Оборудование и материалы.

Персональный компьютер, среда разработки Python.

Указания по технике безопасности:

Соответствуют технике безопасности по работе с компьютерной техникой.

Задания

Языки программирования также позволяют вам определять переменные. Переменные являются именами, которые ссылаются на значения в памяти компьютера. Вы можете определить их для использования в своей программе. В Python символ = применяется для присваивания значения переменной.

В следующей программе целое число 7 присваивается переменной с именем a, затем на экран выводится значение, связанное в текущий момент с этой переменной:

```
>>> a = 7
>>> print(a)
7
```

Сделайте следующее с помощью интерактивного интерпретатора.

1. Как и раньше, присвойте значение 7 имени a. Это создаст объект-«переменную», содержащую целочисленное значение 7.
2. Выведите на экран значение a.
3. Присвойте a переменной b, заставив b прикрепиться к объекту-«переменной», содержащей значение 7.
4. Выведите значение b.

```
>>> a = 7
>>> print(a)
7
>>> b = a
>>> print(b)
7
```

В Python, если вы хотите узнать тип какого-то объекта (переменной или значения), вам следует использовать конструкцию type(объект). Попробуем сделать это для разных значений (58, 99.9, abc) и переменных (a, b):

```
>>> type(a)
<class 'int'>
>>> type(b)
<class 'int'>
>>> type(99.9)
<class 'float'>
>>> type('abc')
<class 'str'>
```

Целые числа

Любая последовательность цифр в Python считается целым числом:

```
>>> 5
5
```

Последовательность цифр указывает на целое число. Если вы поместите знак + перед цифрами, число останется прежним, знак – сделает число отрицательным:

```
>>> 123
123
>>> +123
123
>>> -123
-123
```

С помощью Python вы можете выполнять обычные арифметические действия, как и с обычным калькулятором, используя операторы, показанные в теоретической части работы. Сложение и вычитание будут работать следующим образом:

```
>>> 5+9
14
>>> 100-7
93
>>> 4-10
-6
>>> 5+9+3
17
>>> 4 + 3 - 2 - 1 + 6
10
>>> 5+9  +      3
17
```

Вы можете заметить, что число пробелов в формуле не имеет значения.

Аналогично выглядит операция умножения.

Операция деления чуть более интересна, поскольку существует два ее вида:

- с помощью оператора / выполняется деление с плавающей точкой (десятичное деление);
- с помощью оператора // выполняется целочисленное деление (деление с остатком).

Даже если вы делите целое число на целое число, оператор / даст результат с плавающей точкой:

```
>>> 9/5
1.8
>>> 9//5
1
```

Деление на ноль с помощью любого оператора сгенерирует исключение:

```
>>> 5/0
Traceback (most recent call last):
  File "<pyshell#51>", line 1, in <module>
    5/0
ZeroDivisionError: division by zero
>>> 7//0
Traceback (most recent call last):
  File "<pyshell#52>", line 1, in <module>
    7//0
ZeroDivisionError: integer division or modulo by zero
```

Введите следующие команды:

```
>>> a = 95
>>> a
95
>>> a-3
92
```

Ранее, когда мы выполнили операцию a – 3, мы не присвоили результат переменной a, поэтому ее значение не изменилось. Выведите значение a на экран:

```
>>> print(a)
95
```

Если вы хотите изменить значение переменной a, придется сделать следующее:

```
>>> a = a - 3
>>> print(a)
92
```

Вы можете совместить арифметические операторы с присваиванием, размещая оператор перед знаком =. В этом примере выражение a -= 3 аналогично выражению

a = a– 3. Введите в командную строку:

```
>>> a = 95
>>> a-=3
>>> a
92
```

Это выражение аналогично выражению a = a + 8:

```
>>> a+=8
>>> a
100
```

Аналогично работает умножение:

```
>>> a*=2
>>> a
200
```

Здесь представлен пример деления с плавающей точкой, a = a / 3:

```
>>> a/=3
>>> a
66.66666666666667
```

