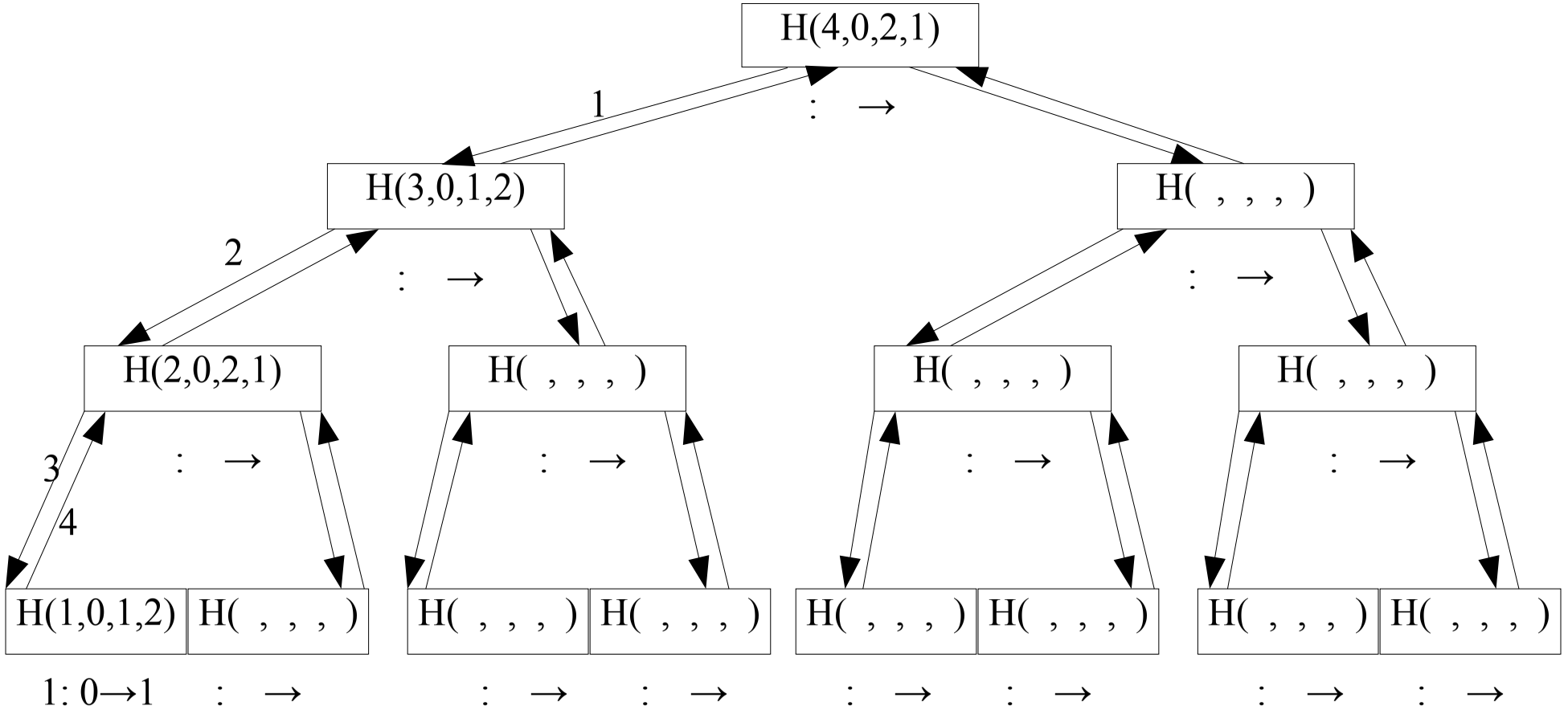
Quiz W5 – CS 102

Name: _____, ID: _____

1) Consider the following code:

```
void hanoi(int numDisks, int start, int end, int temp) {
      if (numDisks == 1) {
            move (start, end);
      } else {
            hanoi (numDisks-1, start, temp, end);
            move (numDisks-1, start, end);
            hanoi (numDisks-1, temp, end, start);
      }
}
void move(int diskIndex, int start, int end) {
      System.out.println(diskIndex + ":" + start + "->" + end);
}
```

Draw the execution tree for `hanoi(4, 0, 2, 1)`. Use the next paper. You can draw it in landscape so that you have enough horizontal space. Draw the current step number next to the arrows representing the moves from one method call to next next/previous, as we did in the class. At a separate place, list all the moves in order.



Sequence: 1: 0→1,

2) Consider the following problem: You are asked to create a class called `PermutationGenerator`. This class is expected to generate all permutations of a given array of ints. You are also asked to design an interface called `IPermutationClient`. The interface should have a single method: `void processPermutation(int[] permutation)`. The `PermutationGenerator` is expected to take the permutation client as an argument to its constructor. It should have a `void generatePermutations(int[] values)` method, which should call (indirectly) the permutation client as it generates each permutation.

a) Write the code for `PermutationGenerator` and `IPermutationClient`. 10 points.

```
// TODO: Interface code




class PermutationGenerator {
      // TODO: Members


      // TODO: Constructor



      public void generatePermutations(int[] values)  {
            generate(values, 0);
      }

      private void generate(int[] values, int i) {
            // The base case is when i reaches n (i.e., values.length).
            // When the base case is reached, call the client.



            // Otherwise, there are (values.length - i) possible choices for the
            // value of the item at ith location.
            // Swap the ith item with the jth one, where j>=i, and recurse









      }
      private void swap(int[] values, int i, int j) {



      }
}
```

b) Write the driver code for printing all permutations of the list: `{1,2,3,4}`. For each permutation, prepend its index at the beginning, as in: `1: [1, 2, 3, 4]`.

```
class PermsDriver implements ____________________
{
    // TODO: member variables



    // TODO: printing logic




    public static void main(String[] args) {
        // TODO: Client instantiation code


        PermutationGenerator gen = new PermutationGenerator(_______);
        gen.generatePermutations(new int[]{1,2,3});
    }
}
```