egyptfwd initiative

UDACITY

fwd initiative

Project-Bike Share Data
Walk-through

Agenda

Project Overview

Code Walkthrough

1

Project Details

- Overview
- The Datasets
- Statistics Computed
- The Files

2

Workspace & Submission

3

Data loading

get_filter and load_data functions

4

Statistics Output

4 functions

5

Interactive Raw Data display

display_raw_data(city)

itida

Agenda

Project Overview

Code Walkthrough

1

Project Details

- Overview
- The Datasets
- Statistics Computed
- The Files

2

Workspace & Submission

3

Data loading

get_filter and load_data functions

4

Statistics Output

4 functions

5

Interactive Raw Data display

display_raw_data(city)

itida

Project overview

The Files:

Inputs:

1. chicago.csv
2. new_york_city.csv
3. Washington.csv

+

1. Raw input

`bikeshare.py`

Outputs :

Interactive script displaying statistics and Data upon request

itida

Project Details

The Datasets:

1. Randomly selected data for the ***first six months of 2017*** are provided for all three cities. All three of the data files contain the same core six (6) columns:
    a. Start Time (e.g., 2017-01-01 00:07:57)
    b. End Time (e.g., 2017-01-01 00:20:53)
    c. Trip Duration (in seconds - e.g., 776)
    d. Start Station (e.g., Broadway & Barry Ave)
    e. End Station (e.g., Sedgwick St & North Ave)
    f. User Type (Subscriber or Customer)
2. The **Chicago** and **New York City** files also have the following two columns:
    a. Gender
    b. Birth Year

Project Details

User Input

User should input three variable

1. City (there are only three options Chicago, New York City and Washington)
2. Time Frame input: Would you like to filter the data by month, day, both or not at all?
3. Month (available options January, February, March, April, May, June)
4. Day (available options all days are avaiable)
5. If he want to display random rows of data.
6. If he want to restart

Project Details

Statistics Computed:

1. **Popular times** of travel (i.e., occurs most often in the start time):
    a. most common month
    b. most common day of week
    c. most common hour of day
2. **Popular stations** and trip:
    a. most common start station
    b. most common end station
    c. most common trip from start to end (i.e., most frequent combination of start station and end station)
3. **Trip duration**:
    a. total travel time
    b. average travel time
4. **User info**:
    a. counts of each user type
    b. counts of each gender (**only available for NYC and Chicago**)
    c. earliest, most recent, most common year of birth (only available for NYC and Chicago)

Project Details

What Software Do I Need to complete this project locally?:

1. You should have **Python 3**, **NumPy**, and **pandas** installed using **Anaconda**
2. A text editor, like **Sublime** or **Atom**.
3. A terminal application

