CSSE 373—Formal Methods in Specification and Design
Exam 1, Fall 2007

Name:_____

This exam is open book and open notes. You may also use your laptop if you wish to typecheck your solutions or review past homework, though you should budget your time very carefully. You may **only** use your laptop to access data on your local hard drive or directly accessible from the course ANGEL page.

All of the problems on the exam concern the example specification given on page 5. You may carefully remove that page from the exam, but should turn it in when done.

If you need extra space for an answer, use the back of a page. Please indicate on the front when you do so.

1	10	
2	10	
3	5	
4	15	
5	15	
6	5	
7	40	
bonus	0	
Total	**100**	

1. [10 points]
Define an initialization schema for the vending machine example.

2. [10 points]
Expand ***Insert_Dime***

3. [5 points]
What is the schema type of ***Insert_Quarter*** ?

4. [15 points]
Calculate the precondition of ***Get_Treat***

5. [15 points]
Using schema operators, define a total operation ***T_Get_Treat*** that uses an output variable, ***report!***, to report success or failure when a button is pushed. You may have to define some additional schemas, but ***T_Get_Treat*** should be defined only with schema operators. A new type that may be useful is given.

```
REPORT ::= success | failure
```

6. [5 points]
What is ***cost |> {100}*** ?

7. [40 points]
Vending machines typically keep stacks of coins for dispensing change, separate from the loose jumble of change inserted by customers. Thus, *the coins returned by a machine are different from the ones inserted*. The machine must keep track of the number of coins available in these return-change stacks.

Rewrite the definitions of ***Vending***, ***Get_Treat***, and your initialization schema for the vending machine so that they keep track of the number of dimes and quarters available for dispensing change. Be sure to make ***Get_Treat*** output the correct change. Include a new schema ***Reload_Change*** that puts additional stacked coins into the machine.

Bonus (5 points): Add a ***Change_Return*** operation to return change without making a purchase.

```
spec

[ BUTTON, TREAT, COIN ]

|  brown, orange, white : BUTTON;
|  fudgebar, dreamsicle, sandwich : TREAT;
|  quarter, dime : COIN

|  get : BUTTON +-> TREAT
|--------------------------------
|  get = { brown -> fudgebar, orange -> dreamsicle,
|          white -> sandwich }

|  cost : TREAT +-> N
|--------------------------------
|  cost = { fudgebar -> 100, dreamsicle -> 75, sandwich -> 100 }

--- Vending ------------------------------------------------
|  value : Z
|--------------------------------
|  0 <= value <= 105
------------------------------------------------------------

--- Insert_Quarter -----------------------------------------
|  Delta Vending;
|  coin? : COIN
|--------------------------------
|  coin? = quarter;
|  value' = value + 25
------------------------------------------------------------

--- Insert_Dime --------------------------------------------
|  Delta Vending;
|  coin? : COIN
|--------------------------------
|  coin? = dime;
|  value' = value + 10
------------------------------------------------------------

--- Get_Treat ----------------------------------------------
|  Delta Vending;
|  button? : BUTTON;
|  dispense! : TREAT
|--------------------------------
|  dispense! = get (button?);
|  value' = value - cost (dispense!)
------------------------------------------------------------

end spec
```