2 Funkce

K čemu slouží funkce?

Funkce umožňují rozdělit kód na jednotlivé samostatné celky, které na základě vstupu vygenerují výstup. Tyto celky se dají v programu volat více než jednou, bez nutnosti je znovu psát

Co je funkce?

Funkce je blok kódu, který slouží určitému účelu. Většinou se využívá, pokud je v programu potřeba více než jednou.

- Název
  - Využívá se při volání funkce
- Argumenty
  - Vstupní hodnoty potřebné pro fungování funkce
- Výstupní hodnota
  - Hodnota, kterou funkce vrací na jejím konci

```
# def -> klíčové slovo definující funkci (v JS "function")
# can_drink -> název funkce
# (age, legal_drinking_age) -> argumenty funkce
# return -> klíčové slovo označující výstupní hodnotu funkce
# True -> výstupní hodnota funkce
# False -> výstupní hodnota funkce
def can_drink(age, legal_drinking_age):
    if age >= legal_drinking_age:
      return True
    else:
      return False
# volání funkce
print(can_drink(19, 18))
# >> True
```

Funkce v pythonu

Funkce v pythonu (a jiných programovacích jazycích) má svá specifika

- Pojmenované parametry
  - Při volání funkce je možné jednotlivé parametry výslovně přiřadit určitému argumentu funkce
- Defaultní hodnoty
  - Využívají se při vytváření funkce - jsou použity v případě, že funkci není dodán parametr
- Libovolný počet argumentů
  - Pokud neznáme počet argumentů, který bude do funkce vložen při jejím zavolání, tak je možné použit klíčová slova: (POZOR! pokud používáme pojmenované i nepojmenované argumenty, musí být klíčová slova jako poslední, jinak přeberou i pojmenované parametry)
- *args (Non-Keyword Arguments)
  - Vložené argumenty budou ve formátu [hodnota1, hodnota2,...]
- **kwargs (Keyword Arguments)
  - Vložené argumenty budou ve formátu {key='value', key='value', ...} geeksforgeeks
- Lokální a globální proměnné
  - Lokální proměnné jsou dostupné pouze ve funkci - globální po celém programu. (POZOR! pokud chci ve funkci upravit globální proměnnou je nutné použít klíčové slovo global)

```
# Defaultní hodnoty, pojmenované parametry

# v tomto případě je legal_drinking_age 18, pokud není napsaná jiná hodnota
def can_drink(age, legal_drinking_age = 18):
    if age >= legal_drinking_age:
      return True
    else:
      return False

print(can_drink(age = 18))
# >> True

#print(can_drink())
# >> error

print(can_drink(age = 18, legal_drinking_age= 19))
# >> False
```

```python
# Libovolný počet argumentů

# *args
def preposition(preposition, *args):
  for arg in args:
    print(f"{preposition}{arg}")
  # funkce nemá výstupní hodnotu

preposition("a", "Petr", "Pavel", "Kolo")
# >> aPetr
# >> aPavel
# >> aKolo

# **kwargs
def kwargs(arg1, **kwargs):
    for key, value in kwargs.items():
        print("%s == %s" % (key, value))

# Driver code
kwargs("Hi", first='world', mid='word', last='or')
# >> first == world
# >> mid == word
# >> last == or
```

```python
# lokální a globální proměnné
text = "Hello World"

def func():
  print(text)

func()
# >> Hello World

def func2():
  a = text
#func2()
#print(a)
# >> error

print(text)
# >> Hello World
def func3():
  global text
  text = "asds"

func3()
print(text)
# >> asds
```

Rekurze

Funkce volá sama sebe, aby nakonec dospěla k výsledku. programiz

- Faktoriál zapsaný pomocí rekurze:

```
def factorial(x):
  if x == 1:
    return 1
  else:
    return (x * factorial(x-1))

input_num = int(input("Číslo pro výpočet faktoriálu > "))
print(f"{input_num}! = {factorial(input_num)}")
```

Funkce main

Main funkce je funkce, která se spustí pokud python interpreter otevře python soubor (využívá se hlavně u programování pro Raspberry Pi a další mikro počítače)

Většina jazyků má přímo nějaké klíčové slovo, které se používá jako název funkce a určuje, že je funkce hlavní.

V pythonu se využívá podmínka, která sice pozná, že se má spustit main, ale až když k ní dojde interpreter

realpython

```
print("baf")
def x():
    print("Hello World!")

if __name__ == "__main__":
    x()
```

MATURITA

- Definice a syntax funkce
- Specifika pro python (pojmenované paramtery, *args, **kwargs, globální/lokální proměnné)
- Rekurze
- Ukázka využití funkce nebo rekurzivní funkce

KÓD Z HODINY

```
# Výpočet nové ceny na základě staré ceny a procentuální slevys
def getPrice(name, old_price, discount):
    new_price = old_price - (old_price*(discount/100))
    return [name, new_price]

while True:
    try:
        name = input("Název> ")
        old_price = int(input("Původní cena> "))
        discount = int(input("Procentuální sleva v procentech> "))
        data = getPrice(name, old_price, discount)
        print(f"Název produktu: {data[0]}, cena po slevě: {data[1]}Kč")
        print("--------")
    except Exception as error:
        print(f"Došlo k chybě: \n{error}")
```