```
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' }

available_cities = ['chicago', 'washington', 'new york city']
while True:
        city = input('Please enter a city you want to analyse from 
washington, new york city and chicago\n').lower()
        if city in available_cities:
            break
        else:
            print("You You entered an invalid month. Choose a corrrect 
city name")

while True:
    month = input('Please specify which month of the year you want to 
analyze, or type "all" to display all months: ').lower()
    month_list = ['january', 'february', 'march', 'april', 'may', 
'june', 'july', 'august', 'september', 'october', 'november', 
'december']
    if month != 'all' and month not in month_list:
        print('You entered an invalid month. Choose a corrrect month 
name')
    else:
        break

while True:
    day = input('Please specify which day of the week you want to 
analyze, or type "all" to display all days: ').lower()
    day_list = ['monday', 'tuesday', 'wednesday', 'thursday', 
'friday', 'saturday', 'sunday']
    if day != 'all' and day not in day_list:
        print('You entered an invalid day. Choose a corrrect day 
name')
    else:
        break

Please enter a city you want to analyse from washington, new york city 
and chicago
chicago
Please specify which month of the year you want to analyze, or type 
"all" to display all months: may
Please specify which day of the week you want to analyze, or type 
"all" to display all days: sunday

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!')
    # get user input for city (chicago, new york city, washington).
HINT: Use a while loop to handle invalid inputs
    while True:
        city = input('Please enter a city you want to analyse from
washington, new york city and chicago\n').lower()
        if city in available_cities:
            break
        else:
            print("You You entered an invalid month. Choose a corrrect
month name")


    # get user input for month (all, january, february, ... ,
december)
    while True:
        month = input('Please specify which month of the year you want
to analyze, or type "all" to display all months: ').lower()
        month_list = ['january', 'february', 'march', 'april', 'may',
'june', 'july', 'august', 'september', 'october', 'november',
'december']
        if month != 'all' and month not in month_list:
            print('You entered an invalid month. Choose a corrrect
month name')
        else:
            break

    # get user input for day of week (all, monday, tuesday, ...
sunday)
    while True:
        day = input('Please specify which day of the week you want to
analyze, or type "all" to display all days: ').lower()
        day_list = ['monday', 'tuesday', 'wednesday', 'thursday',
'friday', 'saturday', 'sunday']
        if day != 'all' and day not in day_list:
            print('You entered an invalid day. Choose a corrrect day
name')
        else:
            break
    print('-'*40)
    return city, month, day

city, month, day = get_filters()

Hello! Let's explore some US bikeshare data!
Please enter a city you want to analyse from washington, new york city
and chicago
chicago
Please specify which month of the year you want to analyze, or type
"all" to display all months: may
Please specify which day of the week you want to analyze, or type
"all" to display all days: sunday
----------------------------------------

city

'chicago'

month

'may'

day

'sunday'

df = pd.read_csv(CITY_DATA[city])

df['Start Time'] = pd.to_datetime(df['Start Time'])

df['Month'] = df['Start Time'].dt.month

df['day_of_week'] = df['Start Time'].dt.day_name()

if month != 'all':
    month_list = ['january', 'february', 'march', 'april', 'may',
'june', 'july', 'august', 'september', 'october', 'november',
'december' ]
    month = month_list.index(month) + 1
    df = df.loc[df['Month'] == month]
df.head()

    Unnamed: 0          Start Time             End Time  Trip Duration
\
1       955915 2017-05-25 18:19:03  2017-05-25 18:45:53           1610

6       961916 2017-05-26 09:41:44  2017-05-26 09:46:25            281

13     1023296 2017-05-30 15:46:18  2017-05-30 15:52:12            354

15      958716 2017-05-25 22:59:33  2017-05-25 23:07:19            466
```

```
16      718598 2017-05-03 13:20:38  2017-05-03 13:31:13          635


                 Start Station                      End Station   User
Type  \
1           Theater on the Lake    Sheffield Ave & Waveland Ave
Subscriber
6         Ashland Ave & Lake St             Wood St & Hubbard St
Subscriber
13  Larrabee St & Kingsbury St                 Clark St & Elm St
Subscriber
15      Clark St & Armitage Ave  Sheffield Ave & Wrightwood Ave
Subscriber
16     Ada St & Washington Blvd                Daley Center Plaza
Subscriber

    Gender  Birth Year  Month day_of_week
1   Female      1992.0      5    Thursday
6   Female      1983.0      5      Friday
13    Male      1985.0      5     Tuesday
15  Female      1985.0      5    Thursday
16    Male      1967.0      5   Wednesday
```

```python
if day != 'all':
    df = df.loc[df['day_of_week'] == day.title()]
df.head()
```

```
     Unnamed: 0          Start Time             End Time  Trip
Duration  \
48       906322 2017-05-21 10:03:55  2017-05-21 10:18:17
862
59       750957 2017-05-07 11:14:27  2017-05-07 11:20:55
388
156      994753 2017-05-28 17:34:53  2017-05-28 17:51:32
999
159      823105 2017-05-14 07:14:56  2017-05-14 07:19:35
279
241      832285 2017-05-14 17:55:22  2017-05-14 18:14:10
1128

                Start Station                 End Station   User Type
\
48   McClurg Ct & Illinois St     McClurg Ct & Illinois St  Subscriber

59   McClurg Ct & Illinois St        Rush St & Superior St  Subscriber

156    Halsted St & Roscoe St         Broadway & Ridge Ave  Subscriber

159   Greenwood Ave & 47th St  Cottage Grove Ave & 47th St  Subscriber
```

```
241    Green St & Randolph St       Clark St & Schiller St  Subscriber


     Gender  Birth Year  Month day_of_week
48     Male      1990.0      5      Sunday
59   Female      1987.0      5      Sunday
156    Male      1970.0      5      Sunday
159    Male      1966.0      5      Sunday
241  Female      1985.0      5      Sunday
```

```
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])
    df['Start Time'] = pd.to_datetime(df['Start Time'])
    df['Month'] = df['Start Time'].dt.month
    df['Week Day'] = df['Start Time'].dt.day_of_week
    df['day_of_week'] = df['Start Time'].dt.day_name()
    df['hour'] = df['Start Time'].dt.hour
    if month != 'all':
        month_list = ['january', 'february', 'march', 'april', 'may',
'june', 'july', 'august', 'september', 'october', 'november',
'december' ]
        month = month_list.index(month) + 1
        df = df.loc[df['Month'] == month]
    if day != 'all':
        df = df.loc[df['day_of_week'] == day.title()]



    return df
```

```
most_common_month = df['Month'].mode()
print(most_common_month)
```

```
0    5
dtype: int64
```

```
most_common_WeekDay = df['day_of_week'].mode()
print(most_common_WeekDay)
```

```
0    Sunday
dtype: object
```

```python
df['hour'] = df['Start Time'].dt.hour
most_common_hour = df['hour'].mode()
print(most_common_hour)
```

```
0    12
dtype: int64
```

```python
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()

    # display the most common month
    most_common_month = df['Month'].mode()
    print(most_common_month)

    # display the most common day of week
    most_common_WeekDay = df['day_of_week'].mode()
    print(most_common_WeekDay)

    # display the most common start hour
    df['hour'] = df['Start Time'].dt.hour
    most_common_hour = df['hour'].mode()
    print(most_common_hour)

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

