Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМУПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра Инновационных

технологий (ФИТ)

Отчет

по лабораторной работе №10

**«СТРОКОВЫЙ ТИП ДАННЫХ»**

по дисциплине «Алгоритмы и структуры данных»

Студент гр. 023

\_\_\_\_\_\_\_\_\_\_\_\_ Иптышев В.А

«\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2025 г.

Доцент кафедры УИ

\_\_\_\_\_\_\_\_\_\_ Лобода Ю.О.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2025 г.

Томск 2025

**Содержание**

[Введение 3](#_Toc197813531)

[I ТЕОРЕТИЧЕСКИЕ ОСНОВЫ ПРИМЕНЕНИЯ ФУНКЦИЙ В PYTHON 4](#_Toc197813532)

[II ПРАКТИЧЕСКАЯ ЧАСТЬ 6](#_Toc197813533)

[2.1. Решения задач на тему «Функции без параметров» 6](#_Toc197813534)

[2.2. Решения задач на тему «Функции с параметрами» 8](#_Toc197813535)

[2.3. Решения задач на тему «Функции с возвратом значения. Часть 1» 11](#_Toc197813536)

[2.4. Решения задач на тему «Функции с возвратом значения. Часть 2» 16](#_Toc197813537)

[2.5. Решения задач на тему «Функции с возвратом значения. Часть 3» 25](#_Toc197813538)

# Введение

При написании кода довольно-таки часто приходится использовать одну и ту же логику в разных местах. Чтобы не повторять код и не усложнять его читаемость, были придуманы функции. Функции помогают выполнять определенную логику и улучшают читаемость кода. Например, в программировании часто приходится сортировать что-либо. Для того, чтобы каждый раз не писать алгоритм сортировки, можно определить этот алгоритм в функции и затем использовать его. Выходит очень удобно.

В Python существует специальный оператор для определения функций: def. После объявления функции мы можем пользоваться ими.

**Целью** лабораторной работы является изучение строкового типа данных в Python.

**Задачи**:

1. Изучить теорию по функциям в Python;
2. Решить задачи на тему функции;
3. Сделать вывод по применению функций в Python.

# I ТЕОРЕТИЧЕСКИЕ ОСНОВЫ ПРИМЕНЕНИЯ ФУНКЦИЙ В PYTHON

В предыдущих уроках мы использовали встроенные в Python функции print(), input(), int(), str(), len() и многие другие. Пришло время начать писать свои собственные функции.

В самом начале курса вам было предложено решить задачу, в которой требовалось изобразить звездный прямоугольник размерами 5×7 (5 строк и 7 столбцов).

Наш первый вариант кода выглядел примерно так (см. рисунок 1.1):

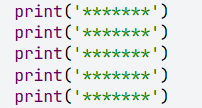


Рисунок 1.1 – пример кода

А теперь представим, что таких прямоугольников нужно изобразить не один, а несколько, скажем, 3 штуки.

И хотя предыдущий код полностью решает поставленную задачу, он не лишен недостатков. Во-первых, он довольно громоздкий из-за повторения части кода, отвечающей за вывод прямоугольника. Во-вторых, если понадобится изменить размеры прямоугольника, придется менять их трижды, в каждой части кода, выводящей прямоугольник.

Вместо повторения кода для вывода прямоугольника, можно перенести его в отдельную функцию и вызвать ее 3 раза.

Для создания функции пишем такой код (см. рисунок 1.2):

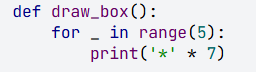


Рисунок 1.2 – пример функции

Когда функция создана, чтобы увидеть результат ее работы, надо вызвать ее по имени.

Теперь, чтобы изобразить 3 прямоугольника, можно написать функцию 3 раза.

Код стал короче, читабельнее (за счет удачного названия функции), а главное, если потребуются иные размеры прямоугольника, достаточно будет изменить только саму функцию draw\_box().

Итак, функция – отдельная, функционально независимая часть программы, выполняющая определенную задачу.

Функции объявляются с помощью ключевого слова def (от англ. define – определять). За ключевым словом def следуют название функции, круглые скобки () и двоеточие : (см. рисунок 1.3).

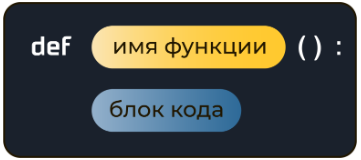


Рисунок 1.3 – структура объявления функции

Со следующей строки идет блок кода – тело функции. Это набор инструкций, составляющих одно целое и выполняющихся каждый раз, когда вызывается функция.

Для вызова функции пишут ее название и круглые скобки.

Было бы намного удобнее, если бы функция draw\_box() выводила прямоугольник с произвольными размерами. И действительно, функции могут принимать входные параметры, что делает их более гибкими.

Функции с параметрами объявляются так же, как функции без параметров, только с указанием в скобках (см. рисунок 1.4):

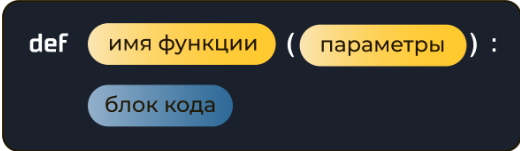


Рисунок 1.4 – структура объявления функции с параметром

Функция с возвратом значения похожа на функцию без возврата значения тем, что:

* это набор инструкций, выполняющий определенную задачу;
* когда нужно выполнить функцию, ее вызывают.

Однако, когда функция с возвратом значения завершается, она возвращает значение в ту часть программы, которая ее вызвала. Возвращаемое из функции значение используется как любое другое: оно может быть присвоено переменной, выведено на экран, использовано в математическом выражении (если это число) и т. д.

Функцию с возвратом значения пишут точно так же, как и без, но она должна иметь инструкцию return.

Вот общий формат определения функции с возвратом значения в Python (см. рисунок 1.5):

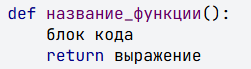


Рисунок 1.5 – объявление функции

Значение выражения, которое следует за ключевым словом return, будет отправлено в ту часть программы, которая вызвала функцию. Это может быть переменная либо выражение, к примеру, математическое.

Python позволяет писать булевы функции, возвращающие либо истину (True), либо ложь (False). Булеву функцию можно применять для проверки условия, тогда значения True и False будут сигнализировать о его выполнении.

Булевы функции широко применяются для упрощения сложных условий, проверяемых в структурах принятия решения и структурах с повторением.

# II ПРАКТИЧЕСКАЯ ЧАСТЬ