Agenda

Project Overview

Code Walkthrough

1

Project Details

- Overview
- The Datasets
- Statistics Computed
- The Files

2

Workspace & Submission

3

Data loading

get_filter and load_data functions

4

Statistics Output

4 functions

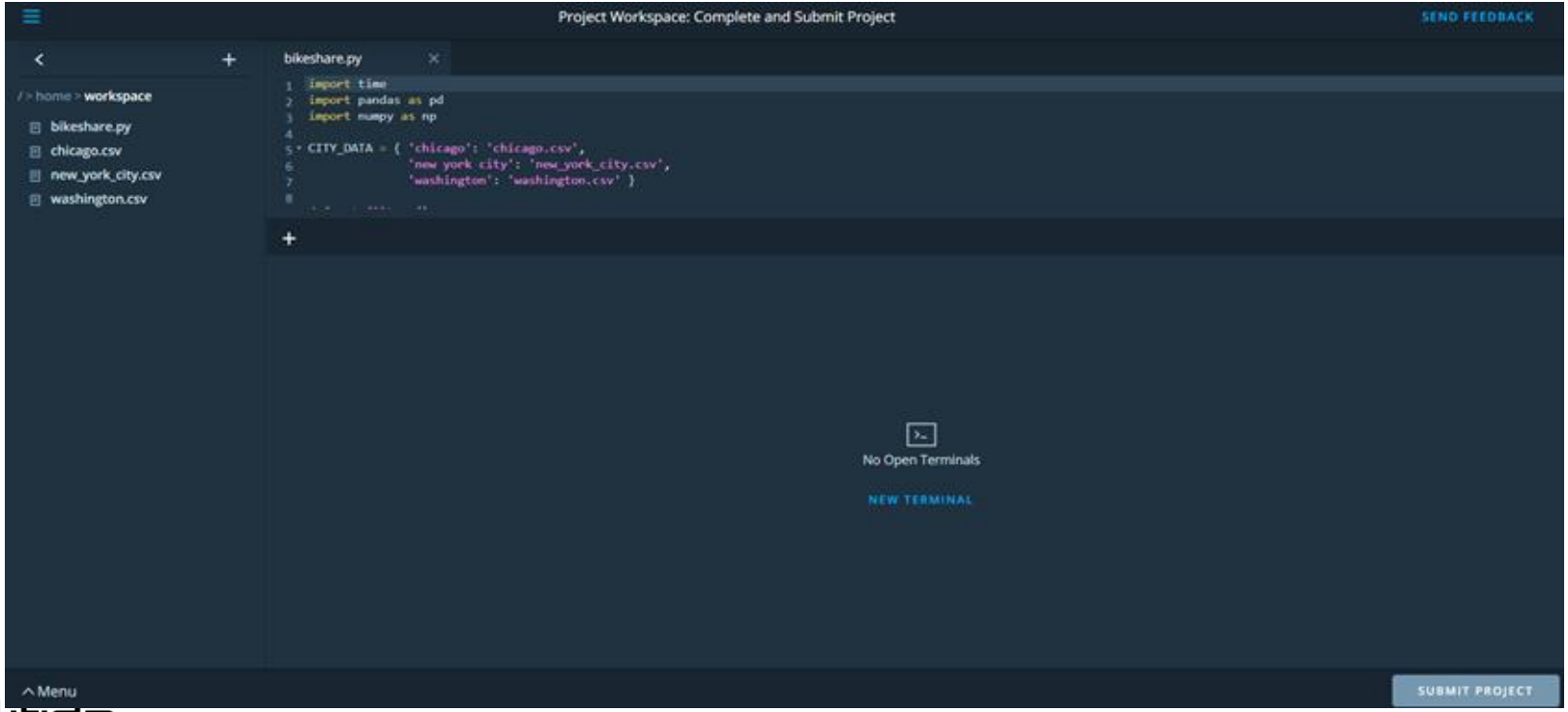
5

Interactive Raw Data display

display_raw_data(city)

itida

Workspace & Submission



Workspace & Submission

Before You Submit::

1. **Check the Rubric**:

   Your project will be evaluated by a Udacity reviewer according to this Project Rubric. Be sure to **review it thoroughly before you submit**. Your project "**meets specifications**" only if it meets specifications **in all the criteria**.

1. **Gather Submission Materials:**

   a. `bikeshare.py:` Your code

   b. `readme.txt:` If you refer to other websites, books, and other resources to help you in solving tasks in the project, make sure that you document them in this file

Agenda

Project Overview

Code Walkthrough

1

Project Details

- Overview
- The Datasets
- Statistics Computed
- The Files

2

Workspace & Submission

3

Data loading

get_filter and load_data functions

4

Statistics Output

4 functions

5

Interactive Raw Data display

display_raw_data(city)

itida
هيئة تنمية صناعة تكنولوجيا المعلومات

Libraries and Data

Inputs:
Raw input (City - Timeframe - Which month /Which day)

Bikeshare.py

Outputs :

Interactive script that answers questions about the dataset

Script Setting Up:

```
import time
import pandas as pd
import numpy as np

CITY_DATA = { 'chicago': 'chicago.csv',
              'new york city': 'new_york_city.csv',
              'washington': 'washington.csv' }
```

Importing the required libraries at the top of the script as per the best practices.

Assigning a dictionary to map the city names' to the corresponding file name path in the file system to access later within the code.

itida

User Questions

There are four questions that user should response:

1. **The City input**: Would you like to see data for Chicago, New York, or Washington?
2. **TimeFrame input:** Would you like to filter the data by month, day, or not at all?
3. **Month input (If they chose month):** Which month - January, February, March, April, May, or June?
4. **Day input (If they chose day):** Which day - Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, or Sunday?

itida

Algorithm

```
def get_filters(df):
    month, day == "all", "all"
    cities = list of available cities
    months = list of first six months
    days = list of all week days
    city = get user input
    while city not in cities:
        city = get a valid input from user
    filter = get input if user wants to filter data
    if filter == 'month' or filter == "all":
        month = get month input
    while month not in months:
        month = get month input
```

