Ярошевич В.А.

090302-ИСТа-о22

ФИТУ 3 курс

**Лабораторная работа №2**

**РАБОТА С ОСНОВНЫМИ ВСТРОЕННЫМИ ФУНКЦИЯМИ**

**Цель работы:** рассмотреть основные встроенные функции языка программирования Python и научиться с ними работать.

**Задание №2:** написать функцию, которая принимает список, который содержит строки и выводит на экран новые списки из входящих строк.

**Краткая теория**

Стандартная библиотека Python предоставляет огромное количество различных модулей и функций, так называемых Built-in Functions. Все эти инструменты поставляются вместе с интерпретатором и их достаточно для того, чтобы немедленно начать работу с языком сразу после установки. Рассмотрим некоторые из основных встроенных функций:

**print (\*objects, sep=’’, end=’\n’, file=sys.stdout, flush=False)**

Данная функция предназначена, как было описано в лабораторной работе №1, для вывода сообщения на стандартное устройство вывода. Однако стоит дополнительно рассмотреть её параметры:

− \*objects – параметр, который принимает объекты для вывода на устройства вывода. Символ «\*» говорит о том, что параметр принимает переменное количество аргументов, например, строки через запятую;

− sep – параметр, аргумент которого предназначен для разделения аргументов параметра \*objects;

− end – параметр, аргумент которого выводится в конце вывода аргументов параметра \*objects;

− file – параметр, принимающий в качестве аргумента только объекты с методом write(string), если он не указывается или указан как None будет использоваться sys.stdout;

− flush – параметр, который предназначен для дополнительного указания функции при использовании параметра file в случае, если он Истина, то производить немедленный сброс данных из буфера в память.

Дополнительно стоит отметить, что оба параметра sep и end должны принимать только строки. Они могут принимать None, что инициирует использование значений по умолчанию. Также стоит учесть, что если \*objects не будет принимать никаких аргументов, то выведется только значение end.

**len(s)**

Данная функция принимает в качестве аргумента последовательность (строка, байты, список, кортежи и другое) или коллекции (словарь, множество и другие). Возвращает длину (число элементов) объекта.

**int([x])**

Возвращает целочисленный объект из числа или строки, переданной в параметр x. В случае, если аргументы не заданы, то возвращает 0. У вещественных чисел удаляется дробная часть.

**float([x])**

Возвращает число с плавающей точкой из числа или строки, переданной в параметр x. В случае, когда аргументом параметра x является строка, то она должна содержать десятичное число с необязательным предшествующим знаком (+ или -). Если аргумент не задан, то возвращается 0.0.

**list([iterable])**

Функция предназначена для конструирования списка. Аргумент параметра iterable может быть последовательностью, контейнером, либо итераторным объектом. Если в качестве аргумента iterable передается уже существующий список, то функция возвращает копию объекта переданного списка. Если заданных аргументов нет, то возвращается новый пустой список.

**dict(\*\*kwarg)**

Функция предназначена для создания словаря. Два символа «\*» означают, что параметр принимает переменное значение именованных аргументов (one=1, two=2). В случае, если аргумент передан параметру \*\*kwarg, то будет создаваться новый словарь с теми же парами ключ-значение. Если ключ в аргументе присутствует более одного раза, то будет использоваться последнее значение ключа переданного аргумента. Если аргумент не задан, то функция вернет новый пустой словарь.

**range(start, stop[, step])**

Функция предназначена для генерации неизменяемой числовой последовательности. Данную функцию можно вызывать тремя способами: ⎯ range(stop). В этом случае вернется ряд чисел, начинающихся с 0 и включающих каждое число до аргумента параметра stop, но не включающее само значение аргумента;

⎯ range(start, stop). В этом случае вернется ряд чисел, начинающихся со значения аргумента параметра stop и включающих каждое число до значения параметра start, но не включающее само значение его аргумента;

⎯ range(start, stop, step). В этом случае функция будет работать так же, как и в предыдущем случае, за исключением того, что аргумент параметра step будет указывать на разницу между двумя соседними числами.

Следует отметить, что данная функция создает не список, а отдельный тип неизменяемой последовательности.

**sum(iterable[, start])**

Функция возвращает сумму значений аргумента параметра iterable, начиная со start. По умолчанию параметр start равен 0.

**Ход работы**

Ознакомившись с материалом, можно приступать к выполнению задания. Сначала мы проходим по каждому элементу списка строк. После каждая строка разбивается на слова с помощью метода split(), который по умолчанию разделяет строки по пробелам. Каждый полученный список слов добавляется в результирующий список. И в конце после этого выводим каждую подстроку в виде списка.

