CSCE 247 - Practice Final
Name:

This is a 150-minute exam. On all essay type questions, you will receive points based on the quality of the answer - not the quantity.

Make an effort to write legibly. Illegible answers will not be graded and awarded 0 points.

There are a total of 10 questions and 100 points available on the test.

Question 1

The following short questions are worth 1 or 2 points each.

2 Points each: (more than one answer may be correct, pick all that apply).

1. Which of the following make sense as classes (rather than objects) in a class diagram?
   a. Homework Assignment
   b. Manton Matthews
   c. Group 5's assignment 5
   d. Person

2. Which of the following coverage criteria **always** requires more test cases than the others?
   a. Statement Coverage
   b. Branch Coverage
   c. Path Coverage
   d. None of the above

1 Point:

3. Requirements-based test cases help the writer clarify the requirements.
   a. True
   b. False

4. An Object is an instantiation of a Class.
   a. True
   b. False

5. The use of global variables generally increases coupling.
   a. True
   b. False

Question 2

Describe the key difference between black-box testing and white-box testing.

Suggested Solution:

Black-box testing involves testing the functionality of a software component without knowing the details of its internal logic - you do not know what it inside its source code, what methods are called, or what objects exist at runtime. White-box testing involves testing the independent logic paths with full implementation knowledge (you can see the code) - however, you do not have full knowledge of the intended functionality (white box tests cannot look for unimplemented code).

Question 3

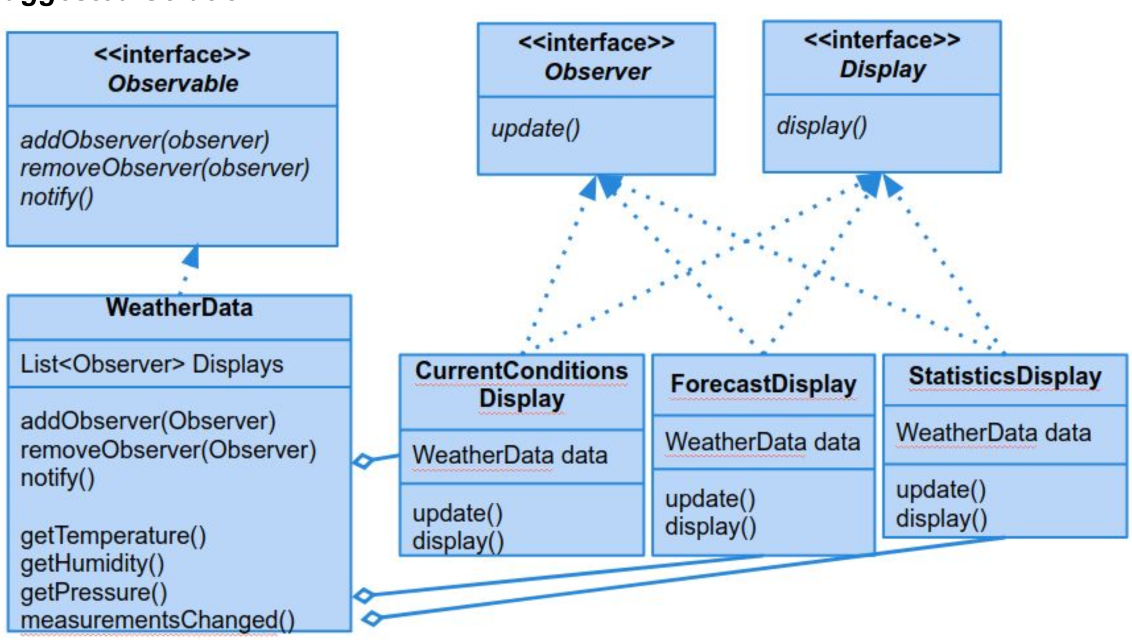
You are designing the software for a weather monitoring station. This station consists of:

- Humidity, Temperature, and Pressure Sensors
- A computer with a monitor

The software for this system must be able to create three displays that can be sent to the monitor: **the current conditions, weather statistics, and a simple forecast.**

Use either the Visitor or Observer pattern to design this software. You can assume that a class called WeatherData exists that stores readings from the sensors (you can add additional functionality to this class).