Algorithm

```
def get_filters(df):
    ……..
    if filter == “day” or filter ==“all”:
        day = get day input
        while day not in days:
            day = get a valid input
    return (city, month, day)
```

Remember:

Any time you ask users for input, there is a chance they **may not enter what you expect**, so your code should **handle unexpected input well without failing**. You need to **anticipate raw input errors** like:

1. Using improper upper or lower case
2. Users misunderstanding what you are expecting.

Interactive Experience

Inputs:
Raw input (City - Timeframe - Which month /Which day)

`bikeshare.py (get_filter())`

Outputs :

Interactive script that answers questions about the dataset

Simple Algorithm to get and check input

```
def get_city_name():
    cities = list of available cities
    city = get user input
    while city not in cities:
        city = get a valid input from user
    return city
```

The `get_filter()` function

Outputs:

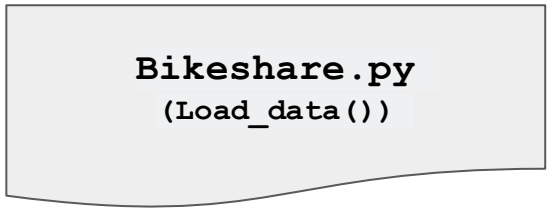
Now, you can set the return statement: `return(city, month, day)`

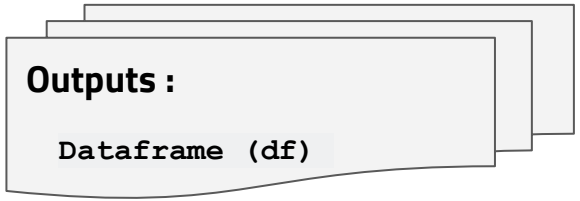
- **Don't forget testing** your script after you are done writing the `get_filters()` function. You **call** it and assign the result to the variable names that will be used as input for the `load_data()` function like this:

```
filtered_values = get_filters()
city, month, day = filtered_values
```

- Also Take EXTRA CARE of the INDENTATION

The `load_data()` Function





The `load_data(city, month, day)`:

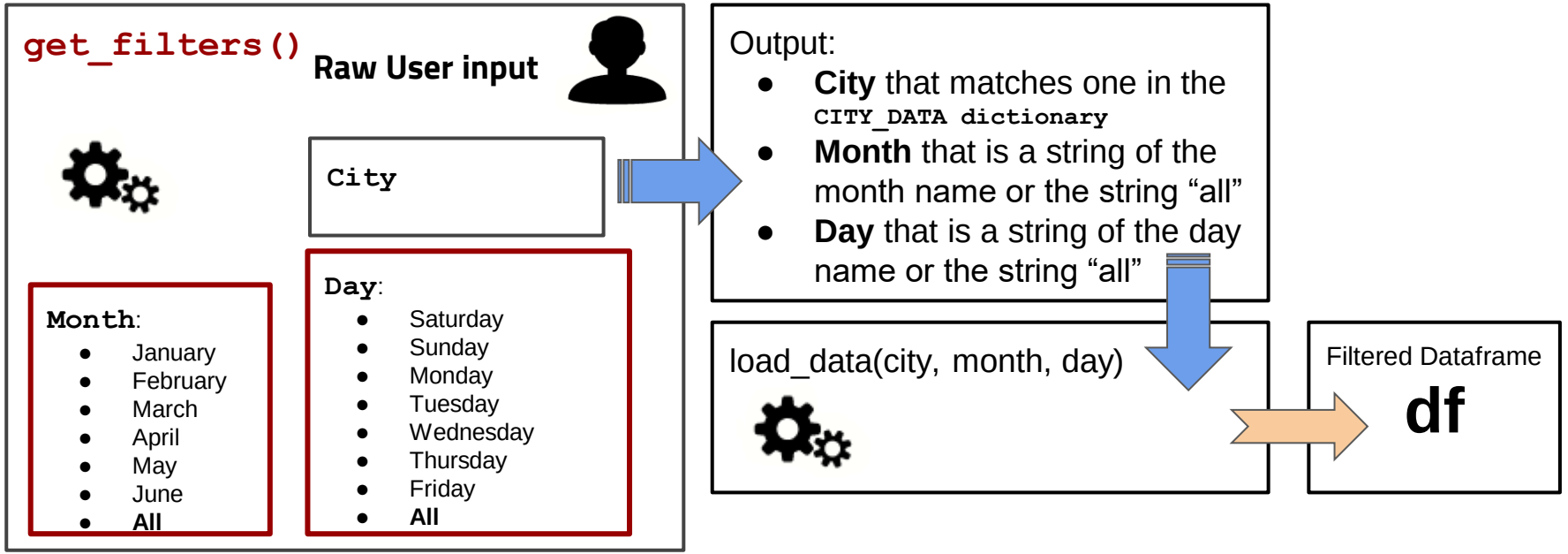
- Load data that match chosen city name
- Convert timestamp column to datetime
- Extract month and day name from timestamp column
- Filter data depending on user choice of both day and month
- This function should return **`df`** which is a dataframe.
- **Don't forget** to call and assign a variable **`df`** to the output

```
load_data(city, month, day)
df = load_data(city, month, day)
```

