bikeshare.py

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

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


def get_filters():
    """
    Asks user to specify a city, month, and day to analyze.

    Returns:
        (str) city - name of the city to analyze
        (str) month - name of the month to filter by, or "all" to apply no month filter
        (str) day - name of the day of week to filter by, or "all" to apply no day filter
    """
    print('Hello! Let\'s explore some US bikeshare data!')
    print('Would you like to see data from Chicago, New York, or Washington?')

    # TO DO: get user input for city (chicago, new york city, washington). HINT: Use a while loop to handle invalid inputs
    #设定四个全局变量，用于其他函数的调用。
    global a_month, a_day, a_city, a_input_1

    #输入城市名字，并且通过while循环处理输入错误的情况。
    city = input()
    while city.lower() not in ('chicago', 'new york', 'washington'):
        print('Make sure that you have input the right name of city.')
        city = input()
    a_city = city

    # TO DO: get user input for month (all, january, february, ... , june)
    print('Would you like to filter the data by month, day, both, or not at all?'
          'Tyepe "none" for no time filter.')

    #输入筛选数据的条件，并且通过while循环处理输入错误的情况。
    input_1 = input()
    while input_1 not in ('month', 'day', 'both', 'none'):
        print('Make sure that you have input the right answer.')
        input_1 = input()

    #判断以month还是day来筛选所看数据。并且通过while循环处理输入错误的情况。
    month = 'all'
    if input_1 in ('month', 'both'):
        print('Which month would you like to see?\n'
              '(january, february, march, april, may or june?)')
        month_list = ['january', 'february', 'march', 'april', 'may', 'june']
        input_2 = input()
        while input_2.lower() not in month_list:
            print('Make sure that you have input the right month.')
            input_2 = input()
        month = input_2
        day = 'all'

    # TO DO: get user input for day of week (all, monday, tuesday, ... sunday)
    if input_1 in ('day', 'both'):
        print('Which day would you like to see?\n'
              '(Monday, Tuesday, Wednesday, Thursday, Friday, Saturday or Sunday?)')
        day_list = ['monday', 'tuesday', 'wednesday', 'thursday', 'friday', 'saturday', 'sunday']
        input_3 = input()
        while input_3.lower() not in day_list:
            print('Make sure that you have input the right day.')
            input_3 = input()
        day = input_3

    if input_1 == 'none':
        day = 'all'

    #给全局变量赋值，用于其他函数的调用。
    a_month = month
    a_day = day
    a_input_1 = input_1

    print('-'*40)
    return city, month, day


def load_data(city, month, day):
    """
    Loads data for the specified city and filters by month and day if applicable.

    Args:
        (str) city - name of the city to analyze
        (str) month - name of the month to filter by, or "all" to apply no month filter
        (str) day - name of the day of week to filter by, or "all" to apply no day filter
```

```
    Returns:
        df - Pandas DataFrame containing city data filtered by month and day
    """
    df = pd.read_csv(CITY_DATA[city.lower()])
    df['Start Time'] = pd.to_datetime(df['Start Time'])
    df['month'] = df['Start Time'].dt.month
    df['day_of_week'] = df['Start Time'].dt.weekday_name
    df['hour'] = df['Start Time'].dt.hour

    #根据筛选数据的条件建立新的数据框。
    if month != 'all':
        months = ['January', 'February', 'March', 'April', 'May', 'June']
        month = months.index(month.title()) + 1
        df = df[df['month'] == month]

    if day != 'all':
        df = df[df['day_of_week'] == day.title()]

    return df


def time_stats(df):
    """Displays statistics on the most frequent times of travel."""

    print('\nCalculating The Most Frequent Times of Travel...\n')
    start_time = time.time()

    #根据不同的筛选条件，输出相应的数据结果。
    # TO DO: display the most common month
    # TO DO: display the most common day of week
    # TO DO: display the most common start hour
    popular_month = df['month'].mode()[0]
    popular_day = df['day_of_week'].mode()[0]
    popular_hour = df['hour'].mode()[0]
    count_month = df['month'].value_counts()[popular_month]
    count_day = df['day_of_week'].value_counts()[popular_day]
    count_hour = df['hour'].value_counts()[popular_hour]

    if a_input_1 in ('none','day'):
        print('Most popular month:{}, Count:{}\n'.format(popular_month, count_month))

    if a_input_1 in ('none','month'):
        print('Most popular day:{}, Count:{}\n'.format(popular_day, count_day))

    print('Most popular hour:{}, Count:{}\n'.format(popular_hour, count_hour))
    print('Filter:{}'.format(a_input_1))

    print("\nThis took %s seconds." % (time.time() - start_time))
    print('-'*40)


def station_stats(df):
    """Displays statistics on the most popular stations and trip."""

