

Függvények

Készítette: Vastag Atila

2020

A programozás annak a művészete, hogy a számítógépet olyan feladatok elvégzésére tanítjuk meg, amiket előzőleg nem tudott végrehajtani. Az egyik legérdekesebb erre szolgáló módszer az, hogy felhasználói függvények formájában új utasításokat illesszünk az általunk használt programozási nyelvbe.

Program írásakor az a célunk, hogy minél rövidebb, gyorsabb, karbantarthatóbb és átláthatóbb kódot írjunk.

Egy probléma hatékony megközelítése gyakran a problémának több, egyszerűbb alproblémára való felbontásából (dekompozíciójából) áll, amiket azután külön vizsgálunk. (Ezek az alproblémák esetleg tovább bonthatók még egyszerűbb alproblémákra és így tovább).

Ha munkánk során egy logikai egészet (ugyanazokat a lépéseket) egynél többször használunk, akkor ezt a logikai egészet függvényekbe kell tenni.

A függvények rövidebb, átláthatóbb és karbantarthatóbb egészekre bontják a programunkat (ha a logikai folyamat változtatásra szorul, akkor csak egy helyen kell kijavítani) így könnyítve azok tesztelését is, mert a függvények külön – külön tesztelhetők, hisz mindegyikük egy, **és csakis egy**, logikailag egész folyamatért kell, hogy felelős legyen.

A függvények műveleteket végeznek, és előállítanak egy új adatot, de nem kötelezően (a függvénynek nem kötelező, hogy visszatérési adata legyen). Ezen új értéket hívjuk a **függvény visszatérési értékének**. A függvény visszatérési értéke gyakorlatilag bármi lehet. Elképzelhető egyszerű adattípusok (logikai, egész, szöveg, ...), de akár összetett adattípusok is (lista, objektum, osztály, ...) is.

A függvény definíciós részének általános alakja:

```
def függvényNeve(paraméterlista : paraméter típus) → visszatérési típus:  
    ...  
    utasításblokk  
    ...
```

1. Függvénynévnek a nyelv foglalt szavai kivételével bármilyen nevet választhatunk azzal a feltétellel, hogy semmilyen speciális vagy ékezetes karaktert sem használhatunk (az aláhúzás karakter « » megengedett). A változónevekhez hasonlóan főként a kisbetűk használata javasolt, nevezetesen a szavak elején (a nagybetűvel kezdődő szavakat fenn fogjuk tartani az osztályok számára, amiket a későbbiekben fogunk tanulmányozni).
2. A **def** utasítás az **if**-hez és a **while**-hoz hasonlóan egy összetett utasítás. Az a sor, amelyik ezt az utasítást tartalmazza kötelezően kettősponttal végződik, ami egy utasításblokkot vezet be, amit nem szabad elfelejtenünk behúzni.

3. A paraméterlista határozza meg, hogy argumentumként milyen információkat kell megadni, ha majd használni akarjuk a függvényt. (A zárójel üresen is maradhat, ha a függvénynek nincs szüksége argumentumokra).
4. Egy függvényt gyakorlatilag úgy használunk, mint akármilyen más utasítást. A programtörzsben a függvényhívás a függvény nevéből és az azt követő zárójelekből áll.
Ha szükséges, a zárójelben adjuk meg azokat az argumentumokat, amiket át akarunk adni a függvénynek. Elvileg a függvénydefinícióban megadott mindegyik paraméter számára meg kell adni egy argumentumot, bár adhatók alapértelmezett értékek ezeknek a paramétereknek (lásd később).

Vegyünk egy egyszerű példát egy függvényen melynek az a feladata, hogy összeadjon két számot !

Az **osszeadas** olyan függvény, amely **float** típusú értéket ad vissza.

Paraméterként két **float** típusú adatot fogad.

```
def osszeadas(elsőOsszeadando: float, másodikOsszeadando: float) -> float :  
    összeg: float = elsőOsszeadando + másodikOsszeadando  
    return összeg
```

```
x: float = 5  
y: float = 10
```

```
eredmeny: float = osszeadas(x, y)  
print(f'{x} + {y} = {eredmeny}')
```

Feladata két
szám
összeadása !

függvény
neve

Paraméter
típusai

visszatérő
érték típusa

paraméterek

```
def osszeadas(elsoOsszeadando: float, masodikOsszeadanto: float) -> float :  
    osszeg: float = elsoOsszeadando + masodikOsszeadanto  
    return osszeg
```

```
x: float = 5  
y: float = 10
```

```
eredmeny: float = osszeadas(x, y)  
print(f"{x} + {y} = {eredmeny}")
```

