25. Обработка исключительных ситуаций в Python. Создание пользовательских исключений и инструкция assert.

При возникновении исключения работа программы прерывается, и чтобы избежать подобного поведения и обрабатывать исключения в Python, существует конструкция try..except.

которая имеет следующее формальное определение:
Общий синтексис конструкции:

Users > georgiydemo > kot > meow.py

```
1    try:
2        #Код, который пытаемся выполнить
3    except expression as identifier:
4        #Ошибка, которая обрабатывается
5    else:
6        #Действие, если нет ошибки
7    finally:
8        #Выполнится всегда
9
```

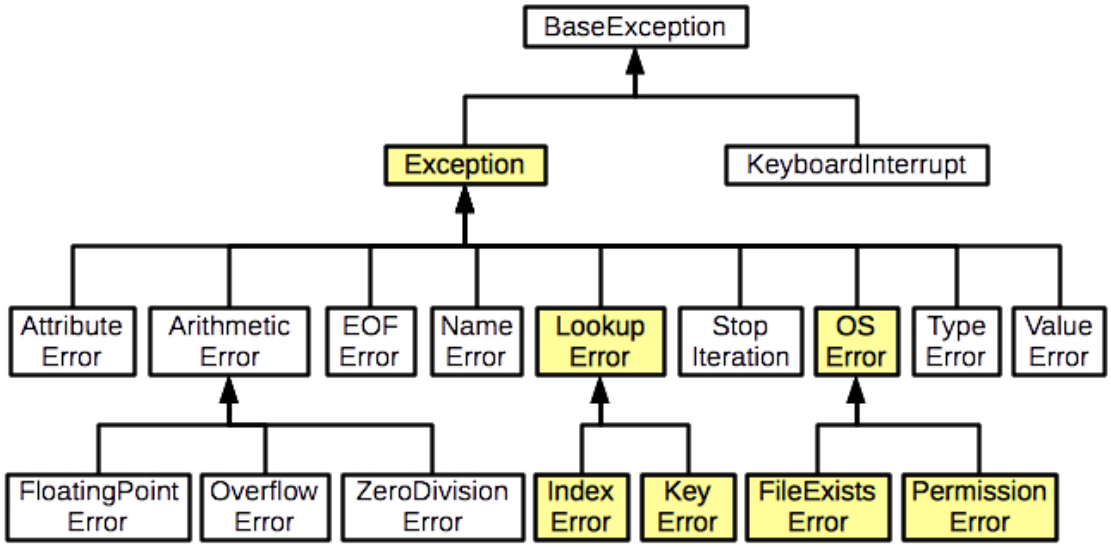
Try - код, который выполняем

В конструкции try/except есть опциональный блок else. Он выполняется в том случае, если не было исключения.

Блок finally - он выполняется всегда, независимо от того, было ли исключение или нет.

Except тип_ошибки as e: - ошибка Exception, которую обрабатываем и представляем ее как e. Представление ошибки необходимо для получения ее типа, подробного описания и прочих полей, связанных с ней.

Виды ошибки в Python3:



Необходимо обратить внимание, что если перехват ошибки не сработал при первом except, то происходит сравнение интерпретатором со вторым и т.д. Причём согласно иерархии выше, можно использовать универсальный перехват с Exception т.к. Все ошибки наследуются от него.

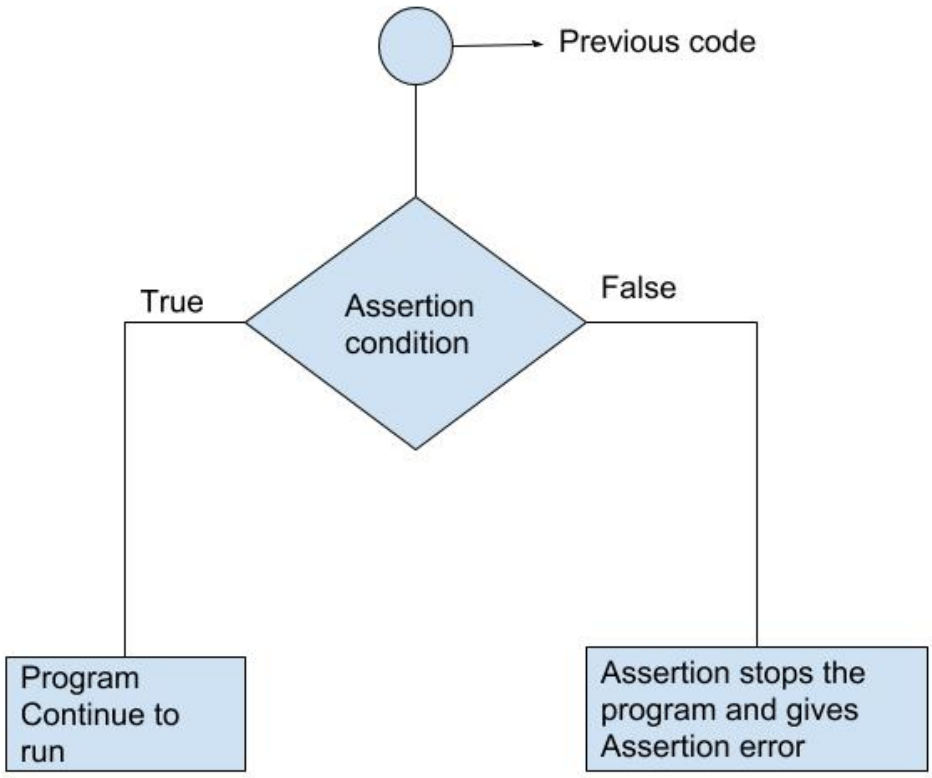
```
try:
    #Код, который пытаемся выполнить
except Exception as e:
    #Ошибка, которая обрабатывается всегда
```

Для того, чтобы сгенерировать явное исключение, в Python применяется оператор raise. Пример собственного исключения:

```
try:
    #Код..
    raise ValueError()
except ValueError as e:
    print("Перехват ошибки")

```

Assert
Assert поверочное выражение, представляет собой средство отладки, которое проверяет условие. Вызывается ошибка AssertionError.
Инструкция assert предназначена для того, чтобы сообщать разработчикам о неустранимых ошибках в программе.



По факту assert можно записать вот так:

```
if __debug__:
    if not expression1:
        raise AssertionError(expression2)
```

Но данный синтаксис упрощается до:

```
a = True
assert a == False #AssertionError
```

Assert широко используется только для отладки, в релизных версиях программ обычно от него избавляются.