```python
most_common_start_station = df['Start Station'].mode()
print(most_common_start_station)
```

```
0    Streeter Dr & Grand Ave
dtype: object
```

```python
most_common_end_station = df['End Station'].mode()
print(most_common_end_)
```

```
0    Streeter Dr & Grand Ave
dtype: object
```

```python
df['Start-End'] = df['Start Station'] + "-" + df['End Station']
common_start_end_station = df['Start-End'].mode()
print(common_start_end_station)
```

```
0    Lake Shore Dr & Monroe St-Streeter Dr & Grand Ave
dtype: object
```

```python
df.head()
```

```
     Unnamed: 0          Start Time             End Time  Trip
Duration  \
48       906322 2017-05-21 10:03:55  2017-05-21 10:18:17
862
59       750957 2017-05-07 11:14:27  2017-05-07 11:20:55
388
156      994753 2017-05-28 17:34:53  2017-05-28 17:51:32
999
159      823105 2017-05-14 07:14:56  2017-05-14 07:19:35
279
241      832285 2017-05-14 17:55:22  2017-05-14 18:14:10
1128

                Start Station                 End Station   User Type
\
48   McClurg Ct & Illinois St     McClurg Ct & Illinois St  Subscriber

59   McClurg Ct & Illinois St        Rush St & Superior St  Subscriber

156    Halsted St & Roscoe St         Broadway & Ridge Ave  Subscriber

159   Greenwood Ave & 47th St  Cottage Grove Ave & 47th St  Subscriber

241    Green St & Randolph St       Clark St & Schiller St  Subscriber


     Gender  Birth Year  Month day_of_week  hour  \
48     Male      1990.0      5      Sunday    10
59   Female      1987.0      5      Sunday    11
156    Male      1970.0      5      Sunday    17
159    Male      1966.0      5      Sunday     7
241  Female      1985.0      5      Sunday    17

                                              Start-End
48   McClurg Ct & Illinois St-McClurg Ct & Illinois St
59      McClurg Ct & Illinois St-Rush St & Superior St
156        Halsted St & Roscoe St-Broadway & Ridge Ave
159  Greenwood Ave & 47th St-Cottage Grove Ave & 47...
241       Green St & Randolph St-Clark St & Schiller St
```

```
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()

    # display most commonly used start station
    most_common_start_station = df['Start Station'].mode()
    print(most_common_start_station)
    # display most commonly used end station
    most_common_end_station = df['End Station'].mode()
    print(most_common_end_)