Suggested Solution:



Question 4

We considered architectural styles, including pipe and filter, layered, and repository.

Suppose that you are to design an automotive system whose subsystems (a-h) are enumerated below. For each style discussed above, choose a subsystem from the set below and describe why this style would be an appropriate structuring mechanism - and why - or describe why this style does not apply to any of the subsystems.

a. On-star communications: manages communications with satellite
b. sensor management: turns noisy sensor data into useful information
c. motion control: operates the motors and provides position and velocity
d. Image processing system to identify highway lanes
e. UX vehicle management involving touch screen
f. Health/status monitoring: checks status of all other subsystems to ensure correct operation
g. Collision avoidance system
h. Dashboard displays

Suggested Solution:
Pipe-and-filter: sensor management; there are several de-noising and sensor fusion transforms that are straightforward to describe using pipe-and-filter architectural styles. Similar arguments could be made for communications with filters for compression / encryption.

Layered: there is a natural layering between supervisory control, navigation control, and motion control. Communications systems themselves are often layered as well (see TCP runs over IP which runs over physical comms)

Repository: Health and status monitoring is often performed using a repository architecture, where vehicle health from many systems is aggregated into one place. Also could be used as a data-plane underlying many of the systems mentioned here.

Question 5

Are path coverage and exhaustive testing the same thing? Motivate your answer.

Suggested Solution

No. Path coverage "only" requires that every path is exercised; it does not require that every input is tested. One can provide path coverage without testing every instance of the inputs that would take you down that path. Thus, problems with divide-by-zero and null-pointer-dereferencing might not be caught.

Question 6

For the following function,

a. Draw the program flow graph for the program.

b. Develop test input that will provide statement coverage. (Input output pairs will be fine.)

c. Develop test input that will provide branch coverage.

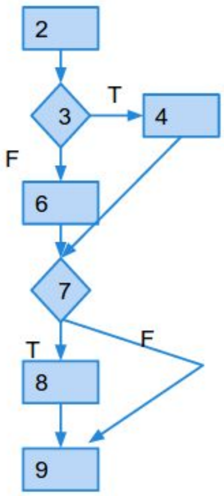
d. Develop test input that will provide full path coverage.

e. Modify the program to introduce a fault so that you can demonstrate that even achieving full path coverage will not guarantee that we will reveal all faults. Please explain how this fault is missed in your example.

```
int findMax(int a, int b, int c)
{
   int temp;
   if (a>b)
      temp=a;
   else
      temp=b;

   if (c>temp)
      temp = c;
   return temp;
}
```

Suggested Solution




```
int findMax(int a, int b, int c) {
 int temp;
 if (a>b)
     temp=a;
 else
     temp=b;
if (c>temp)
     temp = c;
return temp;
}
```

a)
b) (3, 2, 4); (2, 3, 4)
c) (3, 2, 4); (3, 4, 1)
d) (4, 2, 5); (4, 2, 1); (2, 3, 4); (2, 3, 1)
e) If we have (a>b+1) in the first condition as opposed to (a>b), the tests in part D will not reveal this flaw. Only a boundary value test will.

