CSE 231 **Spring 2022**

Computer Project #11

This assignment focuses on the design, implementation and testing of Python classes.

It is worth 50 points (5% of course grade) and must be completed no later than **11:59 PM on Monday, April 25, 2022). 3pts extra credit will be given if submitted by Sunday, April 24, 2022**

Assignment Overview

In this assignment, you will practice with creating your own class as a data structure and implementing some functionality around it.

Assignment Background

You are to design your very own calendar assistant that records and updates events for you. Your program will start a new calendar instance and fill it with events. These events could be anything you would want to mark on a calendar with, mostly time and date specific information rather than lengthy event details.

Important: time will use a 24-hour clock, i.e. no AM or PM (to make your programming easier).

Assignment Specifications

1. Data specifications:
    a. Dates are strings in the format: mm/dd/yyyy
       where:
          1 <= mm <= 12
          1 <= dd <= 31   # to make your programming easier even though it makes February 31 a valid date
          0 <= yyyy <= 9999
    b. Time is a string in the format: hh:mm
       where:
          0 <= hh <= 23
          0 <= mm <= 59

    Note: there is a datetime module that can make date and time standardization code shorter, but the overhead of learning a new module will likely take more time. Your choice.

There are two classes and one driver.
We provide one constant:

```
CAL_TYPE = ['meeting','event','appointment','other']
```

CLASS P11_Event

i. `def __init__(self,date=None,time='9:00',duration=60,cal_type='meeting'):`
All parameters are strings except duration (int) and self.
Initialize the public attributes: date, time, duration, cal_type, valid
If a time is not well formed, assign None.
If a date is not well formed, assign None.
If duration is not a positive integer, assign None.
If cal_type is not in CAL_TYPE, assign None.
If any attributeabove is `None`, valid is `False`, otherwise it is `True`.
Look at item 1 in the Data specifications for a valid data format.

ii. `def get_date(self):`
Return the date

iii. `def get_time(self):`
Return the time

iv. `def get_time_range(self):`
Calculate the end time and return a tuple: `(start_time, end_time).` We assume that duration is in minutes, e.g, 60, 75, 120 minutes. Also, the start_time and end_time are both in minutes.

v. `def __str__(self):`
Return a string formatted as: `01/01/0101: start: 9:00; duration: 60`
That is, a space after each semicolon and colon, except within the time. If any of the used attributes are `None` return the string ‘`None`’

vi. `def __repr__(self):`
PROVIDED. Return a string formatted as (no spaces): `01/01/0101;9:00+60`
If any of the used attributes are `None` return the string ‘`None`’

vii. `def __lt__(self,e):`
Return `True` if self’s time is less than e’s time; `False` otherwise.
If either time is `None`, return `False`.
Hint: convert time to an integer number of minutes, then compare and return the result.
This function is necessary for sorting. We will assume that e is of type Event.

viii. `def __eq__(self,e):`
PROVIDED: Returns `True` if all attributes are equal; `False` otherwise.
This method is needed for tests. We will assume that e is of type Event.

CLASS P11_Calendar():

A calendar is a list of events.

b. def __init__(self):
   Initialize the public attribute `event_list` (which is a list of events) to an empty list.

c. def add_event(self,e):
   Append event e to the list of events attribute if the event doesn't conflict with any events already in the list.
   Return False if there is a conflict; return True otherwise.
   Hint: the event get_time_range() method was created to assist here. It returns the range of time for the event. Draw the ranges in a piece of paper and try to find the cases at which 2 different ranges can overlap.

d. def delete_event(self,date,time):
   Delete the event at the specified date and time.
   Hint: find the index of the specified event and use the list `del` command.
   Return False if unsuccessful, e.g. didn't find an event at that date and time; return True otherwise.

e. def day_schedule(self,date):
   Return a sorted list of events on the date in the date parameter by time in ascending order.
   Return an empty list if the date is not well formatted.

f. def __str__(self):
   Return a string that has an event on each line. Have one header line:
   `'Events in Calendar:\n'`
   Hint: str(e) where e is an event calls the event __str__() method
   Note that __str__ method should return a string. The newline character is `'\n'`

g. def __repr__(self):
   PROVIDED. Return a string of events on one line separated by semicolons.
   Hint: repr(e) where e is an event calls the event __repr__() method.

You will develop a Python program that has following functions:

1. `check_time(time,duration)`
   Return `True` if the time and duration are valid; return `False` otherwise.
   Note that a valid schedule time must start at 6AM at the earlier and end at 5PM at the latest (using 24-hour time). Note that `time` should be well formed.
2. event_prompt()
   Prompt for an event; re-prompt until a valid event is entered. Return the event.
   Prompt for date, time, duration, and cal_type.
   Hint: the `check_time` function was created to help here to check if time and duration are valid.
   The event has an attribute valid to check if the event is valid.

