Python ťahák

Premenné

Premenná je krabička (miesto v pamäti), kde si program ukladá dáta. Každá premenná má svoj `typ` (určuje, aké dáta sa tam dávajú), `meno` (jednoznačný identifikátor) a samozrejme obsahuje `hodnotu` príslušného typu.

Typ	Slovný popis	Príklady
int	celé čísla	1, -10, 0, 123
float	desatinné čísla	0.5, .3, 3.14
bool	pravda/nepravda	`True`/`False`
string	reťazec znakov	„abc“, 'abc'
žiadny	ani jeden z typov	`None`

Premenné v Pythone netreba pred použitím vytvoriť, stačí konkrétnej premennej priradiť hodnotu a premennú za nás vytvorí počítač. Pomocou `=` vieme do premennej uložiť novú hodnotu alebo výraz zložený z matematických operátorov a iných premenných.

```
a = 4   # do a priradime 4
b = 2 * a - 3   # do b priradim hodnotu vyrazu napravo, teda 5
c, d = 10, 12   # vieme robit aj viac priradeni naraz (c = 10 a d = 12)
```

Načítavanie vstupu a výpis výstupu

Na načítavanie používame funkciu `input`. Táto načíta jeden riadok zo vstupu a vráti nám ho ako *reťazec (string)*. Na vypisovanie zas použijeme funkciu `print`. Tej môžeme ako parameter dať napr. premennú(é), ktorú chceme vypísať.

```
a = int(input())   # pokial chceme vytvorit premennu a, typu int, musime retazec skonvertovat na cislo
print(a)   # print po vypisani parametra pokazde vypise aj znak konca riadku, nastavenim end
print(x, end='')   # na prazdny znak sa vieme tohto spravania zbavit.
```

List

Ak chceme naraz vytvoriť a používať množstvo premenných, môžeme použiť `list`. List deklaruje naraz množstvo premenných uložených za sebou, ku ktorým sa dá pristupovať pomocou ich pozície. Pozor, ak máme list veľkosti 100, pozície sú číslované od 0, takže sa viem pozerať len na pozície 0 až 99 a **neviem sa pozerať** na pozíciu 100.

```
# list obsahujuci cisla 1 az 5
A = [1, 2, 3, 4, 5]
# list obsahujuci 47 cisel s hodnotou 10
B = [10] * 47

# vypisanie prveho prvko pola a (1)
print(A[0])
```

A takisto, keďže aj list je len typ premennej, viem robiť listy viacrozmerné – listy listov. Tieto listy, si môžeme predstaviť ako tabuľky. Jedno číslo použijeme na nájdenie riadku, druhé na nájdenie stĺpca.

```
K = [ [None] * 5 for i in range(10) ] # vytvori list velkosti 10 * 5
K[4][5] = 18 # do 5. riadku a 6. stlpca ulozi 15

# POZOR! takto vytvoreny dvojrozmerny list sa nebude spravat ako chceme
ZLE = [[1] * 3] * 5
print(ZLE)   # vypise[[1, 1, 1], [1, 1, 1], [1, 1, 1], [1, 1, 1], [1, 1, 1]]
ZLE[0][1] = 2    # chceme zmenit prvok v 1. riadku a 2. stlpci
print(ZLE)    # vypise [[1, 2, 1], [1, 2, 1], [1, 2, 1], [1, 2, 1], [1, 2, 1]]
```

Problémy pri načítavaní vstupu

V Pythone nevieme načítať časť riadku, vieme načítať iba celý riadok. To robí problémy ak sa na jednom riadku nachádza viacero čísel. V takom prípade potrebujeme riadok „rozbiť“ na jednotlivé čísla. Na rozbíjanie slúži metóda `split` ktorá sa dá zavolať na reťazci a vracia list s rozbitými „úsekmi“.

```
strA, strB = input().split() # ak je input() napr. "14 15" tak input().split() vrati ["14", "15"]
a, b = int(strA), int(strB) # do a sa nacita int("14") teda 14 a do b 15
```

Podmienky

Ak chceme aby sa program správal inak za splnenia istej podmienky, použijeme príkaz `if`. Ten zjednodušene vyzerá `if <podmienka>: <prikaz>`. To znamená, že ak platí `<podmienka>` vykonaj `<príkaz>`. **Nezabudnite**, že porovnávanie dvoch vecí sa vykonáva pomocou **dvoch** rovná sa `(==)`.

```
# ak je x rovne 4, takze podmienka plati a vykona sa prikaz print('x = 4')
if x == 4:
  print('x = 4')

# podmienka if s vetvou else, ktora sa vykona ak podmienka neplati
if p < 10:
  print('p < 10')
else:
  print('p >= 10')

# ak nam nestacia 2 vetvy (if, else), mozeme pridat dalsie pomocou 'elif'
if x % 2 == 0:
  print('x je parne')
elif x % 3 == 1:
  print('x je delitelne 3')
else:
  print('x nie je delitelne ani 2 ani 3')
```

Všimnite si, že každý príkaz za dvojbodkou je na samostatnom riadku a je odsadený od podmienky.

Podmienka je ľubovoľný výraz, ktorý vieme vyhodnotiť ako `True` alebo `False`. Takisto však vieme skladať viac výrazov pomocou logických spojok `and` a `or`. Výraz `x and y` sa vyhodnotí ako `True` ak sú obe podmienky x, y pravdivé. Výraz `x or y` sa vyhodnotí ako `True` ak aspoň jedna z podmienok x, y je pravdivá. V podmienkach môžeme používať zátvorky na zvýšenie prehľadnosti poradia operácii.

```
if (x % 2 == 0) and (x % 3 == 0):
  print('x je delitene 6')

# podmienka vyssie je ekvivalentna vnorenym podmienkam
if x % 2 == 0:
  if x % 3 == 0:
    print('x je delitene 6')

if (meno == 'Jozko') or (meno == 'Maria'):
  print('meno je Jozko alebo Maria')
```

Cykly

Ak chceme aby sa nám nejaká časť programu opakovala viackrát, môžeme použiť cykly. Základný cyklus je príkaz `while`.

Na vypísanie čísiel od 1 po 10 môžeme použiť nasledovný kus programu.

```
i = 1
while i <= 10:
  print(i)
  i += 1
```

Kým platí podmienka ($i <= 10$) sa vykonávajú príkazy v odsadenom tele príkazu `while`. Všimnite si, že posledný príkaz ($i \mathrel{+}= 1$) zvýši premennú i a po 10 opakovaniach cyklu bude $i == 11$, a podmienka prestane platiť.

Kratší spôsob v zápisu využíva príkaz `for <premenna> in <list>`. V Pythone existuje funkcia `range` ktorá má 3 varianty:

- `range(kon)` - vytvorí list s prvkami od 0 po $kon - 1$ vrátane.
- `range(zac, kon)` - vytvorí list s prvkami od zac po $kon - 1$ vrátane.
- `range(zac, kon, k)` - vytvorí list s prvkami od zac po $kon - 1$ vrátane, obsahujúc každé k-te číslo.

Vypísanie čísel od 1 do 10 s príkazom `for`.

```
for i in range(1, 11):
  print(i)
```

Vypísanie párnych čísel medzi 20 až 40 by sa zapísalo:

```
for i in range(20, 41, 2):
  print(i)
```