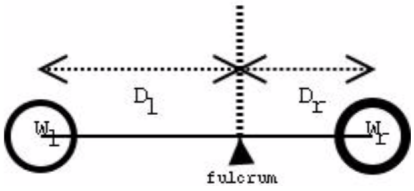
CS302

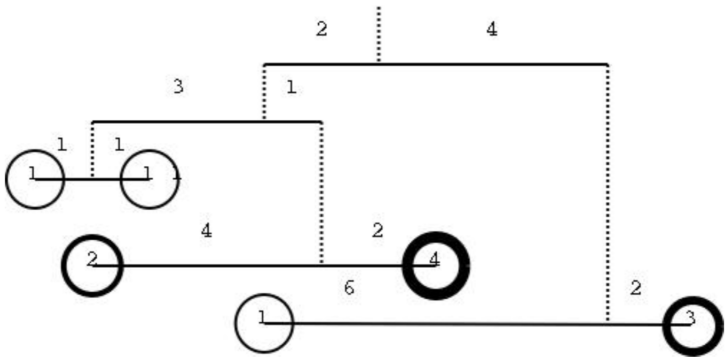
Assignment 5: Balance

Description

Before being an ubiquous communications gadget, a mobile was just a structure made of strings and wires suspending colorful things. This kind of mobile is usually found hanging over cradles of small babies. The figure below illustrates a simple mobile.



It is just a wire, suspended by a string, with an object on each side. It can also be seen as a kind of lever with the fulcrum on the point where the string ties the wire. From the lever principle we know that to balance a simple mobile the product of the weight of the objects by their distance to the fulcrum must be equal. That is $W_l \times D_l = W_r \times D_r$ where D_l is the left distance, D_r is the right distance, W_l is the left weight and W_r is the right weight. In a more complex mobile the object may be replaced by a sub-mobile, as shown in the next figure. In this case it is not so straightforward to check if the mobile is balanced so we need you to write a program that, given a description of a mobile as input, checks whether the mobile is in equilibrium or not.



Input

The input is composed of several lines, each containing 4 integers separated by a single space. The 4 integers represent the distances of each object to the fulcrum and their weights, in the format: W_l D_l W_r D_r If W_l or W_r is zero then there is a sub-mobile hanging from that end and the following lines define the the sub-mobile. In this case we compute the weight of the sub-mobile as the sum of weights of all its objects, disregarding the weight of the wires and strings. If both W_l and W_r are zero then the following lines define two sub-mobiles: first the left then the right one.

Output

For each test case, the output must follow the description below. The outputs of two consecutive cases will be separated by a blank line. Write 'YES' if the mobile is in equilibrium, write 'NO' otherwise.

Specifications

- Comment your code and your functions
- Make sure your code is memory leak free
- Store your binary tree using the Nodes/reference format (not using an array)
- Use a recursive function to traverse the binary tree to determine if the tree/mobile is balanced

Some More Helpful Details

You will have the following custom type, each node in the tree will a `binTreeNode<int>*`, that contains its left and right weight and diameter

```
template <class Type>
struct binTreeNode
{
  Type wL, dL, wR, dR;
  binTreeNode<Type> * left;
  binTreeNode<Type> * right;
};
```

You might want to have the following functions (this is up to you, this is just the way I approached this problem)

- ```
  template <class Type>
  void buildMobile(binTreeNode<Type> * r, ifstream& infile);
  ```

  This function will recursively build your mobile (tree), this uses a variation of preorder to build the tree, you read in one line of input, allocate a node, assign the values read from the line into the correct fields of this node, and then insert into the tree and recursively continue this until the left or right weight is not 0
  ```

- ```
  template <class Type>
  int getWeight(binTreeNode<Type> * r);
  ```

  This function compares `r`'s left side combined weight × `r->dL` with `r`'s right side combined weight × `r->dR`, if they are equal then it returns the sum of `r`'s left side combined weight with `r`'s right side weight, otherwise it returns some special value to its parent to denote the `r` points to a node that does not form a balanced mobile

  Hint: `r`'s left or right side combined weight could be `r->wL` or `r->wR`, but if either one is 0, then you may have to recursively compute the left or right side weight

- ```
  template <class Type>
  void destroyTree(binTreeNode<Type> * r);
  ```

  Deallocates the tree, uses a postorder traversal to do this

Example Output

```
$ cat MobileInput01.txt
0 2 0 4
0 3 0 1
1 1 1 1
2 4 4 2
1 6 3 2

$ cat MobileInput02.txt
0 8 0 5
0 11 0 6
4 1 2 2
8 3 3 8
0 2 0 1
1 8 8 1
6 5 10 3

$ g++ balance.cpp
$ ./a.out

Enter mobile data file: MobileInput01.txt

YES

$ ./a.out

Enter mobile data file: MobileInput02.txt

NO
```

Submission

Upload your source file using the following naming convention `LastName_FirstName_A5.cpp` by the deadline