Following is C# code for branches and balls system.

Firstly I have set the state of each gate switch randomly to left or right & checked the output.

But now the code is showing functionality exactly according to the given tree(As tree image provided in picture). it is working for depth level 4 with 15 balls. Gates open & closed states at start are also according to the picture.  
  
Spent approximately 1 hour to test & write code.

**Following is its output**

F has received ball

N has received ball

B has received ball

I has received ball

H has received ball

O has received ball

C has received ball

K has received ball

E has received ball

M has received ball

A has received ball

J has received ball

G has received ball

P has received ball

D has received ball

L has not received any ball

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace DevTest

{

/// <summary>

/// This class represents tree branch or gate generally known as node

/// </summary>

class Gate

{

public Gate leftChildGate;

public Gate rightChildGate;

public Container leftChildContainer;

public Container rightChildContainer;

public bool isLeftSideOpen;// if false then right side of gate is open

public int depth;

public Gate(int newGateDepth)

{

depth = newGateDepth;

//According to provided image open side of gate predictable for first 3 levels/depth

//for fourth level/depth open side of gate is random

isLeftSideOpen = Convert.ToBoolean(depth % 2);

leftChildContainer = null;

rightChildContainer = null;

}

}

/// <summary>

/// This class is used to represents last nodes of tree i.e containers to received balls

/// Containers are tagged according to provided image

/// </summary>

class Container

{

public char containerTag;

public static char \_containerTag = 'A';

//list to save all containers tags

public static List<char> \_containerTagList = new List<char>();

public Container()

{

containerTag = \_containerTag;

\_containerTagList.Add(\_containerTag);

//Generate next tag for next container

\_containerTag = (char)(((int)\_containerTag) + 1);

}

}

class Tree

{

private int \_systemDepth;

private int \_containerParentNumber = 0;

private Gate \_rootGate;

//Add root node/gate to the tree

public Tree(int systemDepth)

{

\_systemDepth = systemDepth;

\_rootGate = new Gate(1);

AddChilds(\_rootGate);

}

public void AddChilds(Gate parentGate)

{

//Add left & right child if it is not last level/depth

if (parentGate.depth < \_systemDepth)

{

parentGate.leftChildGate = new Gate(parentGate.depth + 1);

AddChilds(parentGate.leftChildGate);

parentGate.rightChildGate = new Gate(parentGate.depth + 1);

AddChilds(parentGate.rightChildGate);

}

//Add left & right container if it is last level/depth

else if (parentGate.depth == \_systemDepth)

{

\_containerParentNumber++;

parentGate.leftChildContainer = new Container();

parentGate.rightChildContainer = new Container();

parentGate.isLeftSideOpen = SetContainerParentOpenSide();

}

}

/// <summary>

/// This function is used to set default open side of last/4th level of gates according to provided image

/// </summary>

/// <returns></returns>

private bool SetContainerParentOpenSide()

{

//\_containerParentNumber is numbering of gates at last/4th level from left to right

switch (\_containerParentNumber)

{

case 1:

case 3:

case 4:

case 7:

return false;

case 2:

case 5:

case 6:

case 8:

default:

return true;

}

}

/// <summary>

/// This function is used to drop a single ball from the root of tree

/// It transfer the ball to the next right or left gate according to available open side

/// untill ball reach to the container

/// </summary>

public void DropBall()

{

//drop of ball started from root

Gate currentGate = \_rootGate;

Container receivedContainer = null;

while (receivedContainer == null)

{

//it is not last gate. move ball to the next gate.

if (currentGate.depth < \_systemDepth)

{

if (currentGate.isLeftSideOpen)

{

currentGate.isLeftSideOpen = !currentGate.isLeftSideOpen;

currentGate = currentGate.leftChildGate;

}

else

{

currentGate.isLeftSideOpen = !currentGate.isLeftSideOpen;

currentGate = currentGate.rightChildGate;

}

}

//it is last gate. After it drop ball in container

else if (currentGate.depth == \_systemDepth)

{

receivedContainer = currentGate.isLeftSideOpen ? currentGate.leftChildContainer : currentGate.rightChildContainer;

currentGate.isLeftSideOpen = !currentGate.isLeftSideOpen;

//if container has received ball remove it from list & display message

Container.\_containerTagList.Remove(receivedContainer.containerTag);

Console.WriteLine(" " + receivedContainer.containerTag + " has received ball");

}

}

}

}

class Program

{

static void Main(string[] args)

{

//Draw a tree having depth 4

Tree tree = new Tree(4);

//Drop 15 balls

for (int count = 1; count <= 15; count++)

tree.DropBall();

//Write containers has not received any ball

Console.WriteLine(Environment.NewLine);

foreach (char containerTag in Container.\_containerTagList)

Console.WriteLine(" " + containerTag + " has not received any ball");

Console.ReadLine();

}

}

}