2096. Step-By-Step Directions From a Binary Tree Node to Another

```
/*
   LCA,Recusrion,backtracking,DFS
   Time complexity: O(n)
   Extra space complexity: O(n)
*/
class Solution {
   public:
      TreeNode* lowest_common_ancestor(TreeNode* root, int p, int q){
         if (!root) return nullptr;

         TreeNode* left = lowest_common_ancestor(root->left,p,q);
         TreeNode* right = lowest_common_ancestor(root->right,p,q);

         if ( left && right || root->val == p || root->val == q) return root;

         return left != nullptr?left:right;
      }

      void find_path(TreeNode* p,int val,std::string& path_tmp,std::string& path){
         if(!p) return;

         if(p->val==val) path=path_tmp;

         path_tmp.push_back('L');
         find_path(p->left,val,path_tmp,path);
         path_tmp.pop_back();

         path_tmp.push_back('R');
         find_path(p->right,val,path_tmp,path);
         path_tmp.pop_back();
      }

      string getDirections(TreeNode* root, int startValue, int destValue){
         TreeNode* lca=lowest_common_ancestor(root,startValue,destValue);

         std::string tmp,path_start_lca;
         find_path(lca,startValue,tmp,path_start_lca);

         for(int i=0;i<path_start_lca.size();++i) path_start_lca[i]='U';

         tmp="";
         std::string path_lca_dest;
         find_path(lca,destValue,tmp,path_lca_dest);

         return path_start_lca+path_lca_dest;
      }
};
```