Függvény
meghívása

A függvény megkapja paraméterként az x és y értékét, ahol **elsőOsszeadando** = x-el és a **masodikOsszeadando** = y-al

3

A függvény elvégzi a két változó összeadását, majd a **return** kulcsszó segítségével visszatér arra a programrészre ahonnan meg lett hívva egy **float** típusú értékkel (**osszeg**), mivel **float** típusú a visszatérő érték.

```
def osszeadas(elsőOsszeadando: float, masodikOsszeadando: float) -> float :
```

4

```
    osszeg: float = elsőOsszeadando + masodikOsszeadando
```

```
    return osszeg
```

Az x és y változók értéket kapnak

```
x: float = 5
```

```
y: float = 10
```

```
eredmeny: float = osszeadas(x, y)
```

```
print(f"x + y = {eredmeny}")
```

1

Meghívásra kerül az **Osszeadas** függvény

2

az **eredmeny** változó felveszi a függvény visszatérő érték értékét (osszeg = 15)

5

Vegyünk egy egyszerű példát egy függvényen melynek az a feladata, hogy kiírjon valamilyen szöveget !

A `helloFelhasznalo` olyan függvény, amelynek nincs visszatérő értéke (**None**).

```
def helloFelhasznalo(felaszنالoNev: str) -> None:  
    print(f"Hello {felaszنالoNev}!")  
  
helloFelhasznalo("Tanar Ur")
```

Paraméterként **string** típusú adatot fogad

Feladata valami kiírás!

függvény
neve

paraméter

Paraméter
típusai

visszatérő
érték
NINCS

```
def helloFelhasznalo(felasznaNev: str) -> None:  
    print(f"Hello {felasznaNev}!")
```

```
helloFelhasznalo("Tanar Ur")
```

Függvény
meghívása

A függvény elvégzi a kiírást, majd a **return** kulcsszó segítségével, mert nincs visszatérő érték, visszatér arra a programrészre ahonnan meg lett hívva

3

A függvény megkapja paraméterként a **nev** változó értékét

4

```
def helloFelhasznalo(felasznoNev: str) -> None:  
    print(f"Hello {felasznoNev}!")
```

Az **nev** változó értékét kap

1

```
nev: strig = "Tanar Ur"
```

Meghívásra kerül az **helloFelhasznalo** függvény

2

```
helloFelhasznalo(nev)
```

Az **elsőOsszeadando** és **masodikOsszeadanto** változók csak a függvény testében léteznek, a függvény végrehajtása után megszűnnek létezni

def **osszeadas**(elsőOsszeadando: float, masodikOsszeadanto: float) -> float :

 osszeg: float = elsőOsszeadando + masodikOsszeadanto

 return osszeg

Ügyelni kell arra is, hogy a programunkban ugyanolyan elnevezésű változó globálisan (látható az egész programban / osztályban ne létezzen mint a paraméterekben megadott változók nevei

x: float = 5

y: float = 10

eredmeny: float = osszeadas(x, y)

print(f"{x} + {y} = {eredmeny}")

Függvényírás szabályai:

- mindig definiálni kell a visszatérő érték típusát
- mindig definiálni kell a függvény nevét. Igyekezzünk olyan nevet adni a függvénynek, amely leírással bír arról, hogy a függvénynek mi is a feladata
- a paraméterek típusát kötelező megadni
- ha a függvénynek van visszatérő típusa definiálva, akkor kötelesek vagyunk **return** kulcsszó segítségével olyan típusú adattal visszatérni a meghívás helyére
- **két, vagy több egyforma elnevezésű függvényt NEM írhatunk melyek különböznek paraméter típusban vagy számban (más nyelveknél ez támogatott)**

FÜGGVÉNY PARAMÉTER ALAPÉRTELMEZETT ÉRTÉKE

```
def fizetes(oraber: int, oraszam: int = 40) -> int:  
    return oraszam * oraber
```

```
    tuloraFizetes: float = fizetes(1200, 60);  
    normalFizetes: float = fizetes(1200);
```

Hogy lehetséges az, hogy a **normalFizetes** függvény meghívható, hisz a **fizetes** függvény két paramétert kér?

