In [6]:
```
import time
```

Handling Exceptions

Try / Except

In [1]:
```
def causeError():
    try:
        return 1/0
    except Exception as e:
        return e

causeError()
```

Out[1]:
```
ZeroDivisionError('division by zero')
```

In [2]:
```
def causeError():
    try:
        return 1/0
    except Exception:
        print('There was some sort of error!')

causeError()
```

```
There was some sort of error!
```

Finally

In [5]:
```
def causeError():
    try:
        return 1/1
    except Exception:
        print('There was some sort of error!')
    finally:
        print('This will always execute!')

causeError()
```

```
This will always execute!
```

Out[5]:
```
1.0
```

In [8]:
```
def causeError():
    start = time.time()
    try:
        time.sleep(0.5)
        return 1/0
    except Exception:
        print('There was some sort of error!')
    finally:
        print(f'Function took {time.time() - start} seconds to execute')

causeError()
```

```
There was some sort of error!
Function took 0.504910945892334 seconds to execute
```

Catching Exceptions by Type

In [13]:
```
def causeError():
    try:
        return 1 + 'a'

    except TypeError:
        print('There was a type error!')
    except ZeroDivisionError:
        print('There was a zero division error!')
    except Exception:
        print('There was some sort of error!')


causeError()
```

```
There was a type error!
```

Custom Decorators

In [16]:
```
def handleException(func):
    def wrapper(*args):
        try:
            func(*args)
        except TypeError:
            print('There was a type error!')
        except ZeroDivisionError:
            print('There was a zero division error!')
        except Exception:
            print('There was some sort of error!')
    return wrapper

@handleException
def causeError():
    return 1/0

causeError()
```

```
There was a zero division error!
```

Raising Exceptions

In [19]:
```
@handleException
def raiseError(n):
    if n == 0:
        raise Exception()
    print(n)

raiseError(1)
```

```
1
```

In [ ]:
```

```