Присвоим значение 13 переменной a, а затем попробуем использовать сокращенный вариант a = a // 4 (целочисленное деление):

```
>>> a = 13
>>> a//=4
>>> a
3
```

Символ % имеет несколько разных применений в Python. Когда он находится между двух чисел, с его помощью вычисляется остаток от деления первого числа на второе:

```
>>> 13 % 5
3
```

Вот так можно получить частное и остаток одновременно:

```
>>> divmod(13,5)
(2, 3)
```

Только что вы увидели кое-что новое: функцию с именем divmod, в которую передаются целые числа 9 и 5, возвращающую двухэлементный результат, называемый кортежем. Кортежи будут рассмотрены в следующих работах.

Системы счисления.

В Python вы можете выразить числа в трех системах счисления помимо десятичной:

- 0b или 0B для двоичной системы (основание 2);
- 0o или 0O для восьмеричной системы (основание 8);
- 0x или 0X для шестнадцатеричной системы (основание 16).

Интерпретатор выведет эти числа как десятичные. Попробуем воспользоваться каждой из систем счисления. Первой выберем десятичное число 10, которое означает «одна десятка и ноль единиц». Затем то же число в двоичной системе:

```
>>> 10
10
>>> 0b10
2
```

Затем попробуйте перевод из восьмеричной и шестнадцатеричной систем:

```
>>> 0o10
8
>>> 0xf
15
>>> 0xef
239
```

Преобразования типов

Для того чтобы изменить другие типы данных на целочисленный тип, следует использовать функцию int(). Она сохраняет целую часть числа и отбрасывает любой остаток.

Простейший тип данных в Python — булевы переменные, значениями этого типа могут быть только True или False. При преобразовании в целые числа они представляют собой значения 1 и 0:

```
>>> int(True)
1
>>> int(False)
0
```

Преобразование числа с плавающей точкой в целое число просто отсекает все, что находится после десятичной запятой:

```
>>> int(98.6)
98
>>> int(1.0e4)
10000
```

Далее рассмотрим пример преобразования текстовой строки (со строкамив Python вы познакомитесь в следующей лабораторной работе, но, возможно, знаете о них из других языков программирования), которая содержит только цифры и, возможно, знаки + и -:

```
>>> int('99')
99
>>> int('-23')
-23
>>> int('+12')
12
```

Применение операции int() к целому числуничего в нём не изменит.

Если вы попробуете преобразовать что-то непохожее на число, сгенерируется исключение:

```
>>> int('10 негритят пошли купаться в море')
Traceback (most recent call last):
  File "<pyshell#93>", line 1, in <module>
    int('10 негритят пошли купаться в море')
ValueError: invalid literal for int() with base 10: '10 негри
в море'
>>> int('')
Traceback (most recent call last):
  File "<pyshell#94>", line 1, in <module>
    int('')
ValueError: invalid literal for int() with base 10: ''
```

Текстовая строка в предыдущем примере начинается с корректных символов-цифр (10), но продолжается теми символами, которые функция int() обработать не может.

Какова размерность целочисленного типа?

В Python 2 размер переменной типа int был ограничен 32 битами. Этого было достаточно для того, чтобы сохранить любое целое число в диапазоне от –2 147 483 648 до 2 147 483 647. Для переменных типа long выделялось еще больше места — 64 бита, что позволяло хранить значения в диапазоне от –9 223 372 036 854 775 808 до 9 223 372 036 854 775 807.

В Python 3 тип long отсутствует, а переменная типа int может быть любого размера — даже больше 64 бит. Поэтому вы можете выполнить арифметические действия наподобие следующих (10**100 называется гугол (googol), компания Google называлась так до того, как решила сменить свое название на более простое):

```
>>> 10**100
10000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000
0000000
```

Вещественные числа

Значениями целочисленного типа являются целые числа, но числа с плавающей точкой (в Python они называются float) имеют десятичную запятую. Числа с плавающей точкой обрабатываются так же, как и целые: вы можете использовать операторы +, –, *, /, //, **, % и функцию divmod().

