Day_04 (Map & Filter File I/O)

Map

Definition

The Python map() method is used to perform a function on every item in a list, dictionary, tuple, or set. The map() method accepts a function and an object to apply that the function will operate on. ... Python includes a built-in method of applying a specific function to all elements within an iterable object: map()

Purpose

map() function returns a map object(which is an iterator) of the results after applying the given function to each item of a given iterable (list, tuple etc.) ... fun : It is a function to which map passes each element of given iterable.

Importance

Python map() function is used to apply a function on all the elements of specified iterable and return map object. Python map object is an iterator, so we can iterate over its elements. We can also convert map object to sequence objects such as list, tuple etc.

Strengths

advantages of a Map include the size property, an easy way to get the number of items in the Map. With an Object, you would be on your own to figure out its size.

Weaknness

Map is pretty awesome function that helps us alot in applying the single function to the whole list.The weakness is the wrong way using it like if you mistakenly give it the wrong function to apply on the list.

Example 30 :

Task:

use map function to map the lambda function that find the length of the string onto the list of strings to find the length of each element string in the list

In [ ]:

```python
checkString='The quick brown fox jumps over the lazy dog'
words=checkString.split()
print(words)
lengths=map(lambda word:len(word),words)
list(lengths)
```

Example 31 :

Task:

map the lambda function that find if the string is palindrom and apply it onto the list

In [ ]:

```python
resultOfPalindrom=map(lambda word:checkPalindrom(word),words)
list(resultOfPalindrom)
```

Example 32 :

Task:

map the greetings function onto the list and print it

In [ ]:

```python
def greetings(student):
    print("Hy {studentName} you has got {studentCGPA} CGPA".format(studentName=student[0],studentCGPA=student[1]))
    return "Hy {studentName} you has got {studentCGPA} CGPA\n".format(studentName=student[0],studentCGPA=student[1])
greetingsStudent=map(greetings,studentsResult)
greetingsStudent=list(greetingsStudent)
```

Filter

Definition

The filter() method constructs an iterator from elements of an iterable for which a function returns true. In simple words, filter() method filters the given iterable with the help of a function that tests each element in the iterable to be true or not.

Purpose

The filter() method filters the given sequence with the help of a function that tests each element in the sequence to be true or not.

Importance

The filter() function in Python is a built-in function that filters a given input sequence(an iterable) of data based on a function which generally returns either true or false for data elements of the input sequence

Strengths

Filters methods belong to the category of feature selection methods that select features independently of the machine learning algorithm model.

filter methods is that they are very fast.

Weakness

the weakness of filter function is that if we will give it wrong condition to apply

It will take large time for large ammount of Data

Example 33 :

Task:

use filter function to filter out the even numbers from the list of fibonachi series

In [ ]:

```python
evenFib=filter(lambda x:x%2,fibList)
list(evenFib)
```

In [ ]:

```python
studentsResult
```

Example 34 :

Task:

Use the filter function to filter out the students that have cgpa greater than 2 from the tuple using lambda functions

In [ ]:

```python
passStudetsList=filter(lambda student:student[1]>2,studentsResult)
```

In [ ]:

```python
list(passStudetsList)
```

File I/O

Definition

A file is some information or data which stays in the computer storage devices. ... Python gives you easy ways to manipulate these files. Generally we divide files in two categories, text file and binary file. Text files are simple text where as the binary files contain binary data which is only readable by computer.

Purpose

Python file handling (a.k.a File I/O) is one of the essential topics for programmers and automation testers. It is required to work with files for either writing to a file or read data from it.

Also, if you are not already aware, I/O operations are the costliest operations where a program can stumble. Hence, you should be quite careful while implementing file handling for reporting or any other purpose. Optimizing a single file operation can help you produce a high-performing application or a robust solution for automated software testing.

Let’s take an example, say, you are going to create a big project in Python that contains a no. of workflows. Then, it’s inevitable for you not to create a log file. And you’ll also be doing both the read/write operations on the log file. Log files are a great tool to debug large programs. It’s always better to think about a scalable design from the beginning, as you won’t regret it later that you didn’t do it.

Importance

File handling is one of the most important parts of any language. Python language supports two types of files. The first one is a text file that store data in the form of text and readable by humans and computers. The second one is binary file that store binary data and readable by computer only.

Strengths

File are used to save the important data produced during the execution of the program like you want to save the username after providing user a input feild to take his name and than to save it into the file to use t later

Weakness

The weakness of the files is that they contain the impotant information if they get deleted all the information will loss and we can mistakenly overwrite the data that also create the data loss

Example 35 :

Task:

Use input function to take input from user and store it into the variable and print it.

In [ ]:

```python
from six.moves import input
string = input("Enter your name: ");
print(string)
```

Example 36 :

Task:

Create a text file and write something into it

In [ ]:

```python
fileOpen = open("Bio.txt", "w")
for greetings in greetingsStudent:
    fileOpen.write(greetings);
fileOpen.close()
```

Example 37 :

Task:

Create a text file and read something from it

In [ ]:

```python
fileOpen = open("Bio.txt", "r+")
str = fileOpen.read();
print (str)
fileOpen.close()
```

In [ ]:

```python
fo = open("Bio.txt", "r+")
str = fo.read(10);
print ("Read String is : \n", str)
position = fo.tell();
print ("Current file position : \n", position)
position = fo.seek(11, 0);
str = fo.read(10);
print ("Again read String is : \n", str)
# Close opend file
fo.close()
```

In [ ]:

```python
import os
os.rename("Bio.txt", "StudentsGreetings.txt")
```

In [ ]:

```python
```