



**ECOM SCHOOL**

המכללה למקצועות הדיגיטל וההייטק

# Errors & File Handling

Python Programming

# | Error Handling Reference



**ECOM SCHOOL**

המכללה למקצועות הדיגיטל וההייטק

# Errors

In general, a programmer cannot program without mistakes. Therefore, there are sometimes situations in programming, in which the software does not know what to do, or something is undefined – In any case, the software crashes!

Take for example a situation where the software downloads a file from the Internet - how should the program behave if the Internet has stopped working? Unless otherwise **is** specified, the software will crash.

Since programmers cannot pre-plan all possible situations (since they are human), Python can be configured to handle such situations using the **Try & Except** concept.



**ECOM SCHOOL**

המכללה למקצועות הדיגיטל וההייטק

```
1 y = 16/0
2 print(y)
```

```
web hacking x
C:\Users\Daniel0z.DESKTOP-KQSGR9V\PycharmProjects\pythonPr
Traceback (most recent call last):
  File "C:\Users\Daniel0z.DESKTOP-KQSGR9V\PycharmProjects\
    y = 16/0
ZeroDivisionError: division by zero
```

# Error Handling Keywords

- In order to use python's errors handling correctly, Python has the following keywords:
  - **try** - Defines a block that lets you test a code for errors.
  - **except** - Defines a code block to run in case of an error in the **try** block.
  - **finally** - Defines a code block to run regardless of the result of the **try-except** blocks.

- **try, except, else, finally** use case:

In this program we want to receive an input of a number from a user, and add it to an integer. So we have `x = 10 | y = input( ) | print(x + y)`.

If we are not using casting we will receive an error and the program will break. however, we can use **try** and **except** to handle the error and the program will not crash.

```
x = 10
y = input()
try:
    print(x+y)
except:
    print("Something went wrong")

print("rest of the code")
```



**ECOM SCHOOL**

המכללה למקצועות הדיגיטל וההייטק

# Multiple Exceptions


```
>_ Terminal

try:
    print(x)
except NameError:
    print("Variable x is not defined")
except Exception as e: #You can use just except as well
    print("Something else went wrong")
    print(f"The error message is: {str(e)}")
```

You can define as many exception blocks as you want, e.g. if you want to execute a special block of code for a special kind of error.

In addition, you can store the error message inside a special variable using as keyword.

# | Else & Finally




>\_ Terminal

```
try:
    print("Hello")
except:
    print("Something went wrong")
else:
    print("Nothing went wrong")
finally:
    print("The code has ended")
```

You can use the `else` keyword to define a block of code to be executed if no errors were raised.

In addition, you can use the `finally` block, if specified, will be executed regardless if the `try` block raises an error or not.

# Raise an exception



>\_ Terminal

```
#Raise an error and stop the program if x is lower than 0
x = -1
if x < 0:
    raise Exception("Sorry, no numbers below zero")

#Raise a TypeError if x is not an integer
x = "hello"
if not type(x) is int:
    raise TypeError("Only integers are allowed")
```

As a Python developer you can choose to throw an exception if a condition occurs.

To throw (or raise) an exception, use the raise keyword.

You can define what kind of error to raise, and the text to print to the user.

# Class Exercise - 1

1. Create a program that accepts two inputs from the user, and divides the first input with the second. Make sure that you handle the zero division.
2. Make another exception for Runtime error.
3. Make exception for one more exception of your choice.

Exercise time – 15 Minutes!



**ECOM SCHOOL**

המכללה למקצועות הדיגיטל וההייטק



# | File Handling Reference



**ECOM SCHOOL**

המכללה למקצועות הדיגיטל וההייטק

# File Handling

- File handling is an important part of any program. Python has several functions for creating, reading, and updating, files.
- Python can be used to access the OS filesystem and to interact with the files.
- Python can be used with files to: Open, read, write, append, delete, create, etc.
- This ability to interact and handle files does not require a module or library and this is a built-in python ability.
- Important to note that the basis of file usage remains the same and permission open file restrictions remain based on the OS.



