Python

1 Datové typy

V Pythonu se datový typ proměnné nemusí deklarovat. Automaticky se vybere ten správný.

Přesto jsou v Pythonu nástroje, které umožní jeho definici nebo konverzi.

LI Číselné typy:

- **1.1.1 Celé číslo (integer)** maximální velikost je dána platformou.
- **1.1.2 Dlouhé celé číslo (long integer)** omezené pouze velikostí dostupné paměti (musí mít na konci znak písmene l)
- **1.1.3 Reálné číslo (float)** číslo s pohyblivou desetinnou čárkou (povolena pouze desetinná tečka)
- **1.1.4 Komplexní číslo** x = 2+3j
- 1.2 **Boolean:** pokud má hodnotu 0 (či je prázdný), je false. Je-li jiná hodnota než 0, je true

1.3 **Řetězce**

Zapisujeme mezi znaky apostrof nebo uvozovky (trojice apostrofů nebo uvozovek umožňuje víceřádkový řetězec). Řetězce jsou číslovány zleva doprava od pozice 0. Řetězce je možné skládat.

1.3.1 Modul string:

Obsahuje funkce pro práci s řetězci. Volání funkcí z tohoto modulu musí předcházet příkaz import string. Modul obsahuje i některé důležité konstanty

Zapisuje se string.funkce

Funkce			
join	skládání řetězců - provede spojení dvou ř. a mezi ně vloží znak mezery		
count(řetězec, podřetězec)	(vrací číslo typu integer) počet všech výskytů podřetězce v řetězci		
Konstanty			
digits	digits obsahuje znaky 0-9 (všechna čísla)		
hexdigits	obsahuje všechny znaky tvořící základ šestnáctkové soustavy		

2 Datové struktury

Můžeme je uvést přímo v programu jako konstanty s použitím závorek. Všechny datové struktury smějí obsahovat libovolné datové typy (akorát u slovníku nesmí být možné měnit datový typ indexů).

2.1 **N-tice ()**

Jedná se o pole pevné délky i obsahu, používá se třeba při volání funkcí. Chová se podobně jako seznam, ale jednotlivé položky se nemohou modifikovat.

Příklad: x = (1,2,3,4,5)

2.2 **Seznam** []: pole, jehož délku i obsah je možné měnit (realizace pole v tomto jazyku)

Příklad: x=[1, 2, 3, 4, 5]

2.3 Slovník (asociativní pole, hashmapa) { }

Mapuje hodnoty na jiné hodnoty, datový typ použitý jako index musí být neměnný tj. číslo, řetězec nebo výše uvedená n-tice.

Příklad: $x = \{1: 'jedna', 2: 'dva', 3: 'tri'\}$ nebo $x = \{("1", "jedna"), ("2", "dva"), ("3", "tri")\}$ či $x = \{("snih", "bily"), ("tma", "cerna"), ("trava", "zelena")\}$

3 Řídící struktury

3.1 Operátory:

Druh	Co obsahuje	Použití v
aritmetické	+, -, *, /, %, **	aritmetických operacích (v příkazech)
logické	and, or, not	podmínkách (možno jimi podmínku i rozšířit)
relační	==, >=, <=, <> (!=)	podmínkách
	=,>,<	aritmetických operacích

3.1.1 Priorita operátorů

- a) provedení knihovního programu
- b) násobení a dělení
- c) sčítání a odčítání

Prioritu operátorů lze měnit pomocí závorek. Hladina vnoření není omezena. Priorita operátorů je stejná jako v běžně používaných zápisech v matematice.

3.2 Větvení

3.2.1 Zápis: if podmínka :

příkazy

elif podmínka:

příkazy

else: příkazy

názvy příkazů je nutné vždy psát malými písmeny

3.2.2 Rozšíření podmínky příkazu:

continue - po zpracování příkazu je proveden skok na počátek podmínky

break - provedením této konstrukce je podmínka okamžitě ukončena

pass - prázdný příkaz

Tyto konstrukce je možné používat i v příkazech while. Pokud je proměnná typu Boolean, pak se podmínka zapisuje: if a:

print " a má hodnotu true"

V podmíněném příkazu lze uvést kombinaci výrazů obsahující logické proměnné i výrazy.

```
I4E\_Polakova\_\: Python
```

3.3 Cykly

3.3.1 while:

zápis: while podmínka:

příkazy

V cyklu while se podmínka vyhodnocuje až za tělem cyklu. To není u jiných jazyků obvyklé.