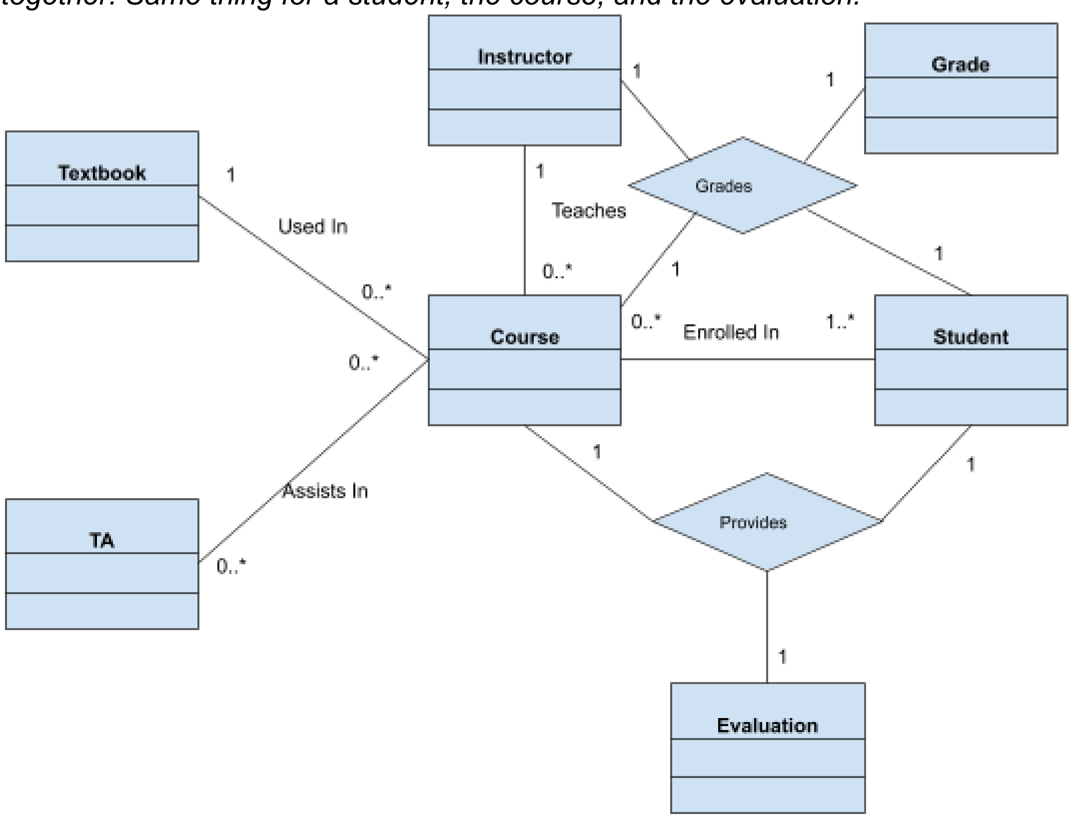
Question 7

Students at the University of South Carolina can be enrolled in more than one class at the time. There is also an option to not be enrolled in any classes (under special circumstances such as completion of all requirements except your PhD dissertation defense). Naturally, we do not offer classes with no students at all.

To equitably allocate teaching effort, there is one instructor assigned to each class (there is no co-teaching). Some instructors might not teach any class (buyout for research for example). Each class uses a textbook (a book that—incidentally—can be used in other classes also).
Depending on class size, there are TAs assisting in the class. A small class gets no TAs, a large class might get several TAs.
When all is done in the class, the instructor assigns the student a grade for the course. In return, each student must fill out a course evaluation form for the course.

