Introduction to Programming and Data Structures

More on File Handling

Malay Bhattacharyya

Associate Professor

MIU, CAIML, TIH
Indian Statistical Institute, Kolkata

September, 2023

Opening/closing a file

Format: `open(filename, mode, encoding = None)`

Examples:

```
fpr = open('filename1.txt', 'r')
fpw = open('filename2.txt', 'w')
fpa = open('filename3.txt', 'a')
fpr.close()
fpw.close()
fpa.close()
```

Modes for opening a file

Mode	Description
'r'	Open file for reading. Raises an I/O error if not found.
'r+'	Open file for reading/writing. Raises an I/O error if not found.
'w'	Open file for writing. Truncates the file if it already exists. Creates a new file if not found.
'w+'	Open file for reading/writing. Truncates the file if it already exists. Creates a new file if not found.
'a'	Open file for writing. The data being written will be inserted at the end of the file. Creates a new file if not found.
'a+'	Open file for reading/writing. The data being written will be inserted at the end of the file. Creates a new file if not found.
'rb'	Open file for reading in binary. Raises an I/O error if not found.
'rb+'	Open file for reading/writing in binary format. Raises an I/O error if not found.
'wb'	Open file for writing in binary. Truncates the file if it already exists. Creates a new file if not found.
'wb+'	Open file for reading/writing in binary. Truncates the file if it already exists. Creates a new file if not found.
'ab'	Open file for appending in binary. Inserts data at the end of the file. Creates a new file if not found.
'ab+'	Open file for reading/appending in binary. Inserts data at the end of the file. Creates a new file if not found.

Encodings for opening a file

Encoding	Description
'utf-8'	Unicode Transformation Format - 8 is a variable-length character encoding standard used for electronic communication.
'ascii'	American Standard Code for Information Interchange is a standard data-encoding format for electronic communication.

Reading from a file – with closing

```
fpr = open('filename.txt', 'r')
# Read size bytes (or all, if empty)
data = fpr.read(size)
# Read and return the next line
dataline = fpr.readline()
# Read and return next n lines (or all, if empty) as list
datalines = fpr.readlines(n)
fpr.close()
```

Reading from a file – without closing

```
with open('filename.txt') as fpr:
    data = fpr.read()
print(data)
```

Reading from a file – without closing

```
with open('filename.txt') as fpr:
    data = fpr.read()
print(data)
```

OR

```
with open('filename.txt') as fpr:
    for line in fpr:
        print(line)
```

Reading from a file – without closing

```
with open('filename.txt') as fpr:
    data = fpr.read()
print(data)
```

OR

```
with open('filename.txt') as fpr:
    for line in fpr:
        print(line)
```

Note: `close()` function is not required to be applied.

Reading from a file – without closing

```
with open('filename.txt') as fpr:
    data = fpr.readlines()
    for line in data:
        word = line.split()
        print(word)
```

Reading from a file – without closing

```
with open('filename.txt') as fpr:
    data = fpr.readlines()
    for line in data:
        word = line.split()
        print(word)
```

Note: `close()` function is not required to be applied.

Writing into a file – with closing

```
fpw = open('filename.txt', 'w')
fpw.write(str) # Writes a sting
fpw.writelines(ls) # Writes a list of strings
fpw.close()
```

Writing into a file – without closing

```
with open('filename.txt') as fpw:
    fpw.write('Welcome!!!')
```

Writing into a file – without closing

```
with open('filename.txt') as fpw:
    fpw.write('Welcome!!!')
```

Note: `close()` function is not required to be applied.

Clearing the internal buffer of a file

```
fpr = open('filename.txt', 'r')
fpr.flush()
... # Internal buffer will not create any problem here
fpr.close()
```

Getting the position of control in a file

```
fpr = open('filename.txt', 'r')
print(fpr.tell())
input = fpr.read(10)
print(fpr.tell())
fpr.close()
```

Getting the position of control in a file

```
fpr = open('filename.txt', 'r')
print(fpr.tell())
input = fpr.read(10)
print(fpr.tell())
fpr.close()
```

Output:

```
0
10
```

Changing the position of control in a file

```
fpr = open('filename.txt', 'r')
# The point of reference set to the beginning of the file
fpr.seek(10, 0)
# The point of reference set to the current position
fpr.seek(10, 1)
# The point of reference set to the end of the file
fpr.seek(10, 2)
fpr.close()
```

Changing the position of control in a file

```
fpr = open('filename.txt', 'r')
# The point of reference set to the beginning of the file
fpr.seek(10, 0)
# The point of reference set to the current position
fpr.seek(10, 1)
# The point of reference set to the end of the file
fpr.seek(10, 2)
fpr.close()
```

Note: `seek(offset, from_what)` takes an `Offset` argument (number of positions to move forward), a `from_what` argument (point of reference), and returns the new absolute position.

Problems

1. Write a program in Python to show its source code as output on stdout.
2. Write a program in Python to reverse the content of a file whose name is provided as command line argument.