УДК 519.688(575.2) (04)

МУСУРАЛИЕВА Д.Э., ОМУРАЛИЕВА Б.Б.
Ж. Баласагын атындагы КУУ/
КНУ им. Ж. Баласагына
MUSURALIEVA D.E., OMURALIEVA B.B.
J. Balasagyn KNU

PYTHON ПРОГРАММАЛОО ТИЛИНДЕГИ НЕГИЗГИ САНДЫК АМАЛДАР

Основные числовые операции на языке программирования Python

Basic numeric operations in the Python programming language

Аннотация: *Бул макалада Python программалоо тилиндеги негизги сандык аракеттер камтылган. Python бардык арифметикалык операцияларды колдойт: арифметика, дайындоо, конверсия функциялары жана сандарды чагылдыруу. Ушул программалоо тилинде программа кодун ишке ашыруунун мисалдары келтирилген.*

Аннотация: *В данной статье рассматриваются основные числовые операции на языке программирования Python. Python поддерживает все распространенные арифметические операции: арифметические операции, арифметические операции с присвоением, функции преобразования чисел и представление числа. Приводятся примеры реализации программного кода на данном языке программирования.*

Abstract: *This article covers basic numeric operations in the Python programming language. Python supports all common arithmetic operations: arithmetic, assignment, conversion functions, and number representation. Examples of the implementation of the program code in this programming language are given.*

Урунттуу сөздөр*: сандык амалдар, Python, арифметикалык амалдар, арифметикалык дайындоо, сандарды өзгөртүү функциялары жана сандарды көрсөтүү.*

Ключевые слова: *числовые операции, Python, арифметические операции, арифметические операции с присвоением, функции преобразования чисел и представление числа.*

Keywords: *numeric operations, Python, arithmetic operations, assignment arithmetic, number conversion functions, and number representation.*

Арифметикалык амалдар. Python тили, ар кандай эсептөө маселелерин чечүү үчүн көптөгөн китепканалардын болушуна байланыштуу, бүгүнкү күндө эң маанилүү пакеттердин бири. Интерактивдүү түрдө ишке киргизилгенде, ал кубаттуу калькуляторго айланат. Бул макалада ушул тилдеги арифметикалык амалдар жөнүндө сөз кылабыз. Арифметикалык амалдарды сандарга карата изилдеп, татаал сандар менен иштөөнү өзүнчө талдайбыз. Ошондой эле, бирдик операциялары, ар кандай эсептөө тутумундагы сандардын чагылдырылышы жөнүндө жана кыскача математика китепканасына токтолобуз. Эгерде кандайдыр бир арифметикалык туюнтманын операнддары катары бүтүндөй сандар гана колдонулса, анда натыйжа дагы бүтүн сан болот. Бөлүнүү операциясы, анын натыйжасында чыныгы сан чыгат. Бүтүн жана чыныгы өзгөрүлмөлөрдү чогуу колдонгондо, натыйжа чыныгы болот.

- +

Эки санды кошуу:
1 чыгаруу (5 + 2) # 7

- -

Эки санды алып салуу:
1 басып чыгаруу (7 - 2) # 5

- *

Эки санды көбөйтүү:

1 чыгаруу (5 * 2) # 10

• /

Эки санды бөлүштүрүү:

1 чыгаруу (8/2) # 4 .0

• //

Эки сандын бүтүн бөлүнүшү:

1

2 чыгаруу (5/2) # 2 .5

print (5 // 2) # 1

Бул операция бөлүүчүнү жокко чыгарып, бөлүнүүнүн бүтүн натыйжасын берет.

• **

Көрсөтүү:

1 басып чыгаруу (8 ** 2) # 8ди 2 денгээлине көтөрүңүз. Жыйынтыгы - 64

• %

Бөлүмдүн калган бөлүгүн алуу:

1 чыгаруу(8 % 2) # 8ди 2ге бөлгөндөн кийин, калганын алыңыз. Жыйынтык - 2

Мындай учурда, 8-дин жакын саны 2 калган жок тарабынан бөлүнүүчү болуп саналат,

6 болот ошондуктан бөлүнүүнүн калган саны 8 - 6 = 2 болот.