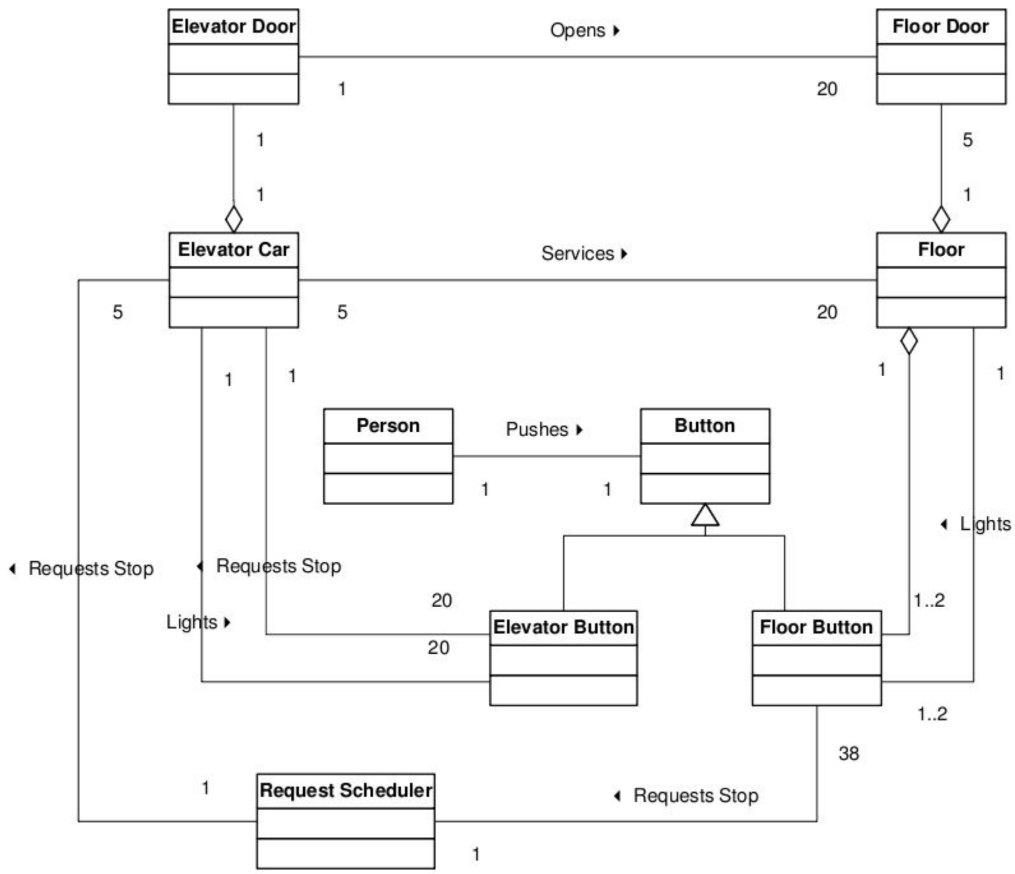
Draw a **class diagram** for the description above. Make sure to show attributes, multiplicities, association names, data attributes, and aggregations/compositions, where appropriate. You may omit operations.

Suggested Solution
This constitutes one solution. Note that a grade, a student, and a course ought to be tied together. Same thing for a student, the course, and the evaluation.



Question 8

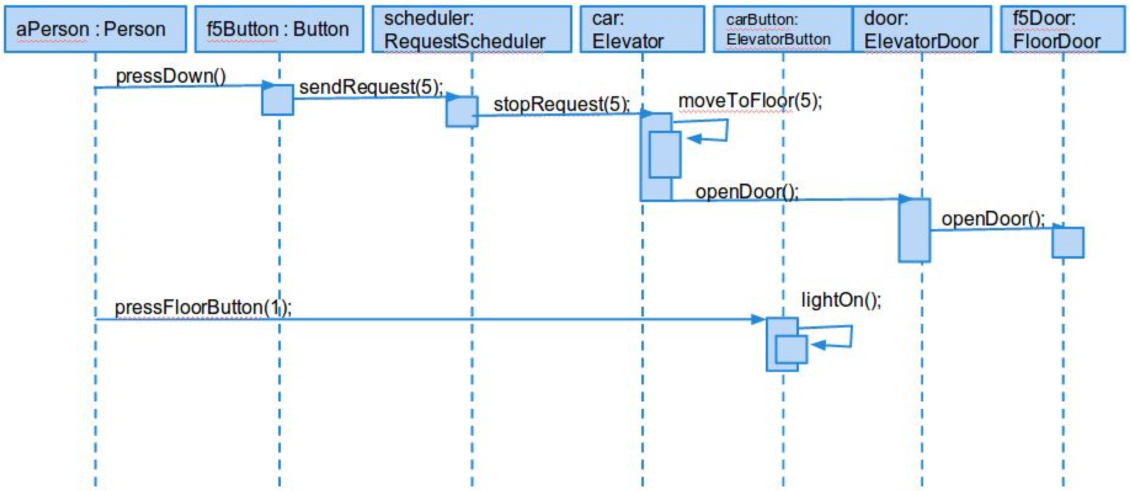
Based on the class diagram below, please draw possible sequence diagrams for the two high-level scenarios below. Note that the operations are not yet defined for the classes. We are drawing these sequence diagrams to help us discover what operations will be needed for each class. Thus, the sequence diagrams will have to contain a little bit more detail than the high-level scenarios we captured when we discussed the use-cases with the customer.