itida

The First Function

```
CITY_DATA = { 'chicago': 'chicago.csv',
              'new york city': 'new_york_city.csv',
              'washington': 'washington.csv' }
```



Agenda

Project Overview → Code Walkthrough

1

Project Details

- Overview
- The Datasets
- Statistics Computed
- The Files

2

Workspace & Submission

3

Data loading

get_filter and load_data functions

4

Statistics Output

4 functions

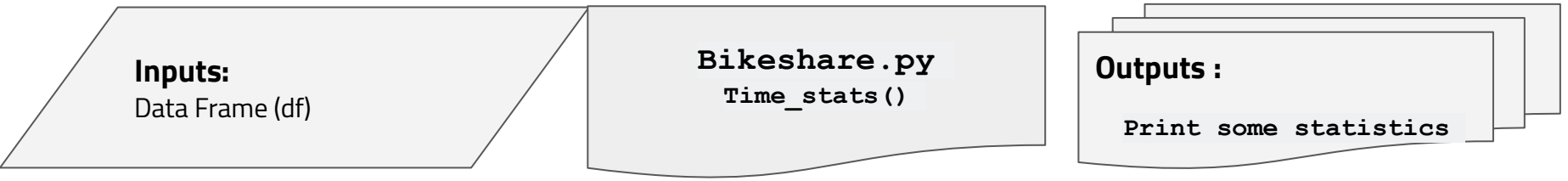
5

Interactive Raw Data display

display_raw_data(city)

itida

Statistics Output (Google is your friend)



`time_stats(df):`

- Extract hour from timestamp column
- Get : (Using **value_counts()** method)
  a. most common month
  b. most common dav of week
  c. most common hour of day

Note

When the user input specifies a particular month **(most common month is meaningless)**
When the user input specifies a particular day **(most common day is meaningless)**

itida

Statistics Output (Google is your friend)

Inputs:
Data Frame (df)

`Bikeshare.py`
`trip_duration_stats()`

Outputs :

`Print statistics`

`trip duration stats(df):`

Get : (Using **value_counts()** method

a. most common start station
b. most common end station
c. most common trip from start to end

Note

most common trip from start to end (i.e., most frequent combination of start station and end station). You can calculate it by adding both start and end stations columns in one columns and get the value counts for this column

Statistics Output (Google is your friend)

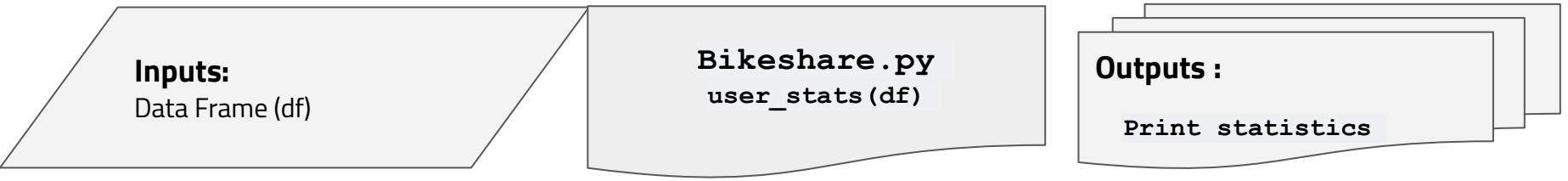
`station_stats(df)`

- Get : (Using **sum and mean** method
  a. total travel time
  b. average travel time

