Log_file_report

March 5, 2018

1 *This program solves the below question from Code Maven*

2 *Python: Exercise: parse hours log file and give report*

3 *https://code-maven.com/slides/python-programming/exercise-parse-log-file*

```
In [1]: import re
        from datetime import datetime
        from collections import defaultdict
```

```
In [2]: # Stored file as array of strings
        text = open("log.txt").read().splitlines()

        # opened file to write the schedule as output
        fh = open("output.txt","w+")
```

```
In [3]: #Regular expression for searching and grouping
        expr = r"(\d\d):(\d\d) (.*)"
        re_compiled=re.compile(expr)
```

```
In [4]: #Default Dictionary to store events as indices
        dictionary = defaultdict(list)
        total_time_min = 0
```

```
In [5]: for line in range(len(text)-1):

            event_1 = re_compiled.search(text[line])

            event_2 = re_compiled.search(text[line+1])

            output = (event_1.group(1)+":"+event_1.group(2)+"-"+
                        event_2.group(1)+":"+event_2.group(2)+" "+
                            event_1.group(3)+"\n")

            fh.write(output)
```

```
            start_time = str(event_1.group(1)+":"+ event_1.group(2))

            end_time = str(event_2.group(1)+":"+ event_2.group(2))

            t1 = datetime.strptime(start_time,"%H:%M")

            t2 = datetime.strptime(end_time,"%H:%M")

            t_hrs = t2 - t1

            t_min = t_hrs.seconds/60

            if event_1.group(3) != "End":
                total_time_min += t_min
                dictionary[event_1.group(3)].append(t_min)

            #print dictionary.keys()
        fh.close()
        print total_time_min
```

```
970
```

```
In [6]: # accessing the Indices
        list_dictionary = list(dictionary.keys())
```

```
In [7]: for iterator in range(len(list_dictionary)):

            if list_dictionary[iterator] != "End":

                individual_time = sum(dictionary[list_dictionary[iterator]])

                individual_percent = (individual_time*100)/total_time_min

                print (list_dictionary[iterator] +"\t"+ str(individual_time) +
                       "\t"+ str(individual_percent)+"%")
```

```
Functions        30        3%
Introduction        100        10%
Dictionaries        15        1%
Lists        60        6%
Break        65        6%
Numbers and strings        55        5%
Solutions        95        9%
Exercises        340        35%
Lists and Tuples        60        6%
Lunch Break        150        15%
```