스타트와 링크

https://www.acmicpc.net/problem/14889

스타트와 링크

N = 6 일 경우

스타트 팀 0,1,2 그러면 링크 팀은 3,4,5

N/2개 뽑는 조합을 구하고

스타트 팀 = $S_{01} + S_{02} + S_{10} + S_{12} + S_{20} + S_{21}$

링크 팀 = $S_{34} + S_{35} + S_{43} + S_{45} + S_{53} + S_{54}$

구해진 조합에서 2개 나열하는 순열을 구한다.

스타트와 링크

```
public static void main(String[] args){
        Scanner sc = new Scanner(System.in);

        N = sc.nextInt();
        mat = new int [N][N];

        R = N / 2;
        team = new int [R];

        for(int i = 0; i < N; i++)
            for(int j = 0; j < N; j++)
                mat[i][j] = sc.nextInt();

        solve(0, 0);

        System.out.println(ans);
        sc.close();
    }
```

0	1	2	3	4	5
1	0	2	3	4	5
1	2	0	3	4	5
1	2	3	0	4	5
1	2	3	4	0	5
1	2	3	4	5	0

스타트와 링크

```
static void solve(int k, int s)
   {
       if (k == R)
       {
           …
       }
       else
           for (int i = s; i < N + (k - R) + 1; i++)
           {
               team[k] = i;
               solve(k + 1, i + 1);
           }
   }
```

N이 6일 때 N/2 개의 조합을 구한다.

```
static void solve(int k, int s)
   {
       if (k == R)
       {
           int start = 0, link = 0;

           int [] chk = new int [20];
           int [] x = new int [20];

           for (int i = 0; i < R; i++)
               chk[team[i]] = 1;
           int tcnt = 0;
           for (int i = 0; i < 20; i++)
               if (chk[i] == 0)
                   x[tcnt++] = i;

           for (int i = 0; i < R - 1; i++)
               for (int j = i + 1; j < R; j++)
                   start += (mat[team[i]][team[j]] + mat[team[j]][team[i]]);

           for (int i = 0; i < R - 1; i++)
               for (int j = i + 1; j < R; j++)
                   link += (mat[x[i]][x[j]] + mat[x[j]][x[i]]);

           ans = ans < Math.abs(start - link) ? ans : Math.abs(start - link);
       }
       else
            …
}
```

team = [0, 1, 2]

x = [3, 4, 5]

$S_{01} + S_{02} + S_{10} + S_{12} + S_{20} + S_{21}$

$S_{34} + S_{35} + S_{43} + S_{45} + S_{53} + S_{54}$

3개에서 2개 나열하는 순열

퇴사

https://www.acmicpc.net/problem/14501

퇴사

	1일	2일	3일	4일	5일	6일	7일
Ti	3	5	1	1	2	4	2
Pi	10	20	10	20	15	40	200

	1일	2일	3일	4일	5일	6일	7일
Ti	7	6	5	4	3	2	1
Pi	10	20	10	20	15	40	200

Ti 와 Pi 두 개의 인자가 있으며 항상 비례적이지는 않다.

	1일	2일	3일	4일	5일	6일	7일
Ti	1	1	1	1	1	1	1
Pi	10	20	10	20	15	40	200

완전검색 : 모든 부분 집합 조사

모든 부분 집합 ➔ Ti로 제외시키고 ➔ Pi로 최적해 구하기

퇴사

```
public static void main(String[] args){
        Scanner sc = new Scanner(System.in);

        N = sc.nextInt();

        for(int i = 0; i < N; i++) {
            Ti[i] = sc.nextInt();
            Pi[i] = sc.nextInt();
        }

        solve(0);

        System.out.println(ans);
        sc.close();
    }
```

	1일	2일	3일	4일	5일	6일	7일
Ti	3	5	1	1	2	4	2
Pi	10	20	10	20	15	40	200

퇴사

	1일	2일	3일	4일	5일	6일	7일
Ti	3	5	1	1	2	4	2
Pi	10	20	10	20	15	40	200

```
static void solve(int k)
    {
        if (k == N)
        {
            …
        }
        else
        {
            Si[k] = true;    solve(k + 1);
            Si[k] = false;   solve(k + 1);
        }
    }
```

