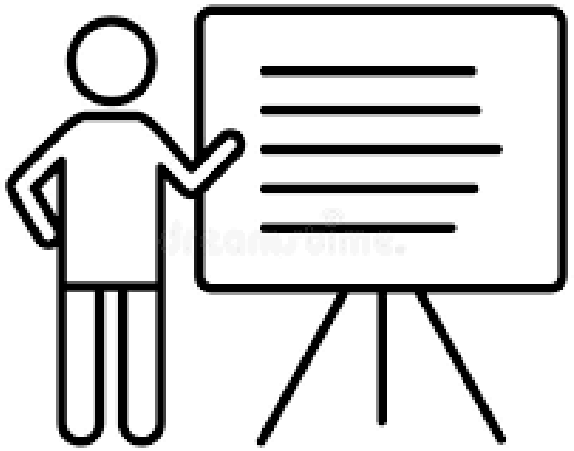
A Python Application for Auditors

Avery Jan

6-30-2022

Outline

- Background
- Application
- Dataset Folders
  -KITH-2017, KITH-2018, KITH-2019
  (4 digits in the name represents the year.)
- Modules
  -utils, pilots, violations, app
- Application Tests on all functions defined
- Sample Output (2017)
- Audit Results 2017, 2018, 2019



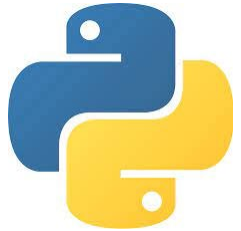
Background

A company that provides insurance for a flight training school wants to ensure that the school complies with the requirements for its insurance. The auditor is given three of a year's worth of records for that school. The audit is divided into three stages, each ensures one of the following:

(1) The students don't take off in bad weather.
(2) The students only fly planes that they are certified for.
(3) The airplanes have regular scheduled maintenance.

This tested application serves as the tool for the auditor to complete the first stage of the audit, “the weather violations”, on 2017, 2018, and 2019 records.

Application

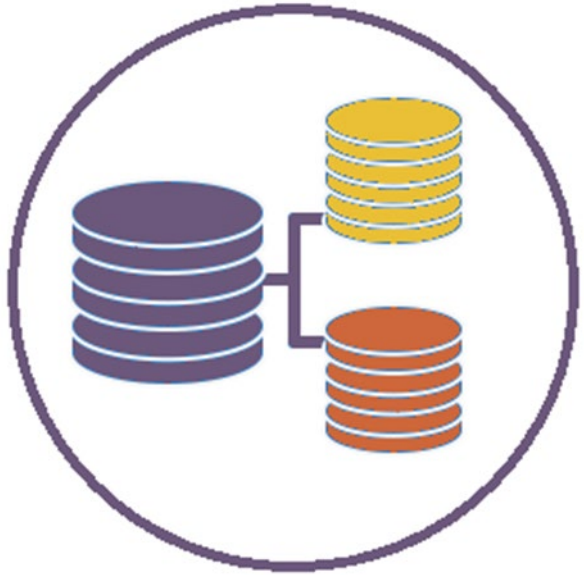


- This application is **a multi-file application** that consists of **four modules**. Each module is in a python file.
- For each year audited, this application retrieves data from five datasets of that year to compare the records of the flight lessons provided by the school against the minimum requirements set by flight regulations.
- Then, the number of weather violations by the school is determined and the information regarding each violation is listed in a file named "output."

Datasets

Folders: KITH-2017, KITH-2018, KITH-2019, each containing these datasets for the year

- **daycycle.json**
  daily sunrise, sunset information
- **lessons.csv**
  records of flight lessons
- **minimums.csv**
  restrictions on weather conditions like ceiling, visibility, wind, and crosswind, for lessons
- **students.csv**
  student credentials like id, certifications, fly with instrument (vfr status), etc.
- **weather.json**
  daily weather report



