§17. Оператор присваивания, ввод и вывод данных

Программа на языке программирования представляет собой последовательность операторов (инструкций, команд), ведущих к решению поставленной задачи.

Оператор - языковая конструкция, представляющая один шаг из последовательности действий или набор описаний.

Присваивание

Запись значения в переменную выполняет оператор присваивания. Общий вид оператора:

<имя переменной> = <значение или вычисляемое выражение>

<куда класть> = < что класть>

Оператор присваивания не имеет словесного обозначения, только символ =

Операция присваивания допустима для всех уже известных вам из параграфа типов данных.

Примеры:

```
a = 25                  # переменная целого типа
b = "Привет"            # переменная строкового типа
c = 1.4 + 5.7 * a       # переменная вещественного типа
d = a < c               # переменная логического типа
A = 25.0                # переменная вещественного типа
e = "Мир! " + b  # переменная строкового типа
f= x * (a + c) / 3
```

В правой части оператора присваивания нельзя указывать переменные, которые не были заранее созданы (определены). Так, для переменных c и d все входящие в соответствующие выражения переменные были заданы выше. Последняя строка содержит ошибку, так как переменная x из правой части ранее не была создана (определена).

В языке Python разрешено **множественное присваивание**. Запись a,b = 19, 25 равносильна паре операторов присваивания:

```
a = 19
b = 25
```

При этом считается, что эти два действия происходят параллельно, т. е. одновременно. Если двум переменным присваивается одно и то же значение, можно применить **множественное присваивание «по цепочке»**: a = b = 5

Эта запись равносильна паре операторов b = 5 и a = 5.

Сокращённая запись	Полная запись
a + = b	a = a + b
a - = b	a = a - b
a * = b	a = a * b
a / = b	a = a / b
a ** = 2	a = a ** 2

Вывод данных

Оператора присваивания достаточно для того, чтобы записать программу преобразования данных, произвести вычисления, но результат этих преобразований виден не будет.

Для вывода данных из оперативной памяти на экран компьютера используется оператор (функция, команда) вывода print ():

print(<выражение 1>, <выражение 2>, ... , <выражение N>)

Здесь в круглых скобках помещается список вывода - список выражений, значения которых выводятся на экран. Это могут быть числовые, символьные и логические выражения, в том числе константы и переменные.

Изучение языков программирования принято начинать с программы, выводящей на экран надпись: «Привет, мир!». На Python соответствующая команда будет иметь вид:

print (‘ Привет, мир! ’)

Последовательность символов, которые должны быть выведены на экран, заключается в двойные кавычки или апострофы и записывается в круглых скобках. В начале строки (левее команды print()) не должно быть пробелов - таково требование языка Python.

Оператор print(' s= ' , s) выполняется так:

1) на экран выводятся символы, заключённые в апострофы: s=

2) на экран выводится значение переменной с именем s.

Если значение переменной s равно 15, и она имеет целочисленный тип, то на экране появится: s= 15

Напишем программу для вычисления значения выражения $\frac{2 * 17{,}56^2}{7 * 2{,}47 * 0{,}43}$

```
print((2 * 17.56 * 17.56) / (7 * 2.47 * 0.43))
```

Так как у нас имеется дробь, то результатом будет являться вещественное число.

При запуске программы на экране будет выведено число 82.94984330235246

Если в качестве операндов некоторого арифметического выражения используются только целые числа, то результат тоже будет целое число. Исключением является операция деления, результатом которой является вещественное число. При совместном использовании целочисленных и вещественных переменных, результат будет вещественным.

Ввод данных

Для ввода в оперативную память значений переменных с клавиатуры используется функция ввода **input()** (от англ. input -*ввод*):

```
a= input ()
```

Пара скобок говорит о том, что мы вызываем функцию. Их надо писать обязательно, даже если в скобках ничего нет.