**ECOM SCHOOL**

המכללה למקצועות הדיגיטל וההייטק

# File Handling

- The key function for working with files in Python is the `open()` function:
  - The `open()` function takes two parameters; **file path**, and **mode**.
- There are four different methods (modes) for opening a file:
  - **"r"** - Read - Default value. Opens a file for reading, error if the file does not exist
  - **"a"** - Append - Opens a file for appending, creates the file if it does not exist
  - **"w"** - Write - Opens a file for writing, creates the file if it does not exist
  - **"x"** - Create - Creates the specified file, returns an error if the file exists
- In addition you can specify if the file should be handled as binary or text mode:
  - **"t"** - Text - Default value. Text mode
  - **"b"** - Binary - Binary mode (e.g. images)



**ECOM SCHOOL**

המכללה למקצועות הדיגיטל וההייטק

```
f = open("demofile.txt")
```

```
#The code above is the same as:
```

```
f2 = open("demofile.txt", "rt")
```

```
#Because "r" for read, and "t" for text are the  
#default values, you do not need to specify them.
```

# File Objects

- Inside the `open()` function we need to pass the full \ relative path of the file as a string and the mode we want to access this file with.

```
f = open("C:\\Users\\Daniel0z.DESKTOP-KQSGR9V\\Desktop\\1.txt", "r")
```

- To avoid special strings problems in the string interpretation, we need to add another "\\" to every "\" character in the string.
- Additional method is to write the letter "r" before the string.

```
f = open(r"C:\Users\Daniel0z.DESKTOP-KQSGR9V\Desktop\1.txt", "r")
```



**ECOM SCHOOL**

המכללה למקצועות הדיגיטל וההייטק

# | Read from the file

```
>_ Terminal

f = open("D:\\myfiles\\welcome.txt", "r")
print(f.read())

#Return the 5 first characters of the file:
f = open("demofile.txt", "r")
print(f.read(5))
```

To open the file, use the built-in `open()` function. The `open()` function returns a file object, which has a `read()` method for reading the content of the file.

By default the `read()` method returns the whole text, but you can also specify how many characters you want to return.

# | Large files & Binary data

- A common practice is to use for loop in order to read each line individually, this is helpful when dealing with big files:

```
a = open("C:\\Users\\User\\OneDrive\\Desktop\\test_file1.txt", 'r')  
  
for line in a:  
    print(a.readline())
```