Si	1	1	1	1	1	1	1

N개에 대한 부분 집합

퇴사

	1일	2일	3일	4일	5일	6일	7일
Ti	3	5	1	1	2	4	2
Pi	10	20	10	20	15	40	200

선택유무조사

Si	1	1	1	1	1	1	1

```
static void solve(int k)
{
    if (k == N) {
        for (int i = 0; i < N; i++) {
            if (Si[i])
                for (int j = i + 1; j < i + Ti[i]; j++)
                    if (j >= N || Si[j])
                        return;
        }
        int tsum = 0;
        for (int i = 0; i < N; i++)
            if (Si[i])
                tsum += Pi[i];
        if (tsum > ans) ans = tsum;
    }
    else
        …
}
```

	1일	2일	3일	4일	5일	6일	7일
Ti	3	5	1	1	2	4	2
Pi	10	20	10	20	15	40	200

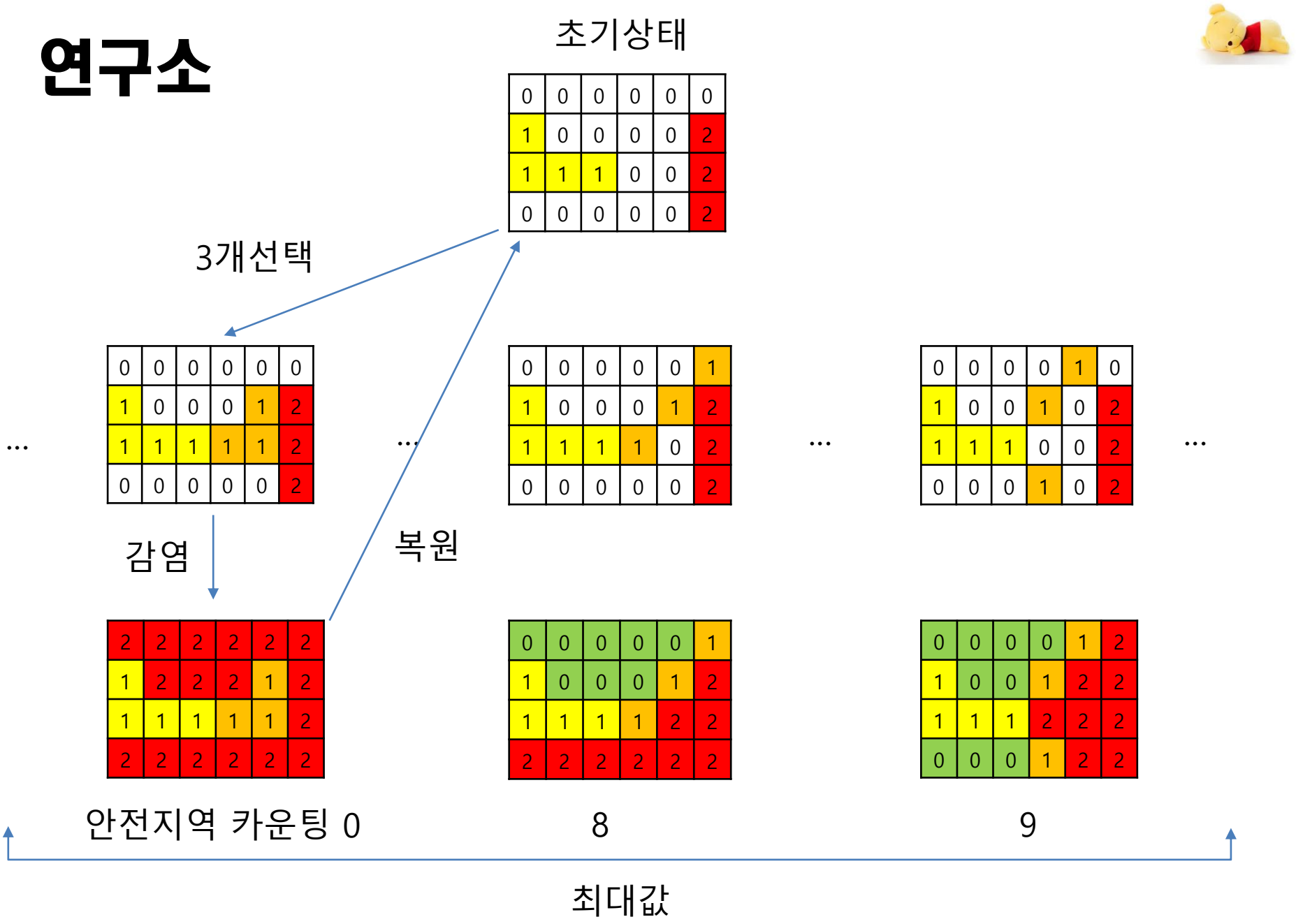
Si	1	0	0	0	1	0	0

10 + 15 = 25 가 최선인가?

