A. **Design**
   1. Modular Design
      Have to give 3 reasons why module design is good and write 2 sentences for each reason.

   2. Integration Testing
      Your given an example graph of a hierarchy of modules
      a) Explain bottom-up integration and then give an example using the graph
      b) Explain top-down integration and then give an example using the graph
      c) Why is bottom-up trash?

B. **UML** (I don't remember all of the relationship numbers but that part is easy, like the 1 to 1 shit)
   1. Class diagram
      Users: costumer and administrator
      Each user has a user ID and password attributes
      Each costumer has a list of orders and credit cards
      There are four types of credit cards: Discover, American Express, MasterCard, Visa
      Each order has detailed-info and shipping-info objects
      Each detailed-info has item
      Each item has a description and weight

      From what I remember there was no description for admin and shipping-info
      might have forgotten a description, but if u know how to draw class diagrams this is easy af

   2. Sequence diagram

      There are three objects: Item, Customer, and PreiceListed

      Given this code that is part of the item object: (don't remember exactly but its close)

```
bool isPrime = customer.checkIfPrime();

getItemPrice(){

    if (isPrime){

            costs = priceListed.getPrice();

    else

            shipping = costumer.getShippingPrice();

            price = priceListed.getPrice();

            costs = shipping + price;

    return costs;

}
```

C. Design Patterns (this is where it got more difficult)

1. Dependency injection (I fucked this one up on the test)

Given this code:

```
Public class TestClass {
        Public static void main(String[] args) {
                GameLogic g = new GameLogic();
                VolleyComm v = new VolleyComm();
                V. sendToServer();
        }
}
Class VolleyComm() {
        Void sendToServer() { }
}
Class GameLogic {
        VolleyComm c;
        GameLogic( ) {
                c = v;
        }
        Void playGame() {
                c.sendToServer();
        }
}
```

Have to INJECT communication dependency in GameLogic() (in the main method)

*I think that 1.6 in the *309 Word Bank Answers* from piazza is the answer*

2. MVC
   a) Create MODEL and VIEW method signatures for a tic-tac-toe game
      Assume controller returns correct response to requests
   b) What would you have to change in the Model to be able to use a different view (say a GUI) ….something like that was asked

D. **Testing**
   1. Coverage
      Given this code:
      If(c >= 9)
          doSomething();
      else
          doSomethingElse();
      a) Create a test case that achieves 100% statement coverage
      b) Create a test case that achieves 100% decision/branch coverage

   2. Oracles (kind of fucked this one up too on the test)

      Given: findPerson(Graph g, int num_min)

      Graph g has n nodes, each node representing a person, if two nodes are connected that means they are friends

      Num_min is the minimum number of friends of the person we are looking for

      Example: 10 nodes and num_min = 2, findPerson would return a node that has at least 2 friends or null

      a) Create a signature for an oracle
      b) Give the logic for this oracle

      * I think that 2.2 in the *309 Word Bank Answers* from piazza is the answer*

E. It was just a blank page that said something like
   "Describe the Scrum process model"

      * 4.1 in the *309 Word Bank Answers* from piazza is the answer*