Note

Don't forget to **test** your code

Statistics Output (Google is your friend)



`user_stats(df):`

Get : (Using **value_counts(), min() and max()** methods

a. counts of each user type
b. counts of each gender (**only available for NYC and Chicago**)
c. earliest, most recent, most common year of birth (only available for NYC and Chicago)

Note

Washington doesn't contain gender and birth_year columns so if you calculate count of gender it will generate error. So take care of this trick

itida

Agenda

Project Overview → Code Walkthrough

1

Project Details

- Overview
- The Datasets
- Statistics Computed
- The Files

2

Workspace & Submission

3

Data loading

get_filter and load_data functions

4

Statistics Output

4 functions

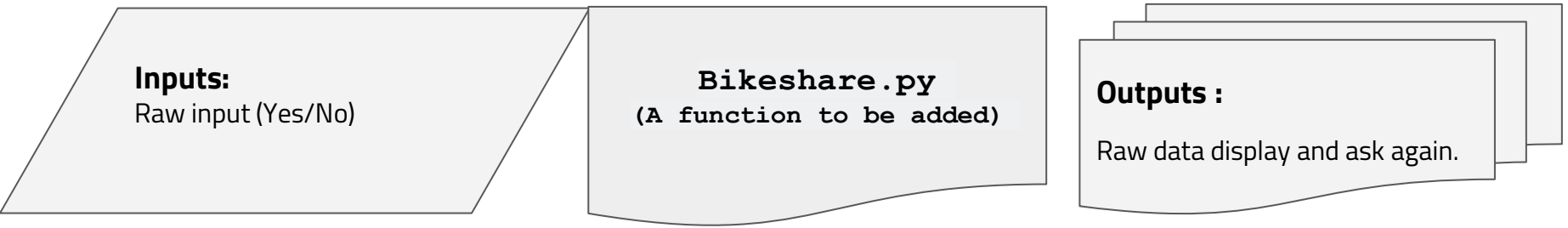
5

Interactive Raw Data display

display_raw_data(city)

itida

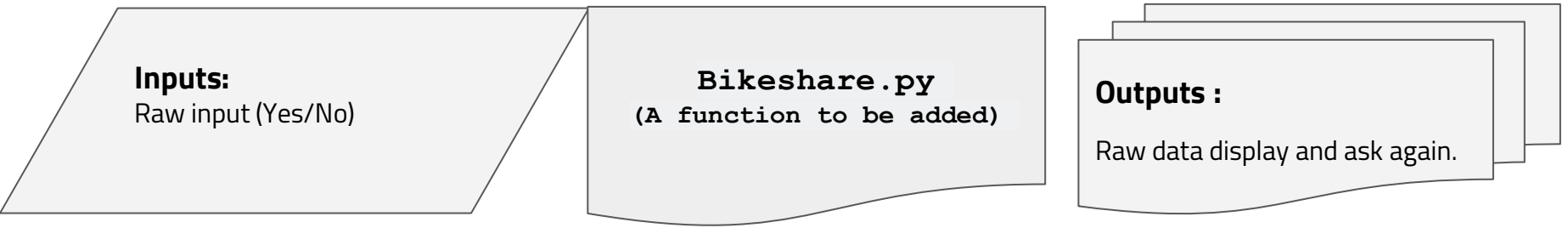
Interactive Raw Data display



The `display_raw_data(city)` function:

Your script also needs to prompt the user whether they would like to see the raw data. If the user answers 'yes,' then the script should print 5 rows of the data at a time, then ask the user if they would like to see 5 more rows of the data. The script should continue prompting and printing the next 5 rows at a time until the user chooses 'no,' they do not want any more raw data to be displayed. Use sample method to print random rows.

Interactive Raw Data display




```
def display_raw_data(df):
    response = get user input if yes or no
    while response == "yes":
        print 5 rows sample of data using sample methos
        response = get user input again
```

Final step

The `main()` function:

```
def main():
    while True:
        city, month, day = get_filters()
        df = load_data(city, month, day)

        time_stats(df, month, day)
        station_stats(df)
        trip_duration_stats(df)
        user_stats(df)
        display_raw_data(city)

        restart = input('\nWould you like to restart? Enter yes or no.\n')
        if restart.lower() != 'yes':
            Break
if __name__ == '__main__':
    main()
```

I know it's a lot of work and mental exercising but enjoy building your pythonista coding Brain!!