**ECOM SCHOOL**

המכללה למקצועות הדיגיטל וההייטק



# | Read Lines

>\_ Terminal

```
f = open("demofile.txt", "r")
print(f.readline()) #Read one line of the file
print(f.readline()) #Read another one line of the file
print(f.readline()) #Read another one line of the file

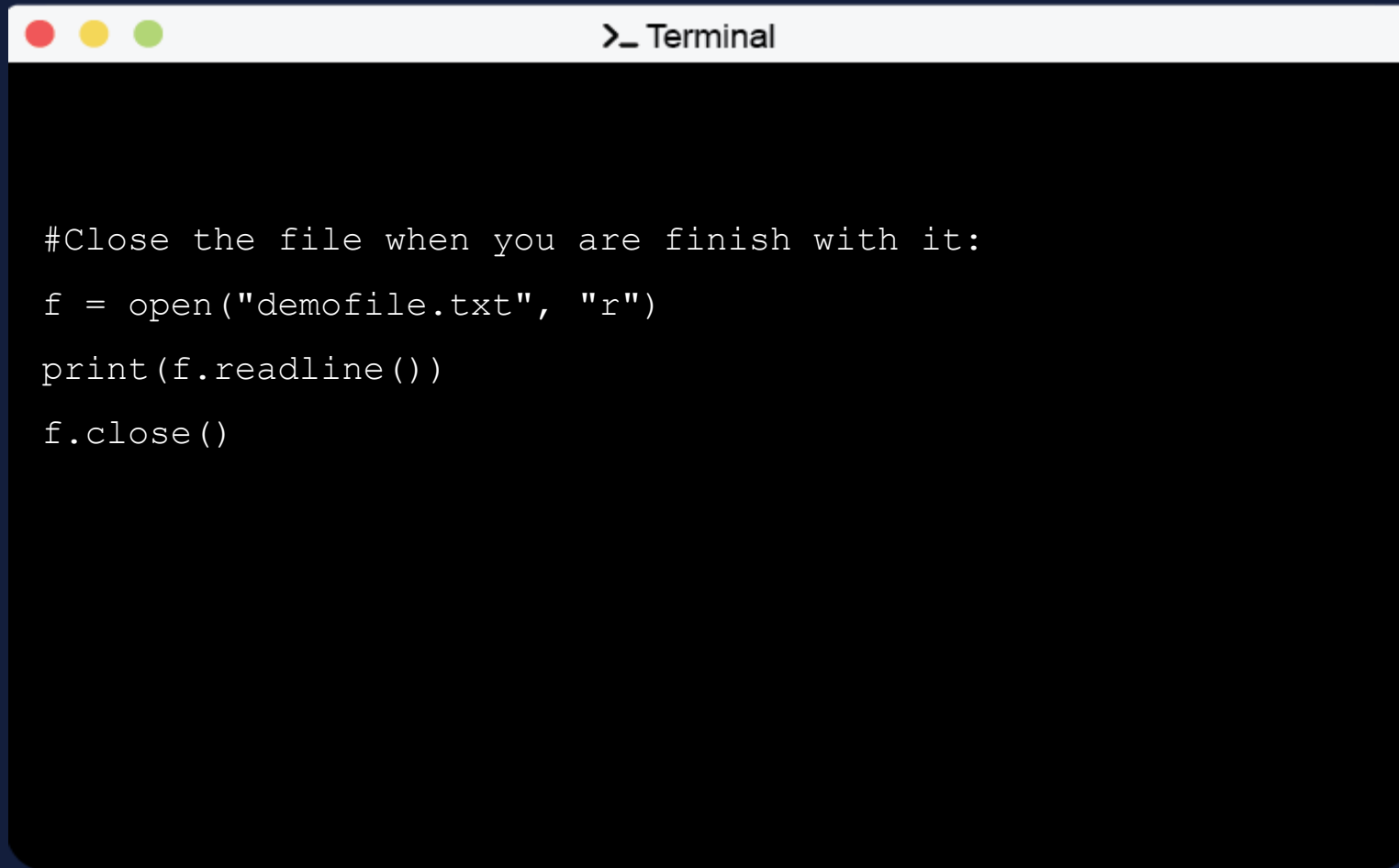
f2 = open("demofile2.txt", "r")
for x in f2: #Loop through the file line by line
    print(x)
```

You can return one line by using the `readline()` method.

By calling `readline()` two times, you can read the two first lines (`readline()` saves the last position used).

Also, by looping through the lines of the file, you can read the whole file, line by line.

