Олексій Васильєв

Мова програмування Python

Київ 2020

Лекція 7.
Числові значення

- Основні числові типи
- Числові літерали
- Системи числення
- Арифметичні оператори
- Побітові оператори
- Дроби
- Комплексні числа

Числові типи

Числові типи:

`int`	цілі числа
`float`	дійсні числа
`complex`	комплексні числа

Крім числових типів `int`, `float` і `complex`, можна використовувати типи `Fraction` і `Decimal` для роботи з раціональними дробами і цілими числами фіксованої точності

Літерали:

- в двійковій системі починаються з префіксу `0b` (або `0B`)
- в вісімковій системі починаються з префіксу `0o` (або `0O`)
- в шістнадцятковій системі починаються з префіксу `0x` (або `0X`)

В шістнадцятковій системі: для позначення чисел від `10` до `15` використовуються букви від `A` до `F` (як великі, так і маленькі)

Цілочислові літерали, задані в системах численнях, відмінних від десяткової, інтерпретуються як цілі числа з відповідним десятковим значенням

Змінні, яким як значення присвоюється двійкове, вісімкове або шістнадцяткове значення, можуть використовуватися у розрахунках як звичні змінні з відповідним цілочисловим значенням

Функції:

`bin()`	отримати двійковий код
`oct()`	отримати вісімковий код
`hex()`	отримати шістнадцятковий код
`int()`	отримати десяткове число

Якщо текстове представлення для числа містить префікс системи числення (`0b`, `0o` або `0x`), то відповідний текстовий вираз можна передати аргументом функції `eval()`

В Python підтримуються системи числення з основою від `2` до `36`. Якщо основа системи числення більше `10`, в якості "цифр" використовуються букви від `A` (значення `10`) до `Z` (значення `35`). За допомогою функції `int()` на основі текстового представлення для числа в довільній системі числення (з основою від `2` до `36`) можна отримати число. Для систем числення з основою `2`, `8` або `16`, можна задавати літерали

Системи числення

Програма (Nums.py)

```
print("Числа задані в різних системах:")
# Число 19 в двійковій системі:
A=0b10011
print("A =",A)
# Число 93 у вісімковій системі:
B=0o135
print("B =",B)
# Число 2603 в шістнадцятковій системі:
C=0xA2B
print("C =",C)
print("C-A-B =",C-A-B)
# Обернене перетворення:
print("Обернене перетворення:")
A=int("10011",2)
print(bin(A),"=",A)
B=int("0o135",8)
print(oct(B),"=",B)
C=int("0xA2B",16)
print(hex(C),"=",C)
D=int("ABC",20)
print("ABC =",D)
```

```
Числа задані в різних системах:
A = 19
B = 93
C = 2603
C-A-B = 2491
Обернене перетворення:
0b10011 = 19
0o135 = 93
0xa2b = 2603
ABC = 4232
```

Оператори

Арифметичні оператори:

- Оператор додавання +. Результат виразу виду A+B з числовими операндами A і B розраховується як сума значень цих операндів. Може використовуватися як унарний оператор (наприклад, вираз виду +A)
- Оператор віднімання –. Результат виразу виду A–B розраховується як різниця значень числових операндів A і B. Може використовуватися як унарний оператор (наприклад, вираз виду –A)
- Оператор множення *. Результат виразу виду A*B з числовими операндами A і B розраховується як добуток значень операндів A і B
- Оператор ділення /. Результат виразу виду A/B розраховується діленням значення оператора A на значення оператора B
- Оператор цілочислового ділення //. Результат виразу виду A//B розраховується діленням націло значення операнду A на значення операнду B: розраховується результат (звичного) ділення значення операнду A на значення операнду B, а отриманий результат округлюється до найближчого цілочислового значення (в напрямку мінус нескінченості)
- Оператор розрахунку від цілочислового ділення % . Результатом виразу виду A%B є залишок від ділення значення операнду A на значення операнду B: значення виразу виду A-(A//B)*B
- Оператор піднесення в степінь **. Результатом виразу виду A**B є число, яке отримується піднесенням значення операнду A в степінь, яка визначається значенням операнду B

Двійкові числа

Принцип бінарного кодування чисел

Позиційне представлення додатних чисел:

$\overline{a_n a_{n-1} \dots a_2 a_1 a_0} = 2^0 a_0 + 2^1 a_1 + 2^2 a_2 + \dots + 2^{n-1} a_{n-1} + 2^n a_n$ **параметри** $a_k = 0,1$**.**

Від'ємні числа:

Додатне число $x = \underbrace{0011010 \dots 011011}_{n}$

Інверсія коду $\sim x = \underbrace{1100101 \dots 100100}_{n}$

Сума чисел $x + \sim x = \underbrace{1111111 \dots 111111}_{n}$

Сума чисел $x + \sim x + 1 = 1\underbrace{0000000 \dots 000000}_{n}$

- **Щоб отримати код від'ємного числа, необхідно взяти код протилежного числа, замінити в ньому нулі на одиниці і навпаки, і прибавити до результату одиницю**

- **Якщо код починається з одиниці, то це від'ємне число. Щоб зрозуміти, яке це число, слід інвертувати код, додати до нього одиницю, отримане значення перевести в десяткову систему і "дописати" знак "мінус"**

Оператори

Побітові оператори:

- Побітове і `&`. Результат виразу виду `A&B` - число: код отримується попарним порівнянням значень бітів в операндах `A` і `B`. Якщо обидва біти дорівнюють `1`, в результаті отримуємо одиничний біт. Інакше отримуємо нульовий біт

