§11. ОБЩИЕ СВЕДЕНИЯ О ЯЗЫКЕ ПРОГРАММИРОВАНИЯ PYTHON

Языки программирования - это формальные языки, предназначенные для записи алгоритмов, исполнителем которых является компьютер. Алгоритмы, записанные на языках программирования, называют программами.

Существует несколько тысяч языков программирования. Один из самых популярных современных языков программирования называется Python (произносится «пайтон», хотя в России многие называют язык просто «питон»). Его разработал в 1991 году нидерландский программист Гвидо ван Россум. Язык Python непрерывно совершенствуется, и сейчас большинство программистов используют его третью версию - Python 3.

Python - язык программирования высокого уровня, предназначенный для решения самого широкого круга задач. С его помощью можно обрабатывать различные данные, проводить математические вычисления, создавать изображения, работать с базами данных, разрабатывать веб-сайты.

Характеристики языка программирования

Каждый язык программирования, как и любой формальный и неформальный язык, имеет свой алфавит, синтаксис и семантику.

Алфавит - набор допустимых символов, которые можно использовать для записи программы.

Синтаксис - система правил образования конструкций языка (правописание)

Семантика - система правил, определяющих смысл и способ употребления конструкций языка (значение слов)

Алфавит и словарь языка Python

Основой языка программирования Python, как и любого другого языка, является **алфавит** - набор допустимых символов, которые можно использовать для записи программы. Это:

• латинские прописные и строчные буквы (A, B, C, "., X, У, Z, a, Ь, c, "., x, y, z);

• арабские цифры (0, 1, 2, 3, 4, 5, 6, 7, 8, 9);

• специальные символы (знак подчёркивания; круглые, квадратные скобки; знаки арифметических операций; # - знак начала однострочного комментария и др.).

В качестве неделимых элементов (составных символов) рассматриваются следующие последовательности символов:

>= не меньше (больше или равно)

<= не больше (меньше или равно);

!= не равно;

" " " или ' ' ' (утроенные двойные или одинарные кавычки, ставящиеся в начале и в конце многострочного комментария).

В языке также существует некоторое количество различных цепочек символов, рассматриваемых как единые смысловые элементы с фиксированным значением. Такие цепочки символов называются **служебными словами**. В таблице приведены некоторые основные служебные слова.

Служебное слово языка Python	Значение служебного слова
and	и
break	прервать
elif	иначе если
else	иначе
False	ложь
float	вещественный тип данных (с плавающей точкой)
for	для
if	если
input	ввод
integer	целый
list	список
not	не
or	или
print	печать
string	строковый тип данных (цепочка символов)
True	истина
while	пока

Для обозначения переменных, программ и других объектов используются **имена** (**идентификаторы**) - любые отличные от служебных слов последовательности букв, цифр и символа подчёркивания, начинающиеся с буквы или символа подчёркивания. Прописные и строчные буквы в именах различаются, например, f и F - две разные переменные. Длина имени может быть любой. Для удобства рекомендуется использовать имена, передающие смысл объектов, с длиной не более 15 символов.

Для основных арифметических операций в языке Python используются следующие обозначения:

+ сложение
- вычитание
* умножение
/ обычное деление
// целочисленное деление
% определение остатка от деления
** возведение в степень

Типы данных, используемые в языке Python

В языке Python используются различные типы данных

Название	**Обозначение**	**Допустимые значения**	**Область памяти**	**Пример значений**
Целочисленный	**int** (от integer)	сколь угодно большие, размер ограничен оперативной памятью		8 -25 0
Вещественный	**float**	Любые действительные числа с дробной частью	чаще 8 байт (точность 15 знаков после запятой)	2.5 -32.456 81.0
Строковый	**str** (от string)	Любые символы из таблицы Unicode (в том числе и русские)		‘abc’ ‘ответ’
Логический	**bool** (от boolean)	True и False	1 байт	True 1 False 0

В вещественном числе (десятичной дроби) целая часть от дробной отделяется точкой, при этом перед точкой и после неё должно быть, по крайней мере, по одной цифре. Пробелы внутри числа недопустимы.

Произвольный набор символов, заключённый в одинарные апострофы или двойные кавычки, считается строковой величиной (строкой). Строка может содержать любые символы, набираемые на клавиатуре, в том числе буквы национальных алфавитов.

В отличие от многих других языков программирования переменные в языке Python не нужно объявлять. Тип переменной определяется автоматически в тот момент, когда ей присваивается новое значение.

Тип каждой переменной может динамически изменяться по ходу выполнения программы. Определить, какой тип имеет переменная в текущий момент, можно с помощью функции (команды) **type** ().

Режимы работы интерпретатора Python

Интерпретатор Python может работать в двух режимах:

• через командную строку (в командном, или интерактивном или непосредственном режиме), когда каждая введённая команда сразу выполняется;

• в программном режиме, когда программа сначала записывается в файл (обычно имеющий расширение .py) и при запуске выполняется целиком.

Оператор присваивания

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.

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.

Пример 1. 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.

*Пример*2. 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

Ввод данных

Для ввода в оперативную память значений переменных с клавиатуры используется оператор (функция, команда) 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

Первая программа на языке Python

Пользуясь рассмотренными операторами, 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.

Для вывода значений переменных

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:

```
#  Программа 1
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
```

Форматирование ввода и вывода данных

Разделитель sep

По умолчанию выводимые выражения в операторе print() разделяются одним пробелом, иначе говоря, разделителем между ними является пробел. Оператор 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.

```
#  Программа 1.1
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)
```

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 sp1it()** (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)

Особые разделители.

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(':')

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.

Контрольные вопросы.

1. Когда появился язык программирования Python? Кто его разработчик?
2. Что такое алфавит, синтаксис и семантика языка программирования? (три определения)
3. Какой знак используется для обозначения однострочного комментария, а какой для многострочного?
4. Как обозначаются основные арифметические операции в языке Python?
5. Какие вы знаете типы данных в языке Python? Какими служебными словами они обозначаются?
6. Как определить, какой тип имеет переменная в текущий момент в языке Python?
7. Что представляет собой программа на языке программирования?
8. Что такое оператор в языке программирования?
9. Как обозначается оператор присваивания в языке Python?
10. Что означает запись x +=y на языке Python? Как это можно записать иначе?
11. Какой оператор используется для вывода информации на экран в языке Python?
12. Какой оператор (функция) используется для ввода значений переменных с клавиатуры в языке Python? Какого типа вводятся значения по умолчанию?
13. Как ввести в переменную X любое целое число? А как вещественное? Запишите соответственные команды.
14. Перепишите программу для вычисления длины окружности и площади круга из параграфа без ошибок, добавьте комментарии к каждой строке (объясните, что каждая строка делает).
15. Для чего используется служебное слово **sep?** Поясните на примерах.
16. Что вы знаете о форматном выводе? Поясните на примерах.
17. Как организовать ввод значений нескольких переменных в одной строке?
18. Как организовать преобразование к числовому (целому, вещественному) типу значений нескольких переменных в одной строке?
19. Как указать особый разделитель для вводимых значений переменных?
20. Как сократить запись считывания нескольких значений и их преобразования в числовой тип? Запишите пример.