При выполнении этой команды (после запуска программы на выполнение) программа ожидает от пользователя ввода последовательности символов с клавиатуры; после того, как пользователь нажимает клавишу **Enter**, набранная им символьная строка вводится в переменную (в «домик» в оперативной памяти компьютера) с именем *a*. Это значит, что в памяти выделяется область необходимого размера, с ней связывается имя *a*, и в этой области сохраняются все полученные символы. То есть по умолчанию, input вводит данные в указанную переменную типа str.

Если мы планируем работать не со строками, а с числами, то сразу же после считывания необходимо выполнить преобразование типов при помощи соответствующей функции:

`a = int(a)` - для целых чисел (преобразует string в integer)

`a = float(a)` - для вещественных чисел (преобразует string в float).

Считывание строк и преобразование типов рекомендуется объединять:

`a = int(input())` - для ввода целого числа в переменную a;

`a = float(input())` - для ввода вещественного числа в переменную a.

Функции int() и float() работают без ошибок, если введённая строка состоит ТОЛЬКО ИЗ ЧИСЕЛ.

Можно совмещать вывод подсказки и ввод данных, указывая текст подсказки в скобках как аргумент функции input ():

```
a = float(input(' Введите длину: '))
```

Программирование линейных алгоритмов

Пользуясь рассмотренными операторами, составим несколько простейших линейных программ.

Программа *ввода вещественного числа и вывода его абсолютной величины* может выглядеть так:

```
x = float(input())
print (abs (x))
```

Составим *программу, вычисляющую длину окружности и площадь круга с радиусом r см.*

Исходным данным в этой задаче является радиус: r. Его значение введём с клавиатуры.

Результатом работы программы должны быть величины c и s:

c - длина окружности и s - площадь круга. c, s и r - величины вещественного типа.

Исходное данное и результаты связаны соотношениями, известными из курса математики:

$c = 2\pi r, \quad s = \pi r^2$

Для ввода используем строку с функцией input() и выводом подсказки:

```
r = float(input('Введите радиус: '))
```

Для подсчета используем оператор присваивания:

```
c = 2*3.14*r
s = 3.14*r**2
```

Для вывода ответа воспользуемся дважды оператором print(). Выведем текстовое сообщение и значение переменной.

```
print('c= ', c)
print('s= ' , s)
```

Программа будет иметь вид:

```
r = float(input('Введите радиус: '))
c  = 2*3.14*r
s  = 3.14*r**2
print('c=', c)
print('s= ', s)
```

Запустив программу на выполнение, мы увидим на экране написанный ранее нами текст «Введите радиус» и компьютер окажется в режиме ожидания. Он будет ждать, пока мы введем запрошенные данные для переменной r с клавиатуры в строку ввода. Пусть это будет число 5.4. Подтвердив ввод клавишей Enter, мы запустим продолжение выполнения программы. Компьютер посчитает и на экране выведет ответ. Таким образом, на экране получится следующий результат (протокол работы программы).

```
Введите радиус:8.5
c= 33.912000000000006
s= 91.56240000000003
```

Запишем на языке Python *программу нахождения суммы цифр вводимого с клавиатуры натурального двухзначного числа.*

Используем тот факт, что положительное двухзначное число можно представить в виде следующей суммы: x = a · 10 + b , где a – количество десятков в числе и его первая цифра, b – количество единиц в числе и его вторая цифра. Если положительное двухзначное число x разделить нацело на 10, то получится первая цифра числа (число десятков). А остаток от такого деления есть последняя цифра числа (число единиц). Например, 25 : 10 = 2 (ост 5) значит, 25 // 10 = 2 ; 25 % 10 = 5

```
x  =  int(input (' Введите двузначное число:   ' ))
a = x  //  10
b  =  x  %  10
s  =  a  +  b
print ( ' сумма цифр = ' ,  s)
```

Эту программу можно записать и короче, сократив количество используемых переменных:

```
x  =  int(input (' Введите двузначное число:   ' ))
print ( ' сумма цифр =   ' ,  (x  //  10)  +  (x  %  10))
```

А если число X трехзначное, то как отделить его цифры?
Команда a = X // 100 отделит число сотен.
Команда b = X % 10 отделит число единиц.
Команда c = (X % 100) // 10 или (X // 10) % 10 отделит число десятков.

