`string_manipulator` Documentation

Kamyar Modjtahedzadeh

Boeing Intelligence & Analytics

March 6, 2024

`string_manipulator` is a Python 3 package written to help Boeing Intelligence & Analytics (BI&A) employees to analyze and manipulate large files.

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(content)
```

Once `content` has been taken in as a parameter, datum of the type `Text` will have been created in Python.

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, $\mathbb{S}$ must be the string containing the substring `word` in σ = `Text(`$\mathbb{S}$`)`.

Example

```python
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

```python
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`.

1.1.4 `Text.split_by_lines`

TEX

2 `ExSpread` **Class**

`ExSpread` is for manipulating data to and/or from spreadsheets. It takes in the path to the input file in the form of a string and an optional parameter to be searched.

> `string_manipulator.ExSpread(fpath: str, search=None)`

Whilst `search` should typically be a string, it can be of any data type.

2.1 **Methods**

2.1.1 `ExSpread.mk_timesheet`

The purpose of this method is to extract timestamps from lines beneath certain strings in `.ascii_out` files (although it will work with other file types).

Let ε be an object of the `ExSpread` class that has a string to be searched passed to it;

> `ε.mk_timesheet(folder: str, fname: str)`

returns: `None`

where `folder` the is path to the directory of the output file and `fname` is the name of the output file which will be a spreadsheet of type `.CSV`.

Example

```
inFile = ExSpread("path/to/inFile", "string to be searched")
inFile.mk_timesheet("path/to/outFile", "outFile_name")
```