EXCEPTIONS

Overview:

- distinguish interrupts and exceptions
- learn Python mechanisms to process exceptions

Exceptions & Interrupts

- both change program flow
- interrupts:

(a) caused by external events
(b) ex: network disruption

- exceptions:

(a) caused by program
(b) ex: division by zero

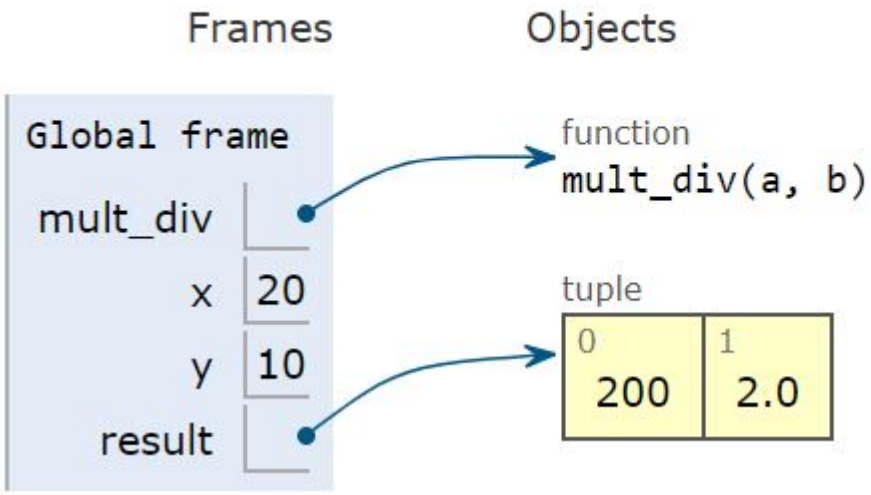
- unhandled exceptions stop execution
- mechanims to "catch" and process exceptions

No Errors

```
def mult_div(a, b):
    mult_result = a * b
    div_result  = a / b
    return mult_result, div_result

result = mult_div(20, 10)
print('result is ', result)
```

```
result is  (200, 2.0)
```

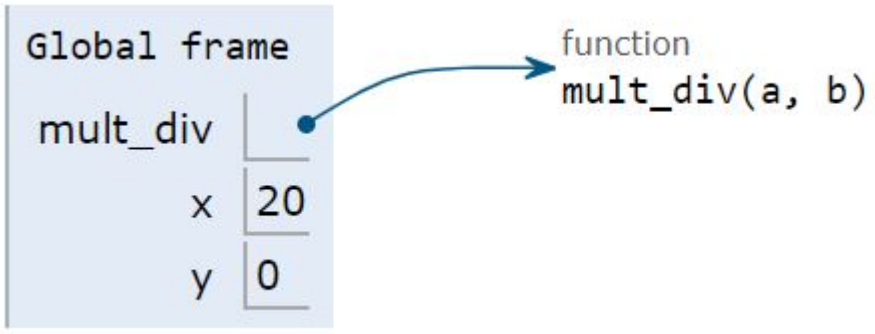


- no errors

An Exception Example

```
def mult_div(a, b):
    mult_result = a * b
    div_result  = a / b
    return mult_result, div_result

result = mult_div(20, 0)
print('result is ', result)
```



- ZeroDivisionError

Raising Exceptions

```
def mult_div(a, b):
    if b == 0:
        raise Exception('divide by zero!')
        mult_result, div_result = None, None
    else:
        mult_result = a * b
        div_result  = a / b
    return mult_result, div_result

result = mult_div(20, 0)
```

- can define exceptions
- Exception: divide by zero!
- raising exceptions stops a program

Handling Exceptions

```
def mult_div(a, b):
    try:
        mult_result = a * b
        div_result  = a / b
    except Exception as e:
        print('Python error:', e)
        print('user-defined error: set to None')
        mult_result, div_result = None, None
    return mult_result, div_result

x = 20; y = 0
result = mult_div(x, y)
print('result is ', result)
```

```
Python error: division by zero
user-defined error: set to None
result is  (None, None)
```

Optional *finally* Clause

```
def mult_div(a, b):
    try:
        mult_result = a * b
        div_result  = a / b
    except Exception as e:
        print('Python error:', e)
        print('user-defined error: set to None')
        mult_result, div_result = None, None
    finally:
            print('execution continues')
    return mult_result, div_result

print('mult_div(20,10) is',mult_div(20,10),'\n')
print('mult_div(20, 0) is',mult_div(20,0))
```

```
execution continues
mult_div(20, 10) is  (200, 2.0)

Python error: division by zero
user-defined error: set to None
execution continues
mult_div(20, 0)  is  (None, None)
```

Exception Examples

Name	Description
Exception	base class
ArithmeticError	errors in computation
ZeroDivisionError	division by zero
ImportError	import statement fails
IndexError	index not in sequence
KeyError	key not in dictionary
NamedError	identifier not found
SyntaxError	error in syntax
IndentationError	improper indentation

Multiple Exceptions

```
try:
     statement(s) - no errors
except exception_group_1 as error_1:
     statement(s) for error_1
      ------------------------------
      ------------------------------
except exception_group_n as error_n:
     statements for error_n
else:
     statement(s) for unknown error(s)
finally:
     final statement(s)
```

- handling multiple exceptions

Exercise(s):

- write a function *ratio_list*() to compute the ratio of first two elements in a list
- the function must be capable to catch the following errors:

(a) IndexError

(b) ZeroDivisionError

- if an exception is generated, function should return *None*