

💡 Funkcie

DEFINOVANIE FUNKCIÍ

📄 **def**

```
def [function_name]([## arguments]):
    [function content]
    ## return [return_variable]
```

PARAMETRE

Typ	SYNTAX
žiadny parameter	()
povinný parameter	(mandatory_param_name)
viacero parameters	(param_1, param_2)
voliteľné parameters	(mandatory_param, optional_param=default_val)
keyword parameters	(**name_of_param_dict)

return STATEMENT

Typ	SYNTAX
jednoduchý	return x
modifikujúci	return x + 1
podmienený	return x if x==0 else y

Viac o funkciách v Pythone sa dozvieš [tu](#) 🔗.

💡 Rekurzívne Funkcie

Ak je funkcia *rekurzívna*, znamená to, že volá samu seba vo svojom tele.


```
def [rec_function_name]([## arguments]):
    if [base_case]:
        return [variable]
    else:
        return [rec_function_name]([## arguments])
```

💡 Kedy použiť Rekurziu

Situácie, kedy je pravdepodobne dobré zvážiť rekurziu, zahŕňajú (ale nie sú obmedzené na):

- 1 Keď sa prichytíš, že píšeš podobné loops vo vnútri loops.
- 2 Keď potrebuješ spracovať problém, ktorého “hlúbku” nepoznáš (ale vieš, že raz skončí).
- 3 Keď sa problém dá rozbiť na menšie identické kúsky.
- 4 Keď dátová štruktúra odkazuje na seba samú (napr. linked lists, grafy, atď.).
- 5 Keď problém zahŕňa backtracking rôznych možných ciest.

💡 Funkcie: Všeobecné Poznámky

- 1 **Telo funkcie nemôže byť prázdne.** Hoci funkcia nepotrebuje ani arguments ani return statements, ak chceš, aby funkcia *nerobila nič*, musíš použiť pass statement (rovnako ako pri **conditionals** ).
- 2 **Vždy sa uisti, že máš správny return type funkcie.** Keď používaš output funkcie na ďalšie výpočty, uisti sa, že typ outputu sedí do zvyšku kódu (inak sa ti môže kód pokaziť alebo nefungovať správne).
- 3 **Vždy sa uisti, že tvoja rekurzia má koniec.** Rovnako ako nekonečné loops, nekonečné volania funkcie sú dobrý spôsob, ako si crashnúť počítač.