`main():`

This function would fire up the program by creating a Calendar and would present the user with five basic options to operate. You program should accept lower and upper case options.
The options would be as follow:

1. To add an event to the calendar:
   prompt for an event, print `"Add Event"`, then simply call calendar's `add_event()` . Print a message based on the status returned.
2. To delete an event:
   print `"Delete Event"` , prompt for date and time, then simply call calendar's `delete_event()` print a message based on the status returned.
3. To list the events for a particular date:
   Print `"List Events",` prompt for a date, Get the calendar events for that date and print them.
   If there are no events, print `"No events to list on "` followed by the date.
   Hint: use calendar's `day_schedule()` to get the list and simply print the events in the list.
4. If the user enters `'Q'` or `'q'`, Exit
5. Show the menu and Re-prompt until the user wants to exit.

Assignment Notes and Hints

1. The coding standard for CSE 231 is posted on the course website:

   http://www.cse.msu.edu/~cse231/General/coding.standard.html

Items 1-9 of the Coding Standard will be enforced for this project.

2. The program will produce reasonable and readable output, with appropriate labels for all values displayed.

3. We providethe following 3 files for you to start with:
   a `proj11.py` program
   a `p11_calendar.py` program
   a `p11_event.py` program

4. If you "hard code" answers, you will receive a grade of zero for the whole project. An example of hard coding is to simply print the approximate value of e rather than calculating it and then printing the calculated average.

Suggested Procedure
it is better to start with coding the event class and then move onto the calendar class. You should code the main in the end. Furthermore, since we are dealing with time and date sensitive items python datetime library is very useful please look for that. The acceptable date format would be year/month/day where the month would be in abbreviated form e.g. feb. Time would be in a 12 hr format with am and pm as input. The duration of an event should be in hours and minutes format.

The last version of your solution is the program which will be graded by your TA.

You should use the ***Mimir*** *system to back up your partial solutions, especially if you are working close to the project deadline. That is the easiest way to ensure that you won't lose significant portions of your work if your machine fails or there are other last-minute problems.*

Assignment Deliverable

The deliverable for this assignment is the following file:

`proj11.py` – the source code for your Python program
`p11_calendar.py –` the calendar class
`p11_event.py –` the event class

Be sure to use the specified file name and to submit it for grading via the **Mimir system** before the project deadline.

Test 1

```
Welcome to your own personal calender.
  Available options:
    (A)dd an event to calender
    (D)elete an event
    (L)ist the events of a particular date
    (Q)uit
Select an option: a
Enter a date (mm/dd/yyyy): 10/21/2020
Enter a start time (hh:mm): 15:00
Enter the duration in minutes (int): 40
Enter event type ['meeting','event','appointment','other']: meeting
Add Event
Event successfully added.
Welcome to your own personal calender.
  Available options:
    (A)dd an event to calender
    (D)elete an event
    (L)ist the events of a particular date
    (Q)uit
Select an option: A
Enter a date (mm/dd/yyyy): 10/21/2020
Enter a start time (hh:mm): 9:00
Enter the duration in minutes (int): 50
Enter event type ['meeting','event','appointment','other']: meeting
Add Event
Event successfully added.
Welcome to your own personal calender.
  Available options:
    (A)dd an event to calender
    (D)elete an event
    (L)ist the events of a particular date
    (Q)uit
Select an option: a
Enter a date (mm/dd/yyyy): 1/2/2000
Enter a start time (hh:mm): 9:00
Enter the duration in minutes (int): 50
Enter event type ['meeting','event','appointment','other']: meeting
Add Event
Event successfully added.
Welcome to your own personal calender.
  Available options:
    (A)dd an event to calender
    (D)elete an event
    (L)ist the events of a particular date
    (Q)uit
Select an option: a
Enter a date (mm/dd/yyyy): 1/2/2000
Enter a start time (hh:mm): 13:00
Enter the duration in minutes (int): 60
Enter event type ['meeting','event','appointment','other']: meeting
Add Event
Event successfully added.
Welcome to your own personal calender.
  Available options:
    (A)dd an event to calender
    (D)elete an event
    (L)ist the events of a particular date
    (Q)uit
Select an option: a
Enter a date (mm/dd/yyyy): 1/2/2000
Enter a start time (hh:mm): 10:00
Enter the duration in minutes (int): 40
Enter event type ['meeting','event','appointment','other']: meeting
Add Event
Event successfully added.
Welcome to your own personal calender.
  Available options:
    (A)dd an event to calender
    (D)elete an event
    (L)ist the events of a particular date
    (Q)uit
Select an option: l
List Events
Enter a date (mm/dd/yyyy): 1/2/2000
1/2/2000: start: 9:00; duration: 50
1/2/2000: start: 10:00; duration: 40
1/2/2000: start: 13:00; duration: 60
Welcome to your own personal calender.
  Available options:
    (A)dd an event to calender
    (D)elete an event
    (L)ist the events of a particular date
    (Q)uit
Select an option: l
List Events
Enter a date (mm/dd/yyyy): 10/21/2020
10/21/2020: start: 9:00; duration: 50
10/21/2020: start: 15:00; duration: 40
Welcome to your own personal calender.
  Available options:
    (A)dd an event to calender
    (D)elete an event
    (L)ist the events of a particular date
    (Q)uit
Select an option: l
List Events
Enter a date (mm/dd/yyyy): 1/21/2020
No events to list on  1/21/2020
Welcome to your own personal calender.
  Available options:
    (A)dd an event to calender
    (D)elete an event
    (L)ist the events of a particular date
    (Q)uit
Select an option: d
Delete Event
Enter a date (mm/dd/yyyy): 10/21/2020
Enter a start time (hh:mm): 15:00
Event successfully deleted.
Welcome to your own personal calender.
  Available options:
    (A)dd an event to calender
    (D)elete an event
    (L)ist the events of a particular date
    (Q)uit
Select an option: l
List Events
Enter a date (mm/dd/yyyy): 10/21/2020
10/21/2020: start: 9:00; duration: 50
Welcome to your own personal calender.
  Available options:
    (A)dd an event to calender
    (D)elete an event
    (L)ist the events of a particular date
    (Q)uit
Select an option: q
```