Ez úgy lehetséges, hogy az **oraszam** paraméternek be van állítva egy alapértelmezett érték (**40**) és ha a **fizetes** függvényt csak egy paraméterrel hívjuk meg (**normalFizetes**), akkor az alapértelmezett értéket veszi, ha két paraméterrel, akkor az **oraszam** paraméter átveszi a függvény meghívásában megadott értéket (**60**).

```
print(tuloraFizetes); //72.000
```

```
print(normalFizetes); //48.000
```

FELADATOK:

1 - Írjunk programot amely összead, kivon, szoroz és eloszt két számot. A matematikai műveleteket függvényekkel oldjuk meg.

2 - Írjunk programot amely megkérdezi a felhasználó nevét majd üdvözlő szöveggel üdvözl.

Egyes logikai egészeket alkotó műveleteket függvényekkel oldjuk meg.

3 - Írjunk programot amely megkérdezi a felhasználó nevét és születési dátumát, majd kiírja : "**Atila ön az idén 39 éves.**"

Egyes logikai egészeket alkotó műveleteket függvényekkel oldjuk meg.

4 - Írjunk programot amely megkérdezi a felhasználó nevét majd üdvözlő szöveggel üdvözl. Az üdvözlő szövegben a nevet olyan színnel írja ki, amelynek a kódja megfelel a név hosszának.

Egyes logikai egészeket alkotó műveleteket függvényekkel oldjuk meg.

5 – Írj egy programot, mely két szöveg típusú adatból meghatározza hány karakterük egyezik meg!

(például: „alma” és „álmatlan” -> 3; „sátortábor” és „bátorság” -> 5; „ágy”, „vágy” -> 3)

Egyes logikai egészeket alkotó műveleteket függvényekkel oldjuk meg.

6 – Írj egy mértékegység konvertáló programot. Az átalakítást Celsius-ból függvényrel oldjuk meg amely két paramétert kap! Az első egy hőmérséklet érték, a második paraméter pedig azt jelzi milyen mértékegységre akarjuk átkonvertálni ('F': Fahrenheit, 'K': Kelvin).
($K = C + 273,15$; $F = 9/5 * C + 32$).

Egyes logikai egészeket alkotó műveleteket függvényekkel oldjuk meg.

7 – Töltsünk fel két 10 elemű tömböt random számokkal. Írjuk ki őket, majd ellenőrizzük mely tömb elemeinek az összege a nagyobb.
Egyes logikai egészeket alkotó műveleteket függvényekkel oldjuk meg.

8 - Írj egy programot mely bekéri két pont koordinátáját (X,Y) koordinátáit és visszaadja a két pont távolságát!
Egyes logikai egészeket alkotó műveleteket függvényekkel oldjuk meg.

9 – Deviza konvertáló programot kell írunk, mely a megadott HUF forintot japán jen, dollár és svájci frankba konvertálja kiválasztás alapján. Az árfolyamokat konstans változóba helyezzük el. Minden konverzió után jelenítsük meg az értéket EUR-ban is, ha tudjuk, hogy $1 \text{ JPY} = 0.75 \text{ EUR}$, $1 \text{ USD} = 0.8 \text{ EUR}$ és $1 \text{ CHF} = 0.55 \text{ EUR}$.
Egyes logikai egészeket alkotó műveleteket függvényekkel oldjuk meg.

10 - Virág és Jázmin számkitalálós játékot készített, a következő módom:

Virág gondolt két számra, az első 0 és 9 közt, a második 40 és 50 közt, majd a számítógép egy véletlen számot (**rnd**) generált titokban e két érték közt, amit Jázmin nem láthatott.

Jázmin feladata az volt, hogy a véletlen számot kitalálja.

A program eredményként Jázmin próbálkozásainak számát írta ki miután kitalálta a számítógép által választott számot. Minden beírt szám után a program Jázmint értesítette, hogy nagyobb vagy kisebb a kitalálandó szám mint amit ő beírt.

11 – Egy kis céget vezetünk, ahol 5 munkás dolgozik nekünk. Minden hét végén el kell számolnunk az alkalmazottak munkabérét. Egy hét 40 munkaórából áll, ez a normál munkidő amit 1000Ft / órában fizetünk ki. Minden túlórát 1500Ft / óra értékben kell kifizetni. Számítsuk ki és írjuk ki a képernyőre, hogy a héten ki hány órát dolgozott, mennyi az össz keresete. A dolgozók nevét és a heti ledolgozott órák számát konzolról kérjük be.