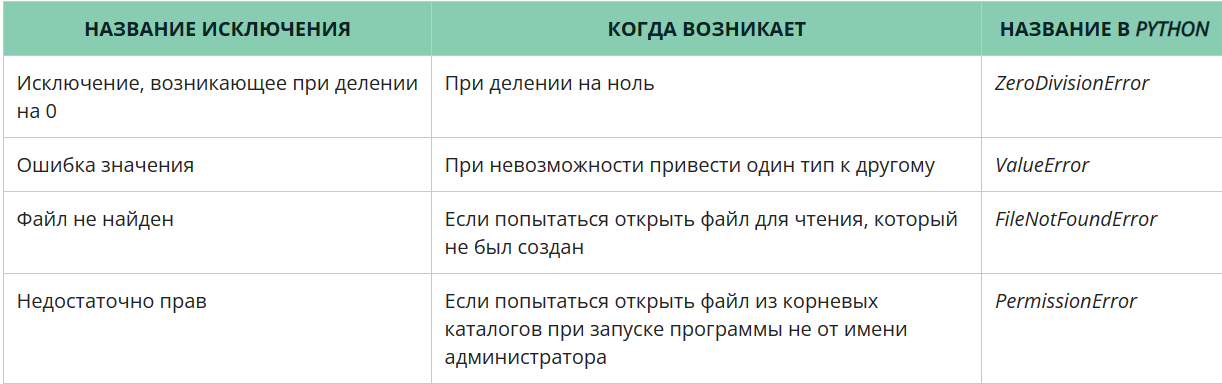
# Блок 0. Введение. Проектирование разработки

## PYTHON - 3. Условные операторы. Включая важную информацию

**Исключения** — это такие ошибки, которые возникают не во время компиляции программы, а в процессе её исполнения, в случаях если что-то идёт не так.

Ошибки бывают двух видов:

* + отлавливаемые — всё, что наследуются от класса Exception;
  + неотлавливаемые — SystemExit, KeyboardInterrupt и т. д.

