**Recursion is hard to understand**

1. Write a Person class that recursively creates children for itself.
   1. The Person class will have a Person array of children. ***Person [] children***.
   2. The Person constructor should take an int as to how many children it should have. *public* ***Person(int numOfChildren)***
   3. Then in the body of the constructor you should instantiate the array (Person array of children) to the size of numOfChildren (passed into the constructor). ***children = new Person[numOfChildren]*** and iterate through the new array and populate it with a new Person. ***Children[i] = new Person()***
   4. Use Random.Next(0, 3) to call these new Person constructors with a number between 0-2. Meaning that these new Persons should have between 0-2 children when they are created (this is the recursion).
   5. NOTE: Just like we did in class, a method recursively calling itself, here the constructor recursively calls itself (stopping only when Random.Next(0, 3) = 0 and it creates no children).
   6. Keep a static int in Person class that you increment in the Person constructor, so at the end you can track how many Persons were created.
   7. In the main method, create one Person that should have 1 child (new Person(1) ) and then print how many total Persons were created.
2. E.C. Take it a step further, after you created this chain of Persons, recursively call a method that will track how many leaf nodes there are. Meaning how many Persons are there that don’t have any children.
   1. Do it with a static int in Person that you increment as you recursively call a method to call on the first created person, that he calls on his children, that they call on theirs, etc.
   2. Any person in this chain discovers he doesn’t have children he should increment that static int.
   3. Then print this int.

Try your best. We will review in class………..