Programovací jazyk Python

# Proměnné, deklarace proměnné a datové typy, operátory, print, komentáře.

Proměnné jsou označené oblasti paměti, které se používají k ukládání dat. K přiřazení hodnoty k proměnné se používá znaménko = (rovnítko). Pokus o použití nedefinované proměnné, způsobí chybu. Proměnné mají různé datové typy.

Proměnou pro celé číslo vytvoříme zkratkou int (z anglického integer). Velikost tohoto čísla je limitovaná pamětí počítače.

Proměnným s desetinnými místy ukládáme zkratkou float (floating point number). U desetinných čísel se nepíše čárka, ale tečka. Tato čísla nemohou být neomezeně přesná (4 bajty). Čísla začínající nulou mohou být psána bez nuly – 0.25 nebo .25

Řetězec (zkratka str z anglického string) je definovaný jako sekvence znaků. Je zahájen a ukončen dvojitými (“) nebo jednotnými (‘) uvozovkami. Pokud je nutné, aby náš řetězec obsahoval dvojité uvozovky, napíšeme před ně obrácené lomítko (/). Program nám jinak vytvoří dva řetězce. Pokud definujeme dvě čísla jako řetězce, jejich sčítáním je spojíme. *“25“ + “25“ = 2525 25 + 25 = 50*

Operátory slouží k operacím s hodnotami.

Základní aritmetické výpočty provedeme znaménky + (sčítání), - (odčítání), \* (násobení), / (dělení).

Pomocí znaménka % vydělíme dvě čísla. Výsledkem je zbytek celočíselného dělení – 7%2 = 1

Umocnění provádíme pomocí \*\*. Například: 3\*\*2= 9

Pomocí operátorů můžeme také porovnávat. Dvojité rovnítko (==) znamená “Je rovno“. Není rovno napíšeme jako vykřičník a rovnítko ( != ).

Pro porovnávání větší než a menší než používáme <>. Pro větší nebo rovno a menší nebo rovno používáme <= a =>.

Za pomoci funkce **print** je možné si zobrazit hodnoty jednotlivých proměnných- *print(x)* . Pokud chceme vytisknout text dáme jej do uvozovek- *print(“text“) .* Pro vytisknutí textu s proměnou uvnitř oddělíme text a proměnou čárkou – *print(“text“,x ,“text“)*

# Seznamy, slovníky, n-tice, sady.

Seznam je proměnitelná sekvence objektů v pythonu. Seznamy lze vytvořit vložením různých hodnot oddělených čárkou do hranatých závorek- *list1 = [1, 2, 3]*

N-tice jsou skoro jako seznamy. Vytvoří se stejně, jen vynecháme hranaté závorky. N-tice nejdou měnit. Dají se ale používat v cyklu.

Sady jsou neuspořádané sbírky jedinečných prvků. Pro použití se odstraňují duplikáty. *x = [1, 1, 2]*

*Set(x)*  Příkazem set uděláme ze seznamu sadu. Toto nám odstraní duplikáty. Odstranění duplikátu nefunguje pro řetězce.

Příkazem .add přidáme položku- *sada.add (“položka“)*  a pomocí .discard položku odstraníme*.*

Slovníky jsou seznamy prvků, které jsou odděleny čárkami. Každý prvek je kombinací hodnoty a unikátního klíče. Klíče a elementy jsou odděleny dvojtečkami. Celý slovník je napsán ve složených závorkách. K prvkům slovníku lze přistupovat, aktualizovat je a mazat. Ze slovníku lze vypsat jednotlivé části.

# Moduly, importování modulů

Po ukončení a následném otevření interpretru Pythonu se dříve vytvořené definice, funkce a proměnné se ztratí. Proto při vytváření delšího nebo složitějšího programu je důležité používat textový editor. Když je program připraven, jednoduše jej předejte interpretu k provedení. Tímto vytvoříme typ programu, který se nazývá **skript.** S většími a složitějšími programy je nutné je rozdělit na menší programy pro snadnější údržbu a orientaci. Takto programátor vytvoří hlavní program, do kterého importuje všechny menší potřebné programy. Tyto menší programy se nazývají **Moduly**.

I u menších programů je výhodné importovat existující moduly. Programy je takto snadné znovu využít u jiných projektů. Velmi časté je i využívání cizích volně dostupných modulů k vylepšení vlastního programu. Python má velkou a aktivní komunitu vývojářů, kteří píší a sdílí moduly.

# Větvení programu IF-THEN

Bloky IF THEN umožňují programu se rozhodovat na základě výsledku výrazu. Například pokud se X rovná nule vytiskni “nula“.

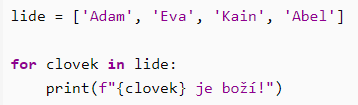
Python také implementuje dvě podstruktury **ELSE** a **ELIF.**

**ELSE** umožňuje specifikovat akci, která má být provedena, pokud je podmínka nepravdivá (X se nerovná 0). Tato část je volitelná.

**ELIF** (ELSE IF) se používá ke stanovení druhé podmínky v případě, že první podmínka je nepravdivá. Může existovat nula nebo více ELIF.

# Smyčky (FOR, WHILE)

Smyčka **For** opakuje libovolné sekvence (seznam nebo řetězce) v pořadí, v jakém se objevují v sekvenci. Například při použití s funkcí print u seznamu jsem smyčka pokaždé vypíše jiné jméno. Začne prvním jménem, pokračuje druhým pak třetím...



Zdroj: https://www.lovelydata.cz/blog/python-for-loops-smycky/

Smyčku lze předčasně ukončit za pomoci **break**. Užitečnou funkcí je také funkce **range**, která udává počet opakování smyčky.

Na rozdíl od smyčky for, která pracuje s prvky z existujícího pole, smyčka typu **while** provádí blok kódu, dokud není splněna nějaká podmínka.

Tělo smyčky je nutné odsadit. Odsazení je způsob seskupování příkazů v Pythonu. Každý řádek musí být odsazen o stejnou hodnotu. Odsazením určíme konec smyčky a začátek další části kódu.