연구소

https://www.acmicpc.net/problem/14502

연구소



연구소

```
public static void main(String[] args){
        Scanner sc = new Scanner(System.in);

        N = sc.nextInt(); M = sc.nextInt();

        mat = new int [N][M];
        backup_mat = new int [N][M];

        for (int i = 0; i < N; i++)
            for (int j = 0; j < M; j++)
                backup_mat[i][j] = mat[i][j] = sc.nextInt();

        for (int i = 0; i < N; i++) {
            for (int j = 0; j < M; j++) {

                if (mat[i][j] == 2) {
                    virus_pos[viruscnt][0] = i;
                    virus_pos[viruscnt++][1] = j;
                }
                else if (mat[i][j] == 0){
                    safe_pos[safecnt][0] = i;
                    safe_pos[safecnt++][1] = j;
                }
            }
        }

        solve(0, 0);

        System.out.println(ans);
        sc.close();
    }
```

0	0	0	0	0	0
1	0	0	0	0	2
1	1	1	0	0	2
0	0	0	0	0	2

0	0	0	0	0	0
1	0	0	0	0	2
1	1	1	0	0	2
0	0	0	0	0	2

0, 0	0, 1	0, 2	0, 3	0, 4	...	3, 4

연구소

```
static void solve(int k, int s)
{
    if (k == 3)
    …
    else {
        for (int i = s; i < safecnt + (k - 3) + 1; i++) {
            combi[k] = i;
            solve(k + 1, i + 1);
        }
    }
}
```

0	1	2

0, 0	0, 1	0, 2	0, 3	0, 4	…	3, 4

안전영역 개수에서 임의
3개 선택 ➔ 조합 생성

0	0	0	0	0	0
1	0	0	0	0	2
1	1	1	0	0	2
0	0	0	0	0	2

연구소

```
static void solve(int k, int s)
{
    if (k == 3) {
        int x, y;

        for (int i = 0; i < 3; i++) {
            x = safe_pos[combi[i]][0];
            y = safe_pos[combi[i]][1];
            mat[x][y] = 1;
        }

        for (int i = 0; i < viruscnt; i++) {
            x = virus_pos[i][0];
            y = virus_pos[i][1];
            virus_infact(x, y);
        }

        int tsafecnt = 0;
        for (int i = 0; i < N; i++)
            for (int j = 0; j < M; j++)
                if (mat[i][j] == 0)
                    tsafecnt++;
        if (ans < tsafecnt) ans = tsafecnt;

        for (int i = 0; i < N; i++)
            for (int j = 0; j < M; j++)
                mat[i][j] = backup_mat[i][j];
    }
    else
        …
```

벽 세우기

0	0	0	0	0	0
1	0	0	0	1	2
1	1	1	1	1	2
0	0	0	0	0	2

안전영역세기

감염시키기

2	2	2	2	2	2
1	2	2	2	1	2
1	1	1	1	1	2
2	2	2	2	2	2

초기 상태로 복원

0	0	0	0	0	0
1	0	0	0	0	2
1	1	1	0	0	2
0	0	0	0	0	2

연구소

```
static void virus_infact(int x, int y)
    {
        mat[x][y] = 2;
         …
        int xx, yy;

        for (int i = 0; i < 4; i++) {
            xx = x + dx[i];
            yy = y + dy[i];
            if (xx < 0 || xx >= N || yy < 0 || yy >= M) continue;
            if (mat[xx][yy] == 0)
                virus_infact(xx, yy);
        }
    }
```

