Files, Errors and Exceptions

Content Source:
Complete Python Bootcamp: Go from zero to hero in Python
By
Jose Portilla

File Handling

- `myfile = open('help.txt')`

  Create a text file named help.txt

- `myfile = open('help.txt')`
- `myfile.read()`
- `myfile.read()`
- `myfile.seek(0)`
- `contents = myfile.read()`
- `contents`
- `myfile.seek(0)`
- `myfile.readlines()`

File Locations

- `myfile = open("C:\\Users\\Username\\Folder\\help.txt")`

  OR

- `myfile = open("/Users/Username/Folder/help.txt")`
- `myfile.close()`
- `myfile = open('help.txt')`
- `with open('help.txt') as my_new_file:`

  `contents=my_new_file.read()`

- `contents`

Writing to a File

- `with open('help.txt', mode='r') as my_new_file:`

  `contents=my_new_file.read()`

- `contents`
- `with open('help.txt', mode='w') as my_new_file:`

  `contents=my_new_file.read()`

  `ERROR`

- `with open('help.txt', mode='a') as my_new_file:`

  `contents=my_new_file.read()`

Reading, writing, appending modes

- mode='r' read only
- mode='w' write only (will overwrite files or create new!)
- mode='a' append only (will add on to the files)
- mode='r+' reading and writing
- mode='w+' writing and reading (will overwrite files or create new!)

- ```
  with open('help.txt', mode='r') as f:
      f.read()
  ```
- ```
  with open('help.txt', mode='a') as f:
      f.write('\nThis line was appended')
  ```
- ```
  with open('help.txt', mode='w') as f:
      f.write('I have overwritten everything')
  ```
- ```
  with open('clap.txt', mode='r') as f:
      f.write('Wow this works!')
  ```

Errors and Exception Handling

Errors

- Errors will creep into your code, like it or not.
- Error handling is an attempt to plan for possible errors that may arise in our code in the future.
- Currently the code we write, if there is an error in the program the program stops it’s execution and gives an error statement.
- In error handling, the script will report some error encountered and then move with the other parts of the script which may not be affected by this error.

Keywords in Error Handling

- try : This is the block of code that MAY lead to an error
- except : This is the block of code that executes when the try block code HAS an error.
- finally : This is a final block of code that gets executed regardless of any error.

```
def add(n1,n2):
    print(n1+n2)


add(10,20)
Number1 = 10
Number2 = input(“Enter a no.”)
add(Number1,Number2)
print(“Hey! What happened”)
```

```
try:
    Result = 10 + 10
except:
    print(“Hey incorrect addition”)


Result
```

```
try:
    Result = 10 + ‘10’
except:
    print(“Hey incorrect addition”)


Result
```

```
try:
    Result = 10 + ‘10’
except:
    print(“Hey incorrect addition”)
else:
    print(“You did well”)
    print(Result)
```

```
try:
    f = open('testfile','w')
    f.write("Write a test line")
except TypeError:
    print("There was a type error!")
except OSError:
    print("There was a an OS error!")
except:
    print("All other exceptions")
finally:
    print("I always run")
```

```
try:
    f = open(‘testfile’,’w’)
    f.write(“Write a test line”)
except:
    print(“There was an error!”)
finally:
    print(“I always run”)
```

```
try:
    f = open(‘testfile’,’r’)
    f.write(“Write a test line”)
except:
    print(“There was an error!”)
finally:
    print(“I always run”)
```

```
def ask_for_int():
    try:
        result=int(input(“Please provide a number:”))
    except:
        print(“That is not a number”)
    finally:
        print(“End of try/except/finally”)
ask_for_int()
```

```
def ask_for_int():
    while True:
        try:
            result=int(input(“Please provide a number:”))
        except:
            print(“That is not a number”)
            continue
        else:
            print(“That was a number”)
            break
        finally:
            print(“End of try/except/finally”)
            print(“I will always run at the end”)
ask_for_int()
```

```
def ask_for_int():
    while True:
try:
        result=int(input(“Please provide a number:”)
    except:
        print(“That is not a number”)
        continue
    else:
        print(“That was a number”)
        break

ask_for_int()
```

Problem 1

```
for i in [‘a’, ‘b’, ‘c’,’d’]:
    print(i**2)
```

Solution 1

```
try:
    for i in [‘a’,‘b’,‘c’]
        print(i**2)
except:
    print(“Some Error Occured”)
```

Solution 1 - Refined

```
try:
    for i in [‘a’,‘b’,‘c’]
        print(i**2)
except TypeError:
    print(“Type Error Occured!”)
except:
    print(“All other exceptions”)
```

Problem 2

```
x=5
y=0
z=x/y
```

Solution 2

```
try:
x=5
y=0
z=x/y
except:
    print(“Error!”)
finally:
    print(“All done”)
```

Solution 2

```
try:
x=5
y=0
z=x/y
except ZeroDivisionError:
    print(“Trying to divide by zero!”)
except:
    print(“Error!”)
finally:
    print(“All done”)
```

What does this code do?

```
def ask():
    waiting = True
    while waiting:
        try:
            n=int(input(“Enter a number:”))
        except:
            print(“Please try again! \n”)
            continue
        else:
            waiting = False
    print(n**2)
ask()
```