# Tutorial II

# Intermediate Python Programming

## 7 – Exception Handling.

repl.it

Resources created by Andy Colley (@MrAColley)

# Learning Goals/Objectives

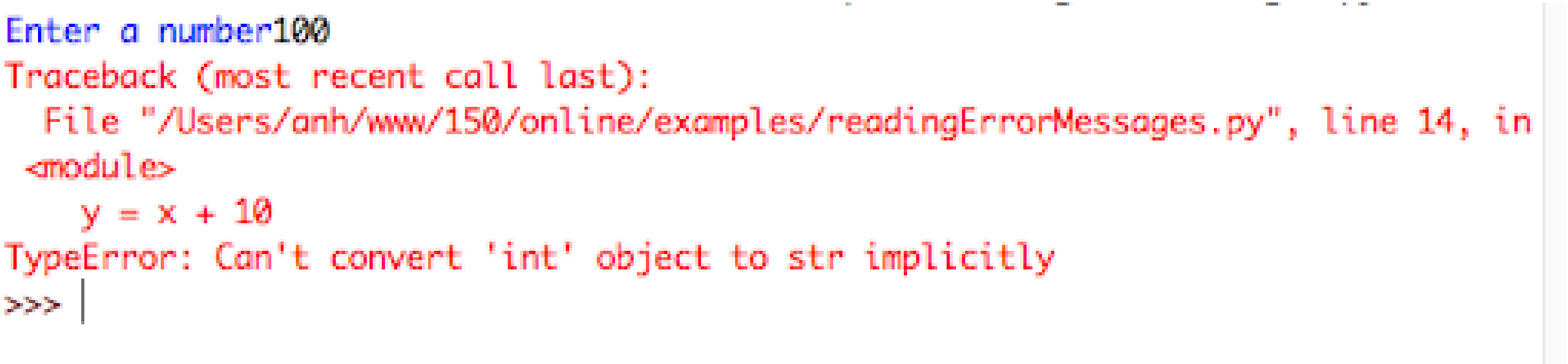
Be able to read, comprehend, trace, adapt and create Python code that:

- Catches general **exceptions**
- Catches value exceptions
- Outputs suitable error messages

Resources created by Andy Colley (@MrAColley)

# What Is An Exception?

- An exception is something that happens during a program’s **run that disrupts the flow of instructions.**
- They usually produce error messages - these are generated by an **exception handler.**
- It’s possible to code our own exception handlers to prevent the program crashing in certain circumstances.



```
Enter a number100
Traceback (most recent call last):
  File "/Users/anh/www/150/online/examples/readingErrorMessages.py", line 14, in
 <module>
    y = x + 10
TypeError: Can't convert 'int' object to str implicitly
>>> 
```

Resources created by Andy Colley (@MrAColley)

# Basic Exception Handling

Resources created by Andy Colley (@MrAColley)

# Exceptions - How To Code

```
try:
    Run this code in normal circumstances
except:
  Run this code when there is an exception
```

Resources created by Andy Colley (@MrAColley)

# Exceptions - How To Code

```
name = “Dave”

try:
    print(name)
except:
  print(“The variable has not been assigned”)
```

Resources created by Andy Colley (@MrAColley)

# Exceptions - How To Code

```
name = “Dave”


try:
    print(name)
except NameError:
  print(“The variable has not been assigned”)
except:
    print(“Something else went wrong)
```

Resources created by Andy Colley (@MrAColley)

# Exceptions - How To Code

```
name = “Dave”
try:
    print(name)
except NameError:
  print(“The variable has not been assigned”)
except:
    print(“Something else went wrong)
else:
    print(“nothing went wrong”)
```

Resources created by Andy Colley (@MrAColley)

# Exceptions - How To Code

```
name = “Dave”
try:
     print(name)
except NameError:
  print(“The variable has not been assigned”)
except:
     print(“Something else went wrong)
else:
     print(“Nothing went wrong”)
finally:
     print(“The try except has finished”)
```

Resources created by Andy Colley (@MrAColley)

# Value Errors

Resources created by Andy Colley (@MrAColley)

# Value Errors - How To Code

1. Cast the input to the desired data type.

```
try:
   num1 = int(input(“Type a number
   between 1 and 10”))
except ValueError:
      print(“Hey, that wasn’t a number!”)
else:
      print(“You typed “ + num1)
```

2. Use ‘**ValueError**’ in your except.

Resources created by Andy Colley (@MrAColley)

# Built-in Exceptions

repl.it

Resources created by Andy Colley (@MrAColley)

# Base Classes for Exceptions

See https://docs.python.org/3/library/exceptions.html for a full list so called „**base classes**“ of Python exceptions

- Some Examples

**exception EOFError**
Raised when the input() function hits an end-of-file condition (EOF) without reading any data. (N.B.: the io.IOBase.read() and io.IOBase.readline() methods return an empty string when they hit EOF.)

**exception KeyboardInterrupt**
Raised when the user hits the interrupt key (normally Control-C or Delete). During execution, a check for interrupts is made regularly. The exception inherits from BaseException so as to not be accidentally caught by code that catches Exception and thus prevent the interpreter from exiting.

**exception TypeError**
Raised when an operation or function is applied to an object of inappropriate type. The associated value is a string giving details about the type mismatch.

**exception FileNotFoundError**
Raised when a file or directory is requested but doesn't exist. Corresponds to errno ENOENT.

**exception NameError**
Raised when a local or global name is not found. This applies only to unqualified names. The associated value is an error message that includes the name that could not be found.