SYSTEM VERILOG

CONSTRAINTS – Part 2

Write a constraint to generate even number in odd locations and odd number in even locations

```
class constraint_11;
        rand bit [3:0] da[];
        constraint c1{da.size == 10;}
        constraint c2{foreach(da[i])
                                if(i%2 == 0)
                                        da[i]%2 == 1;
                                else if(i%2 == 1)
                                        da[i]%2 == 0; }

endclass

constraint_11 c1;

module test();
        initial
                begin
                        repeat(5)
                                begin
                                        c1=new;
                                        assert(c1.randomize());
                                        $display("da: %p",c1.da);
                                end
                end
endmodule
```

Write a constraint to generate a pattern 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89 (Fibonacci Series)

```
class constraint_12;
    rand int da[];
    constraint c1{da.size == 12;}
    constraint c2{foreach(da[i])
                    if(i==0)
                        da[i] == 0;
                    else if(i==1)
                        da[i] == 1;
                    else
                        da[i] == da[i-2] + da[i-1]; }
endclass

constraint_12 c1;

module test();
    initial
        begin
            c1=new;
            assert(c1.randomize());
            $display("da: %p",c1.da);
        end
endmodule
```

Write a constraint to generate a pattern 123123123123

```
class constraint_13;
      rand int da[];
      constraint c1{da.size == 12;}
      constraint c2{foreach(da[i])
                              da[i] == ((i%3) + 1); }
endclass

constraint_13 c1;

module test();
      initial
            begin
                  c1=new();
                  assert(c1.randomize());
                  $display("da: %p",c1.da);
            end
endmodule
```

Write a constraint to generate a pattern 100100100100

```
class constraint_14;
      rand int da[];
      constraint c1{da.size == 12;}
      constraint c2{foreach(da[i])
                              if(i % 3 == 0)
                                    da[i] == 1;
                              else
                                    da[i] == 0; }
endclass

constraint_14 c1;

module test();
      initial
            begin
                  c1=new;
                  assert(c1.randomize());
                  $display("da: %p",c1.da);
            end
endmodule
```

Write a constraint to generate a pattern 001002003004005

```
class constraint_15;
      rand int da[];
      constraint c1{da.size == 15;}
      constraint c2{foreach(da[i])
                        if(i % 3 == 2)
                              da[i] == i / 3 + 1;
                        else
                              da[i] == 0; }
endclass

constraint_15 c1;

module test();
      initial
            begin
                  c1=new();
                  assert(c1.randomize());
                  $display("da: %p",c1.da);
            end
endmodule
```

Write a constraint to generate a pattern 1221122112211…

```
class constraint_16;
      rand int da[];
      constraint c1{da.size == 15;}
      constraint c2{foreach(da[i])
                              if(i%4==0 || i%4==3)
                                    da[i] == 1;
                              else
                                    da[i] == 2; }
endclass

constraint_16 c1;

module test();
      initial
            begin
                  c1=new;
                  assert(c1.randomize());
                  $display("da: %p",c1.da);
            end
endmodule
```

Generate an array of 10 elements where even indices are 0, and odd indices are random values between 1 and 9.

```
class constraint_17;
      rand int da[];
      constraint c1{da.size == 10;}
      constraint c2{foreach(da[i])
                        if(i%2 == 0)
                              da[i] == 0;
                        else
                              da[i] inside {[1:9]}; }
endclass

constraint_17 c1;

module test();
      initial
            begin
                  repeat(5)
                        begin
                              c1=new;
                              assert(c1.randomize());
                              $display("da: %p",c1.da);
                        end
            end
endmodule
```

Write a constraint for payload of size 10 and each value in the payload should be greater than the previous value by 5.

```
class constraint_18;
    rand int da[];
    constraint c1{da.size == 10;}
    constraint c2{foreach(da[i])
                    if(i == 0)
                        da[i] == 0;
                    else
                        da[i] == da[i-1] + 5; }
endclass

constraint_18 c1;

module test();
    initial
        begin
            c1=new;
            assert(c1.randomize());
            $display("da: %p",c1.da);
        end
endmodule
```

Write a constraint to generate a pattern 5, -10, 15, -20, 25, -30, 35, -40, 45, -50.

```
class constraint_19;
      rand int da[];
      constraint c1{da.size == 10;}
      constraint c2{foreach(da[i])
                        if(i%2 == 0)
                              da[i] == 5 * (i+1);
                        else
                              da[i] == -5 * (i+1);
endclass

constraint_19 c1;

module test();
      initial
            begin
                  c1=new;
                  assert(c1.randomize());
                  $display("da: %p",c1.da);
            end
endmodule
```

Write a constraint to generate even numbers in the range 15 to 60 using fixed array, dynamic array and queue.

```
class constraint_20;
	rand int fixed[15];
	rand int da[];
	rand int queue[$];
	constraint c1{da.size == 15;
				queue.size == 15;}
	constraint c2{foreach(fixed[i])
				fixed[i] inside {[15:60]} &&
fixed[i]%2==0;}
	constraint c3{foreach(da[i])
				da[i] inside {[15:60]} && da[i]%2==0;}
	constraint c4{foreach(queue[i])
				queue[i] inside {[15:60]} &&
queue[i]%2==0;}
endclass

constraint_20 c1;

module test();
	initial
		begin
			c1=new;
			assert(c1.randomize());
			$display("fixed: %p, da: %p, queue:
%p",c1.fixed,c1.da,c1.queue);
		end
endmodule
```