Indexing reports:

My code for testing it: (you can find it on `index.py`)

I have broken it into different parts to be viewed more easily:

- **Preparation:**

```
if __name__ == '__main__':
    prepreprocessed_documents_path =
project_root+'/data/IMDB_Preprocessed.json'
    with open(prepreprocessed_documents_path, 'r') as f:
        preprocessed_documents = json.load(f)

    index = Index(preprocessed_documents)
```

- **Checking add/remove:**

Code:

```
print("Cheking add/remove:")
index.check_add_remove_is_correct()
```

Output:

```
Cheking add/remove:
Add is correct
Remove is correct
```

- **Storing and loading the index files:**

Code:

```
index.store_index(project_root+'/index', Indexes.DOCUMENTS.value)
index.store_index(project_root+'/index', Indexes.STARS.value)
index.store_index(project_root+'/index', Indexes.GENRES.value)
index.store_index(project_root+'/index', Indexes.SUMMARIES.value)
index.load_index('index')
```

- **Checking if index loaded correctly:**

Code:

```
print("Checking if index loaded correctly (documents):")
print(index.check_if_index_loaded_correctly(
        Indexes.DOCUMENTS.value,
        index.index[Indexes.DOCUMENTS.value]))

print("Checking if index loaded correctly (stars):")
print(index.check_if_index_loaded_correctly(
        Indexes.STARS.value,
        index.index[Indexes.STARS.value]))

print("Checking if index loaded correctly (genres):")
print(index.check_if_index_loaded_correctly(
        Indexes.GENRES.value,
        index.index[Indexes.GENRES.value]))

print("Checking if index loaded correctly (summaries):")
print(index.check_if_index_loaded_correctly(
        Indexes.SUMMARIES.value,
        index.index[Indexes.SUMMARIES.value]))
```

Output:

```
Checking if index loaded correctly (documents):
True
Checking if index loaded correctly (stars):
True
Checking if index loaded correctly (genres):
True
Checking if index loaded correctly (summaries):
True
```

- **Checking if indexing is good (documents) for 2 different words:**

Code:

```
print("Checking if indexing is good (documents):")
print('Word: tt1160419')
print(index.check_if_indexing_is_good(Indexes.DOCUMENTS.value, 'hello'))
print('Word: tt15239678')
print(index.check_if_indexing_is_good(Indexes.DOCUMENTS.value, 'word'))
```

Output:

```
Checking if indexing is good (documents):
Word: tt1160419
Brute force time:  8.034706115722656e-05
Implemented time:  3.814697265625e-06
Indexing is correct
Indexing is good
True
Word: tt15239678
Brute force time:  4.291534423828125e-05
Implemented time:  7.152557373046875e-07
Indexing is correct
Indexing is good
True
```

- **Checking if indexing is good (stars) for 2 different words:**

Code:

```
print("Checking if indexing is good (stars):")
print('Word: ben')
print(index.check_if_indexing_is_good(Indexes.STARS.value, 'ben'))
print('Word: bob')
print(index.check_if_indexing_is_good(Indexes.STARS.value, 'bob'))
```

Output:

```
Checking if indexing is good (stars):
Word: ben
Brute force time:  4.1961669921875e-05
Implemented time:  2.86102294921875e-06
Indexing is correct
Indexing is good
True
Word: bob
Brute force time:  0.0003216266632080078
Implemented time:  1.1920928955078125e-06
Indexing is correct
Indexing is good
True
```

- **Checking if indexing is good (genres) for 2 different words:**

Code:

```
print("Checking if indexing is good (genres):")
print('Word: history')
print(index.check_if_indexing_is_good(Indexes.GENRES.value, 'history'))
print('Word: comedy')
print(index.check_if_indexing_is_good(Indexes.GENRES.value, 'comedy'))
```

Output:

```
Checking if indexing is good (genres):
Word: history
Brute force time:  2.09808349609375e-05
Implemented time:  2.86102294921875e-06
Indexing is correct
Indexing is good
True
Word: comedy
Brute force time:  1.1920928955078125e-05
Implemented time:  5.9604644775390625e-06
Indexing is correct
Indexing is good
True
```

- **Checking if indexing is good (summaries) for 2 different words:**

Code:

```
print("Checking if indexing is good (summaries):")
print('Word: murder')
print(index.check_if_indexing_is_good(Indexes.SUMMARIES.value, 'murder'))
print('Word: happy')
print(index.check_if_indexing_is_good(Indexes.SUMMARIES.value, 'happy'))
```

Output:

```
Checking if indexing is good (summaries):
Word: murder
Brute force time:  7.390975952148438e-05
Implemented time:  7.152557373046875e-06
Indexing is correct
Indexing is good
True
Word: happy
Brute force time:  0.0006542205810546875
Implemented time:  2.86102294921875e-06
Indexing is correct
Indexing is good
True
```

As you can see, my code on `index.py` has passed all the tests.