    # display most frequent combination of start station and end
station trip
    df['Start-End'] = df['Start Station'] + "-" + df['End Station']
    common_start_end_station = df['Start-End'].mode()
    print(common_start_end_station)

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

```
total_travel_time = df['Trip Duration'].sum()
print(total_travel_time)
```

```
10795584
```

```
total_mean_time = df['Trip Duration'].mean()
print(total_mean_time)
```

```
1279.7041251778094
```

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

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

    # display total travel time
    total_travel_time = df['Trip Duration'].sum()
    print(total_travel_time)

    # display mean travel time
    total_travel_time = df['Trip Duration'].mean()
    print(total_travel_time)


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

```
user_type_counts = df['User Type'].value_counts()
print(user_type_counts)
```

```
Subscriber    4551
Customer      3885
Name: User Type, dtype: int64
```

```
if 'Gender' in df.columns:
    gender_counts = df['Gender'].value_counts()
    print(gender_counts)
else:
    print('The data is not available for gender')
```

```
Male      3131
Female    1419
Name: Gender, dtype: int64
```

```
if 'Birth Year' in df.columns:
    earliest_year_of_birth = df['Birth Year'].min()
    print(earliest_year_of_birth)
else:
    print('The data is not available for birth year')
```

```
1901.0
```

```
if 'Birth Year' in df.columns:
    most_recent_year_of_birth = df['Birth Year'].max()
    print(most_recent_year_of_birth)
else:
    print('The data is not available for birth year')
```

```
2000.0
```

```
if 'Birth Year' in df.columns:
    most_common_year_of_birth = df['Birth Year'].mode()
    print(most_common_year_of_birth)
else:
    print('The data is not available for birth year')
```

```
0    1992.0
dtype: float64
```

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

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

    # Display counts of user types
    user_type_counts = df['User Type'].value_counts()
    print(user_type_counts)


    # Display counts of gender
    if 'Gender' in df.columns:
        gender_counts = df['Gender'].value_counts()
        print(gender_counts)
    else:
        print('The data is not available for gender')
    # Display earliest, most recent, and most common year of birth
    if 'Birth Year' in df.columns:
        earliest_year_of_birth = df['Birth Year'].min()
        print(earliest_year_of_birth)
        most_recent_year_of_birth = df['Birth Year'].max()
        print(most_recent_year_of_birth)
        most_common_year_of_birth = df['Birth Year'].mode()
        print(most_common_year_of_birth)
    else:
        print('The data is not available for birth year')

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


view_data = input('\nWould you like to view five rows of individual
trip data? Enter yes or no\n')
start_loc = 0
while view_data =='yes':
    print(df.iloc[start_loc:(start_loc+5)])
    start_loc += 5
    view_data = input("Would you like to continue?").lower()
```

```
Would you like to view five rows of individual trip data? Enter yes or
no
yes
     Unnamed: 0          Start Time             End Time  Trip
Duration  \
48       906322 2017-05-21 10:03:55  2017-05-21 10:18:17
862
59       750957 2017-05-07 11:14:27  2017-05-07 11:20:55
388
156      994753 2017-05-28 17:34:53  2017-05-28 17:51:32
999
159      823105 2017-05-14 07:14:56  2017-05-14 07:19:35
279
241      832285 2017-05-14 17:55:22  2017-05-14 18:14:10
1128

                Start Station                End Station   User Type
\
48   McClurg Ct & Illinois St    McClurg Ct & Illinois St  Subscriber

59   McClurg Ct & Illinois St       Rush St & Superior St  Subscriber

156    Halsted St & Roscoe St        Broadway & Ridge Ave  Subscriber
```

```
159   Greenwood Ave & 47th St  Cottage Grove Ave & 47th St  Subscriber

241    Green St & Randolph St       Clark St & Schiller St  Subscriber


     Gender  Birth Year  Month day_of_week  hour  \
48     Male      1990.0      5      Sunday    10
59   Female      1987.0      5      Sunday    11
156    Male      1970.0      5      Sunday    17
159    Male      1966.0      5      Sunday     7
241  Female      1985.0      5      Sunday    17

                                             Start-End
48   McClurg Ct & Illinois St-McClurg Ct & Illinois St
59      McClurg Ct & Illinois St-Rush St & Superior St
156        Halsted St & Roscoe St-Broadway & Ridge Ave
159  Greenwood Ave & 47th St-Cottage Grove Ave & 47...
241      Green St & Randolph St-Clark St & Schiller St
Would you like to continue?no
```

```python
def display_data(df):
    display_data = input('\nWould you like to view five rows of
individual trip data? Enter yes or no\n')
    start_loc = 0
    while display_data =='yes':
        print(df.iloc[start_loc:(start_loc+5)])
        start_loc += 5
        display_data = input("Would you like to continue?").lower()
```