

Обработка исключений

Роли исключений

обработка ошибок времени выполнения

оповещения о событиях

обработка специальных случаев

освобождение ресурсов

нетривиальный контроль потока выполнения

▲try/except – перехватывает исключения в блоке, с возможностью продолжить выполнение

▲try/finally – выполняет операторы из блока finally независимо от того, произошло ли исключение

▲raise – создаёт событие исключения

▲assert – создаёт событие исключения в зависимости от выполнения условий

▲with/as – управление контекстом ресурсов

Исключения

События, которые могут менять поток управления программой, обычно сигнализируют о том, что что-то пошло не так

```
try:
```

```
    #начало блока
```

```
    l[0]
```

```
except IndexError:
```

```
    #ждём исключение
```

```
    print('an error occurred')
```

```
l[0] = 10
```

```
#выполнение продолжается
```

Перехват
исключений

```
try:
    #начало блока
    raise IndexError
    #создание события
except IndexError:
    #ждём исключение
    print('an error occurred')
```

Создание
события
исключения

try:

#начало блока

l = [1,2,3,4,5]

#создание ресурса

except IndexError:

#ждём исключение

print('an error occurred')

finally:

del l

#освобождение памяти

Освобождение
памяти

```
try:
    ... statements ...           #основная часть
except event1:
    ... statements ...           #если произошёл event1
except (event2, event3):
    ... statements ...           #если произошли event2 или event3
except event4 as error:
    ... statements ...           #присваивание event4 переменной
except:
    ... statements ...           #если произошли любые другие
                                события
else:
    ... statements ...           #если исключений не было
finally:
    ... statements ...           #выполняется всегда
```

Готов ко всему

Исключения на практике