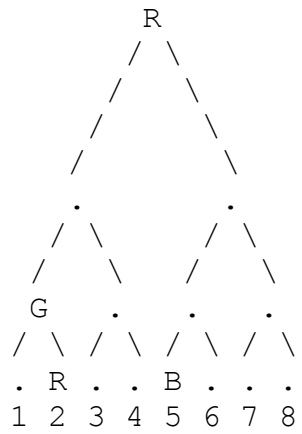
ICS 212 Homework 8

TreeBall in C++

Complete the TreeBall game we started in lab point exercises a couple months ago. However, use C++ idioms for all input, output, constants, etc.

In TreeBall, the user is presented with a perfect (complete) binary tree and a number of input options corresponding to each of the tree's leaves:



Each turn, the game generates a ball of a random color at the root of the tree. (Here, I'm using R, G, B for red, green, and blue.) The user then selects which leaf node to send the ball towards. The ball stops early if there is another ball blocking its path. For example, last turn the user must have sent the G ball to leaf 2, but it stopped early because there was already a R ball there. If the user had instead sent the ball to leaf 1, it would have made it.

The user's goal is to make an unbroken path of the same color from the root to one or more leaves. In this case, the player wins the game. If a ball cannot be sent from the root along the path selected by the user, the game ends in a loss. (For more examples, see some sample game sessions.) If it is not possible to send the current ball along any path from the root, the program should not ask for input.

First, don't forget to design before you start coding. As discussed during the lab exercises, an array-based tree is probably the best approach for this problem--though this is not the only possible solution. C code for creating and printing such a tree is available here.

You should also give some thought on paper on how you will direct a ball to the correct leaf node and how you will determine that a path of the same color exists from the root to a leaf node. The code for each of these tasks is shorter and less complicated than the printing code, but both will still require some careful thought.

You should write your code such that, by changing a single const variable, you can change the number of levels in the tree. (4 levels, as shown above, is a good default, but your code should be able to handle a change to any value between 2 and 6, inclusive, and still work correctly.) Also, there should be a single place where it is possible to change the ball colors, including how many colors, are used.

The primary purpose of this assignment is to practice pure C++ syntax. Do not use #define, malloc, free, or or any functions from C's <stdio.h> (which includes printf, scanf, getchar, puts, etc.) Instead, use const variables, new, delete, and <iostream>'s cout and cin. You may also use any C++ data structures, such as vector or string, though this is not required. (You can just use a char[] or char* for the tree.) If you need other C libraries, such as <stdlib.h> for rand() or <math.h> for pow(), use the C++ versions instead, such as <cmath> and <cstdlib>. You do not need to use OOP for this assignment; we'll get to that in the next one.

What to Submit

Besides your source code, also include transcripts for game sessions that demonstrate the following:

- you win
- you lose because you send a ball from the root into a full path
- you lose because both paths from the root are full
- you place a ball in every node of the tree [can be combined with one of above]

One way to do this is to copy-and-paste the contents out of the Windows command prompt window. To do this, you have to right-click and select Mark, then select what you want to copy, and right-click again to actually copy it. Paste this into a plain text (.txt) file.

Alternatively, you can do this very easily on uhunix. At the prompt, type:

```
script game.txt
```

You should then see a message that says: "Script started, file is game.txt". From this point on, everything you type or see on your screen is also being copied into game.txt. (You can use a different filename if you want to.) You should compile your program and play the games you want to record. When you are done, type:

```
exit
```

You will then see "Script done, file is game.txt". Download the game.txt from uhunix and include it in your submission to Tamarin.

Good luck!