3.3.2 for:

Jeho provádění je poněkud odlišné od konstrukce v jazyce PASCAL. Je možné jej doplnit o konstrukci else.

zápis: seznam = [1,2,3,4,5] - hodnota prvků v položce seznam může být prakticky libovolná

for i in seznam:

příkazy

Je možné si příkaz přizpůsobit tak, aby se podobal zápisu v Pascalu:

for i in range (4):

příkazy

V tomto případě nabývá proměnná i postupně hodnot 0,1,2,3.

Cyklus typu FOR může pracovat pouze s celými čísly (nesmí být desetinná).

V příkazu range mohou být i záporná čísla. Např. je někdy vhodné zpracovávat jednotlivé bity v bytu z leva.

V tomto případě je zápis následující: for i in range (7,-1,-1):

příkazy

34 Funkce

Musí být vždy deklarována před jejím použitím. Tato definice může být umístěna kdekoliv v programu.

zápis: def jméno(seznam parametrů) :

tělo funkce

return (zde může být seznam parametrů) - zde končí deklarace

jméno(parametry)

příkazy

3.5 Práce se soubory

příkazy:

otevření souboru nazev = open ("nazev1", "kod")

uzavření souboru nazev.close() - nazev musí být stejný jako nazev použitý v deklaraci open

čtení ze souboru readline - přečte aktuální řádek a uloží ho jako řetězec do proměnné

readlines - uloží celý obsah souboru po jednotlivých řádcích do proměnné

a = jmeno.readline/s()

zápis do souboru nazev.write(a) - proměnná a obsahuje řetězec znaků, které budou zapsány jako

řádek do souboru

nazev - symbolické jméno, které přiřadím souboru

nazev1 - skutečné jméno souboru v adresáři včetně přípony(může být i proměnná bez apostrofů, která obsahuje textový řetězec názvu souboru)

kod - udává činnost, kterou budu se souborem provádět.

- r otevřen pouze pro čtení
- w otevřen pouze pro zápis
- a otevřen pro přidání záznamu za již existujícím záznamem
- r+ otevřen pro čtení i pro zápis.

3.6 Matematická knihovna math

obsahuje funkce math- volána příkazem import math

- sin, cos, tan goniometrické funkce
- sqrt druhá odmocnina
- pow mocnina čísla
- ..

Lze použít knihovnu pro práci s komplexními čísly cmath., což vede ke zjednodušení výpočtů při práci s komplexními čísly. Každá funkce z knihovny math má svůj protějšek z cmath (např. sqrt – csqrt apod.)

3.7 KNIHOVNA RANDOM

Používá se pro generování pseudonáhodných čísel v zadaném intervalu či rozložení. Pseudonáhodné číslo není úplně vygenerováno náhodně, ale je zde jistá vazba na nastavení generátoru náhodných čísel či vazba na předchozí generované číslo.

Volání funkcí z tohoto modulu musí předcházet příkaz import random.

3.8 Funkce knihovny:

- seed(x) provede počáteční inicializaci generátoru náhodných čísel (není-li x uvedeno pro inicializaci se vezme okamžitý údaj z generátoru náhodných čísel)
- randit(min,max) vrací náhodné číslo z uzavřeného intervalu min, max (číslo je celé)
- chr(x) převádí čísla na znaky, je vnitřní součástí prostředí jazyka Python (x-číslo či jiný znak)
- ...

4 Struktura programu

Jedná se o moderní skriptovací programovací jazyk, který je objektově orientovaný (strukturovaný - **záleží na odsazování**). Kromě objektového programování je schopen také pracovat jako klasický programovací jazyk (slučuje klasické i objektově orientované programování). Je jistým pokračovatelem jazyka PASCAL, používá některé shodné konstrukce a podporuje strukturu.

Jistým vylepšením jazyka Python oproti jazyku PASCAL je absence příkazu skoku (ten strukturované programování jistě nepodporuje). Kromě výše uvedených vlastností jazyka Python podporuje práci v počítačových sítích. Je obohacen o velké množství knihovních programů, které každý uživatel může přímo využívat resp. může stávající knihovní programy rozšiřovat o další moduly.

- 4.1 **Proměnné** prakticky libovolné jméno (doporučuje se používat písmen malé abecedy)
- 4.2 **Přiřazení** x = 0, můžeme použít i vícenásobného přiřazení: x = y = z = 0
- 4.3 **Tisk na obrazovku -** příkaz print

chceme-li na jeden řádek zobrazit více proměnných, pak je oddělujeme čárkami

I4E_Polakova_ Python

4.4 **Vstup dat** - raw input (ze systémového zařízení)

Přečteme řetězec znaků, který bude ukončen klávesou ENTER.

4.5 **Komentáře** # - následující znaky, až do konce řádku, se ignorují (taky možné použít apostrofy)