Бир нече арифметикалык операцияларды ырааттуу колдонуу, аларды аткаруу алардын артыкчылыгына ылайык жүргүзүлөт. Биринчи кезекте жогорку артыкчылыктуу операциялар жасалат. Төмөндөө тартибинде операциялардын артыкчылыгы төмөнкү 1-таблицада көрсөтүлгөн .

Операциялар	Багыты
**	Оңдон солго
* / //%	Солдон оңго
+ -	Солдон оңго

1-таблица. Иштин артыкчылыктары

Төмөнкү туюнтма аткарылсын:

1

2 чи саны = 2 + 5 * 6 ** 2 + 8

басып чыгаруу (номери) # 190

Бул жерде учурда башынан саптуу (6 ** 2) жогорку артыкчылык менен иш катары жүзөгө ашырылат, анда натыйжасы болуп саналат (4 * 5 36 көбөйтүлгөн), андан тышкары (2 + 18 0) пайда болот жана андан кийин кошумча (182 + 9).

Иш-аракеттердин тартибин кайрадан аныктоо үчүн кашаа колдонсо болот:

1

2 саны = (5 + 3) * (6 ** 2 + 9)

басып чыгаруу (номери) # 360

Арифметикалык амалдарга бүтүн жана бөлчөк сандар да катыша ала тургандыгын белгилей кетүү керек. Эгерде бир амалга бүтүндөй (int) жана калкып чыккан чекиттин номери (float) катышса , анда бүтүн сан float түрүнө чыгарылат .

Тапшыруу менен арифметикалык амалдар

Бир катар атайын операциялар операциянын жыйынтыгын биринчи операндга ыйгарууга мүмкүндүк берет . (2-таблица)

+ =
Кошуунун натыйжасын берүү
- =
Чакыруунун жыйынтыгын дайындоо

* =
Көбөйтүүнүн натыйжасын берүү
/ =
Бөлүнүүнүн натыйжасын берүү
// =
Бүтүн бөлүүнүн натыйжасын берүү
** =
Сандын кубаттуулугун берүү
% =
Бөлүнүүнүн калган бөлүгүн берүү

2-таблица . Атайын операциялар

Операциялардын мисалдары :

```
1
2
3
4
саны = 5
саны + = 5
басып чыгаруу (сан) # 10
саны - = 2
басып чыгаруу (сан) # 8
саны * = 5
басып чыгаруу ( номери ) # 40
```

Номерди өзгөртүү функциялары

Pythondo орнотулган бир катар функциялар сандар менен иштөөгө мүмкүнчүлүк берет. Атап айтканда, int () жана float () функциялары , тиешелүүлүгүнө жараша , int жана float түрлөрүнө маани берүүгө мүмкүнчүлүк берет .

Мисалы, бизде төмөнкүдөй код бар дейли:

```
1
2
3     first_numb er = "4"
экинчи_сандар = 5
үчүнчү_сандар = биринчи_сандар + экинчи_сандар
```

"4" + 5 =9 болот деп күтүп жатабыз. Бирок, бул код өзгөчө кырдаалды жаратпайт , анткени биринчи сан сапты билдирет. Баары иштеши үчүн, int () функциясын колдонуп, сапты санга айлантуу керек :

```
1
2
3
4   first_number = "4"
экинчи_сандар = 5
үчүнчү_сандар = int (биринчи_сандар) + экинчи_сандар
басып чыгаруу ( ҮЧҮНЧҮ, _ саны ) # 9
```

Float () функциясы окшош ыкма менен иштейт , ал өзгөрүлмө чекиттин номерине өтөт. Бирок жалпысынан, бөлчөк сандар менен, алар менен иштөөнүн натыйжасы толугу менен так болбой калышы мүмкүн экендигин эске алуу керек. Мисалы:

```
1
2
3
4   биринчи_сандар = 4.0001
экинчи_сандар = 9
үчүнчү_сандар = биринчи_сандар / экинчи_сандар
```

басып чыгаруу (ҮЧҮНЧҮ, _ саны) # 0.80002000000000008

Бул учурда, биз 0.80002 санын алабыз деп күтүп жатабыз, бирок аягында, нөлдөрдүн катарынан кийин, дагы төртөө пайда болот. Же дагы бир сөз:

1 -жөнү (4,0001 + 0,1) # 4,1001000000000003

Бул учурда, натыйжаны тегеректөө үчүн round () функциясын колдонсок болот :

```
1
2
3
4       биринчи_сандар = 4.0001
экинчи_сандар = 0.1
үчүнчү_сандар = биринчи_сандар + экинчи_сандар
басып чыгаруу ( тегерек ( үчүнчү _ саны , 8)) # 4.1001
```

Функциянын биринчи параметри - тегеректелүүчү номер, экинчиси - алынган сан канча ондук белгилерден турушу керек.

