinu toga što trebate dolazi u paketu
- Neke opcije ipak treba uključiti – moduli
- Moduli predstavljaju skup funkcionalnosti koje su izdvojene u zasebnu cjelinu.
  - Moduli imaju svoje specifične funkcije
- Neki su uključeni odmah, a neke module treba uključiti po potrebi
- Dio rezultata `help(print)`:

```
Help on built-in function print in module builtins:
```

Moduli

- Uključivanje modula se radi na početku datoteke pomoću ključne riječi **`import`** iza koje dolazi naziv modula
- Kasnije možemo koristiti ugrađene funkcije iz tog modula
- Neki od ugrađenih modula:
  - **Random** – funkcije za generiranje nasumičnih brojeva
  - **Math** – matematičke funkcije kao potenciranje, kvadriranje, trigonometrijske fukcije i sl.
  - **Datetime** – funkcije za upravljanje tipom varijabli za pohranu vremena (datuma i vremena kao sati, minuta, sekundi)
  - **Os** – funkcije za rad s nekim elementima operativnog sustava na kojem se pokreće Python program. Na primjer datoteke, mape, konzola i sl.
- Ako postoje „ugrađeni” moduli, znači da postoje i „vanjski” moduli. To su moduli koji ne dolaze s instalacijom Pythona i potrebno ih je naknadno instalirati.
  - Takve module ćemo najviše koristiti kod obrade velike količine podataka, a neke i ranije

Korisnički definirane funkcije

- Funkcije koje ovisno o potrebama programa radimo sami
- Kreiranje funkcije u Pythonu:

```
def naziv_funkcije(argumenti):
    """ Docstring """
    instrukcije
```

- Kreiranje funkcije počinje ključnom riječi **def**
- Nakon def dolazi **naziv funkcije**, po pravilima koja vrijede za varijable
- Iza naziva funkcije slijedi **lista argumenata**, međusobno odvojenih zarezom te grupiranih unutar zagrada
- Na kraju prve linije obavezna **:**
- Odmah ispod deklaracije funkcije, **opcionalno** se dodaje ***Docstring***. Opis što radi funkcija te koje argumente prima.
- I nakon toga piše se kôd koji radi ono zbog čega smo i kreirali funkciju.

Zadaci

- Preraditi prethodne vježbe tako da sada koriste funkcije:
  - Je li riječ palindom
  - Baza vozila firme
  - Kamen-Škare-Papir
  - Izradite aplikaciju za konverziju mjernih jedinica tako da korisnik bira koju jedinicu će konvertirati u koju
  - Izradite aplikaciju za preračunavanje potrošnje goriva automobila u kune uz mogućnost izbora izračuna koliko maksimalno auto smije trošiti na 100 km, ako je ciljana mjesečna potrošnja X kuna?

Korisnički definirane funkcije – vježbe

- Kupovina u online dućanima je postala svakodnevica. Za to nam je potrebna kreditna kartica. Ali Internet nije sigurno mjesto.
  Napišite program koji će uneseni broj kartice korisnika zaštititi tako da sve znakove osim zadnja četiri maskira pomoću # znakova. Primjer:
  - Broj: 521478523691234, treba biti: ##########1234
  - **Dodatak:**
  - Dodajte mogućnost da ako je korisnik unio broj kartice s "-" znakovima da se onda ti znakovi ne zaštićuju.
    3698-521-47852, treba biti: ####-###-#7852
  - Dodatno neka korisnik bira kojim znakom će zaštiti broj kartice.

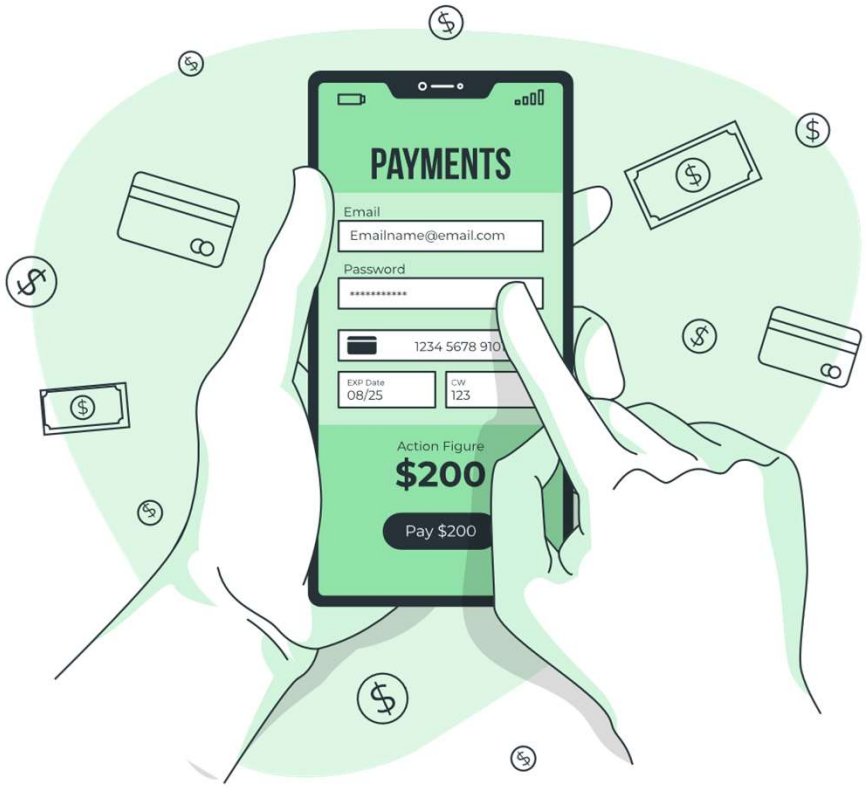

Illustration by Freepik Storyset