

Основные понятия модуля

Python — это высокоуровневый интерпретируемый кроссплатформенный язык программирования.

Актуальная версия Python — **Python 3**

[Документация по Python](#) 

Переменная — именованная область памяти компьютера, адрес которой позволяет получить доступ к данным.

Алгоритм — набор последовательных действий, направленных на достижение поставленной цели или решение конкретной задачи.

Функция — фрагмент кода, к которому можно обратиться из любого другого места.

Функции, как правило, возвращают некоторое значение в качестве результата работы.

Аргументы — данные, которые необходимы функции для её работы.

Python — язык с **неявной сильной динамической типизацией**.

Динамическая типизация

Тип переменной определяется во время выполнения программы

Сильная типизация

Нельзя совершать операции над объектами разного типа без приведения их к одному типу

Неявная типизация

Не надо указывать тип переменной при её объявлении

Строки — неизменяемый тип данных, предназначенный для хранения текстовой информации.

Ввод и вывод информации

<code>print(аргумент)</code>	Печатает на экране данные, которые мы передали. Это может быть переменная или выражение.
<code>print(аргумент_1, аргумент_2, ... , аргумент_n)</code>	Печатает переданные значения через пробел.
<code>input(подсказка_для_пользователя)</code>	<p>Функция для ввода информации от пользователя.</p> <p>В круглых скобках можем написать строку-подсказку для пользователя о том, какую именно информацию мы ожидаем от него получить.</p>

Codeboard

RUN	Запускает выполнение кода. Так вы можете проверить, как работает ваш код до отправки решения.
TEST	Запускает тесты для вашего решения. Таким образом проверяется логика решения.
SUBMIT	Отправляет код на проверку и запускает дополнительные тесты => результат и начисление баллов (при верном решении).

Присваивание

<code>a = 5</code>	Переменной <code>a</code> присвоили значение 5.
<code>b = a</code>	Переменной <code>b</code> присвоили значение переменной <code>a</code> .
<code>a, b = 5, 6</code>	Множественное присваивание: переменной <code>a</code> присвоили значение 5, переменной <code>b</code> присвоили значение 6.

Правила именования переменных

- Название переменной должно состоять только из букв, цифр и знаков подчёркивания `_`.
- Название переменной не может начинаться с цифры.

Типы данных

Тип данных	Изменяемость	Класс	Пример
Целые числа	-	<code>int</code>	73 0
Числа с плавающей точкой	Нет	<code>float</code>	3.14 -2.79
Строки	Нет	<code>str</code>	"Hello, world!" "5"
Логические переменные	Нет	<code>bool</code>	True False
Списки	Да	<code>list</code>	[1,2,3,4]
Кортежи	Нет	<code>tuple</code>	('a','b','c')
Словари	Да	<code>dict</code>	{ 'a' : 1, 'b' : 2 }
Множества	Да	<code>set</code>	{ 'a', 1, 'b', 2 }

Определение типа переменной и идентификатора объекта

- `type(n)` — тип переменной `n`.
- `id(n)` — уникальный идентификатор объекта, который хранится в переменной `n`.

Операции с целыми и вещественными числами

Сложение	+	$7+5 = 12$ $3.14+1 = 4.14$
Вычитание	-	$7-5 = 2$ $3.14-1 = 2.14$
Умножение	*	$7*5 = 35$ $3.14*2 = 6.28$
Возведение в степень	**	$7**5 = 16807$ $3.14**2 = 9.8596$
Деление	/	$5/2 = 2.5$ $3.14/2 = 1.57$
Целочисленное деление	//	$7 // 5 = 1$ $3.14 // 2 = 1.0$
Остаток от деления	%	$7 \% 5 = 2$ $3.14 \% 2 = 1.14$

Округление чисел

- `round(значение, количество_знаков_после_запятой)` — округляет число к заданной точности.

Значения логического типа данных

- `True` — Истина
- `False` — Ложь

Строки

<code>s = "Hello!"</code>	Задаём строку
<code>s[начало:конец:шаг]</code>	Срез строки
<code>s = "Hel" + "lo!"</code>	Сложение строк
<code>s = "Hello!"*n</code>	Дублирование значения строки n раз
<code>len(s)</code>	Длина строки
<code>find(substr)</code>	Метод для поиска подстроки в строке Пример вызова: <code>s.find('e')</code> возвращает индекс символа 'e' в строке s
<code>isdigit()</code>	Метод возвращает <code>True</code> , если строка состоит только из цифр
<code>isalpha()</code>	Метод возвращает <code>True</code> , если строка состоит только из букв
<code>isalnum()</code>	Метод возвращает <code>True</code> , если строка состоит только из букв и цифр
<code>upper()</code>	Метод возвращает новую строку в верхнем регистре
<code>lower()</code>	Метод возвращает новую строку в нижнем регистре
<code>split(разделитель)</code>	Метод разбивает строку на части по разделителю (по умолчанию — пробел) и возвращает результат в виде списка
<code>'строка-разделитель'.join(список)</code>	Метод объединяет элементы списка в строку, вставляя между ними строку-разделитель

Форматирование строк

Форматирование строк используется, когда нам необходимо вставлять в шаблон строки разные данные.

Это можно сделать и с помощью соединения частей строк и данных, но с помощью приёмов форматирования делать это можно гораздо удобнее.

Способ создания форматированной строки	Пример задания шаблона
Метод <code>format()</code>	'The {} currency rate on the date {} is {:.3f}'.format(currency, cur_date, rate)
f-строки	f'The {currency} currency rate on the date {cur_date} is {rate:.3f}'