Project 3: Creating a Class Hierarchy
(50 Points)

In this project, you will create a class hierarchy that can be used to test the counter from Project 2.

```
// data types

typedef logic[7:0] data_t;
typedef enum {ld, inc, op} op_t;

// Please define a class called ctr_op_t with two data members of type
data_t and op_t and a
// method called convert2string that returns a string with the value of
the object data members
// You can use this call to $sformat to format the string
//
//       $sformat(s,"data %2h   op: %3s", <your data_t name> ,<your op_t
name>);

// Please define a class called ld_op_t which extends ctr_op_t and which
constrains the op to be "ld"

// Please define a class called small_data_t extends ctr_op_t and
// which constrains the data_t variable to be less than 8'd101


module top;
   ctr_op_t ctr_op;
   ld_op_t ld_op;
   small_data_t small_data;

   initial begin : oop_example
      ctr_op = new(); ld_op = new() ; small_data = new();

      $display("*** no constraint ***");
      repeat (5) begin : no_constraint
         assert(ctr_op.randomize());
         $display(ctr_op.convert2string);
      end

      $display("*** inc only ***");
      repeat(5) begin : inc_only
         assert(ctr_op.randomize()
                // Please add a constraint using a "with" statement
                // that limts the op_t data member to be "inc" only
                );
         $display(ctr_op.convert2string);
      end

      $display("*** load only ***");
      repeat (5) begin : load_only
         assert(ld_op.randomize());
         $display(ld_op.convert2string);
      end

      $display("*** small only ***");
      repeat (5) begin : small_only
```

```
         assert(small_data.randomize());
         $display(small_data.convert2string);
      end
   end // initial begin
endmodule // top
```