READ ME FIRST
- This is a 90 minute, open notes, open books, open computer exam.
- Your exam will be graded based on what is submitted with this exam, not what you have on the computer.
- Don’t spend too much time on any one problem.
- Amount of time spent on a problem is not necessarily proportional to the points.
- Scan through the entire test and do the easy problems first.
- If something is not clear, ASK.
- BE NEAT. We cannot give you points for something that we can’t read.
- Write down your assumptions.

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>20 points</td>
<td>Indexed Face Sets</td>
</tr>
<tr>
<td>2</td>
<td>20 points</td>
<td>What happens?</td>
</tr>
<tr>
<td>3</td>
<td>20 points</td>
<td>On/Off</td>
</tr>
<tr>
<td>4</td>
<td>20 points</td>
<td>Proto</td>
</tr>
<tr>
<td>5</td>
<td>20 points</td>
<td>Loop and Twist</td>
</tr>
<tr>
<td></td>
<td>100 points</td>
<td>GRAND TOTAL</td>
</tr>
</tbody>
</table>
1. **Indexed Face Sets (20 points)**

Create a world that contains a tetrahedral (4-faced) shape using the coordinates given below (15 points). Add a different color to each face (5 points). Hint: one of your homeworks asked you to do a hexahedral (6-faced) shape. Watch the orientation of the faces, and watch your VRML syntax!

<table>
<thead>
<tr>
<th>Point</th>
<th>Coordinate</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>(0,2,0)</td>
</tr>
<tr>
<td>1</td>
<td>(-1,0,1)</td>
</tr>
<tr>
<td>2</td>
<td>(0,0,-1)</td>
</tr>
<tr>
<td>3</td>
<td>(1,0,1)</td>
</tr>
</tbody>
</table>
2. **What happens? (20 points)**

Given the VRML world below. First draw the world as it would appear when first displayed by the browser (5 points). Then, describe *concisely* what happens when the user clicks on the visible object (15 points).

```vrml
#VRML V2.0 utf8
DEF RedSphere Transform{
  children Shape{
    geometry Sphere{
      radius 1
    }
    appearance Appearance{
      material Material{
        diffuseColor 1 0 0
      }
    }
  }
}

DEF BlueBox Transform{
  children Shape{
    geometry Box{
      size 1 1 1
    }
    appearance Appearance{
      material Material{
        diffuseColor 0 0 1
      }
    }
  }
}

DEF ClickMe TouchSensor{
}

DEF Timer TimeSensor{
}

DEF DoSomething PositionInterpolator{
  key [0, 0.2, 0.4, 0.6, 0.8, 1]
  keyValue [1.0 1.0 1.0,
              1.4 1.4 1.4,
              1.8 1.8 1.8,
              2.2 2.2 2.2,
              2.6 2.6 2.6,
              3.0 3.0 3.0]
}
```

ROUTE ClickMe.touchTime TO Timer.set_startTime
ROUTE Timer.fraction_changed TO DoSomething.set_fraction
ROUTE DoSomething.value_changed TO BlueBox.set_scale
3. **On/Off (20 points)**

Complete the following world such that the light bulb will turn on when the viewer comes within 4 units from it, and turn off when the viewer is more than 4 units from it.

```vrml
#VRML V2.0 utf8

Transform{
    children Shape{
        geometry Sphere{
            radius 1
        }
        appearance Appearance{
            material DEF LightBulb Material{
                diffuseColor 0.2 0.2 0.2
            }
        }
    }
}

DEF TurnOn ColorInterpolator{
    key [0, 0.5, 1.0]
    keyValue [0.2 0.2 0.2, 0.6 0.2 0.2, 1.0 0.2 0.2]
}

DEF TurnOff ColorInterpolator{
    key [0, 0.5, 1.0]
    keyValue [1.0 0.2 0.2, 0.6 0.2 0.2, 0.2 0.2 0.2]
}
```
4. **Proto (20 points)**
   Complete the PROTO Arrow node below such that:
   (a) It will have 3 interface fields: arrowPosition, arrowDirection, arrowColor
   (b) (10 points). Its default length should be 1. Default orientation should be pointing up along positive Y-axis. The base of the arrow should be at the origin. The default color is set gray. The length of the arrow is measured from the base to the tip of the arrowhead. Set the arrow body radius to 0.05, arrowhead height to 0.2, and the arrowhead radius to 0.1.
   (c) (10 points). User should be able to create instances of an Arrow node that can be positioned, oriented, and set to some desired color.

```
#VRML V2.0 utf8

PROTO Arrow [
    field SFVec3f arrowPosition 0 0 0
    field SFRotation arrowDirection 0 0 1 0
    field SFColor arrowColor 0.5 0.5 0.5
]
{
    DEF MAIN Transform{
        children[
            DEF ArrowBody Transform{

                }, ### End of ArrowBody
```


DEF ArrowHead Transform {

} ### End of ArrowHead
]
### End of children

### Rest of MAIN Transform node

} ### End of MAIN Transform node

Arrow{ ### SAMPLE: green diagonal arrow from 2,2
arrowColor 0 1 0
arrowPosition 2 2 0
arrowDirection 0 0 1 0.707
}"
5. Loop and Twist (20 points)
The following VRML world will make the jet loop through the ring (from homework 5). Modify the world so that the jet twists along its own axis as it is looping through the ring. Allow one complete twist per loop. That is, the pilot will start to roll until he's upside down, then continue rolling until he's upright again, all the while he's looping around the ring.

```vrml
#VRML V2.0 utf8
Transform {  
  children [  
    ### RING ###  
    Inline {  
      url "ring.wrl"
    },  
    ### JET ###  
    Transform {  
      children [  
        # ====== timer ======  
        DEF Timer TimeSensor {  
          loop TRUE  
          cycleInterval 4
        },  
        # ====== circular path =====  
        DEF Path PositionInterpolator {  
          key [ 0.0, 0.2, 0.4, 0.6, 0.8, 1.0 ]  
          keyValue [ 0 0 0, 1.382 0 1.902, 3.618 0 1.176, 3.618 0 -1.176, 1.382 0 -1.902, 0 0 0 ]
        },  
        # ====== make object turn along path =====  
        DEF Turn OrientationInterpolator {  
          key [ 0.0, 0.2, 0.4, 0.6, 0.8, 1.0 ]  
          keyValue [ 0 1 0 0, 0 1 0 1.256, 0 1 0 2.512, 0 1 0 3.768, 0 1 0 5.024, 0 1 0 0 ]
        }
      ]
    ]
  ]
}
```
# ======= make object turn on its axis ====

```c
DEF TurnObject Transform {
    children [
        Transform {
            translation -2 0 -2
            rotation 0 1 0 1.57
            center 2.0 0.0 0.0
            children Transform {
                scale 0.1 0.1 0.1
                children Inline {
                    url "f16.wrl"
                }
            }
        }
    ]
}
```

### ================ routes ================

ROUTE Timer.fraction_changed TO Path.set_fraction
ROUTE Timer.fraction_changed TO Turn.set_fraction

ROUTE Path.value_changed TO TurnObject.set_translation
ROUTE Turn.value_changed TO TurnObject.set_rotation