    print('\nCalculating The Most Popular Stations and Trip...\n')
    start_time = time.time()

    #输出最热门的起始地点和终止地点，以及最热门的骑车区间。
    # TO DO: display most commonly used start station
    popular_start_station = df['Start Station'].mode()[0]
    count_start_station = df['Start Station'].value_counts()[popular_start_station]
    print('Most popular start station:{}, Count:{}, Filter:{}'.format(popular_start_station, count_start_station, a_input_1))

    # TO DO: display most commonly used end station
    popular_end_station = df['End Station'].mode()[0]
    count_end_station = df['End Station'].value_counts()[popular_end_station]
    print('Most popular end station:{}, Count:{}, Filter:{}'.format(popular_end_station, count_end_station, a_input_1))

    # TO DO: display most frequent combination of start station and end station trip
    df['trip'] = "'" + df['Start Station'] + "'" + ', ' +  "'" + df['End Station']+  "'"
    popular_trip = df['trip'].mode()[0]
    count_trip = df['trip'].value_counts()[popular_trip]
    print('Most popular trip:({}), Count:{}, Filter:{}'.format(popular_trip, count_trip, a_input_1))

    print("\nThis took %s seconds." % (time.time() - start_time))
    print('-'*40)


def trip_duration_stats(df):
    """Displays statistics on the total and average trip duration."""

    print('\nCalculating Trip Duration...\n')
    start_time = time.time()

    #输出所选时间段内总的骑车时间和平均每次骑车时间。
    # TO DO: display total travel time
    total_travel_time = df['Trip Duration'].sum()
    count_travel_time = df['Trip Duration'].count()
```

```
    # TO DO: display mean travel time
    mean_travel_time = df['Trip Duration'].mean()
    print('Total Duration:{}, Count:{}, Avg Duration:{}, Filter:{}'.format(total_travel_time, count_travel_time, mean_travel_time, a_input_1))

    print("\nThis took %s seconds." % (time.time() - start_time))
    print('-'*40)


def user_stats(df):
    """Displays statistics on bikeshare users."""

    print('\nCalculating User Stats...\n')
    start_time1 = time.time()

    #输出骑车用户的类型以及各类型的数量。
    # TO DO: Display counts of user types
    count_subscribers = df['User Type'].value_counts()['Subscriber']
    count_customers = df['User Type'].value_counts()['Customer']
    print('Suscriber:{}, Customers:{}, Filter:{}'.format(count_subscribers, count_customers, a_input_1))
    print("\nThis took %s seconds." % (time.time() - start_time1))

    #根据输入的城市，输出骑车用户数量关于性别的和出生年份的分布。（跳过Washington）
    # TO DO: Display counts of gender
    if a_city.title() != 'Washington':
        print('Would you like to view the information about the gender of users? Type "yes" or "no".')
        if input() == 'yes':
            print('\nCalculating Gender Stats...\n')
            start_time2 = time.time()
            coutn_male = df['Gender'].value_counts()['Male']
            coutn_female = df['Gender'].value_counts()['Female']
            print('Male:{}, Famale:{}, Fileter:{}'.format(coutn_male, coutn_female, a_input_1))
            print("\nThis took %s seconds." % (time.time() - start_time2))

    # TO DO: Display earliest, most recent, and most common year of birth
    if a_city.title() != 'Washington':
        print('Would you like to view the information about the birth information of users? Type "yes" or "no".')
        if input() == 'yes':
            print('\nCalculating Birth Stats...\n')
            start_time3 = time.time()
            year_earliest = df['Birth Year'].min()
            year_most_recent = df['Birth Year'].max()
            year_most_common = df['Birth Year'].mode()[0]
            print('Earliest:{}, Most recent:{}, Most common:{}, Fileter:{}'.format(year_earliest, year_most_recent, year_most_common, a_input_1))
            print("\nThis took %s seconds." % (time.time() - start_time3))

    print('-'*40)


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

        time_stats(df)
        station_stats(df)
        trip_duration_stats(df)
        user_stats(df)

        restart = input('\nWould you like to restart? Enter yes or no.\n')
        if restart.lower() != 'yes':
            break


if __name__ == "__main__":
    main()
```