Python обработка исключений

Ошибки

Варкалось. Хливкие шорьки
Пырялись по наве,
И хрюкотали зелюки,
Как мюмзики в мове.

К неработоспособности программы приводят ошибки (спасибо, Кэп). Как правило ошибки делят на **синтаксические** и **семантические**.

Нарушение семантики обычно означает, что, хотя выражения написаны верно с точки зрения синтаксиса языка, программа не работает так, как от нее ожидалось.

В общем случае при возникновении ошибки интерпретатор останавливает выполнение программы и выводит сообщение, указав на место возникновения ошибки.

Исключения

Обработка исключений – механизм реакции программы на ошибки выполнения (исключения).

```
>>> 42 / 0
Traceback (most recent call last):
  File "/usr/lib/python3.5/code.py", line 91, in runcode
    exec(code, self.locals)
  File "<input>", line 1, in <module>
ZeroDivisionError: division by zero
```

Все исключения, которые может выбросить программа, построены в иерархию. Общим предком считается класс BaseException. От него наследуются все остальные.

Виды исключений

Примеры исключений:

ZeroDivisionError - деление на ноль.

AttributeError - объект не имеет данного атрибута (значения или метода).

NameError - не найдено переменной с таким именем.

SystemError - внутренняя ошибка.

TypeError - операция применена к объекту несоответствующего типа.

ValueError - функция получает аргумент правильного типа, но некорректного значения.

Дерево можно посмотреть здесь: http://bit.ly/2kjEslE

Или здесь:

Оператор try

```
try:
    Операторы
except Исключение:
    Обработка исключения
else: # выполняется, когда в try не возникло исключения
    Действия
finally: # выполняется в любом случае
    Действия
```

Оператор except может обрабатывать как одиночное исключение, так и их список (в скобках).

Если не указано имя исключения, обрабатываться будут все.

Оператор try

```
try:
    a = float(input("Введите делимое: "))
    b = float(input("Введите делитель: "))
    c = a / b
    print("Частное: " , c)
except ValueError:
    print("Нельзя вводить строки")
except ZeroDivisionError:
    print("Нельзя делить на ноль")
```

Или…

```
except (ValueError, ZeroDivisionError):
    print("Нельзя вводить строки или делить на ноль")
```

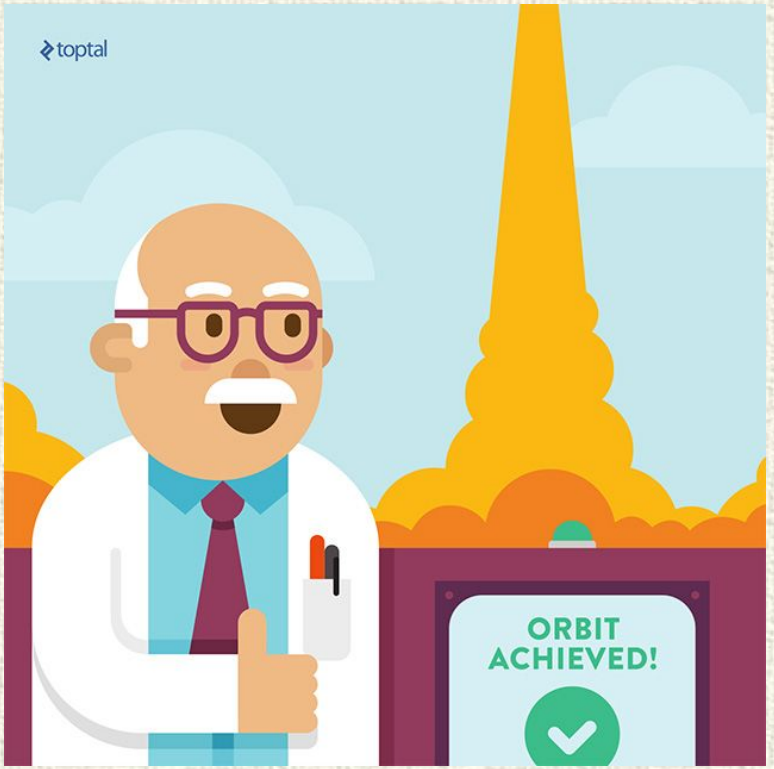
Оператор try

```
try:
    n = input('Введите целое число: ')
    n = int(n)
except ValueError:
    print("Вы что-то попутали с вводом")
    3 / 0
except ZeroDivisionError:
    print("Деление на ноль")
else:
    print("Все нормально. Вы ввели число", n)
finally:
    print("Конец программы")
```

Оператор raise

Оператор raise позволяет выбрасывать собственные исключения.

raise Имя_Исключения(аргументы)



Оператор try

```
try:
    numberl = int(input("Введите первое число: "))
    number2 = int(input("Введите второе число: "))
      if number2 == 0:
          # выбрасываем исключение сами с сообщением
          raise MyException("Второе число не должно быть 0")
      print("Частное:", numberl/number2)
except ValueError:
    print("Введены некорректные данные")
except MyException:
    print("Завершение программы")
```

```
Введите первое число: 42
Введите второе число: 0
Второе число не должно быть 0
Завершение программы
```

Достоинства исключений

- Исключения отделяют код обработки ошибок от алгоритма программы, тем самым повышая разборчивость, надежность и расширяемость кода.
- Генерация исключения – единственный чистый способ сообщить об ошибке из конструктора.
- Исключения трудно игнорировать, в отличие от кодов ошибок.
- Исключения легко передаются из глубоко вложенных функций.
- Исключения могут быть, и часто являются, определяемыми пользователем типами, несущими гораздо больше информации, чем код ошибки.

Недостатки исключений

- Исключения нарушают структуру кода, создавая множество скрытых точек выхода, что затрудняет чтение и изучение кода.
- Исключения легко вызывают утечки ресурсов, особенно в языке, не имеющем встроенного сборщика мусора и блоков finally(в конце).
- Обработка исключений может потребовать большого количества ресурсов.
- Тяжело научиться писать безопасный код исключений.
- Исключения тяжело ввести в устаревший код.