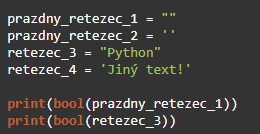
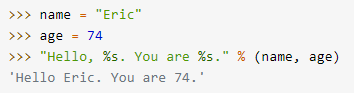
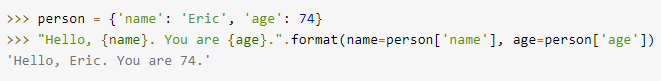
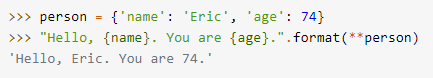
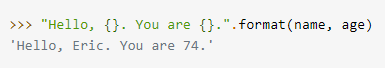
**13. PRG – Řetězce a práce s nimi**

* textové řetězce v pythonu slouží k uchovávání textu
* značí se dvojitými nebo jednoduchými uvozovkami
* trojitými uvozovkami se značí víceřádkový řetězec, při printu se zobrazí i s zalomeními
* řetězec je iterovatelný objekt – skládá se z jednotlivých znaků, ke kterým lze samostatně přistupovat (cyklem, nebo z nich vytvářet seznamy (každý znak se uloží samostatně)), každý znak má index, který začíná od 0
* pokud je řetězec prázdný, vyhodnotí se hodnotou False, pokud v něm něco je tak True

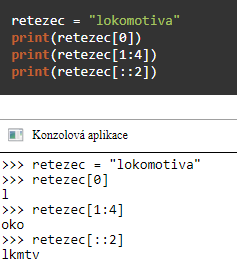
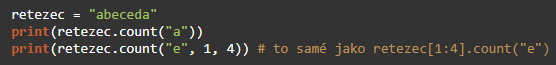
F-string, format():

* **%-formát** – nejstarší a nejméně využívaný kvůli nepřehlednosti programu
* **format()** – ještě se používá, ale je horší jak formát f-string; základní formát je se složenými závorkami, kdyžtak můžeme i měnit pořadí zobrazení prvků podle indexu, protože v tom *format(name, age)* jsou ty proměnné uloženy jako n-tice
  + popřípadě se dá použít i slovník, to se do *format()* dosadí klíče a do řetězce se doplní hodnoty
  + můžeme i určovat datový typ proměnné, *retezec = “{0:3.1f}“.format(a)*, v tomto případě je proměnná typu float a bude mít tři znaky, jeden před des. tečkou, jeden je des. tečka a jedno desetinné číslo
* **f-string** – *retezec = f“Text {promenna}“* – místo promenna se do řetězce dosadí daná proměnná, v současnosti nejvyužívanější a nejpřehlednější

Speciální (escape) znaky:

* \n – nový řádek
* \t – tabulátor
* \\ - zpětné lomítko (jedno)
* uvozovky – pokud řetězec označujeme dvojitými uvozovkami, můžeme uvnitř něj použít jednoduché uvozovky a naopak

Operace s řetězci:

* len() – zjistí délku řetězce; len(‘Řetězec‘)
* slučování – dva řetězce jdou spojit do jednoho pomocí +
* replikace – řetězec lze i replikovat, stačí jej vynásobit číslem, kolikrát jej chceme replikovat
* ord() – převede znak na jeho číselnou ASCII hodnotu
* chr() – převede ASCII hodnotu znaku na samotný znak
* porovnávání - řetězce se dají porovnávat klasickými porovnávacími operátory; porovnávají se podle své ASCII hodnoty, tzn a < b
* řetězce lze i porovnávat, jestli jeden řetězec obsahuje jiný řetězec (vrátí buď True nebo False)
* ořezávání – stejně jako ze seznamu lze i z řetězce extrahovat jednotlivé prvky, k tomuto účelu se používají hranaté závorky a :
* count() – vrátí počet podřetězců v určitém řetězci
* endswith() – vrátí True/False podle toho, jestli řetězec končí zadaným podřetězcem
* find() – vrátí index nejlevější (první) pozice podřetězce v daném řetězci, pokud není nalezen vrací -1
* index() – dělá to samé jako find(), jen pokud nenalezne podřetězec vyvolá výjimku
* isalpha() – vrací True, pokud jsou všechny znaky v řetězci písmena, False pokud ne
* isdigit() / isnumeric() – vrátí True, pokud jsou všechny znaky čísla 0-9
* islower() – vrací True, pokud jsou všechny znaky v řetězci malá písmena
* isupper() – vrací True, pokud jsou všechny znaky v řetězci malá písmena
* lower() – převede znaky v řetězci na malá písmena !neuloží to do původního řetězce
* upper() – převede znaky v řetězci na velká písmena !neuloží to do původního řetězce
* replace() – najde podřetězec a nahradí ho jiným

Datum a čas:

* import datetime nebo time
* získání aktuálního času je *cas = datetime.now()*, obsahuje všechno den, měsíc, rok, hodinu…
* zobrazení jednotlivých elementů je poté *cas.hour, cas.minute, cas.second*