Листинг кода:  
def split\_strings\_to\_lists(string\_list):

result = []

for string in string\_list:

result.append(string.split())

for lst in result:

print(lst)

input\_strings = ["Первая строка", "Вторая строка", "Третья строка"]

split\_strings\_to\_lists(input\_strings)

**Результаты выполнения кода:**

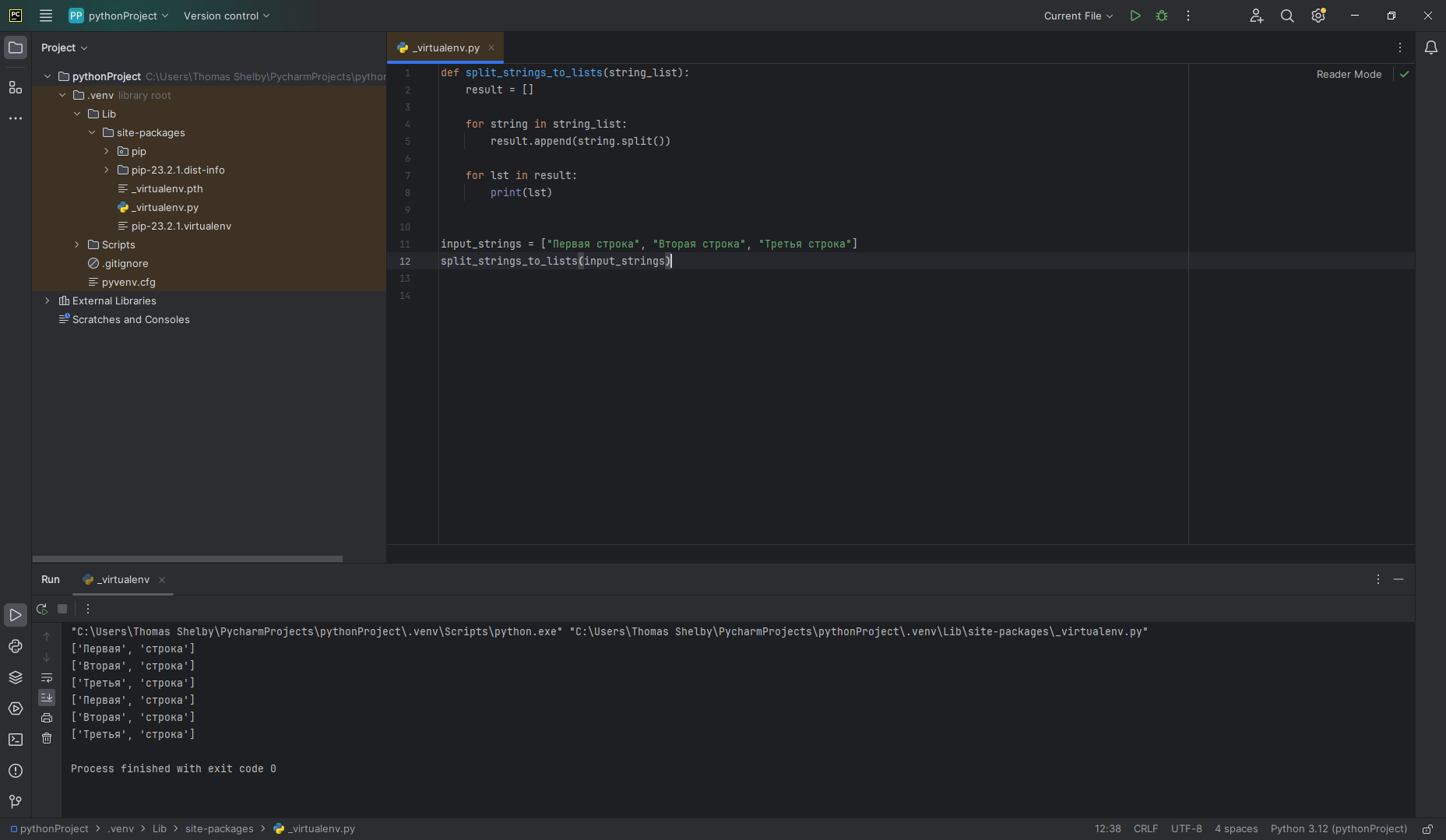


Рис. 1 - результаты работы кода

**Вывод**: в результате работы мы рассмотрели основные встроенные функции языка программирования Python и научились с ними работать.