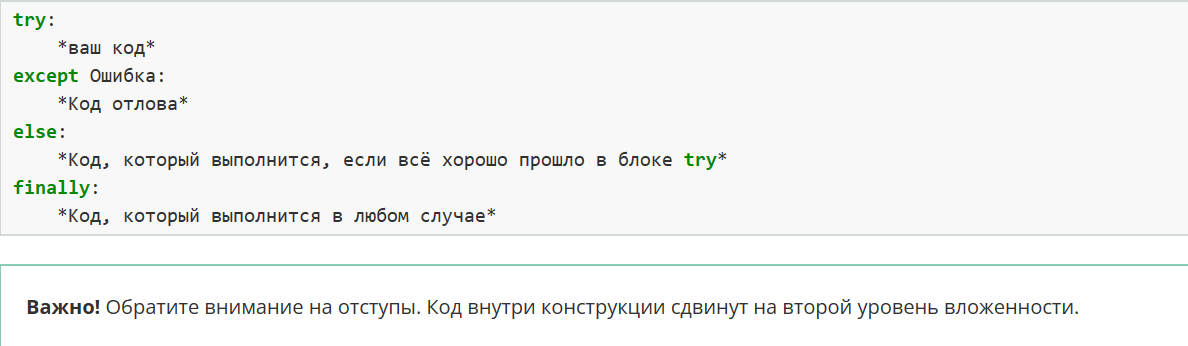


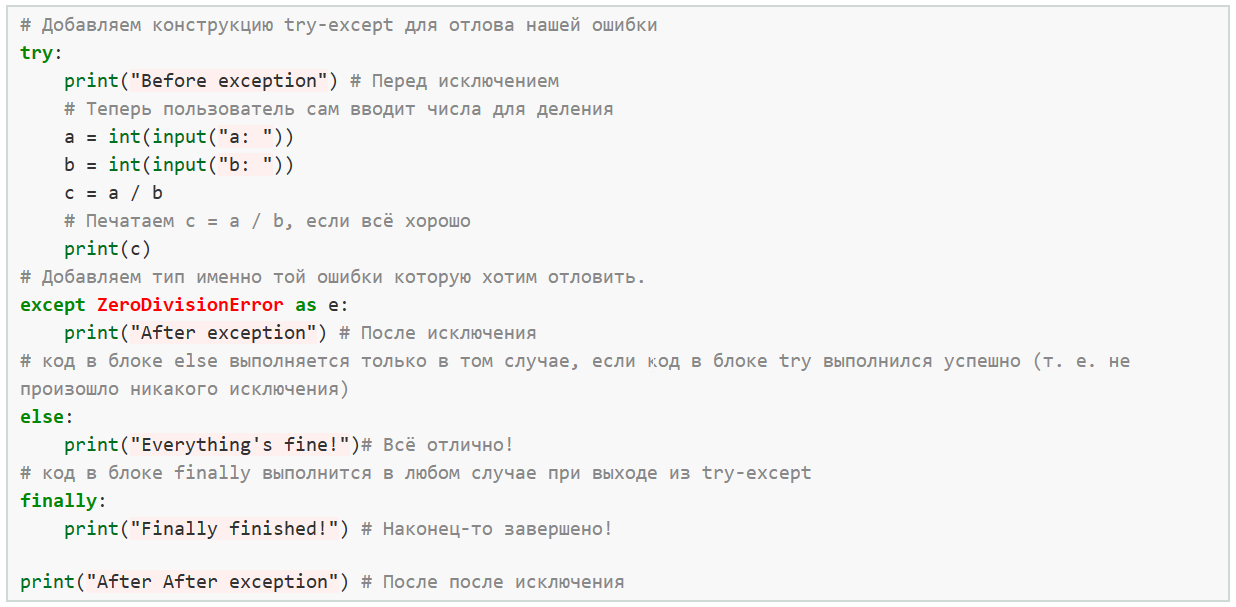
Как же сделать так, чтобы программа не вылетала при ошибке и продолжала работу? Очень просто! Для этого и нужна конструкция try-except.

**Примечание.** В блоке except мы указали имя исключения, которое мы ожидаем — *ZeroDivisionError* (ошибка деления на 0). С помощью ключевого слово as мы дали этому объекту имя (по сути записали в переменную с именем e). Далее в этом же блоке мы можем воспользоваться переменной e и вывести её на экран.

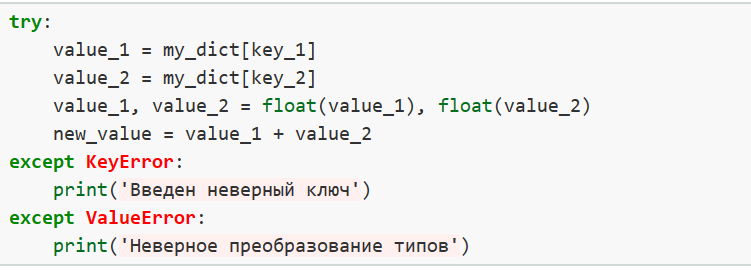
В блок try помещается «опасный» кусок кода, который может вызывать исключения. А в блоке except указывается класс ошибки, которую мы хотим отловить, а затем помещается код, который нужно выполнить в случае возникновения ошибки.

Ну, конечно же, это ещё не всё. Есть также блоки finally и else. Если вы хорошо помните тему условия, то там тоже было ключевое слово else. Код в блоке else выполнялся, если не отработал ни один из блоков if-elif. С try-except происходит нечто похожее — блок else подключается тогда, когда не отработал блок except

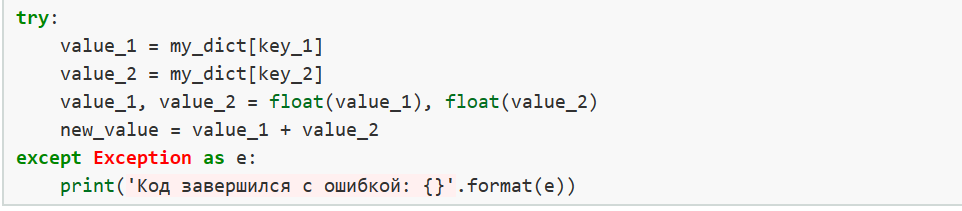




Если в вашем коде потенциально может возникнуть больше ошибок, то можете добавить дополнительные блоки except и обрабатывать их. Обратите внимание, ошибки обрабатываются последовательно.

