```
open and close
myfile = open("filename.txt")
"r"read mode
"w"write mode
"a"append mode
"r+"write/read mode
"wb"write binary mode
....
myfile.close()
The argument of the open function is the path to the file.
working with files
It is good practice to avoid wasting resources by making sure
that files are always closed after they have been used.
try:
        f = open("filename.txt")
    print(f.read())
finally:
        f.close()
An alternative way of doing this is using with statements.
This creates a temporary variable (often called f), which is
only accessible in the indented block of the with statement.
The file is automatically closed at the end of the with
statement, even if exceptions occur within it.
with open("filename.txt") as f:
    print(f.read())
```


## open and close

myfile = open("filename.txt")
"r"read mode
"w"write mode
"a"append mode
"r+"write/read mode
"wb"write binary mode
....
myfile.close()
The argument of the open function is the path to the file.

## working with files

It is good practice to avoid wasting resources by making sure
that files are always closed after they have been used.
try:
f = open("filename.txt")
print(f.read())
finally:
f.close()

An alternative way of doing this is using with statements.
This creates a temporary variable (often called f), which is only accessible in the indented block of the with statement. The file is automatically closed at the end of the with statement, even if exceptions occur within it.
with open("filename.txt") as f:
print(f.read())

## By Nima (nimakarimian)

cheatography.com/nimakarimian/
www.nimakarimian.ir

Published 26th June, 2020.
Last updated 26th June, 2020.
Page 1 of 1 .

```
```

reading

```
```

reading
file = open("filename.txt", "r")
file = open("filename.txt", "r")
cont = file.read()
cont = file.read()
print (cont)
print (cont)
file.close()
file.close()
file = open("filename.txt", "r")
file = open("filename.txt", "r")
print(file.read(16))
print(file.read(16))
This determines the number of bytes that
This determines the number of bytes that
should be read.
should be read.
To retrieve each line in a file, you can
To retrieve each line in a file, you can
use the readlines method
use the readlines method
file = open("filename.txt", "r")
file = open("filename.txt", "r")
print(file.readlines())
print(file.readlines())
file.close()
file.close()
>>>
>>>
['Line 1 text \n', 'Line 2 text \n',
['Line 1 text \n', 'Line 2 text \n',
'Line 3 text']
'Line 3 text']
>>>
>>>
. . . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . .
You can also use a for loop to iterate
You can also use a for loop to iterate
through the lines in the file:
through the lines in the file:
file = open("filename.txt", "r")
file = open("filename.txt", "r")
for line in file:
for line in file:
print(line)
print(line)
file.close()
file.close()
>>>
>>>
Line 1 text
Line 1 text
Line 2 text
Line 2 text
Line 3 text
Line 3 text
>>>

```
```

>>>

```
```

