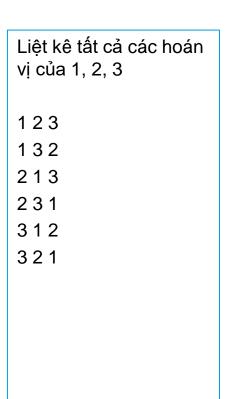
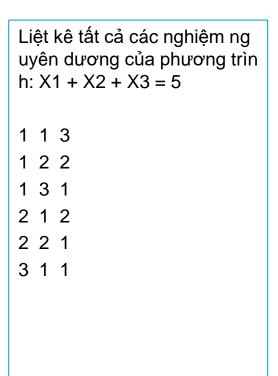
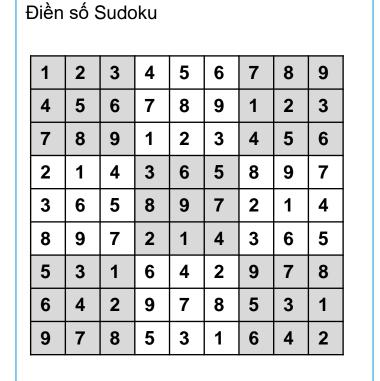
• Bài toán duyệt (liệt kê/vét cạn) tất cả các phương án (cấu hình tổ hợp) thỏa mãn các điều kiện (ràng buộc đặt ra)

Liệt kê tất cả các xâu n hị phân độ dài 3
000
001
010
011
100
101
110
111







- Bài toán duyệt (liệt kê/vét cạn) tất cả các phương án (cấu hình tổ hợp) thỏa mãn các điều kiện (ràng buộc đặt ra)
- Lời giải được biểu diễn bởi bộ các biến quyết định (X1, X2, . . ., Xn), mỗi biến Xk có thể nhận các giá trị từ tập Ak
- Xét lần lượt từng biến (giả sử lần lượt xét các biến theo thứ tự X1, X2, . . ., Xn)
- Với mỗi biến thì duyệt tất cả các giá trị có thể gán cho biến đó (kiếm soát ràng buộc)

```
1. Try(k){ // thứ giá trị cho Xk
    for each v in Ak do
       if check(v, k){ // kiểm tra xem v có thể gán cho Xk mà ko vi phạm ràng buộc
3.
         Xk = v; // gán v cho Xk
         [Cập nhận một số cấu trúc dữ liệu liên quan]
         if k = n then Solution();
         else Try(k+1);
         [Khôi phục trạng thái cấu trúc dữ liệu được cập nhật ở dòng 5
8.
9.
10. }
Main(){
   Try(1);
```

Liệt kê các xâu nhị phân độ dài n

```
public class Main {
    static int[] X;
    static int n;
    public static void solution(){
        for(int i = 1; i <= n; i++) System.out.print(X[i]);</pre>
        System.out.println();
    public static void Try(int k){
        for(int v = 0; v <=1; v++){
            X[k] = v;
            if(k == n) solution();
            else Try(k+1);
    public static void main(String[] args) {
        n = 3;
        X = new int[n+1];
        Try(1);
```

Liệt kê các xâu nhị phân độ dài n

```
public class Main {
    static int[] X;
    static int n;
    public static void solution(){
        for(int i = 1; i <= n; i++) System.out.print(X[i]);</pre>
        System.out.println();
    public static void Try(int k){
        for(int v = 0; v <=1; v++){
            X[k] = v;
            if(k == n) solution();
            else Try(k+1);
    public static void main(String[] args) {
        n = 3;
        X = new int[n+1];
        Try(1);
```

• Liệt kê các hoán vị của 1, 2, . . ., n. Sử dụng mảng đánh dấu: mark[v] = true nếu v đã xuất hiện, và mark[v] = false, ngược lại

```
public class Main {
    static int[] X;
    static boolean[] mark;
   static int n;
    public static void solution(){
        for(int i = 1; i \le n; i++) System.out.print(X[i] + " ");
        System.out.println();
    public static void Try(int k){
        for(int v = 1; v <= n; v++){
            if(!mark[v]) {
                X[k] = v;
                mark[v] = true;
                if (k == n) solution();
                else Try(k + 1);
                mark[v] = false;
    public static void main(String[] args) {
        n = 3;
        X = new int[n+1]; mark = new boolean[n+1];
        for(int v = 1; v \leftarrow n; v++) mark[v] = false;
        Try(1);
```

• Liệt kê các hoán vị của 1, 2, . . ., n. Sử dụng mảng đánh dấu: mark[v] = true nếu v đã xuất hiện, và mark[v] = false, ngược lại

```
public class Main {
    static int[] X;
    static boolean[] mark;
   static int n;
    public static void solution(){
        for(int i = 1; i \le n; i++) System.out.print(X[i] + " ");
        System.out.println();
    public static void Try(int k){
        for(int v = 1; v <= n; v++){
            if(!mark[v]) {
                X[k] = v;
                mark[v] = true;
                if (k == n) solution();
                else Try(k + 1);
                mark[v] = false;
    public static void main(String[] args) {
        n = 3;
        X = new int[n+1]; mark = new boolean[n+1];
        for(int v = 1; v \leftarrow n; v++) mark[v] = false;
        Try(1);
```

```
123
132
213
231
312
321
```