Test 2

```
Welcome to your own personal calender.
  Available options:
    (A)dd an event to calender
    (D)elete an event
    (L)ist the events of a particular date
    (Q)uit
Select an option: a
Enter a date (mm/dd/yyyy): 10-21-2020
Enter a start time (hh:mm): 15:00
Enter the duration in minutes (int): 40
Enter event type ['meeting','event','appointment','other']: meeting
Invalid event. Please try again.
Enter a date (mm/dd/yyyy): 10/21/2020
Enter a start time (hh:mm): 9:00
Enter the duration in minutes (int): 50
Enter event type ['meeting','event','appointment','other']: meeting
Add Event
Event successfully added.
Welcome to your own personal calender.
  Available options:
    (A)dd an event to calender
    (D)elete an event
    (L)ist the events of a particular date
    (Q)uit
Select an option: a
Enter a date (mm/dd/yyyy): 1/2/2000
Enter a start time (hh:mm): 2:00
Enter the duration in minutes (int): 50
Enter event type ['meeting','event','appointment','other']: meeting
Invalid event. Please try again.
Enter a date (mm/dd/yyyy): 1/2/2000
Enter a start time (hh:mm): 1300
Enter the duration in minutes (int): 60
Enter event type ['meeting','event','appointment','other']: meeting
Invalid event. Please try again.
Enter a date (mm/dd/yyyy): 1/2/2000
Enter a start time (hh:mm): 16:00
Enter the duration in minutes (int): 70
Enter event type ['meeting','event','appointment','other']: meeting
```

```
Invalid event. Please try again.
Enter a date (mm/dd/yyyy): 1/2/2000
Enter a start time (hh:mm): 9:00
Enter the duration in minutes (int): 30
Enter event type ['meeting','event','appointment','other']: meeting
Add Event
Event successfully added.
Welcome to your own personal calender.
  Available options:
    (A)dd an event to calender
    (D)elete an event
    (L)ist the events of a particular date
    (Q)uit
Select an option: q
```

Grading Rubric

```
Computer Project #11                             Scoring Summary
General Requirements:
  ( 3 pts) Coding Standard 1-9
     (descriptive comments, function headers, mnemonic identifiers,
     format, etc...)

Implementation:

( 5 pts)  Test 1

( 3 pts)  Test 2

Main

 ( 3 pts) check_time

 ( 3 pts)  event_prompt

Event class

 ( 6 pts)  __init__

 ( 1 pts)  get_date

 ( 1 pts)  get_time

 ( 3 pts)  get_time_range

 ( 2 pts)  __str__

 ( 4 pts)  __lt__

Calendar class

 ( 1 pts)  __init__

 ( 5 pts)  add_event

 ( 4 pts)  delete_event

 ( 4 pts)  day_schedule

 ( 2 pts)  __str__


Note: hard coding an answer earns zero points for the whole project
-10 points for not using main()
```