Student Name: _____

(**print** name) _____

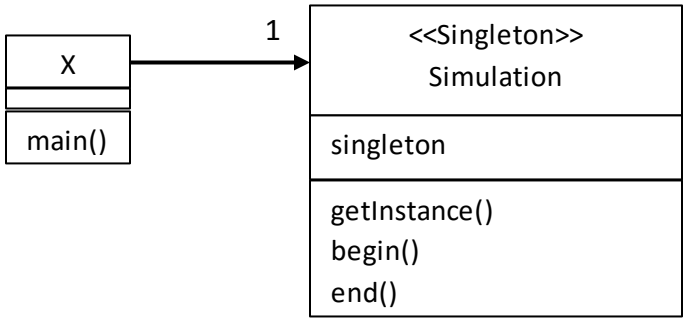
Student No: _____

Place answers on the test paper.
Sequence Diagrams must have **activation boxes** and all messages must be named. Returns do not need to be included.
There are 4 questions and 25 marks.

1. (1 mark) Answer the following:

   a) What does GoF stand for? _____

2. (4 marks) Suppose Simulation is implemented with the Singleton design pattern (lazy instantiation is used), and suppose it is necessary for a method *main()* of a class *X* to send a *begin*() message and then an *end*() message to the simulation object. Use a sequence diagram to show all messages that are sent to accomplish this. Assume Simulation has not been used prior to this. The class diagram:



3. (10 marks) The text's example for the Command design pattern has a Remote Control With Undo with several rows of on and off buttons and an undo button. The undo button can only undo the last command that was executed. There is a garage door receiver with two methods, up() and down(), that cause a garage door to open and close respectively. Suppose the first row of buttons on the remote control are set so that onButtonWasPushed(0) causes the garage door to open and offButtonWasPushed(0) causes the garage door to close. See the class diagram in the Appendix.
   Consider the following four statements and assume there are no other buttons pushed. Draw a sequence diagram to show all messages sent when the following four statements are executed in the Remote Loader:

```
remoteControl.undoWasPushed();
remoteControl.onButtonWasPushed(0);
remoteControl.offButtonWasPushed(0);
remoteControl.undoWasPushed();
```

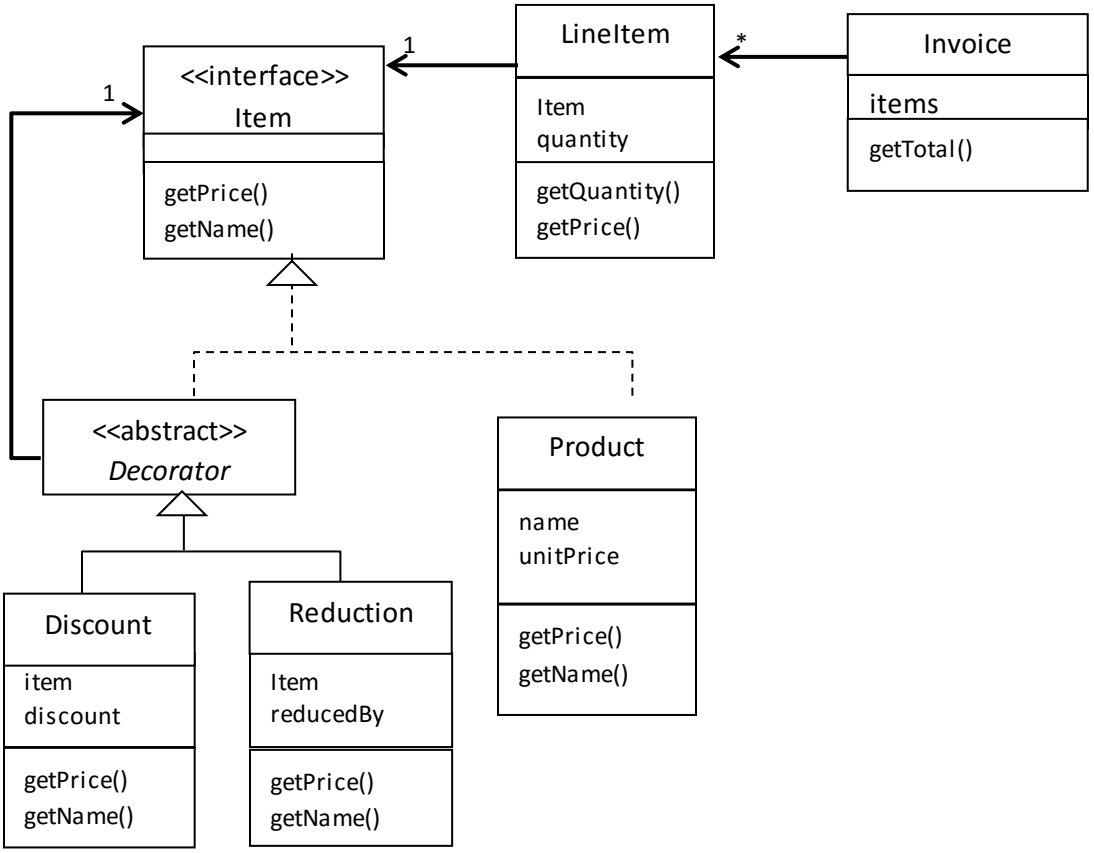
4. (10 marks) A class diagram for managing invoices is in the Appendix. The appendix also contains pseudocode for pertinent methods. An invoice consists of line items where each line item refers to a product that may be decorated in accordance with the Decorator design pattern.
    - A product could have one or more discounts to be applied. A discount reduces the price by a percentage, for example 10%.
    - A product could have one or more reductions to be applied. A reduction reduces the price by a specific amount, for example $1.00

    Consider an invoice with two line items:

    - The first line item is for two copies of a product named *Intro to Java* which has a unit price of $10; this line item has
        - a Reduction where the reducedBy amount is $2, and
        - a Discount where the percent discount is 10%.
    - The second line item is for two copies of a product named *Java 8* which has a unit price of $10; this line item is not reduced in price nor does it have a discount.

    Use a sequence diagram to show all messages sent when the invoice receives the message getTotal() and determines its total.

Class diagram for Invoices (Decorator pattern)




```
getTotal()  in Invoice is:
  total = 0
  for each LineItem:
       total = total + lineItem.getPrice() * lineItem.getQuantity()
   return total

getPrice() in LineItem is:
       return item.getPrice()

getQuantity() in LineItem is:
       return  quantity

getPrice() in Product is:
       return  unitPrice

getPrice() in Discount is:
       return  item.getPrice() * (1 - (discount/100))

getPrice() in Reduction is:
       return  item.getPrice() - reducedBy
```

Class Diagram for Remote Control With Undo plus Garage (Command pattern)