Branch: master

Find file Copy path

Gitam-Skill-Enhancement-May-2019 / PythonProgramming / PythonNotebooks / 16 May 2019.ipynb

Akash-Sinha commit on 17 May 2019

3230124 8 days ago

1 contributor

Raw Blame History

388 lines (387 sloc) 8.07 KB

Problem Solving and Programming ¶

Day No -

Date -

Day Objectives

1. File Handling
2. External Libraries
3. Functional Programming

Problem 1 :

Problem Statement

Define a function to read data from a text file

Constraints

Test Cases

- Test Case 1
- Test Case 2
- Test Case 3

In [15]:
```
def readFileData(filename):
    with open(filename, 'r') as f:
        #for line in f:
         #   print(line, end = '')
        filedata = f.read()

    return filedata


def writeIntoFile(filename, data, mode):
    with open(filename, mode) as f:
        f.write(data)
    return

#writeIntoFile('DataFiles/fileWrite.txt', 'Second line \n', 'a')
```

In [ ]:
```

```

Problem 2 :

Problem Statement

Define a function to generate a Marks data file(text file) for 1300 students such that each mark is entered in a new line. Marks range from 0 to 100 (inclusive) as random numbers

Constraints

Test Cases

- Test Case 1
- Test Case 2
- Test Case 3

In [39]:
```
import random

def generateMarksData(n, filename):
    with open(filename, 'w') as marksfile:
        for i in range(0, n):
            marks = random.randint(0, 100)
            marksfile.write(str(marks)+'\n')
    return

generateMarksData(1300, 'marksData.txt')
```

In [ ]:
```

```

Problem 3 :

Problem Statement

Generate a report on the marks data with the following indicators

- Highest Mark :
- Lowest Mark :
- Average mark :
- No of students with distinction(>80) :
- No of students with first class(>60) :
- No of students with second class(>50) :
- No of students with third class(>40) :
- No of students failed(<40) :

Constraints

Test Cases

- Test Case 1
- Test Case 2
- Test Case 3

In [53]:
```
import re, timeit
def generateMarksReport(marksfile):
    start = timeit.default_timer()
    marksdata = readFileData(marksfile)
    #print(marksdata[2])
    marksdata = re.split(r'\n', marksdata)
    marksdata = list(map(int, marksdata[:len(marksdata)-1]))
    #print(marksdata[len(marksdata)-2])
    #print(type(marksdata[0]))
    print(max(marksdata))
    return timeit.default_timer() - start

generateMarksReport('marksData.txt')
```

```
100
```

Out[53]:
```
0.0013619729998026742
```

In [ ]:

Problem 1 :

Problem Statement

Map example

Constraints

Test Cases

- Test Case 1
- Test Case 2
- Test Case 3

In [52]:
```
import timeit
def square(n):
    return n * n
st = timeit.default_timer()
li = [1, 2, 3, 4, 5, 6]
#s = str(li)
s = list(map(str, li))

s = [float(i) for i in s]
print(timeit.default_timer()-st)
s
```

```
0.00010992699935741257
```

Out[52]:
```
[1.0, 2.0, 3.0, 4.0, 5.0, 6.0]
```

In [58]:
```
import re, timeit
def generateMarksReport(marksfile):
    start = timeit.default_timer()
    marksdata = readFileData(marksfile)
    #print(marksdata[2])
    marksdata = re.split(r'\n', marksdata)
    marksdata = list(map(int, marksdata[:len(marksdata)-1]))
```

```
    #print(marksdata[len(marksdata)-2])
    #print(type(marksdata[0]))
    #print(max(marksdata))
    return marksdata

marksdata = generateMarksReport('marksData.txt')

def distinction(mark):
    return mark >= 80

dis = sum(map(distinction, marksdata))

failed = sum(1 for i in marksdata if i < 40)
failed
```

Out[58]: 516

In [ ]:

In [ ]:

In [ ]:

```
### Problem 1 :
#### Problem Statement

#### Constraints

#### Test Cases
* Test Case 1
* Test Case 2
* Test Case 3
```

In [60]:

```
import numpy as np   # Importing libraries

a = np.array([[0, 1, 2], [3, 4, 5]])

print(a)
type(a)
```

```
[[0 1 2]
 [3 4 5]]
```

Out[60]: numpy.ndarray

In [ ]:

In [ ]:

```
### Problem 1 :
#### Problem Statement

#### Constraints

#### Test Cases
* Test Case 1
```