گزارش پروژه بازیابی اطلاعات

فاز۱ : بخش ۲ – شاخص مکانی

برای پیاده سازی این بخش ابتدا فایلی که در بخش قبلی پیش پردازش شده بود را باز میکنیم:

```
#opening preprocessed file
file_path = 'C:/Users/Samin/Desktop/University/Term ۷/Information
Retrieval/Project/Data/IR_data_news_۱۲k_preprocessed.json'

try:
    with open(file_path, 'r', encoding='utf-۸') as f:
        preprocessed_data = json.load(f)
        print("File opened successfully!")
except IOError:
    print("Error opening file.")
```

در قدم بعدی میخواهیم یک ساختمان داده ای در نظر بگیریم که امکانات خواسته شده در شاخص مکانی را داشته باشد. برای اینکار یک دیکشنری تعریف میکنیم به نام positional_index_dic در این دیکشنری هر کلمه به صورت یک key ذخیره خواهد شد که دیکشنری های درونی بعنوان value آن خواهیم داشت.

در دیکشنری درونی برای هر داکیومنت docID آن به عنوان کلید و value آن یک دیکشنری دیگری خواهد بود که کلید countرا دارد که نشان دهنده تکرار آن کلمه در آن داکیومنت خاص است و کلید positions را که لیستی از مکان های حضور لغت در این داکیومنت را ذخیره میکند.

کلید دیگر دیکشنری درونی کلید total را داریم که تعداد حضور کلمه در کل داکیومنت هارا بصورت یک دیکشنری درونی با کلید count و مقدار تعداد تکرار ها در آن ذخیره شده است.

مثلا

```
positional_index_dic['مهر']
```

کلمه مهر را در دیکشنری در نظر میگیریم و داکیومنتی با داک آیدی ۱۰۵۳ را در دیکشنری درونی آن صدا میزنیم:

```
print(positional_index_dic['مهر']['۱۰۵۳'])
```

پاسخ به صورت زیر خواهد بود:

```
{'count': 2, 'positions': [23, 128]}
```

همانطور که مشاهده میشود در این داکیومنت دوبار تکرار این کلمه را داشتیم که در موقعیت های ۲۳ و ۱۲۸ اتفاق افتاده است.

با فراخوانی کلید total برای این کلمه

```
print(positional_index_dic['مهر']['total'])
```

خواهیم داشت:

```
{'count': 418}
```

همانطور که مشاهده میشود این کلمه مجموعا در کل داکیومنت ها ۴۱۸ بار تکرار شده است.

اما برای پیاده سازی این ساختمان داده در دیکشنری کلی به صورت زیر عمل خواهیم کرد:

- ابتدا دیکشنری را تعریف میکنیم:

```
-  positional_index_dic = {}
```

- روی فایل پیش پردازش شده حرکت میکنیم و روی هر داکیومنت مجددا حرکت میکنیم تا به هر کلمه پیش پردازش شده در آن برسیم

```
-  for docID, doc in preprocessed_data.items():
       for position, term in enumerate(doc['content']):
```

- برای هرکلمه چک میکنیم که در لیست دیکشنری موجود بوده یا نه، اگر نبوده آن را اضافه میکنیم:

```
-  if term not in positional_index_dic:
       positional_index_dic[term] = {}
```

- برای این کلمه چک میکنیم که آیا این داکیومنت در لیست کلید های دیکشنری درونی اش اضافه شده یا نه، اگر نشده اضافه میکنیم:

```
-  if docID not in positional_index_dic[term]:
       positional_index_dic[term][docID] = {'count': ۰, 'positions': []}
```

- حالا اگر ساختمان داده مدنظر را نداشته بودیم برای این کلمه ساخته ایم و اگر داشته ایم باید اطلاعات را در آن اضافه کنیم، پس به تعداد تکرار کلمه در این داکیومنت یکی اضافه کرده و موقعیت آن را به لیست positions اضافه میکنیم:

```
-  positional_index_dic[term][docID]['count'] += ۱
   positional_index_dic[term][docID]['positions'].append(position)
```

- در قدم بعدی اگر کلید total برای این کلمه تعریف نشده بود آن را تعریف میکنیم، سپس به تعداد تکرار کلمه در کل داکیومنت ها می افزاییم:

```
-   if 'total' not in positional_index_dic[term]:
        positional_index_dic[term]['total'] = {'count': ۰}
    positional_index_dic[term]['total']['count'] += ۱
```

با حرکت روی تمامی اطلاعات داده های پیش پردازش شده دیکشنری با شاخص مکانی ساخته میشود، حالا آن را ذخیره میکنیم:

```
# save positional index dic as a JSON file
output_file_path = 'C:/Users/Samin/Desktop/University/Term ۷/Information
Retrieval/Project/Data/IR_data_news_۱۲k_positional_index_dic.json'
with open(output_file_path, 'w', encoding='utf-۸') as f:
    json.dump(positional_index_dic, f, ensure_ascii=False, indent=٤)
```