Для того чтобы преобразовать другие типы в тип float, следует использовать функцию float().

Математические функции

Python имеет привычный набор математических функций вроде квадратного корня, косинуса и т. д. Они будут рассмотрены в лабораторных работах, где будет рассмотрено применение Python в науке.

Упражнения

В этойработе были показаны атомы Python: числа, строки и переменные. Используя полученные знания, выполним несколько небольших упражнений по работе с ними с помощью интерактивного интерпретатора.

1. Сколько секунд в часе? Используйте интерактивный интерпретатор как калькулятор и умножьте количество секунд в минуте (60) на количество минут в часе (тоже 60).
2. Присвойте результат вычисления предыдущего задания (секунды в часе) переменной, которая называется seconds_per_hour.
3. Сколько секунд в сутках? Используйте переменную seconds_per_hour.
4. Снова посчитайте количество секунд в сутках, но на этот раз сохраните результат в переменной seconds_per_day.
5. Разделите значение переменной seconds_per_day на значение переменной seconds_per_hour. Используйте деление с плавающей точкой (/).
6. Разделите значение переменной seconds_per_day на значение переменной seconds_per_hour. Используйте целочисленное деление (//). Совпадает ли полученный результат с ответом на предыдущее упражнение, если не учитывать символы .0 в конце?

***Содержание отчета*:выполненную работу сохранить в папке под своей фамилией и показать преподавателю**

Контрольные вопросы

1. Каковы особенности синтаксиса языка Python?
2. Какие вычислительные операции применимы к целочисленным типам данных? К вещественным?
3. Что такое строковый тип? Какие операции применимы к строковому типу?
4. Каковы особенности ввода данных в Python?
5. Каковы особенности вывода данных в Python?
6. Что такое IDLE?
7. Какой командой определяется тип переменной?
8. Как применить операцию деления нацело?
9. Что произойдет при делении на 0 с помощью оператора /? С помощью //?
10. Если ввести последовательность команд

```
>>> a = 95
>>> a
95
>>> a – 3
92
```

Что будет выведено после команды print(a)?

11. Как теперь получить a = 92?
12. Что означает выражение a -= 3?
13. Какие способы вывода значения переменной на экран в IDLEвы знаете?
14. Что будет выведено в результате?

```
>>> a = 13
>>> a//=4
>>> a
```

15. Что получится в результате вычисления выражения:

```
>>> 13 % 5
3
```

16. Что это за операция? Что получается в результате?

```
>>> divmod(13,5)
(2, 3)
```

17. Что будет выведено в результате команды

```
>>> 0o10
```

18. Что будет выведено в результате

```
>>> 0xf
```

19. Что будет выведено в результате команды

```
>>> 0b10
```

20. Что будет выведено в результате команды

```
>>> 0b100
```

21. Что будет выведено в результате применения функции:

```
>>> int(5.2e5)
```

22. Что будет выведено в результате команды:

```
>>> True + 2
```

23. Что будет выведено в результате команды:

```
>>> 10**100
```

Список литературы, рекомендуемый к использованию по данной теме:

1. Михеева, Е. В. Информационные технологии в профессиональной деятельности : учеб. пособие / Е.В. Михеева. - 14-е изд., стер. - М. : Академия, 2016. - 384 с.
2. Гохберг, Г. С. Информационные технологии : учебник / Г.С. Гохберг, А.В. Зафиевский, А.А. Короткин. - 9-е изд., перераб. и доп. - М. : Академия, 2014. - 240 с.
3. Хлебников, А. А. Информационные технологии : учебник / А. А. Хлебников. – М. :КноРус, 2014. – 472 с.
4. Любанович Билл. Простой Python. Современный стиль программирования. — СПб.: Питер, 2016. — 480 с.

Лабораторная работа 2. Работа со строками в языке Python.

Цель работы: