Eng591 - Virtual Reality Programming

First Mid-Term Exam

Fall 1999

You are allowed to use your coursepack only, (including any notes written therein.)

No other assistance or references are permitted. You are not permitted to use computers.

I have neither given nor received aide on this examination,
nor have I concealed any violation of the honor code.

Name (Printed): *Answer Key*

Signature:

Question 0: (No points, just for fun): If you are in a room with a chimpanzee, an orangutan, and an ape, which is the most intelligent primate in the room? I AM !

1. Short Answer / Multiple Choice / True or False (5 points each)

1A. It is desired to place an object 10 units above the origin, with X,Z coordinates of (20, 30), and rotated 60 degrees about the axis (1, -2, 2). What are the translation and orientation values necessary to achieve this, (assuming the object is originally at the origin)?

```
WTp3 translation = { 20.0f, -10.0f, 30.0f };

WTq orientation = { 0.16667f, -0.3333f, 0.3333f, 0.866f };
```

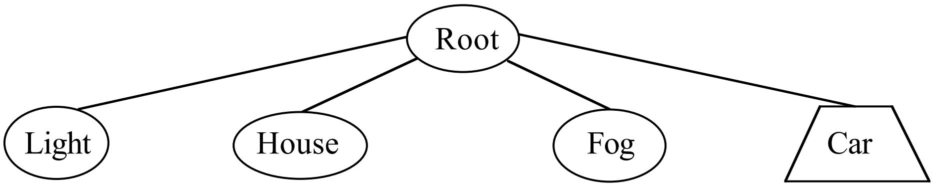
1B. True or False: Assuming the correct values have been entered above, (and a valid file exists), the following code will place a butterfly in the correct position:

```
WTnode *node;
node = WTnode_load( Root, "BUTTERFLY.DXF", 1.0 );
WTnode_settranslation( node, translation );
WTnode_setorientation( node, orientation );
```

a. True

[✓ b. False] (Need to use WTmovnode_load, not WTnode_load)

1C. In the following scene graph, which object(s) will be shrouded in fog?



a. The house.

[✓ b. The car.]

c. Both.

d. Neither.

e. None of the above.

1D. For a switch node, which combination(s) of its children can be visible simultaneously? Circle all valid possibilities.

[✓ a. None visible.]

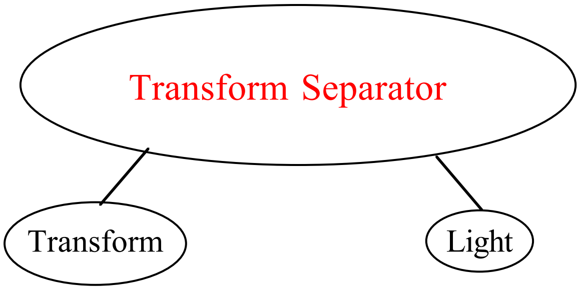
[✓ b. All visible.]

[✓ c. Any single child.]

d. Any combination of children.

e. None of the above.

1E. What is the correct type of node to use as the parent of a movable light? I.e. what type of node should be used in the following scene graph segment?



2. Scene Graph Construction - (25 points)

On a separate page, draw a scene graph for a simulation of the solar system, having the following components as discussed below:

1. One Sun, stationary at the origin.

2. Nine planets, named Mercury, Venus, Earth, Mars, Jupiter, Saturn, Uranus, Neptune, and Pluto. These planets will move in orbit about the sun.

3. One moon orbiting Earth, named Moon.

4. Four moons orbiting Jupiter, named Io, Europa, Ganymede, and Callisto. (Jupiter actually has 16 moons. The solar system has over 60 total. The four listed here can be seen in common consumer-grade telescopes.)

5. One ring surrounding Saturn. (The ring rotates at a different velocity from the planet.)

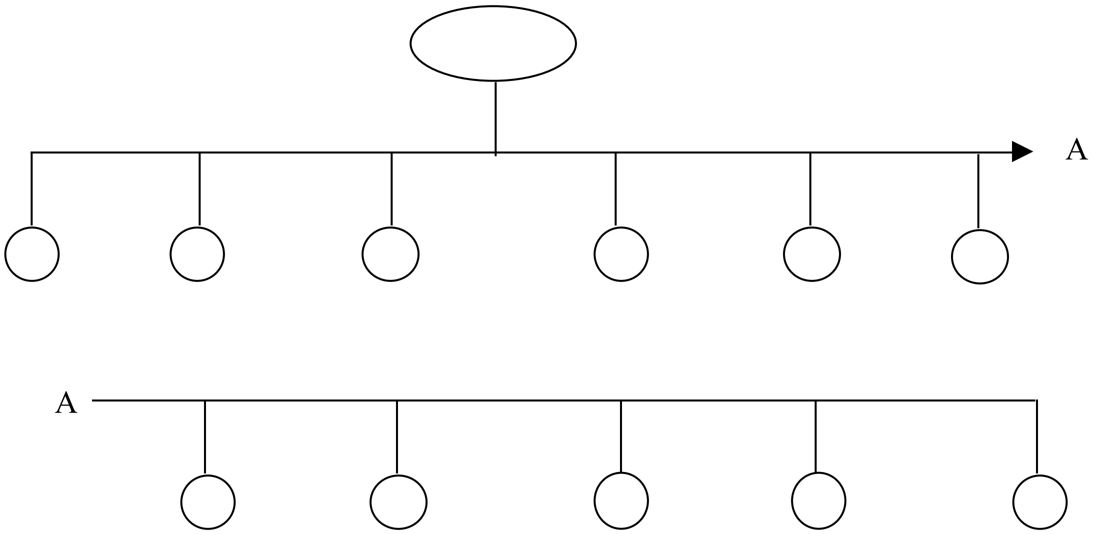
6. N stars, all having identical geometries, but different locations.

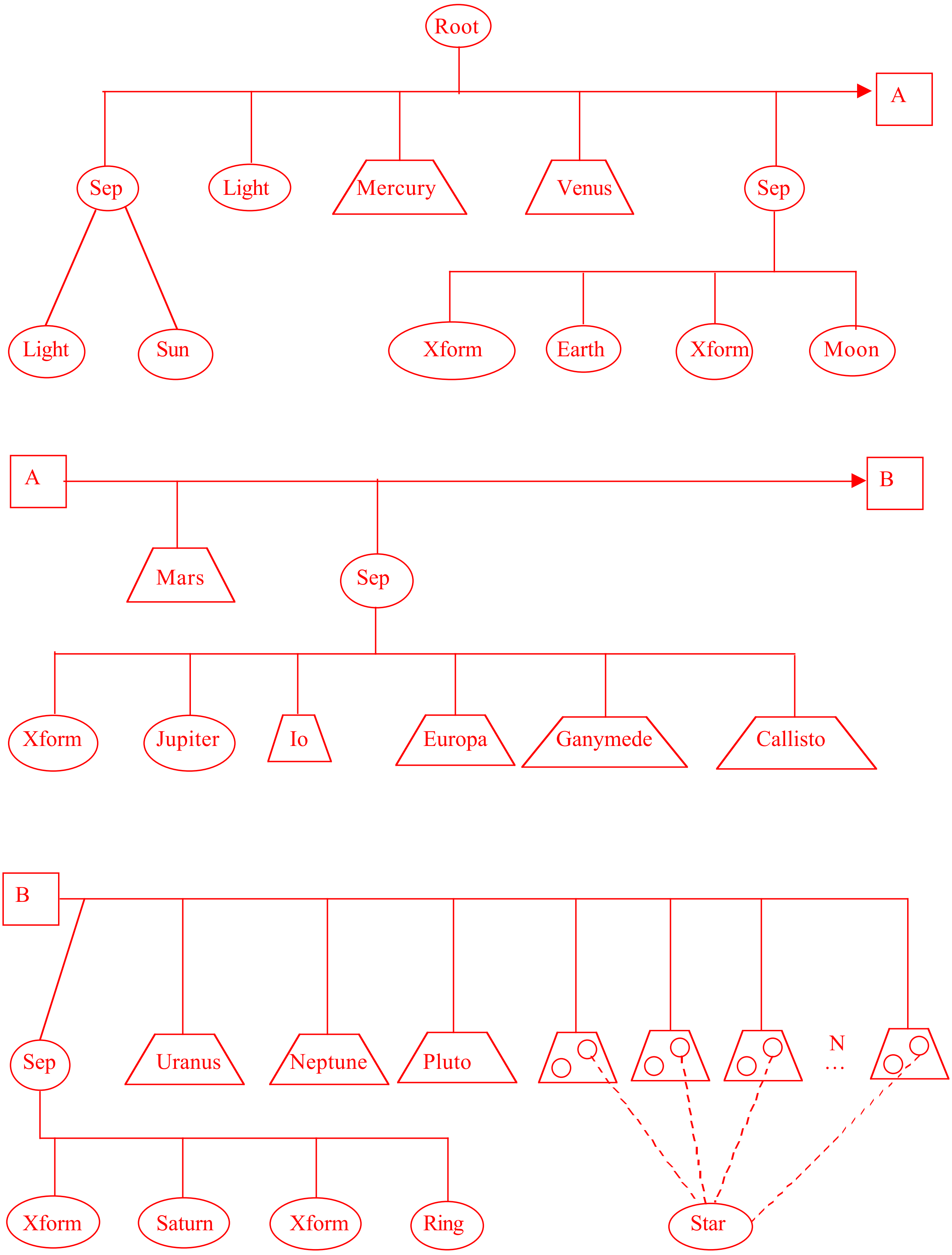
7. One point light source, located at the origin. This light will shine on everything except the sun (which surrounds it.)

8. Another light node that will illuminate the sun only. You do not need to concern yourself with what kind of light this is - only that it only shines on the sun.

9. Simple movable nodes can be indicated by ⏢ . More complicated structures should be shown as separate nodes.

10. Your scene graph may not fit easily on a page without using continuations. You can continue your scene graph thusly:




Root
A
Sep
Light
Mercury
Venus
Sep
Light
Sun
Xform
Earth
Xform
Moon
A
B
Mars
Sep
Xform
Jupiter
Io
Europa
Ganymede
Callisto
B
Sep
Uranus
Neptune
Pluto
N
. . .
Xform
Saturn
Xform
Ring
Star

3. Original Code Development - (50 points)

Given the scene graph and diagrams on the following pages, write the code for a function to load a simple merry go round into a WorldToolKit simulation. Additional information:

1. The WTnode *root is passed into the function as an argument.

2. In the scene graph, "S" indicates a separator node, "T" a transform node, "LOD" a level-of-detail node, "G" the platform geometry node, and "H" a horse geometry node.

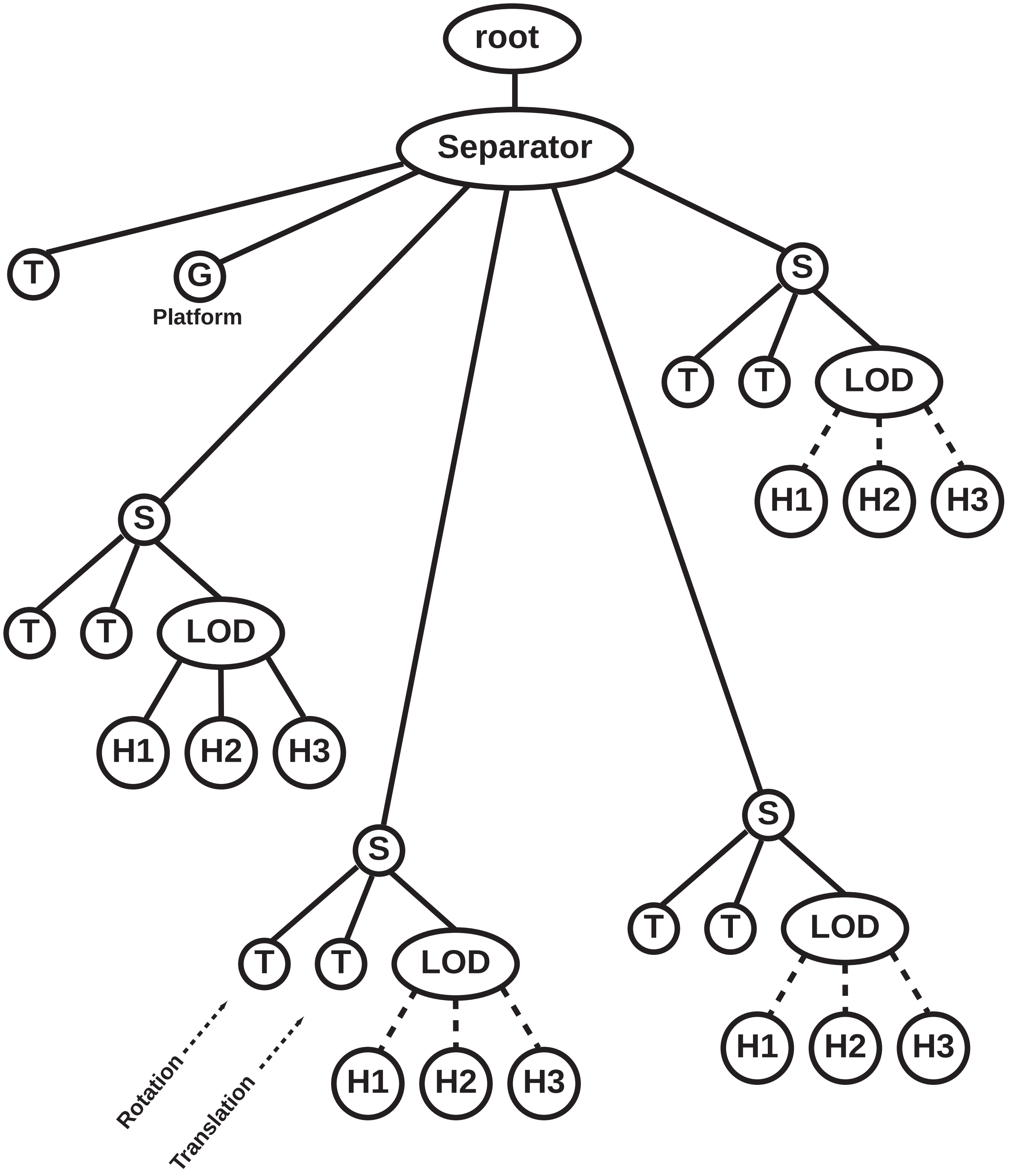
3. The first transform node places the entire merry-go-round in place, and rotates the merry-go-round as the simulation runs.

4. The platform geometry for the merry-go-round is located in the file PLATFORM.NFF. The platform has four poles, arranged as shown in the diagram, which remain stationary with respect to the platform. More specifically, the poles are 10 units away from the origin, along the positive and negative X and Z axes. The four horses will slide up and down on these poles.

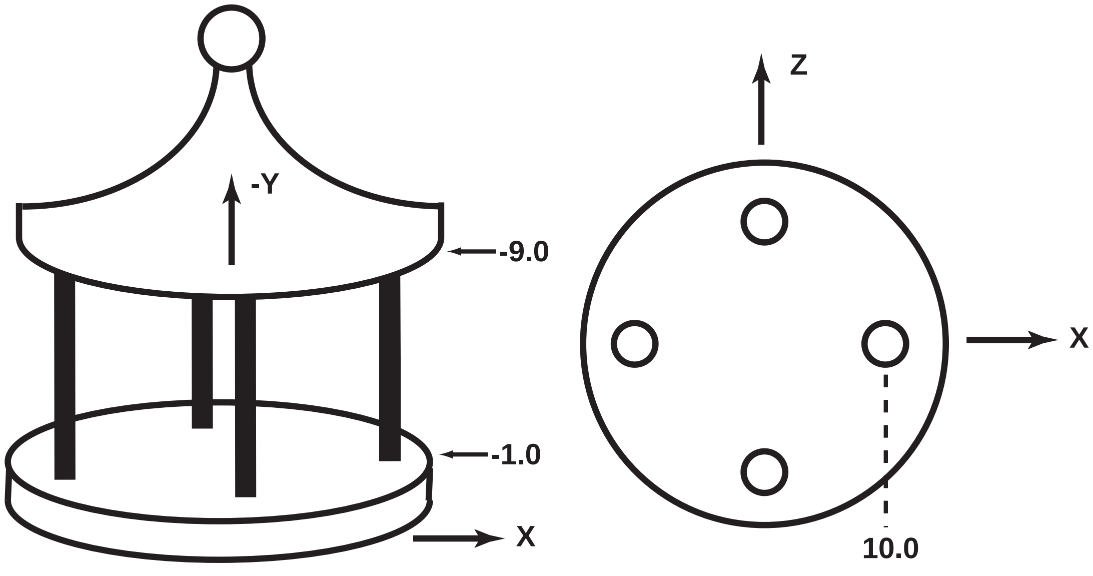
5. Each horse has two transform nodes. The first rotates the horse for it's proper orientation. The second translates the horse from the origin to it's pole. As the simulation runs, the horses will move up and down, in the range from -3 to -6. Initially, the horses should be placed at different heights within this range, as indicated on the diagram.

6. The horse geometries are located in three files, HORSE1.NFF, HORSE2.NFF, and HORSE3.NFF, which are decreasingly detailed versions of the same horse. (I.e. HORSE1 is more detailed than HORSE2, which is more detailed than HORSE3.) Each of these geometries should only be loaded into memory once, and placed on the scene graph in as many locations as is necessary.

7. Feel free to annotate the scene graph with additional labels, if it will make your code easier to write and / or follow.



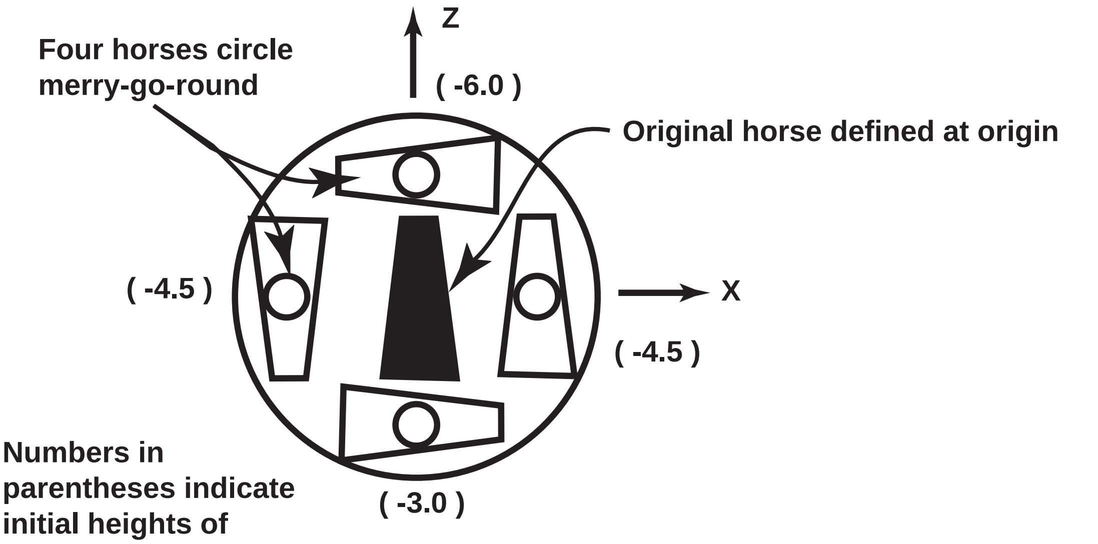
root
Separator
T
G
Platform
S
T
T
LOD
H1
H2
H3
S
T
T
LOD
H1
H2
H3
S
T
T
LOD
H1
H2
H3
Rotation
Translation
S
T
T
LOD
H1
H2
H3



PLATFORM.NFF

Not to Scale

HORSE*.NFF: A carousel horse, centered about the origin, facing forward along the positive Z axis. The length, width, and height of the horse are 4 units, 1 units, and 2 units respectively. I.e. the X coordinates of the horse extend from -0.5 to 0.5; the Y coordinates from -1.0 to 1.0, and the Z coordinates from -2.0 to 2.0.



Four horses circle merry-go-round

Original horse defined at origin

Numbers in parentheses indicate initial heights of each horse.

Not to Scale

```
/* Declare Global Variables Here */

WTnode *Merry = NULL, *Horses[ 4 ] = { NULL, NULL, NULL, NULL };
```

```
void loadMerryGoRound( WTnode *root ) {

     /* Declare local variables here */

     int i;

     WTnode *node = NULL, *sep = NULL, *lod = NULL, *sep2 = NULL,
            *horse1 = NULL, *horse2 = NULL, *horse3 = NULL;

     WTp3 offsets[ 4 ] =
          { {  10.0f, -4.5f,   0.0f },
            {   0.0f, -6.0f,  10.0f },
            { -10.0f, -4.5f,   0.0f },
            {   0.0f, -3.0f, -10.0f } };

     WTq rotations[ 4 ] =
          { { 0.0f,  0.0f,    0.0f, 1.0f    },
            { 0.0f, -0.7071f, 0.0f, 0.7071f },
            { 0.0f,  1.0f,    0.0f, 0.0f    },
            { 0.0f,  0.7071f, 0.0f, 0.7071f } };

     WTp3 center = { 20.0f, -5.0f, 30.0f };

     float lodRanges[ 3 ] = { 20.0, 50.0, 1.0e6 };


/* Continue local variables on additional page(s) as necessary */
     /* Begin code on a fresh page */
     /* Code begins here */

     sep = WTsepnode_new( root );
     Merry = WTxformnode_new( sep );
     WTnode_load( sep, "PLATFORM.NFF", 1.0f );

     for( i = 0; i < 4; i++ ) {

          sep2 = WTsepnode_new( sep );

          node = WTxformnode_new( sep2 );
          WTnode_setorientation( node, orientations[ i ] );

          Horses[ i ] = WTxformnode_new( sep2 );
          WTnode_settranslation( Horses[ i ], offsets[ i ] );

          lod = WTlodnode_new( sep2 );

          /* The following two lines were excluded from the exam
             question.  They are included in the solution only
             for completeness. */

          WTlodnode_setcenter( lod, center );
          WTlodnode_setrange( lod, lodRanges, 3 );

          if( i == 0 ) {

               horse1 = WTnode_load( lod, "HORSE1.NFF", 1.0f );
               horse2 = WTnode_load( lod, "HORSE2.NFF", 1.0f );
               horse3 = WTnode_load( lod, "HORSE3.NFF", 1.0f );

          } else {

               WTnode_addchild( lod, horse1 );
               WTnode_addchild( lod, horse2 );
               WTnode_addchild( lod, horse3 );

          } /* End of if-else block */

     } /* End of for loop through four horses */

} /* End of function to load merry go round */
```