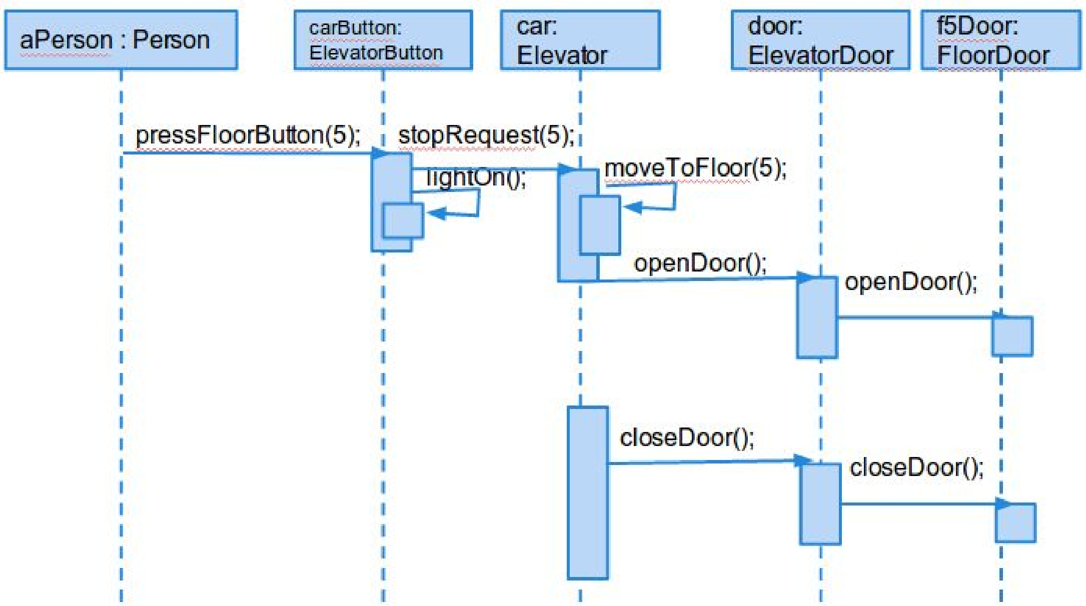


Scenario 1 (Requesting a Ride Down):
A person approaches the elevator on the fifth floor. She wants to go down so she presses the “down” button next to the elevators. She waits until an elevator arrives and the doors open. She enters the elevator and presses the elevator button for the ground floor (floor 1). The light next to the button for the first floor is lit.

Scenario 2 (Getting Off at a Floor):
A person is standing in the elevator with the door closed. The person pushes the elevator button for floor 5(and there are no other requests). The elevator stops at the fifth floor, opens the doors, and the person steps out. The elevator doors close.

Suggested Solution:





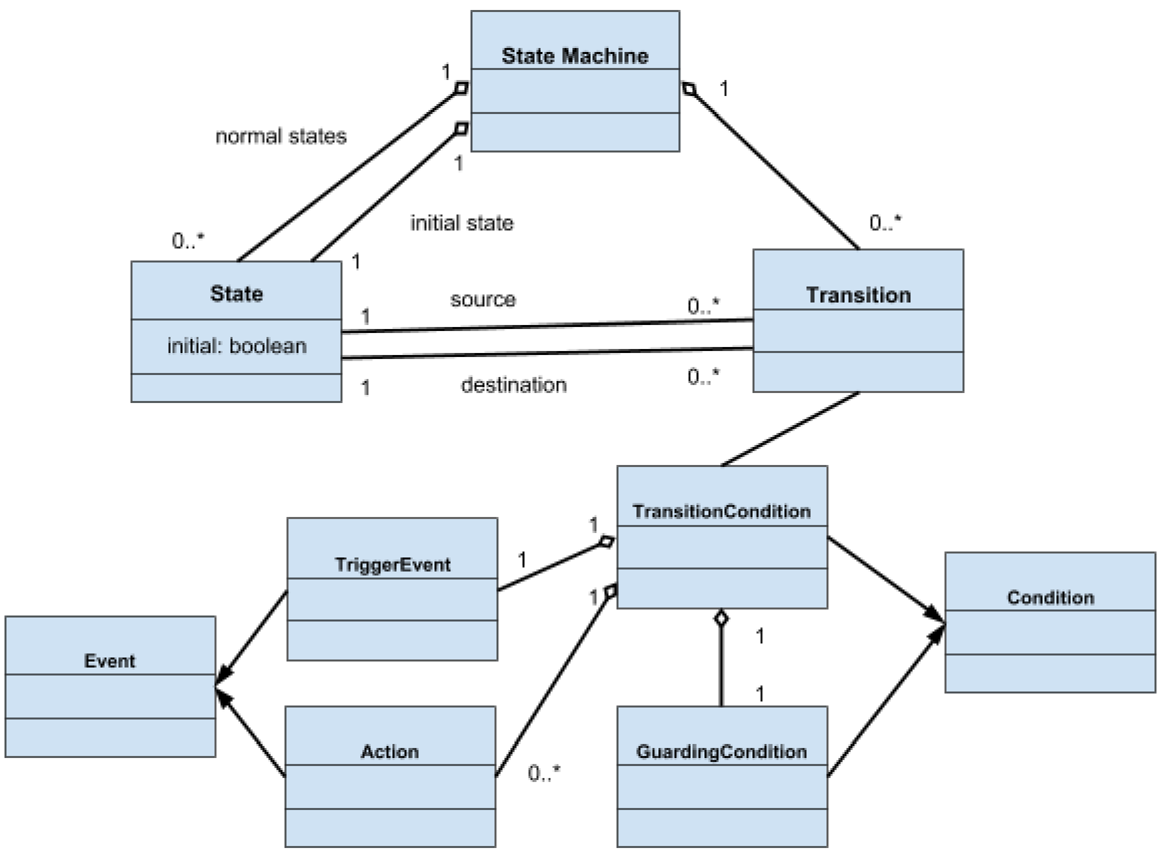
Some variation OK as long as they make it clear that they have understood the fundamentals of a sequence diagram.

Question 9

You are developing software that will simulate and execute finite state machines.
A state machine consists of states and transitions. One state is special and designated to be the initial state (this is where we always start). Besides this, the initial state is just like all other states.
The transitions have transition conditions associated with them. A transition condition consists of a trigger event, a guarding condition, and a possibly empty set of actions (actions are events generated as a result of taking the transition).

Draw a **class diagram** for the description above. Make sure to show attributes, multiplicities, association names, data attributes, and aggregations/compositions, where appropriate. You may omit operations.

Suggested Solution
This constitutes one solution. Make sure you read the problem description carefully. Here I would like to see TriggerEvent and Action both inherit from a parent, as it is mentioned that actions are events. TransitionCondition and GuardingCondition can also inherit from a common parent, as both are conditions.



Question 10

You are building a web store that you feel will unseat Amazon as the king of online shops. Your marketing department has come back with figures stating that - to accomplish your goal - your shop will need an **availability** of at least 98.5%, a **probability of failure on demand** of less than 0.1, and a **rate of fault occurrence** of less than 2 failures per 8-hour work period.

You have recently finished a testing period of one week (seven full 24-hour days). During this time, 972 requests were served to the page. The product failed a total of 64 times. 37 of those resulted in a system crash, while the remaining 27 resulted in incorrect shopping cart totals. When the system crashes, it takes 3 minutes to restart it.

1. What is the rate of fault occurrence?
2. What is the probability of failure on demand?
3. What is the availability?
4. What additional information would you need to calculate the mean time between failures?
5. Is the product ready to ship? If not, why not?

Suggested Solution:

1. *64/168 hours = 0.38 per hour = 3.04 per work period*
2. *64/972 = 0.066*
3. *37*3 = 111 minutes downtime. 111/10080 minutes = 0.011. Avail = 98.9%*
4. *The times that failures occurred. You know how many, but not when.*
5. *No. Avail is good, ROCOF is not.*