Given an array of pairs, find all symmetric pairs in it

Two pairs (a, b) and (c, d) are said to be symmetric if c is equal to b and a is equal to d. For example (10, 20) and (20, 10) are symmetric. Given an array of pairs find all symmetric pairs in it.

It may be assumed that first elements of all pairs are distinct.

Example:

```
Input: arr[] = {{11, 20}, {30, 40}, {5, 10}, {40, 30}, {10, 5}}
Output: Following pairs have symmetric pairs
        (30, 40)
        (5, 10)
```

We strongly recommend you to minimize your browser and try this yourself first.

A **Simple Solution** is to go through every pair, and check every other pair for symmetric. This solution requires O(n^2) time.

A **Better Solution** is to use sorting. Sort all pairs by first element. For every pair, do binary search for second element in the given array, i.e., check if second element of this pair exists as first element in array. If found, then compare first element of pair with second element. Time Complexity of this solution is O(nLogn).

An **Efficient Solution** is to use Hashing. First element of pair is used as key and second element is used as value. The idea is traverse all pairs one by one. For every pair, check if its second element is in hash table. If yes, then compare the first element with value of matched entry of hash table. If the value and the first element match, then we found symmetric pairs. Else, insert first element as key and second element as value.

Following is Java implementation of this idea.

```
// A Java program to find all symmetric pairs in a given array of pairs
import java.util.HashMap;

class SymmetricPairs {

    // Print all pairs that have a symmetric counterpart
    static void findSymPairs(int arr[][])
    {
        // Creates an empty hashMap hM
        HashMap<Integer, Integer> hM = new HashMap<Integer, Integer>();

        // Traverse through the given array
        for (int i = 0; i < arr.length; i++)
        {
            // First and second elements of current pair
            int first = arr[i][0];
            int sec   = arr[i][1];

            // Look for second element of this pair in hash
            Integer val = hM.get(sec);

            // If found and value in hash matches with first
            // element of this pair, we found symmetry
            if (val != null && val == first)
               System.out.println("(" + sec + ", " + first + ")");

            else  // Else put sec element of this pair in hash
               hM.put(first, sec);
        }
    }

    // Drive method
    public static void main(String arg[])
    {
        int arr[][] = new int[5][2];
        arr[0][0] = 11; arr[0][1] = 20;
        arr[1][0] = 30; arr[1][1] = 40;
        arr[2][0] = 5;  arr[2][1] = 10;
        arr[3][0] = 40;  arr[3][1] = 30;
        arr[4][0] = 10;  arr[4][1] = 5;
        findSymPairs(arr);
    }
}
```

Output:

```
Following pairs have symmetric pairs
(30, 40)
(5, 10)
```

Time Complexity of this solution is O(n) under the assumption that hash search and insert methods work in O(1) time.