Modules

Folder: auditor

1. **utils.py**

   To read and write files and to process dates and times

2. **pilots.py**

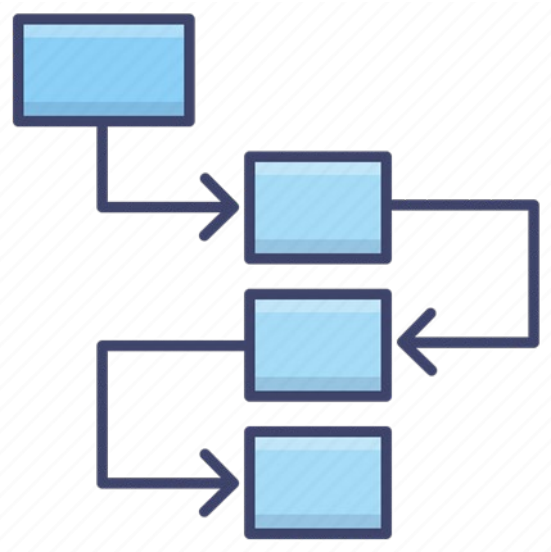
   To find information about students' certifications and the conditions they're allowed to fly

3. **violations.py**

   To check for violations of weather restrictions

4. **app.py**

   To runs the application or test it



utils.py

Import: csv, json, daytime, pytz, from dateutil.parser import parse

- **read_csv(filename)**
  Reads the contents of the file "filename" and returns the contents as a 2-dimensional list.
- **write_csv(data,filename)**
  Writes the given "data" out as a CSV file "filename".
- **read_json(filename)**
  Returns the contents read from the JSON file "filename".
- **str_to_time(timestamp,tzsource=None)**
  Returns the datetime object for the given timestamp or None if the timestamp is invalid.
- **daytime(time, daycycle)**
  Returns True if the "time" takes place during the day.
- **get_for_id(id, table)**
  Returns (a copy of) a row with the given "id" from the "table."

pilots.py

Import: utils

- **get_certification(takeoff, student)**

  Returns the certification classification for this "student" at the time of "takeoff."

- **get_best_value(data, index, maximum=True)**

  Returns the 'best' value from a given column in a 2-dimensional nested list.
  This is a helper function of get_minimums(cert, area, instructed, vfr, daytime, minimums).

- **get_minimums(cert, area, instructed, vfr, daytime, minimums)**

  This is **the main function**. It returns the most advantageous minimums for the given flight category. Parameter "minimums" is a table of allowed minimums (a 2d-list). This is a complicated function that (1) searches the table (minimums.csv) for rows that match this function's parameters and stores them in a 2-d list, which will be used as the parameter "data" of the get_best_value function (2) determines the minimums of ceiling and visibility and the maximums of windspeed and crosswind (These are what the most advantageous minimums refer to.) by calling the above helper function, get_best_value.

violations.py

Import: utils, pilots, os.path

- **bad_visibility(visibility,minimum)**

  Returns True if the visibility measurement violates the minimum, False otherwise.
- **bad_winds(winds,maxwind,maxcross)**

  Returns True if the wind measurement violates the maximums, False otherwise.
- **bad_ceiling(ceiling,minimum)**

  Returns True if the ceiling measurement violates the minimum, False otherwise.
- **get_weather_report(takeoff,weather)**

  Returns the most recent weather report at or before take-off.
- **get_weather_violation(weather,minimums)**

  Returns a string representing the type of weather violation (empty string if flight is ok).
- **list_weather_violations(directory)**

  Returns the (annotated) list of flight lessons that violate weather minimums.

app.py

Import: utils, tests, os.path, violations

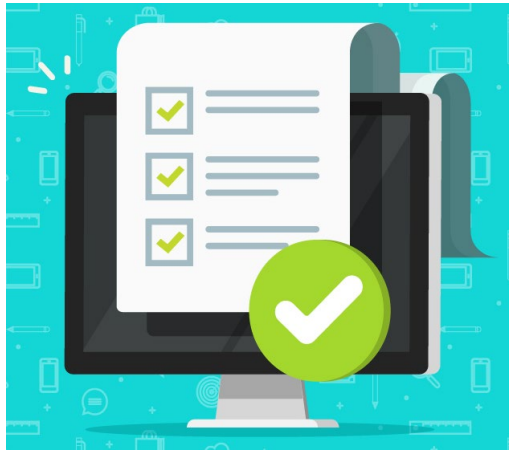
- **discover_violations(directory,output)**

  It searches the dataset directory for any flight lessons that violate regulations. If any of such lessons is found, information about the lesson will be included in the file "output.csv" (see Sample Output). In addition, it prints a message that states "# (the number) violations found." (see Audit Results). If no such lessons are found, the message states "No violations found." instead.