Санды көрсөтүү

Сандык өзгөрмөнүн кадимки аныктамасында ага ондук маани берилет. Бирок Pythondo ондуктан тышкары экилик, сегиздик жана он алтылык тутумдарды колдонсок болот.

Бинардык тутумдагы санды аныктоо үчүн, 0 жана b префикси анын маанисинин алдына коюлат :

```
1    x = 0 b 101    # 101   5ке барабар
```

Сегиздик тутумдагы санды аныктоо үчүн, 0 жана префикстин маанисинин алдына жайгаштырылат:

```
1      a = 0 o 11      # 11 сегиздик 9 болот
```

Он алтылык тутумундагы бир санды аныктоо үчүн, анын маанисинин алдында 0 жана префикси x :

```
1       y = 0x0 a          #  алтылыкта 10
```

Башка өлчөө тутумдарындагы сандар менен арифметикалык амалдарды жүргүзсөңүз болот:

```
1
2
3
4       x = 0b101 # 6
y = 0x0a # 12
z = x + y # 18
print ("{0} бинардык түрдө {0: 08b} алтылыкта {0: 02x} сегиздикте {0: 02o}". формат (z))
```

Эсептөө ар түрдүү системаларында бир катар өзгөрүүлөр үчүн, формат милдети колдонулат сап боюнча аталган. Бул сапка ар кандай форматтар өткөрүлүп берилет.

"{0:08 b }" бинардык тутуму үчүн, 8 саны санда канча белги болушу керектигин көрсөтөт.

Эгерде номерге талап кылынгандан ашыкча белгилер көрсөтүлсө, анда керексиз кызмат орундары нөл менен толтурулат. Он алтылык үчүн формат "{0:02 x }". Бул жерде бардыгы бирдей - номер жазуусу эки белгиден турат, эгер бир белгинин кереги жок болсо, анын ордуна нөл киргизилет. Ал эми сегиздик белгилөө үчүн "{0:02 o }" форматы колдонулат.

Скриптеги иштин натыйжасы: 15 сегиздиктеги алтылык ичиндеги 00001111 экилик санында 17

Демек, бул программалоо тилинин синтаксиси кадимки математикалык операцияларга окшош, мында өзгөчө кыйынчылыктар болбойт. Тилдин негизги операторлору, сөз айкаштары жана көптөгөн камтылган маалыматтардын түрлөрү каралып, Python менен иштөө принциптери кыскача түшүндүрүлүп, расмий Python программалоо стилинин эрежелери келтирилди.

Пайдаланылган адабияттар:

1. PyCharm - интеллектуалдык Python IDE [Электрондук ресурс]. - URL: https://jetbrains.ru/products/pycharm/
2. Tkinter Python китепканасы боюнча курсу [Электрондук ресурс]. - URL: https://ru.wikipedia.org/wiki/Python
3. Грамаков Д.А. Жалпы жана педагогикалык кесиптик билим берүү тутуму үчүн программалоо тилдерин тандоо. // Заманбап маалыматтык технологиялар жана IT билим берүү. IV Эл аралык илимий-практикалык конференциясынын материалдары. М., Москва мамлекеттик университети. 2009.
4. Билл Лубанович Жөнөкөй Python. Заманбап программалоо стили. - SPb.: Питер, 2016. - 480 б.: - ("Bestseppers O'Reilly" сериясы).
5. Lutz M. Python программалоосу, II том, 4-басылышы. - Per. англис тилинен - SPb.: Symbol-Plus, 2011. - 992 б.

Рецензент: Мекенбаев Б.Т. - *физика - математика илимдеринин кандидаты, Билим берүүдөгү заманбап информациялык технологиялар институтунун информациялык технологиялар, менеджмент жана укук кафедрасынын доценти.*