А как *перевернуть трехзначное число* (например, было 123, а стало 321)?
После отделения цифр, собрать его «наоборот», сделав единицы сотнями, а сотни единицами.

```
X1 = b * 100 + c * 10 + a
```

Составим программу для *перевода сантиметров в метры и сантиметры*.

Например, 435 сантиметров = 4 метра 35 сантиметров.

Этого можно добиться, если

1) число 435 поделим нацело на 100, 435 // 100 = 4 метра.
2) возьмем остаток от деления 435 на 100, 435 % 100 = 35 сантиметров.
3) Выведем оба ответа.

```
A = int(input('введите число сантиметров'))
print(A, ' сантиметров = ', A // 100, ' метров ', A % 100, ' сантиметров ')
```

Форматирование ввода и вывода данных

Ввод нескольких данных в одной строке

Каждый оператор ввода input() захватывает только одну строку данных, причем захватывает её целиком. Для того чтобы ввести в одной строке два целых числа, разделённых пробелом (например, 10 ␣ 20), используют **функцию** split() (от англ. *split* - расщепить). Можно воспользоваться следующей последовательностью команд:

```
a, b = input().split()        #ввод двух строковых величин, разделенных пробелом.
a, b  =  int (a),  int (b )    #преобразование к целому типу
```

Аналогично организуется считывание трёх и более переменных:

```
a, b, c  =  input().split()
```

Для преобразования к целому типу переменных a, b, c можно использовать конструкцию:

```
a, b, c  =  int(a), int(b), int(c)
```

Сократить запись считывания нескольких значений и их преобразования в числовой тип можно с помощью **функции map**, которая применяет к каждому элементу списка заданное правило.

```
a, b, c = map(int, input().split())
```

Здесь с помощью функции map организовано применение функции int() к каждому элементу вводимого списка.

Составим *программу для вычисления значения арифметического выражения* $x = \frac{-b+\sqrt{|b^2-4ac|}}{2a}$

```
from math import *
a,b,c = map(float, input('введите значения для a  b  c ').split() )
x = (-b + sqrt(abs(b**2 - 4 * a * c)))/ (2 * a)
print('x=', x)
```

Особые разделители для входных данных.

Теперь рассмотрим ситуацию, когда входные данные заданы в одной строке, но разделены особыми разделителями, отличными от пробела. Типичным примером таких входных данных являются показания времени (10:33). В таких случаях надо для split() указывать конкретный символ разделителя, взятый в двойные или одинарные кавычки.

В этом примере: `hours, minutes  = input().split(':')`

Разделитель sep

По умолчанию выводимые выражения в функции print() разделяются одним пробелом, иначе говоря, разделителем между ними является пробел. Функция print() вставляет между выводимыми значениями так называемый **разделитель** (сепаратор, от англ. *separator*). Его можно изменять, указывая новый разделитель после служебного слова sep.

Вариант организации вывода	**Оператор вывода**	**Результат**
По умолчанию	print (1, 20, 300)	1 20 300
Убрать разделители — пробелы	print (1, 20, 300, sep=' ')	120300
Добавить другой разделитель (например, запятую)	print (1, 20, 300, sep=', ') print (a, b, c , sep=', ')	1, 20, 300
Вывод каждого значения с новой строки	print(l , 20 , 300, sep= ' \n ') print (a, b, c , sep=' \n ')	1 20 300

Форматный вывод

Предположим, что мы работаем с натуральными числами, каждое из которых меньше 100. Тогда на одно число на экране достаточно выделить 3 позиции: две позиции на запись самого числа и ещё одну позицию на пробел слева, разделяющий числа.

Записывается это так: print('{: 3}{: 3}{: 3} '.format (a, b, c))

Это форматный вывод: строка для вывода строится с помощью функции **format**. Аргументы этой функции (a, b и c) - это имена тех величин, значения которых выводятся; они указываются в круглых скобках.