# Close Files

A terminal window with a title bar containing three colored circles (red, yellow, green) and the text ">\_ Terminal". The terminal has a black background with white text. It contains a comment and three lines of Python code: opening a file named 'demofile.txt' in read mode, reading the first line, and closing the file.

```
>_ Terminal

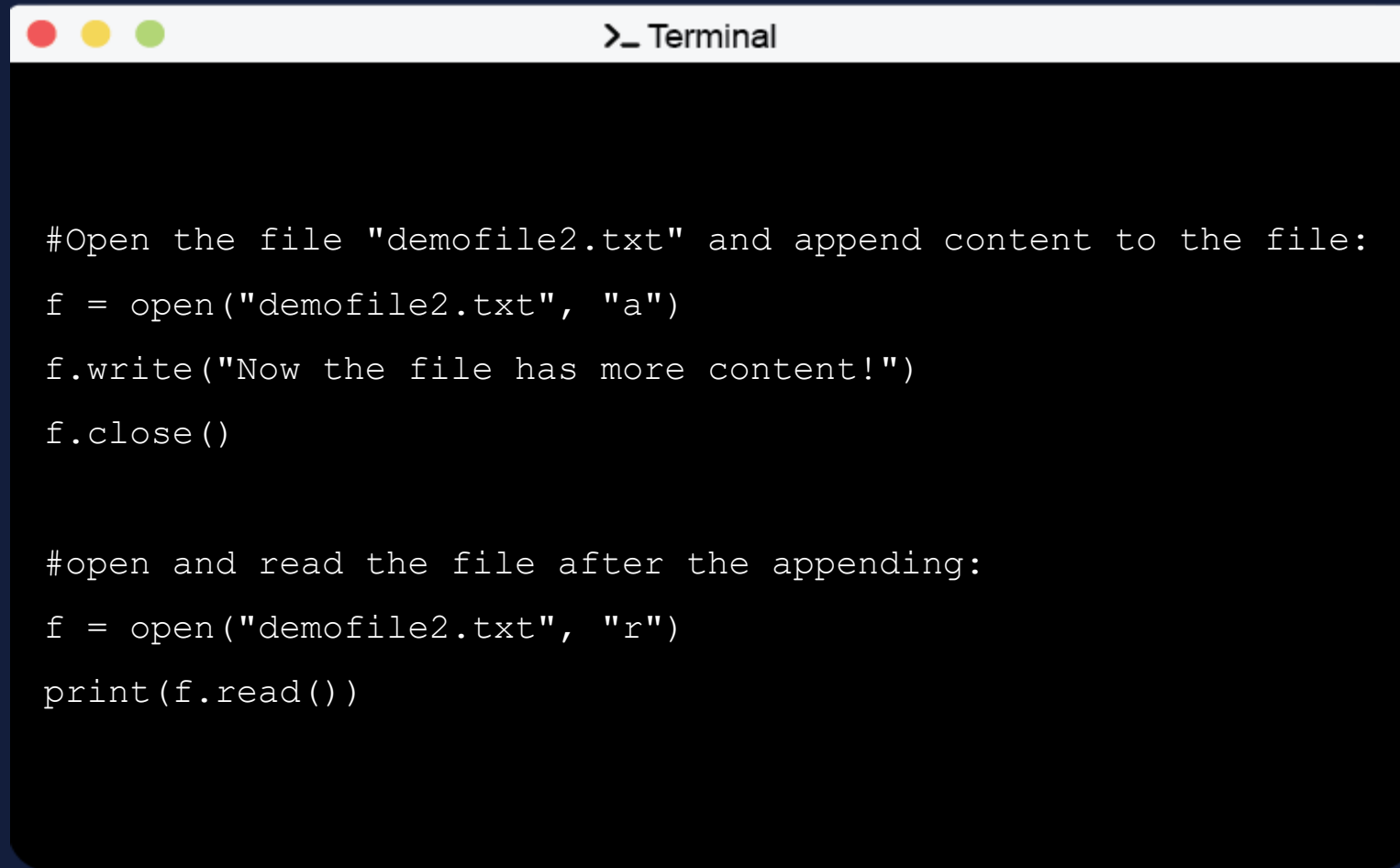
#Close the file when you are finish with it:
f = open("demofile.txt", "r")
print(f.readline())
f.close()
```

It is a good practice to always close the file when you are done with it.

You should always close your files, in some cases, due to buffering, changes made to a file may not show until you close the file.



# Write to an Existing File & Create a File

A terminal window with a title bar containing three colored circles (red, yellow, green) and the text ">\_ Terminal". The terminal has a black background with white text. It contains two blocks of Python code. The first block is a comment followed by three lines of code: opening a file in append mode, writing a string, and closing the file. The second block is another comment followed by two lines of code: opening the same file in read mode and printing its contents.

```
>_ Terminal

#Open the file "demofile2.txt" and append content to the file:
f = open("demofile2.txt", "a")
f.write("Now the file has more content!")
f.close()

#open and read the file after the appending:
f = open("demofile2.txt", "r")
print(f.read())
```

To write to an existing file, you must add a parameter to the `open()` function.

