**Переменные**

Если в вашей программе присутствуют большие вычисления, то неудобно их записывать в виде одного длинного выражения, лучше вычислять его значения по частям, используя переменные для хранения промежуточных результатов.

**Переменная** — это некоторая ячейка в памяти, в которой хранится какое-то значение любого из возможных типов, например, число или строка. В переменную можно записать какое-то значение (это называется **присвоить переменной значение**). Для присваивания значения переменной используется операция =, где слева от знака присваивания записывается имя переменной, справа — присваиваемое значение, которое может быть числом, строкой, другой переменной, каким-то выражением. Значение переменной можно использовать при вычислениях или вывести его на экран при помощи функции **print**().

**Пример**

Пусть длина школьного урока хранится в переменной lesson\_time, продолжительность перемены между уроками — rest\_time, а количество уроков — n. Хотим вычислить сколько времени занимает заданное количество уроков, которое будет храниться в переменной total\_time.

>>>lesson\_time = 45

>>>rest\_time = 15

>>>n = 6

>>>total\_time = lesson\_time \* n + (n - 1) \* rest\_time

>>>**print**(total\_time)

345

Имя переменной (также называется **идентификатором**) — последовательность букв, цифр и символа подчёркивания, первым символом должна быть буква или символ подчёркивания. В языке Python допускаются буквы не латинского алфавита в качестве имён переменных, но следует ограничиваться только латинскими символами.

Имена переменных должны быть осмысленными и понятными, например, lesson\_time или rest\_time.

**Пример**

>>>a = 14

>>>v = a

>>>a = 7

>>>a = a + 3

>>>v = a + 3

>>>**print**(a, v)

10 13

Посмотрим, как работает данная программа. В первой строке переменной a присваивается значение 14, затем переменной v присваивается значение переменной a, то есть 14. При этом переменные существуют независимо друг от друга, поэтому при выполнении присваивания a = 7 значение переменной v не изменится.

При выполнении присваивания a = a + 3 вычисляется сначала правая часть выражения, равная в данном случае 10, после чего это значение присваивается переменной. При выполнении присваивания v = a + 3, аналогично вычисляется сначала правая часть выражения. В данном случае в переменной a хранится значение, равное 10, значит правая часть выражения будет равна 13. Именно это значение и будет хранится после присваивания в переменной v.

После этого значения переменных a и v можно вывести на экран при помощи функции **print**(a, v), которую мы позже рассмотрим подробнее.

**Типы данных**

**Типы данных**

Если мы присваиваем переменной a значение, равное 5, то переменная будет иметь значение типа int, который используется для целых чисел. При умножении, вычитании и сложении чисел этого типа мы получаем числа того же типа.

Мы уже сталкивались с тем, что при выполнении операции деления результатом будет дробное число. Если переменной b присвоить значение 10 / 2, при выводе на экран результата увидим число 5.0, которое равно числу 5, но имеет другой тип. Переменная b имеет значение типа float. Это специальный тип данных для представления действительных чисел, и при умножении, вычитании и сложении чисел этого типа мы получаем числа того же типа.

Еще один тип данных используется для работы со строками. Если мы присваиваем переменной с значение, равное 'Hello', то эта переменная будет иметь значение типа str. Строки тоже можно складывать. Однако операция сложения для целых чисел и для строк работает по-разному: для чисел это сложение, а для строк — **конкатенация**.

**Основные операции над строками:**

A + B—  конкатенация, то есть соединение строк путём размещения строки B сразу после строки A;

A \* n—  повторение строки nn раз, значение nn должно быть целого типа.

**Преобразование типов**

Иногда полезно целое число записать как строку. И, наоборот, если строка состоит из цифр, то полезно эту строку представить в виде числа, чтобы дальше можно было выполнять арифметические операции с ней. Для этого используются функции, имя которых совпадает с именем типа, то есть int, float, str. Например, int('123') вернёт целое число 123, а str(123) вернёт строку '123'.

**Пример:**

>>>n = 5

>>>price = 15

>>>s = str(n) + " ручек стоят " + str(n \* price) + " рублей"

>>>**print**(s)

5 ручек стоят 75 рублей

**Пример:**

>>>**print**(str(1 + 1) \* int('1' + '1'))

22222222222

Результатом будет строка '2', повторенная 1111 раз.

**Ввод переменных**

### Ввод данных: функция input()

Все ранее рассмотренные программы выдавали один и тот же результат при каждом своем запуске. Для изменения входных данных нужно было вручную изменять значения переменных. Этого можно избежать, считывая данные с клавиатуры.

Для считывания строки с клавиатуры используется функция input(), которая возвращает значение считанной строки. Например, можно считать значения двух строк следующим образом:

a = input()

b = input()

Правда, функция input() возвращает текстовую строку, а нам бывает нужно сделать так, чтобы переменные имели целочисленные значения. Поэтому сразу же после считывания выполним преобразование типов при помощи функции int и запишем новые значения в переменные a и b.

a = int(a)

b = int(b)

Можно объединить считывание строк и преобразование типов, если вызывать функцию int для того значения, которое вернёт функция input():

a = int(input())

b = int(input())

Далее в программе можно использовать значения переменных a и b для вычисления новых переменных или для вывода результата на экран. Теперь мы можем, не меняя исходного кода программы, многократно использовать её для решения различных задач. Для этого нужно запустить программу и после запуска ввести с клавиатуры два числа, нажимая после каждого клавишу Enter. Затем программа сама выведет результат.

### Вывод данных: функция print()

Функция **print**() может выводить не только значения констант или переменных, но и значения выражений. Например, допустима такая запись:

**print**(a + b)