Assignment 1 - Coding component

● Graded

Student

Tyler NGUYEN

Total Points

100 / 100 pts

Autograder Score

80.0 / 80.0

Passed Tests

1.1) Simple test case I (10/10)
1.2) Simple test case II (10/10)
1.3) First Edge Case (15/15)
1.4) Second Edge Case (15/15)
1.5) Huge time test (30/30)

Question 2

Clean Code **20** / 20 pts

✔ **– 0 pts** Correct

– 10 pts No document in code.

Autograder Results

1.1) Simple test case I (10/10)

1.2) Simple test case II (10/10)

1.3) First Edge Case (15/15)

1.4) Second Edge Case (15/15)

1.5) Huge time test (30/30)

Submitted Files

scheduler.py

Download

```python
#CPSC 413 Assignment 1 question 5
#Author: Tyler Nguyen
#UCID: 30158563
class Scheduler(object):
    def __init__(self, classlist):
        self.classlist = classlist

    def schedule(self, start_time, end_time):

        sorted_classes_by_end_time = list(self.classlist)
        #classes that will be sorted by their end times (ET)
        for i in range(len(sorted_classes_by_end_time)):
        #iterate over the whole list of classes
            for j in range(0, len(sorted_classes_by_end_time) - i - 1):
                #will go through the list up to the unsorted position, as i increases j will
                #decrease since the end of the list becomes sorted
                if sorted_classes_by_end_time[j][1] > sorted_classes_by_end_time[j + 1][1]:
                    #checks for the end time of the current class is > the end time of the next class
                    temp = sorted_classes_by_end_time[j]
                    #start of swap
                    sorted_classes_by_end_time[j] = sorted_classes_by_end_time[j + 1]
                    sorted_classes_by_end_time[j + 1] = temp
                    #end of swap

        max_amount_of_classes = 0
        #initialize a counter in order to track the number of classes that can be scheduled
        last_class_end_time = start_time - 10
        #makes the end time of the last class to 10 units before starting time which lets the
        #first class to be considered if it starts exactly at start time
        for class_start_time, class_end_time in sorted_classes_by_end_time:
            #iterate through all the classes to determine if they can be attended
            if class_start_time >= last_class_end_time + 10 and class_end_time <= end_time:
                #will check if the class will start at least 10 units after the last class's end time
                #and if it finishes on or before the end time which will ensure classes do not overlap
                #as well as if they have a correct gap between them
                last_class_end_time = class_end_time
                #updates the last class to the end time of the current class which means it is the
                #last attended class
                max_amount_of_classes += 1
                #increment the total amount of classes that can be scheduled
        return max_amount_of_classes
        #return the max amount of classes that can be scheduled
```