1. **Havent’t used, but found this functionality useful**

Using with to open multiple files at once in a single line

**Example:**

with open(file1,’r’,encoding=”utf-8”) as first\_file, open(file2,’w’) as second\_file:

for line in first\_file:

second\_file.write(line)

1. **Abstract Methods.**

Class Parser():

def parse(self,contents):

raise NotImplementedError

When a class inherits Parser it should implement and overwrite parse method, otherwise an exception is raised.

This way of implementing abstract method has a disadvantage. Say the Current\_incident() class which is inheriting the Parser class forgot to implement the parse method. Then an error will be raised only when that method is accessed.

Example:Parser()

Will not create any error, Error will be raised only when

Parser.parse() is called.

This can be avoided using ABC module which is in-built in python.

From abc import ABCMeta, abstractmethod

Class Parser(metaclass=ABCMeta):

def parse(self,contents):

“Method is implemented in child class”

By using this module, when you try to instantiate Parser() class, you will get an error, if the method is not implemented in inherited class.

1. **Doesn’t** automatically return last expression in a method or function, which is very awkward in functional programming. Returns None instead

**Python string function split()** doesn’t support multiple delimiter

For example,

Time\_value=’12-04-1990 12:34:40’

Time\_value.split(‘- :’)

This doesn’t work. Split() function works only with single delimiter. To deal with case, regular expressions should be used.

1. In python, when using built-in functions knowing the function is not enough, you need to even know how to use it. In python, code is not read from left to right, so I was confused while using split function, whether to use str.split() or split(str). Also, if I want get length of the list returned by split(), I am not sure whether to use len(list) or list.len().

I have also checked, how it works in ruby. I found that in ruby the code is read from left to right. So all built in functions are used as function\_name(). So I just have to know the function\_name and don’t have to worry about the syntax. This prevents python from providing pipe like functionality which is very important in functional programming languages.

Str=”12-04-1990”

In python: len(str.split(‘-‘))

In Ruby: str.split(‘-‘).length

1. To manipulate the lists, Python has list comprehensions which are easy to use and compact. They can even be nested, which makes them more powerful.
2. Python allows importing selected modules from a file. You need not to import complete file when you need just a module in it, you can just say “from file import module”
3. Python uses indentation to indicate blocks of code, unlike other languages which uses {} or end to indicate blocks. Python uses ‘:’ to indicate start of code block.

Comparisons:

Os.path.join() In python: Used for working on OS independent file paths.

1. Os.path.join(“/Users/WRPS\_DATA”, “/Current\_incidents/preprocess”)

‘/Current\_incidents/preprocess’

File.join() in Ruby:

1. File.join(“/Users/WRPS\_DATA”, “/Current\_incidents/preprocess”)

‘/Users/WRPS\_DATA/Current\_incidents/preprocess’

joinPath() in php:

1. joinPath(“/Users/WRPS\_DATA”, “/Current\_incidents/preprocess”)

‘/Users/WRPS\_DATA/Current\_incidents/preprocess’

1. No access modifiers, whereas ruby has access modifiers.

Urllib.request

Provides a simple and straight forward way to fetch data from urls. Deals with blocking connections by having an optional “timeout” parameter in seconds. However, this works only with HTTP, HTTPS, FTP and FTPS connections only.

Os.makedirs()

Easy to built-in function to create directories recursively. Creates all the intermediate directories needed to create leaf directory.

Pymongo :

Pymongo provides a simple interface for interacting with MongoDB from Python.

Bottle:

Supports MVC framework. Bottle has route module that specifies which URL request has to be handled by which python function. It also has run module which is used to run the application in development server.

Bottle has built-in template engine called Simple Template Engine. Template() function can be used to render a template. Python expression can be written in {{ }}, as long as it evaluate to a string in Simple Template Engine. Python code can be embedded in the template by starting the code with %. Also python code blocks can be included in <% and %>

Libraries Used:

os ------- This module provides a portable way of using operating system dependent functionality.

time ----- This module provides various time-related functions

datetime---- This module is used for manipulating dates and times

urllib.request ----- This module has functions for opening urls.

re --------- This module provides regular expression matching operations

json -----Changing the data format into Javascript object notation.

shutil ---- This module allows operations on files and collection of files

abc --- This provides a way for defining abstract base classes

pymongo --- This module has tools for working with MongoDB

matplotlib ---- Provides functions for plotting

**Framework:**

bottle ----- It is a lightweight WSGI micro web-framework for python.