```
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' }
MONTH_DATA = { 'january', 'february', 'march', 'april', 'may', 'june', 'all'}
WEEK_DATA = { 'sunday', 'monday', 'tuesday', 'wednesday', 'thursday', 'friday', 'saturday', 'all' }

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!')
    # TO DO: get user input for city (chicago, new york city, washington). HINT: Use a while loop to
handle invalid inputs
    city_name = ' '
    while city_name.lower() not in CITY_DATA:
        city_name= input ("\nPlease enter a City name. \nChoose chicago, new york city or
washington.\n ")
        if city_name.lower() in CITY_DATA:
            city=city_name.lower()
        else:
           print ("The city you chose is not yet on our Data base, please enter again")

    # TO DO: get user input for month (all, january, february, ... , june)
    month_name=' '
    while month_name.lower() not in MONTH_DATA:
        month_name = input ("\nPlease enter a specified month: \n")
        if month_name.lower() in MONTH_DATA:
            month=month_name.lower()
        else:
             print("\nsorry this month is not on our database, please enter again.\n")

    # TO DO: get user input for day of week (all, monday, tuesday, ... sunday)
    day_name= ' '
    while day_name.lower() not in WEEK_DATA:
        day_name= input ("\n Please specify a day of the week or all.\n")
        if day_name.lower() in WEEK_DATA:
            day=day_name.lower()
        else:
            print (" please re enter a day.")

    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], index_col=0)
    df['Start Time'] = pd.to_datetime(df['Start Time'])

    # extract month, day of week, hour from Start Time to create new columns
    df['month'] = df['Start Time'].dt.month
    df['day_of_week'] = df['Start Time'].dt.weekday_name
    df['hour'] = df['Start Time'].dt.hour

    if month != 'all':
        months = ['january', 'february', 'march', 'april', 'may', 'june']
        month = months.index(month) + 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()

    # TO DO: display the most common month
    df['month'] = df['Start Time'].dt.month
    common_month = df['month'].mode()[0]
    print("the most common month is: ", common_month)

    # TO DO: display the most common day of week
    df['day_of_week'] = df['Start Time'].dt.weekday_name
    common_weekday = df['day_of_week'].mode()[0]
    print ("the most common day is: ", common_weekday)

    # TO DO: display the most common start hour
    df['hour'] = df['Start Time'].dt.hour
    common_hour = df['hour'].mode()[0]
    print ("the most common hour is: ", common_hour)

    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()

    # TO DO: display most commonly used start station
    common_start_station = df['Start Station'].mode()[0]
    print("most common start station is: ", common_start_station)

    # TO DO: display most commonly used end station
    common_end_station = df['End Station'].mode()[0]
    print("most common end station is: ", common_end_station)
    # TO DO: display most frequent combination of start station and end station trip
    frequent_combination = (df['Start Station'] +" "+ df['End Station']).mode()[0]
    print("the most frequent combined stations are: " , frequent_combination)

    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()

    # TO DO: display total travel time
    total_travel_time= df['Trip Duration'].sum()
    print ("total travel time is: " , total_travel_time)

    # TO DO: display mean travel time
    mean_travel_time= df['Trip Duration'].mean()
    print ("the mean travel time is: " , mean_travel_time)

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


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

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

    # TO DO: Display counts of user types
    user_type= df['User Type'].value_counts()
    print (" number of users is: " , user_type)

    # TO DO: Display counts of gender
    if city == 'chicago.csv' or city == 'new_york_city.csv':
        gender = df['Gender'].value_counts()
        print ("number of user gender is: ", str(gender))

    # TO DO: Display earliest, most recent, and most common year of birth
    if city == 'chicago.csv' or city == 'new_york_city.csv':
        earliest_year= df['Birth Year'].min()
        print ("the earliest year of birth is: " , earliest_year)
        recent_year = df['Birth Year'].max()
        print ("the reacent year of birth is: ", recent_year)
        common_year = df['Birth Year'].mode()[0]
        print ("the most common year of birth is: ", common_year)

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

def display_raw_data(df):

    print(df.head())
    next = 0
    while True:
        view_raw_data = input('\nWould you like to view next five row of raw data? Enter yes or no.
\n')
        if view_raw_data.lower() != 'yes':
            return
        next = next + 5
        print(df.iloc[next:next+5])


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, city)

        while True:
            view_raw_data = input('\nWould you like to view first 5 row of raw data? Enter yes or no.
\n')
            if view_raw_data.lower() != 'yes':
                break
            display_raw_data(df)
            break

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


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