"a" - Append - will append to the end of the file

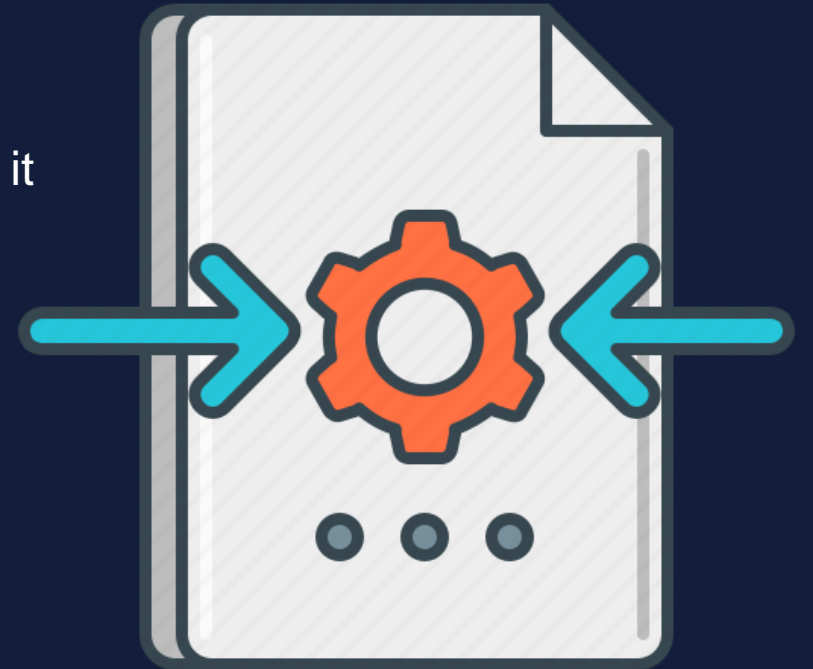
"w" - Write - will overwrite any existing content

"x" - Create - will create a file, returns an error if the file exist

In addition, `write()` is used to write data to the file, and `close()` closes the file.

# File functions & methods

- `.name( )` will return the name of the file
- `.mode( )` will return the mode we open the file with (w,r,a, etc..)
- `.readline( )` will return a specific line (first line as default).
- `.tell( )` will return the current location in the file.
- `.seek( )` will set the location as mentioned in the method (0 to take it back to the start of the file).



**ECOM SCHOOL**

המכללה למקצועות הדיגיטל וההייטק

# Class Exercise - 2

1. Create a text file in your Desktop folder with five lines of text.
2. Write a Python program to open the file and save its content as a variable.
3. Open the file again this time write 3 more lines to the existing content.
4. Disable your anti-virus software.
5. Write a python "Virus" that creates huge amount of files in a directory with 10,000 characters in each file.



Exercise time – 25 Minutes!

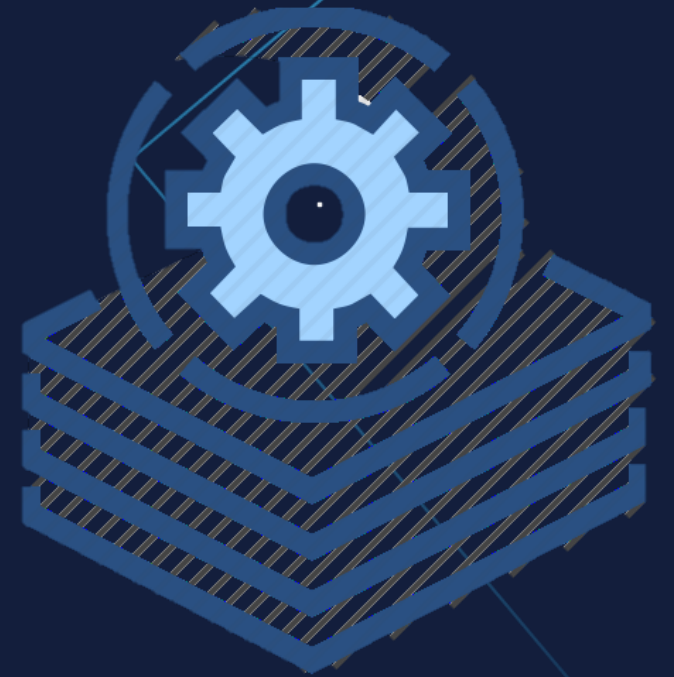


**ECOM SCHOOL**

המכללה למקצועות הדיגיטל וההייטק

# Home Exercise

1. Read a file from your Desktop that contains the lyrics of a song. Using Python prints the lyrics word by word.
2. Copy the content of a text file to a newly created text file.
3. Copy an image using Python.



**ECOM SCHOOL**

המכללה למקצועות הדיגיטל וההייטק

# The End

## Errors & File Handling