ENSF 310: Lab #5

Design, implementation and testing of a simple software system

Due: Tuesday, October 26th

You will have two lab periods to work on this lab. You should be able to finish Part 1 during the first lab session and start on Part 2.

You may work individually or with a partner. If you work with a partner, both of you must make contributions to the lab assignment, and both of you must push changes to your shared repository.

Part 1: Software Design (40 points)

Your team has been hired to develop the back end to a ticket checking and capacity management system for a local ski resort. The ski hill has already purchased equipment to scan tickets and passes and tickets are sold at the ticket counter using existing equipment and an existing user interface. Your system should

- Maintain a list or database of ski pass holders
- Generate and validate daily ski tickets. You do not have to handle the actual payments or multi-day tickets.
- Allow no more than 200 skiers on the mountain on any particular day
- All ski area users have a profile with personal information and an email address and phone number

You do not have to provide an interactive user interface beyond what is needed to test functionality. For example, you can assume that when a ticket is scanned, a function such as `Ticket.scan(ticket)` is called, and to issue a new ticket, a function such as `Ticket.issue()` is called.

1. Requirements

Develop a set of testable requirements based on the given information. Clearly state all assumptions you have to make. You will have to develop and demonstrate a test for each of your requirements based on your implementation.

Requirements:

- ... **your solution here**

2. Software Design

Detail the class and function level design of your solution.

For example:

- class: `Pass`
    - variables:
        - `PassType` :
        - ...
    - functions
        - `validate(date)` : returns `True` if valid, ...
        - ...
- ... **your solution here**

Part 2: Implementation (40 points)

Implement your design in a separate python file. Place any testing / debugging code in this Python notebook -- your testing code should demonstrate functionality of your soution.

In [ ]:

Part 3: Testing (20 points)

Develop a series of tests to test that your solution meets the requirements you set out in part 1. You must test each requirement.

In [ ]: