Hacettepe University
Computer Engineering Department

Programming in Python

BBM103 Introduction to Programming Lab 1
Week 10

Fall 2017
Exceptions

- Built-in Exceptions
- User-defined Exceptions
Built-in Exceptions

The simplest way to handle exceptions is with a "try-except" block:

Example 1:

```python
(x, y) = (5, 0)
try:
    z = x/y
except ZeroDivisionError:
    print ("divide by zero")
```

Output: divide by zero
Example 2: except ValueError:

```python
first_number = input("First number: ")
second_number = input("Second number: ")

try:
    number1 = int(first_number)
    number2 = int(second_number)
    print(number1, "/", number2, ":", number1 / number2)
except ValueError:
    print("Error! Please enter number!")
```
Example 3: except ZeroDivisionError:

```python
def main():
    first_number = input("First number: ")
    second_number = input("Second number: ")
    try:
        number1 = int(first_number)
        number2 = int(second_number)
        print(number1, "/", number2, "+", number1 / number2)
    except ValueError:
        print("Error! Please enter number!")
    except ZeroDivisionError:
        print("You can't divide a number to 0!")
```

main()
Example 4: except (ValueError, ZeroDivisionError)

```python
first_number = input("First number: ")
second_number = input("Second number: ")

try:
    number1 = int(first_number)
    number2 = int(second_number)
    print(number1, "/" , number2, "/equal/ ", number1 / number2)
except (ValueError, ZeroDivisionError):
    print("Error!")
```
Example 5: try... except... as...

```python
first_number = input("First number: ")
second_number = input("Second number: ")

try:
    number1 = int(first_number)
    number2 = int(second_number)
    print(number1, "/", number2, ":", number1 / number2)
except (ValueError, ZeroDivisionError) as error:
    print("Error!")
    print("Original error message: ", error)
```
Example 6: try... except... else...

```python
for arg in sys.argv[1:]:
    try:
        f = open(arg, 'r')
    except IOError:
        print('cannot open', arg)
    else:
        print(arg, 'has', len(f.readlines()), 'lines')
    f.close()
```
```python
Example 7: try... except... finally...

    try:
        file = open("dosyaadı", "r")
    except IOError:
        print("error!")
    finally:
        file.close()
```
Some Examples using Exceptions

except IOError:
    print('An error occurred trying to read the file.')

except ValueError:
    print('Non-numeric data found in the file.')

except ImportError:
    print('NO module found«)

except EOFError:
    print('Why did you do an EOF on me?')

except KeyboardInterrupt:
    print('You cancelled the operation.')

except:
    print('An error occurred.')
Example 8:

```python
tr_character = "şçüöî"  
password = input("Enter your password: ")
for i in password:
    if i in tr_character:
        raise TypeError("You can't use Turkish characters in password!")
    else:
        pass

print("Password is excepted!")
```
Example 9:

```python
try:
    while True:
        if int(input('Guess a number: ')) == 5:
            raise ZeroDivisionError
except ZeroDivisionError:
    print ('You got it!')
```
Example 10:

```python
import sys

try:
    f = open('myfile.txt')
    s = f.readline()
    i = int(s.strip())
except OSError as err:
    print("OS error: \{0\}\.format(err))
except ValueError:
    print("Could not convert data to an integer.\")
except:
    print("Unexpected error:", sys.exc_info()[0])
raise
```
User-Defined Exceptions

Example 11:

```python
class MyException(Exception):
    def __init__(self, t=0):
        self.numtries = t

try:
    for tries in range(1, 6):
        if int(input('Guess a number: ')) == 5:
            raise MyException(tries)
except MyException as e:
    print ('You got it in only %d tries!' % e.numtries)
else:
    print ('Too bad, you ran out of tries!')
```
Example 12 user-defined exceptions

```python
class Error(Exception):
    """Base class for other exceptions"""
    pass

class ValueTooSmallError(Error):
    """Raised when the input value is too small"""
    pass

class ValueTooLargeError(Error):
    """Raised when the input value is too large"""
    pass

# our main program
# user guesses a number until he/she gets it right

# you need to guess this number
number = 10
```

This example continues in the next slide
```python
Example 12 continued

while True:
    try:
        i_num = int(input("Enter a number: "))
        if i_num < number:
            raise ValueTooSmallError
        elif i_num > number:
            raise ValueTooLargeError
        break
    except ValueTooSmallError:
        print("This value is too small, try again!")
        print()
    except ValueTooLargeError:
        print("This value is too large, try again!")
        print()
print("Congratulations! You guessed it correctly.")
```
Assert

assert <some_test>, <message>

Example 13:

```python
def test_set_comparison():
    set1 = set("1308")
    set2 = set("8035")
    assert set1 == set2

test_set_comparison()
```

Output:

```
C:\Users\necva\Desktop> py deneme.py
Traceback (most recent call last):
  File "deneme.py", line 8, in <module>
    test_set_comparison()
  File "deneme.py", line 4, in test_set_comparison
    assert set1 == set2
AssertionError
```
Example 14:

```python
array = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

def number(input):
    assert (input in array)

number(10)
number(5)
```

Output:

```
C:\Users\necva\Desktop>py deneme.py
Traceback (most recent call last):
  File "deneme.py", line 7, in <module>
    number(10)
  File "deneme.py", line 4, in number
    assert (input in array)
AssertionError
```
Example 15:

```python
def func(a, b):
    max = 0
    if a < b: max = b
    if b < a: max = a
    print(max)
    assert (max == a or max == b) and max >= a and max >= b

func(10, 15)
```

Output:

```
C:\Users\necva\Desktop>py deneme.py
15
```