1. "가장 가까운 큰수" 정답코드

```
class Solution {
        int answer, target, m;
        ArrayList<Integer> nums;
        int[] ch;
        boolean flag;
        public void DFS(int L, int number){
                if(flag) return;
                if(L == m){
                        if(number > target){
                                answer = number;
                                flag = true;
                        }
                }
                else{
                        for(int i = 0; i < m; i++){
                                if(ch[i] == 0){
                                        ch[i] = 1;
                                        DFS(L + 1, number * 10 + nums.get(i));
                                        ch[i] = 0;
                                }
                        }
                }
        }
        public int solution(int n){
                answer = 0;
                flag = false;
                nums = new ArrayList<>();
                target = n;
                int tmp = n;
                while(tmp > 0){
                        int t = tmp%10;
                        nums.add(t);
                        tmp = tmp / 10;
                }
                nums.sort((a, b) -> a - b);
                m = nums.size();
                ch = new int[m];
                DFS(0, 0);
                if(flag == false) return -1;
                return answer;
        }
}
```

2. "줄다리기" 정답코드

```
class Solution {
        int[] ch;
        int[][] relation;
        int answer;
        Stack<Integer> pm;
        public void DFS(int L){
                if(L == 7) answer++;
                else{
                        for(int i = 1; i < 8; i++){
                                if(!pm.empty() && relation[pm.peek()][i] == 1) continue;
                                if(ch[i] == 0){
                                        ch[i] = 1;
                                        pm.push(i);
                                        DFS(L + 1);
                                        ch[i] = 0;
                                        pm.pop();
                                }
                        }
                }
        }
        public int solution(int[][] fight){
                answer = 0;
                pm = new Stack<>();
                relation=new int[8][8];
                for(int[] x : fight){
                        relation[x[0]][x[1]] = 1;
                        relation[x[1]][x[0]] = 1;
                }
                ch = new int[8];
                DFS(0);
                return answer;
        }
}
```

▶▶ Comment :

if(!pm.empty() && relation[pm.peek()][i] == 1) continue;

pm.peek()번 학생과 i번 학생이 서로 싫어하는 학생이면 해당 순열의 경우는 더 이상 만들지 않습니다.

3. "바둑대회" 정답코드

```
class Solution {
        int n, answer = 1000000000;
        int[] ch;
        public void DFS(int L, int s, int[][] cans){
                if(L == n/2){
                        ArrayList<Integer> A = new ArrayList<>();
                        ArrayList<Integer> B = new ArrayList<>();
                        for(int i = 0; i < n; i++){
                                if(ch[i] == 1) A.add(i);
                                else B.add(i);
                        }
                        int Asum = 0, Bsum = 0;
                        for(int i = 0; i < L; i++){
                                Asum += cans[A.get(i)][0];
                                Bsum += cans[B.get(i)][1];
                        }
                        answer = Math.min(answer, Math.abs(Asum-Bsum));
                }
                else{
                        for(int i = s; i < n; i++){
                                ch[i] = 1;
                                DFS(L + 1, i + 1, cans);
                                ch[i] = 0;
                        }
                }
        }
        public int solution(int[][] cans){
                n = cans.length;
                ch = new int[n];
                DFS(0, 0, cans);
                return answer;
        }
}
```

4. "팰린드롬의 경우수" 정답코드

```
class Solution {
        Deque<Character> tmp;
        ArrayList<String> res;
        HashMap<Character, Integer> sH;
        int len;
        public void DFS(){
                if(tmp.size() == len){
                        String Ts = "";
                        for(char x : tmp) Ts += x;
                        res.add(Ts);
                }else{
                        for(char key : sH.keySet()){
                                if(sH.get(key) == 0) continue;
                                tmp.addFirst(key);
                                tmp.addLast(key);
                                sH.put(key, sH.get(key) - 2);
                                DFS();
                                tmp.pollFirst();
                                tmp.pollLast();
                                sH.put(key, sH.get(key) + 2);
                        }
                }
        }
        public String[] solution(String s){
                tmp = new LinkedList<>();
                res = new ArrayList<>();
                sH = new HashMap<>();
                len = s.length();
                for(char x : s.toCharArray()){  sH.put(x, sH.getOrDefault(x, 0)+1); }
                int odd = 0;
                char mid = '#';
                for(char key : sH.keySet()){
                        if(sH.get(key) % 2 == 1){
                                mid = key;
                                odd++;
                        }
                }
                if(odd > 1) return new String[]{};
                if(mid != '#'){
                        tmp.add(mid);
                        sH.put(mid, sH.get(mid) - 1);
                }
                DFS();
                String[] answer = new String[res.size()];
                for(int i = 0; i < res.size(); i++) answer[i] = res.get(i);
                return answer;
        }
}
```

5. "IP 주소" 정답코드

```
class Solution {
        LinkedList<String> tmp;
        ArrayList<String> res;
        public void DFS(int start, String s){
                if(tmp.size() == 4 && start == s.length()){
                        String Ts = "";
                        for(String x : tmp) Ts += x+".";
                        res.add(Ts.substring(0, Ts.length()-1));
                }
                else{
                        for(int i = start; i < s.length(); i++){
                                if(s.charAt(start) == '0' && i > start) return;
                                String num = s.substring(start, i + 1);
                                if(Integer.parseInt(num) > 255) return;
                                tmp.add(num);
                                DFS(i + 1, s);
                                tmp.pollLast();
                        }
                }
        }
        public String[] solution(String s){
                tmp = new LinkedList<>();
                res = new ArrayList<>();
                DFS(0, s);
                String[] answer = new String[res.size()];
                for(int i = 0; i < res.size(); i++) answer[i] = res.get(i);
                return answer;
        }
}
```

▶▶ Comment :

if(s.charAt(start) == '0' && i > start) return;

‘0’ 으로 시작하면서 두자리 숫자 이상이면 이 경우로 재귀를 더 진행하지 않습니다.

6. "알파코드" 정답코드

```
class Solution {
        int[] dy;
        public int DFS(int start, String s){
                if(dy[start] > 0) return dy[start];
                if(start < s.length() && s.charAt(start) == '0') return 0;
                if(start == s.length() -1 || start == s.length()) return 1;
                else{
                        int res = DFS(start + 1, s);
                        int tmp = Integer.parseInt(s.substring(start, start + 2));
                        if(tmp <= 26) res += DFS(start + 2, s);
                        return dy[start] = res;
                }
        }
        public int solution(String s){
                dy = new int[101];
                int answer = DFS(0, s);
                return answer;
        }
}
```

▶▶ Comment :

if(start < s.length() && s.charAt(start) == '0') return 0;

'0' 으로 시작하는 경우에는 0을 반환합니다.