In [1]:
```
a = 8/0
```

```
---------------------------------------------------------------------------
ZeroDivisionError                         Traceback (most recent call last)
<ipython-input-1-f45c6a53e9b9> in <module>
----> 1 a = 8/0

ZeroDivisionError: division by zero
```

Python Try Except

The `try` block lets you test a block of code for errors.

The `except` block lets you handle the error.

The `finally` block lets you execute code, regardless of the result of the try- and except blocks.

Exception Handling

When an error occurs, or exception as we call it, Python will normally stop and generate an error message.

These exceptions can be handled using the try statement:

In [2]:
```
try:
    a = 8/0
except:
    print("An exception occurred")
```

```
An exception occurred
```

Since the `try` block raises an error, the `except` block will be executed.

Without the `try` block, the program will crash and raise an error:

Many Exceptions

You can define as many exception blocks as you want, e.g. if you want to execute a special block of code for a special kind of error:

In [3]:
```
try:
    a = 8/0
except ZeroDivisionError:
    print("ZeroDivisionError occred")
except:
    print("An exception occurred")
```

```
ZeroDivisionError occred
```

Else

You can use the `else` keyword to define a block of code to be executed if no errors were raised:

In [4]:
```
try:
    print(Hello)
except:
    print("Something went wrong")
else:
    print("## Nothing went wrong")
```

```
Something went wrong
```

In [5]:
```
try:
    print("Hello")
except:
    print("Something went wrong")
else:
    print("## Nothing went wrong")
```

```
Hello
## Nothing went wrong
```

Finally

The `finally` block, if specified, will be executed regardless if the try block raises an error or not.

In [6]:
```
try:
    print(x)
except:
    print("Something went wrong")
finally:
    print("The 'try except' is finished")
```

```
Something went wrong
The 'try except' is finished
```

Raise an exception

As a Python developer you can choose to throw an exception if a condition occurs.

To throw (or raise) an exception, use the `raise` keyword.

In [7]:
```
# example 1
x = -1

if x < 0:
  raise Exception("Sorry, no numbers below zero")
```

```
---------------------------------------------------------------------------
Exception                                 Traceback (most recent call last)
<ipython-input-7-74a7167e8c47> in <module>
      3
      4 if x < 0:
----> 5   raise Exception("Sorry, no numbers below zero")

Exception: Sorry, no numbers below zero
```

In [8]:
```
# example 2
x = "hello"

if not type(x) is int:
  raise TypeError("Only integers are allowed")
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-8-ab20f9cc7363> in <module>
      3
      4 if not type(x) is int:
----> 5   raise TypeError("Only integers are allowed")

TypeError: Only integers are allowed
```

In [ ]:
```
```