- **execute(args)**

  It executes the application or prints an error message if executed incorrectly.

Application Tests



* Credit: Test codes were provided by Dr. Walker M. White at Cornell University

```
Last login: Wed Jun 29 19:50:07 2022 from 192.168.10.93
codio@modemsardine-stadiumtraffic:~/workspace$ python3 auditor --test
Testing module utils
  utils.read_csv passed all tests
  utils.write_csv passed all tests
  utils.read_json passed all tests
  utils.str_to_time passed all tests
  utils.daytime passed all tests
  utils.get_for_id passed all tests
Testing module pilots
  pilots.get_certification passed all tests
  pilots.get_best_value passed all tests
  pilots.get_minimums passed all tests
Testing module violations
  violations.bad_visibility passed all tests
  violations.bad_winds passed all tests
  violations.bad_ceiling passed all tests
  violations.get_weather_report passed all tests
  violations.get_weather_violation passed all tests
  violations.list_weather_violations passed all tests
Testing module app (this may take a while)
  app.discover_violations passed all tests
  app.execute passed all tests
The application passed all tests
codio@modemsardine-stadiumtraffic:~/workspace$
```

Sample Output

Folder: auditor
File: output.csv
Dataset: KITH-2017

	A	B	C	D	E	F	G	H
1	STUDENT	AIRPLANE	INSTRUCTOR	TAKEOFF	LANDING	FILED	AREA	REASON
2	S00687	548QR	I061	2017-01-08T14:00:00-05:00	2017-01-08T16:00:00-05:00	VFR	Pattern	Winds
3	S00758	548QR	I072	2017-01-08T09:00:00-05:00	2017-01-08T11:00:00-05:00	VFR	Pattern	Visibility
4	S00880	133CZ	I072	2017-01-08T14:00:00-05:00	2017-01-08T16:00:00-05:00	VFR	Pattern	Winds
5	S00971	426JQ	I072	2017-01-12T13:00:00-05:00	2017-01-12T15:00:00-05:00	VFR	Pattern	Ceiling
6	S00922	133CZ	I053	2017-01-18T11:00:00-05:00	2017-01-18T13:00:00-05:00	VFR	Practice Area	Weather
7	S00974	426JQ	I061	2017-01-18T11:00:00-05:00	2017-01-18T13:00:00-05:00	VFR	Pattern	Ceiling
8	S00803	426JQ	I072	2017-01-20T14:00:00-05:00	2017-01-20T16:00:00-05:00	VFR	Pattern	Ceiling
9	S00717	426JQ	I061	2017-01-22T10:00:00-05:00	2017-01-22T12:00:00-05:00	VFR	Pattern	Ceiling
10	S00591	446BU		2017-01-23T09:00:00-05:00	2017-01-23T11:00:00-05:00	VFR	Pattern	Ceiling
11	S00789	684TM	I061	2017-01-25T13:00:00-05:00	2017-01-25T15:00:00-05:00	VFR	Practice Area	Ceiling
12	S00822	254SE	I076	2017-02-02T13:00:00-05:00	2017-02-02T15:00:00-05:00	VFR	Practice Area	Visibility
13	S00789	254SE	I076	2017-02-02T09:00:00-05:00	2017-02-02T11:00:00-05:00	VFR	Practice Area	Ceiling
14	S00990	133CZ	I076	2017-02-08T09:00:00-05:00	2017-02-08T12:00:00-05:00	VFR	Practice Area	Ceiling
15	S00978	133CZ	I076	2017-02-09T12:00:00-05:00	2017-02-09T14:00:00-05:00	VFR	Practice Area	Weather
16	S01000	738GG	I076	2017-02-13T10:00:00-05:00	2017-02-13T12:00:00-05:00	VFR	Pattern	Weather

Audit Results

2017: 93 violations found
2018: 169 violations found
2019: 223 violations found

```
Last login: Wed Jun 29 18:48:27 2022 from 192.168.10.93
codio@modemsardine-stadiumtraffic:~/workspace$ python3 auditor KITH-2017
93 violations found.
codio@modemsardine-stadiumtraffic:~/workspace$ python3 auditor KITH-2018
169 violations found.
codio@modemsardine-stadiumtraffic:~/workspace$ python3 auditor KITH-2019
223 violations found.
codio@modemsardine-stadiumtraffic:~/workspace$
```