`string_manipulator` Documentation

Kamyar Modjtahedzadeh

Boeing Intelligence & Analytics

March 4, 2024

`string_manipulator` is a Python 3 package written to help Boeing Intelligence & Analytics (BI&A) employees to analyze and manipulate large files. As of March 4, 2024 it only consists of one class: the `Text` class.

1 `Text` Class

The `Text` class should take in an object, which for the purpose of BI&A's work it is usually in the shape of a file converted into a string or into a list of strings; whether that is a `.txt` type file, `.dat` file, or whatever file that has raw string data.

```
string_manipulator.Text(object)
```

Once the `object` has been taken in as a parameter, datum of the type `Text` will have been created in Python. Based on the current implementations of the methods in `Text`, it is best that `object` be a string.

1.1 Methods

1.1.1 `Text.find_string`

As its name suggests, the `find_string` method can find substrings.

```
σ.find_string(word)
```

returns: `tuple`

where σ is an object of class `Text`. The returned tuple is of length 2 and its first element is the amount of times the string `word` appears in `object`, its second element is a list of each position of each occurrence of `word[0]`. Please note that with the way `find_string` is implemented, $\mathcal{S}$ must be the string containing the substring `word` in σ `= Text(`$\mathcal{S}$`)`.

Example

```
with open("inFile", "r") as f:
    data = f.read().lower()

text = Text(data)
text.find_string(word)
```

1.1.2 `Text.split_by_size`

The `split_by_size` method splits larger files into smaller files based on the measure of space consumed by the file.

```
φ.split_by_size(size, ext, folder, fname)
```

returns: `None`

where φ is an object of class `Text`. `size` is the *maximum* size of the output files in bytes,[1] `ext` is the type of the output files, `folder` is the path to the output files, and `fname_`ι are the names of the output files where ι starts from 0. Please note that with the way `find_string` is implemented, $\mathcal{P}$ must be the string path to the input file in φ `= Text(`$\mathcal{P}$`)`.

Example

```
inFile = Text("path/to/file")
inFile.split_by_size(size, ext, folder, fname)
```

1.1.3 `Text.divide_by_lines`

LaTeX

[1] Each file will be maxed out at approximately `size` bytes, but the last file should be $\leq$ `size` depending whether or not the original file size is divisible by `size`.