- Побітове або `|`. Результат виразу виду `A|B` - число: код отримується попарним порівнянням значень бітів в операндах `A` і `B`. Якщо хоча б один біт дорівнює `1`, в результаті отримуємо одиничний біт. Інакше в результаті отримуємо нульовий біт

- Побітове виключне або `^`. Результат виразу виду `A^B` - число: код отримується попарним порівнянням бітів в операндах `A` і `B`. Якщо значення бітів різні, то в результаті отримуємо одиничний біт. Інакше в результаті отримуємо нульовий біт

- Побітовий зсув вліво `<<`. Результат виразу виду `A<<n` - число: код отримується зсувом вліво коду числа `A` на кількість позицій, які визначаються значенням операнду `n`. Молодші біти заповнюються нулями. Старші біти втрачаються

- Побітовий зсув вправо `>>`. Результат виразу виду `A>>n` - число: код отримується зсувом вправо коду числа `A` на кількість позицій, яка визначається значенням операнду `n`. Старші біти заповнюються значенням знакового біту. Молодші біти втрачаються

- Побітове інвертування `~`. Результат виразу виду `~A` - число: код отримується з коду числа `A` заміною нулів на одиницю і одиниць на нулі

Побітові операції

Програма (BinOps.py)

```
# Змінні:
A=13
print("A =",A)
B=7
print("B =",B)
# Побітові операції:
print("A&B =",A&B)
print("A|B =",A|B)
print("A^B =",A^B)
print("~A+1 =",~A+1)
# Побітові зсуви:
print("B>>1 =",B>>1)
print("B<<2 =",B<<2)
```

```
A = 13
B = 7
A&B = 5
A|B = 15
A^B = 10
~A+1 = -13
B>>1 = 3
B<<2 = 28
```

```
13 = 1101
 7 = 0111
```

```
& 1101
  0111
  0101 = 5
```

```
| 1101
  0111
  1111 = 15
```

```
^ 1101
  0111
  1010 = 10
```

```
A    000...001101
~A   111...110010
~A+1 111...110011
```

```
B    000...00111
B>>1 000...00011 = 3
```

```
B    000...00111
B>>2 000...11100 = 28
```

Дроби і числа

```
from fractions import Fraction
from decimal import Decimal
print("Дробові значення:")
A=Fraction(2,5)
print("A =",A)
B=Fraction(3,7)
print("B =",B)
C=A+B
print("A+B =",C)
print("Дійсні числа:")
X=2/5
print("X =",X)
D=X+B
print("X+B =",D)
print("Числа заданої точності:")
A=Decimal('1.01')
print("A =",A)
B=Decimal('2.02')
print("B =",B)
C=A+B
print("A+B =",C)
print("1.01+2.02 =",1.01+2.02)
```

- **Реалізація дробів:**
  тип `Fraction`
  з модуля `fractions`

- **Числа фіксованої точності:**
  тип `Decimal`
  з модуля `decimal`

Програма (FracDec.py)

```
Дробові значення:
A = 2/5
B = 3/7
A+B = 29/35
Дійсні числа:
X = 0.4
X+B = 0.8285714285714285
Числа заданої точності:
A = 1.01
B = 2.02
A+B = 3.03
1.01+2.02 = 3.0300000000000002
```

Комплексні числа

Уявна одиниця: $i^2 = -1$
Комплексне число: $z = x + iy$
Комплексно спряжене: $z^* = x - iy$
Модуль: $|z| = \sqrt{zz^*} = x^2 + y^2$
Тригонометрична форма: $z = |z| \exp(i\phi)$
Аргумент: $\cos(\phi) = \frac{x}{|z|}$ **и** $\sin(\phi) = \frac{y}{|z|}$
Формула Ейлера: $\exp(i\phi) = \cos(\phi) + i\sin(\phi)$

- **Уявна частина визначається за допомогою суфіксу `j` або `J`**

Наприклад: `3+4j`

- **Функція `complex`:**

Наприклад: `complex(3,4)`

```
# Комплексні числа:
A=3+4j
print("A =",A)
print("Re(A) =",A.real)
print("Im(A) =",A.imag)
print("|A| =",abs(A))
B=-1+1j
print("B =",B)
C=complex(0,1)
print("C =",C)
# Арифметичні операції:
print("A+B =",A+B)
print("A*C =",A*C)
print("A/B =",A/B)
```

Програма (Complex.py)

```
A = (3+4j)
Re(A) = 3.0
Im(A) = 4.0
|A| = 5.0
B = (-1+1j)
C = 1j
A+B = (2+5j)
A*C = (-4+3j)
A/B = (0.5-3.5j)
```

Завдання - 1

Напишіть програму, в якій користувач уводить основу для системи числення (2, 8 або 16) і число (в десятковій системі), а програма відображає це число у відповідній системі числення

Завдання - 2

Напишіть програму, в якій користувач уводить ціле число і номер біту, значення якого потрібно визначити в програмі

Домашнє завдання

[1] Напишіть програму, в якій користувач уводить два комплексних числа, а програма розраховує суму, різницю, добуток і частку цих чисел. Серед отриманих значень визначити ті, у яких найбільший і найменший модуль

[2] Напишіть програму, в якій користувач уводить два раціональних дроби, а програма розраховує суму, добуток, різницю і частку цих дробів, серед отриманих значень знаходить найбільше і найменше, і відображає результати розрахунків