DFS로 순회

0	0	0	0	0	0
1	0	0	0	1	2
1	1	1	1	1	2
0	0	0	0	0	2

(1, 5)에서 감염

2	2	2	2	2	2
1	2	2	2	1	2
1	1	1	1	1	2
0	0	0	0	0	2

연구소

```
def virus_infact(x, y):
    mat[x][y] = 2
    for dx, dy in ((0, 1), (0, -1), (1, 0), (-1, 0)):
        xx, yy = x + dx, y + dy
        if not (0 <= xx < N and 0 <= yy < M): continue
        if not mat[xx][yy]:
            virus_infact(xx, yy)
```

0	0	0	0	0	0
1	0	0	0	1	2
1	1	1	1	1	2
0	0	0	0	0	2

(1, 5)에서 감염

2	2	2	2	2	2
1	2	2	2	1	2
1	1	1	1	1	2
0	0	0	0	0	2

치킨배달

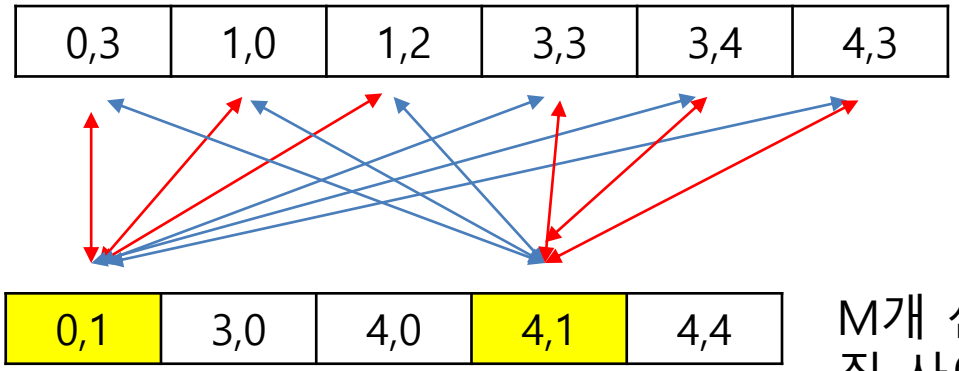
https://www.acmicpc.net/problem/15686

치킨배달

M = 2 (2개 치킨 집만 선택)

	0	1	2	3	4
0	0	2	0	1	0
1	1	0	1	0	0
2	0	0	0	0	0
3	2	0	0	1	1
4	2	2	0	1	2

집위치



모든 집과

M개 선택한 치킨 집 사이의 거리 중 최소 거리 선택

치킨집위치

치킨 집에서 임의 2개 선택

$_5C_2$

최소거리의 합 구하기

합 중 최소 구하기

치킨배달

	0	1	2	3	4
0	0	2	0	1	0
1	1	0	1	0	0
2	0	0	0	0	0
3	2	0	0	1	1
4	2	2	0	1	2

```
public static void main(String[] args){
        …

    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++) {

            if (mat[i][j] == 1) {
                home[homecnt][0] = i;
                home[homecnt++][1] = j;
            }
            else if (mat[i][j] == 2) {
                chicken[chickencnt][0] = i;
                chicken[chickencnt++][1] = j;
            }
        }
    }

    for (int i = 0; i < chickencnt; i++)
        for (int j = 0; j < homecnt; j++)
            dist[i][j] = Math.abs(chicken[i][0] - home[j][0]) +
                         Math.abs(chicken[i][1] - home[j][1]);

    solve(0, 0);

        …
}
```

0,3	1,0	1,2	3,3	3,4	4,3

0,1	3,0	4,0	4,1	4,4

치킨 집에서 M개 선택하는 조합 생성

치킨배달

```
public static void main(String[] args){
        …

    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++) {

            if (mat[i][j] == 1) {
                home[homecnt][0] = i;
                home[homecnt++][1] = j;
            }
            else if (mat[i][j] == 2) {
                chicken[chickencnt][0] = i;
                chicken[chickencnt++][1] = j;
            }
        }
    }

    for (int i = 0; i < chickencnt; i++)
        for (int j = 0; j < homecnt; j++)
            dist[i][j] = Math.abs(chicken[i][0] - home[j][0]) +
                         Math.abs(chicken[i][1] - home[j][1]);

    solve(0, 0);

        …
}
```