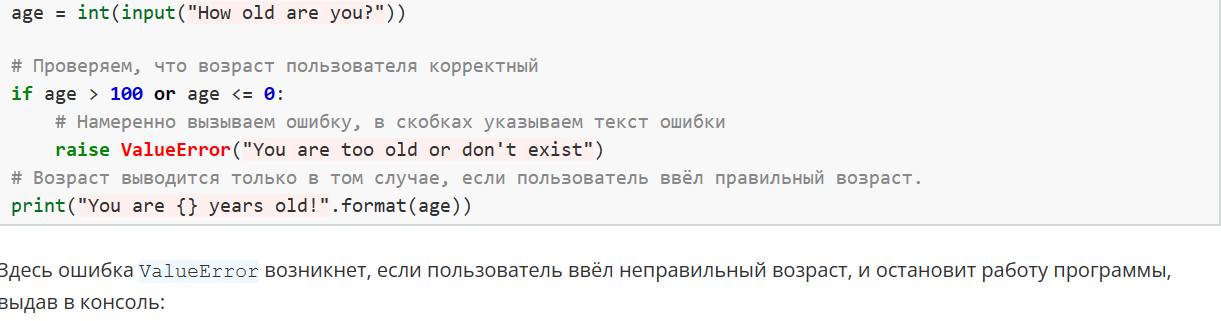


Можно отлавливать не конкретный тип ошибки, а все виды исключений, которые могут возникнуть в коде. Для этого вместо конкретного типа ошибки нужно указать Exception. Код будет выглядеть следующим образом:

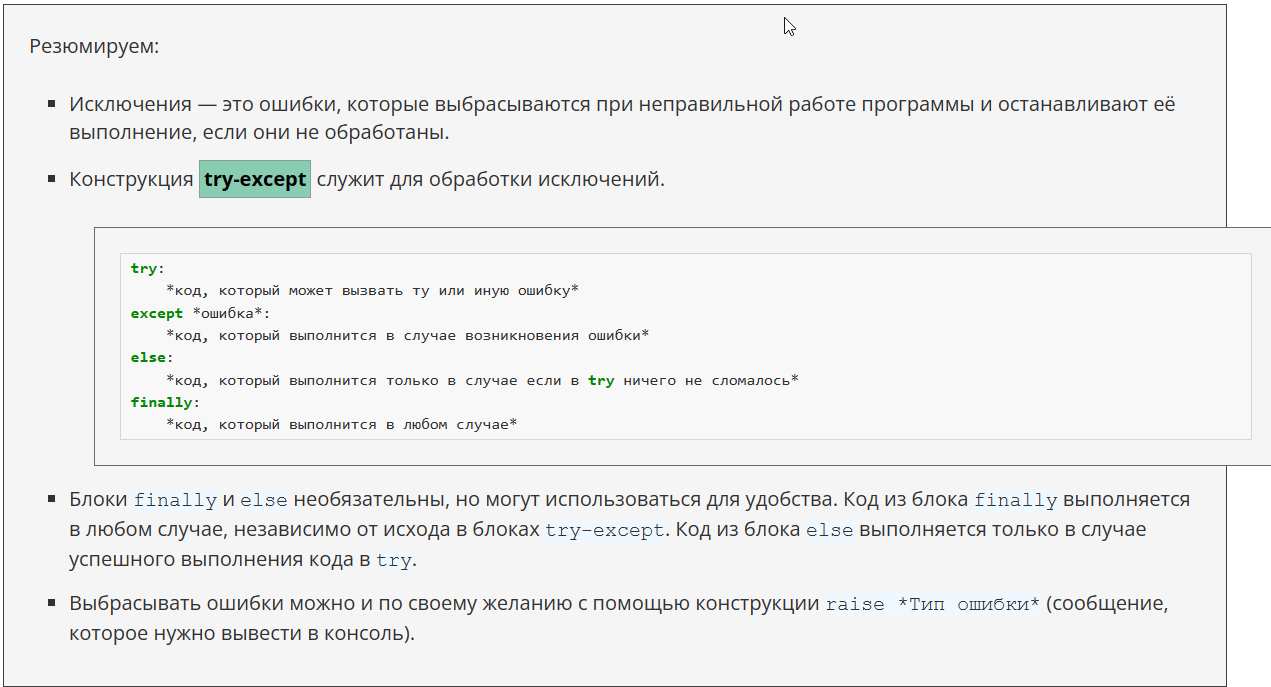


Исключение Exception — это специальный тип исключения, который является родительским для всех видов исключений.

Мы можем вызывать ошибки самостоятельно с помощью конструкции raise. Как правило, это используется для отладки кода и остановки программы в критических ситуациях.



В консоль выводится именно то сообщение, которое вы передадите в аргумент конструктора класса исключения. Если не хотите передавать сообщения, просто оставьте скобки пустыми. Стоит отметить, что отлавливать вызываемые с помощью raise ошибки тоже можно.



## PYTHON - 4. Циклы

## PYTHON - 5.1. Введение в функции Python

## PYTHON - 5.2. Продвинутое использование функций в Python

## PYTHON - 6. Практика

## PYTHON - 7\*(Bonus). Как писать красивый код на Python