Символьная строка слева от точки - это форматная строка, которая определяет формат вывода, т. е. как именно значения величин будут представлены на экране. Фигурные скобки обозначают место для вывода очередного элемента; число после двоеточия - количество позиций, которые отводятся на число; на первом месте выводится значение a, на втором - значение b, на третьем - c.

Если цифр в числе меньше, чем зарезервированных под него позиций на экране, то свободные позиции дополняются пробелами слева от числа. Если указанное в формате вывода число меньше, чем необходимо, то оно автоматически будет увеличено до минимально необходимого.

Например, числа 12, 5 и 15 будут выведены так: ␣12 ␣ ␣ 5 ␣ 15

Числа 12, 5 и 1500 будут выведены следующим образом: ␣ 12 ␣ 51500

(␣ обозначение пробела)

Для вывода вещественного числа в списке вывода для каждого выражения указываются два параметра:

1) общее количество позиций, отводимых на число;

2) количество позиций в дробной части числа:
d - целые числа (int);
f - вещественные (float);
e - экспоненциальный формат.

Фрагмент программы	Результат выполнения оператора вывода
`a = 4` `print ("a=", "{:5d}{:5d}".format(a, a * a))`	a= □□□□4□□□16
`a = 1 / 3; b = 1 / 9` `print ("{:7.3f}{:7.4f}".format(a, b))`	□□0.333□0.1111
`a = 1 / 3` `print ("{:10.3e}".format(a))`	□3.333e-01

Улучшим внешний вид выводимого результата в ранее составленной программе для вычисления длины окружности и площади круга, использовав вывод по формату.

```
r = float(input('Введите радиус: '))
c  = 2*3.14*r
s  = 3.14*r**2
print ("c=", "{: 7.4f}".format(c))
print ("s=", "{: 7.4f}".format(s))
```

Протокол работы программы станет следующий:

```
Введите радиус:8.5
c= 53.3800
s= 226.8650
```

Данная программа позволяет вычислять длину окружности и площадь круга для любого значения r. Она решает не единичную задачу, а целый класс подобных задач. В программе понятно и удобно организован ввод исходных данных и вывод получаемых результатов. Это обеспечивает дружественность пользовательского интерфейса.

При выполнении очередного оператора print() по умолчанию вывод продолжается в новой строке. Чтобы убрать переход к новой строке, используется параметр end:

```
print (a,  end=" ")  #  убран переход на новую строку
print(b)
```

Контрольные вопросы.

1. Что представляет собой программа на языке программирования?
2. Что такое оператор в языке программирования?
3. Как обозначается оператор присваивания в языке Python?
4. Что означает запись x +=y на языке Python? Как это можно записать иначе?
5. Какой оператор используется для вывода информации на экран в языке Python?
6. Какая функция используется для ввода значений переменных с клавиатуры в языке Python? Какого типа вводятся значения по умолчанию?
7. Как ввести в переменную X любое целое число? А как вещественное? Запишите соответственные команды.
8. Перепишите программу для вычисления длины окружности и площади круга из параграфа без ошибок, добавьте комментарии к каждой строке (поясните, что каждая строка делает).
9. Используя команды и примеры, написанные в параграфе, запишите программу, вычисляющую произведение цифр трехзначного числа. Прокомментируйте каждую строку программы. *(2б)*
10. Используя команды и примеры, написанные в параграфе, запишите программу, переворачивающую трёхзначное число. *(2б)*
11. Как организовать ввод значений нескольких переменных в одной строке? Запишите пример.
12. Как организовать преобразование к числовому (целому, вещественному) типу значений нескольких переменных в одной строке? Запишите примеры.
13. Как сократить запись считывания нескольких значений и их преобразования в числовой тип? Запишите пример.
14. Запишите программу для вычисления значения выражения $d = \frac{\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}}{2}$. Используйте ввод данных в одной или двух строках. *(2б)*
15. Как указать особый разделитель для входных данных? Приведите пример.
16. Для чего используется служебное слово **sep?** Поясните на примерах.
17. Что вы знаете о форматном выводе? Поясните на примерах.