모든 집과 모든 치킨 집 사이의 거리를 구해 놓기

치킨집 \ 집						
	2	2	2	5	6	6
	6	2	4	3	4	4
	7	3	5	4	5	3
	6	4	4	3	4	2
	5	7	5	2	1	1

치킨배달

집에서 최소거리 치킨집 선택

예

0	2

2	2	2	5	6	6
6	2	4	3	4	4
7	3	5	4	5	3
6	4	4	3	4	2
5	7	5	2	1	1

2+2+2+4+5+3=18

```
static void solve(int k, int s){

    if (k == M){
        int tsum = 0;
        for (int h = 0; h < homecnt; h++){
            int tmin = Integer.MAX_VALUE;
            for (int c = 0; c < M; c++)
                tmin = Math.min(tmin, dist[combi[c]][h]);
            tsum += tmin;
        }
        ans = Math.min(ans,  tsum);
    }
    else
        for (int i = s; i < chickencnt + k - M + 1; i++){
            combi[k] = i;
            solve(k + 1, i + 1);
        }
}
```

M길이 조합 구하기

인구이동

https://www.acmicpc.net/problem/16234

인구이동

L = 10, R = 50


국경
개방
인구
이동
10 100 20 90
80 100 60 70
70 20 30 40
50 20 100 10
10 100 20 90
80 100 60 70
70 20 30 40
50 20 100 10
10 100 50 50
50 50 50 50
50 50 50 50
50 50 100 50
10 100 50 50
50 50 50 50
50 50 50 50
50 50 100 50
30 66 66 50
30 66 50 50
50 50 62 50
50 62 62 62
30 66 66 50
30 66 50 50
50 50 62 50
50 62 62 62
48 48 54 54
54 54 54 50
54 54 54 54
54 54 62 54

인구이동

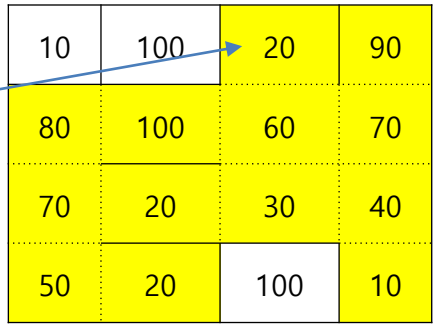
```
public static void main(String[] args){
        …

    int cnt = 0;

    while (true)
    {
        visited = new boolean [N][N];
        moved = false;

        for (int i = 0; i < N; i++)
            for (int j = 0; j < N; j++)
                if (!visited[i][j])
                 bfs(i, j);

        if (moved) cnt++;
        else  break;
    }

    System.out.println(cnt);
}
```

매번 visited 새로 만들기

10	100	20	90
80	100	60	70
70	20	30	40
50	20	100	10

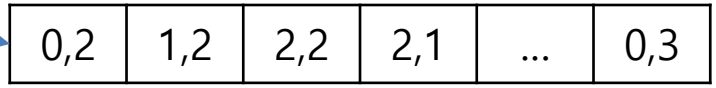
인구 이동이 없으면 중단

인구이동




```
static void bfs(int x, int y){
   int xx, yy;
   f = r = -1;
   qx[++r] = x; qy[r] = y;
   visited[x][y] = true;

   while (f != r){
      x = qx[++f]; y = qy[f];

      for (int i = 0; i < 4; i++){
         xx = x + dx[i];
         yy = y + dy[i];

         if (xx < 0 || xx >= N || yy < 0 || yy >= N) continue;

         int t = Math.abs(mat[x][y] - mat[xx][yy]);

         if (!visited[xx][yy] && L <= t && t <= R){
             visited[xx][yy] = true;
             qx[++r] = xx; qy[r] = yy;
         }
      }
  }

…
}
```

이동한 정점 큐에 저장되어 있음



조건에 맞으면 이동

인구이동

```
static void bfs(int x, int y){

…

    if (r > 0){
        int tsum = 0;
        for (int i = 0; i <= r; i++)
            tsum += mat[qx[i]][qy[i]];

        for (int i = 0; i <= r; i++)
            mat[qx[i]][qy[i]] = tsum / (r + 1);
        moved = true;
    }
}
```

10	100	20	90
80	100	60	70
70	20	30	40
50	20	100	10

10	100	50	50
50	50	50	50
50	50	50	50
50	50	100	50

0,2	1,2	2,2	2,1	…	0,3

인구 이동 있었음