

백준 1197, 최소 스패닝 트리

step 1 면적을 계산

NET: 흰 스파츌로의 대역
WCF: B6 각성 강우 블록간 Vector

Sum: 4000 원 가격

2008-2009 프리미엄 학기 가을학기

vec	1	2	1	3	2	3
	1		3		2	

NST	1	2	3
Graph	(1)	(2)	(3)

Step 2 모든 칸선을 가로막 으로 치우치게 정렬,

vec	1	2	2	3	1	3
	1		2		3	

MST	1	2	3	Graph
	(1)	(2)	(3)	

Step 3 모든 Vec토리에서 정점 1과 2를 Union-Find 알고리즘으로 연결하는
과정, 단위 / → 2개 정점 1과 연결

vec	1	2	2	3	1	3
	1		2		3	

NST

2	2	3
---	---	---

Graph

sum21

Union-Find 06.22(3)

34 [Find \(2nd\)](#) < 0.9 3E-32

(ATT TIME LIMITS) 8 AM 42

if ($MST[\alpha] == \alpha$)
 f

```
    return MST[i];  
}  
return MST[st = find(MST[st])]
```

• [View Details](#)

```

    void Union(LIST &a, LIST &b)
{
    a = find(a);
    b = find(b);
    if (a != b) // a와 b가 서로 다른 경우

```

Step 4 그 다음, VCC에서 나오는 두 출력은 Union-Find 알고리즘의 두 연결 요소

vec	1	2	2	3	1	3
	1		2		3	

MST	3	3	3
-----	---	---	---

Graph

step 4 마우스 우클릭 후 [x] 버튼 클릭(과 함께 드롭다운 메뉴로 접근하기 가능)

vec	1	2	2	3	1	3
.	1	.	2	.	3	.

MST	3	3	3
-----	---	---	---

Graph

백준 1516, 토마토

step1 번역 초기화
day: 오늘도가 목은데 걸리는 148, 053 3334
box: 흐기 토마토상회, 품목 알면
9: 정식을 토마토 쪽을 찾는 문제

box	0	-1	0	0	0	0
	-1	0	0	0	0	0
	0	0	0	0	0	0
	0	0	0	0	0	1

<i>day</i>	0	0	0	0	0
	0	0	0	0	0
	0	0	0	0	0
	0	0	0	0	0

9

--	--	--	--	--	--	--

step2 box 뒤에를 확인하여 1인 경우 차포를 1개 push

box	0	-1	0	0	0	0
	-1	0	0	0	0	0
	0	0	0	0	0	0
	0	0	0	0	0	1
	(5,3)					

day	0	0	0	0	0	0
	0	0	0	0	0	0
	0	0	0	0	0	0
	0	0	0	0	0	0

9 (5,35)

Step3 9에서 차트를 꺼내어 출판권을 갖추
box 끝에 예전것을 빼고 깊은 10 번정
수로, 아래에 출판권 + 1을 표시한다.

box	0	-1	0	0	0	0
	-1	0	0	0	0	0
	0	0	0	0	0	$(-1, 2)$
	0	0	0	0	1	$(1, -2)$
	0	0	0	0	$(-1, 3)$	$(3, 3)$

day	0	0	0	0	0	0
	0	0	0	0	0	0
	0	0	0	0	0	1
	0	0	0	0	1	0

9	(4,3)	(5,2)	•	•	•	•	•
4	front : (5,2)						

Step4 Step3을 반복 (99 번째로 가 0를 떠나기)

box	0	-1	0	0	0

-1	0	0	0	0	0

0	0	0	0	1	1
	.	.	.	(4,2)	(4,1)
0	0	0	1	1	1
	.	.	.	(3,3)	(3,2)
0	0	0	1	1	1
	.	.	.	(2,3)	(2,2)

day	0	0	0	0	0	0
	0	0	0	0	0	0
	0	0	0	0	2	1
	0	0	0	2	1	0

q. $\text{front} = (4, 3)$

Step5 허용률을 높이기 위해 첫 번째 `bias` 값이 0인 것은 찾을 수 없는 곳이므로 이를 허용률을 높여 두 번째 `day`를 찾을 때 찾을 확률

box	0	→				
-1						

day	0	0	6	5	4	3
	0	6	5	4	3	2
	6	5	4	3	2	1
	5	4	3	2	1	0

백준 2178, 미로 탐색

Step 1 맵과 초기화

- 이미지: 현재 칸의 상태를 같은 배열(0=벽, 1=통로, 2=출구, 3=목표)으로 표시하는 배열
- dist: 시작 위치에서 최단거리를 저장하는 배열(0으로 초기화)
- q: 찾을 수 있는 경로 좌표를 담은 Queue

arr	<table border="1"> <tr><td>1</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>0</td><td>1</td><td>1</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>0</td><td>1</td><td>1</td></tr> </table>	1	0	1	1	1	1	1	0	1	0	1	0	1	0	1	0	1	1	1	1	1	0	1	1	dist	<table border="1"> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> </table>	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	1	1	1																																														
1	0	1	0	1	0																																														
1	0	1	0	1	1																																														
1	1	1	0	1	1																																														
0	0	0	0	0	0																																														
0	0	0	0	0	0																																														
0	0	0	0	0	0																																														
0	0	0	0	0	0																																														

q	<table border="1"> <tr><td> </td><td> </td><td> </td><td> </td><td> </td><td> </td></tr> </table>						

Step 2 시작 좌표 L(1,1)을 큐에 push 한다.

제일 먼저 이미 찾자고 했던 것이다.

arr	<table border="1"> <tr><td>1</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>0</td><td>1</td><td>1</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>0</td><td>1</td><td>1</td></tr> </table>	1	0	1	1	1	1	1	0	1	0	1	0	1	0	1	0	1	1	1	1	1	0	1	1	dist	<table border="1"> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> </table>	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	1	1	1																																														
1	0	1	0	1	0																																														
1	0	1	0	1	1																																														
1	1	1	0	1	1																																														
0	0	0	0	0	0																																														
0	0	0	0	0	0																																														
0	0	0	0	0	0																																														
0	0	0	0	0	0																																														
		push	<table border="1"> <tr><td>1</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>0</td><td>1</td><td>1</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>0</td><td>1</td><td>1</td></tr> </table>	1	0	1	1	1	1	1	0	1	0	1	0	1	0	1	0	1	1	1	1	1	0	1	1																								
1	0	1	1	1	1																																														
1	0	1	0	1	0																																														
1	0	1	0	1	1																																														
1	1	1	0	1	1																																														

q	<table border="1"> <tr><td>1</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td></tr> </table>	1	0	1	1	1	1
1	0	1	1	1	1		

Step 3 arr[1][1]가 찾자고 했던 것이다. dist[1][1] += 1를 한다.

이제 큐에 pop한다.

arr	<table border="1"> <tr><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>0</td><td>1</td><td>1</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>0</td><td>1</td><td>1</td></tr> </table>	0	0	1	1	1	1	1	0	1	0	1	0	1	0	1	0	1	1	1	1	1	0	1	1	dist	<table border="1"> <tr><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>2</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> </table>	1	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1																																														
1	0	1	0	1	0																																														
1	0	1	0	1	1																																														
1	1	1	0	1	1																																														
1	0	0	0	0	0																																														
2	0	0	0	0	0																																														
0	0	0	0	0	0																																														
0	0	0	0	0	0																																														
		push	<table border="1"> <tr><td>1</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>0</td><td>1</td><td>1</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>0</td><td>1</td><td>1</td></tr> </table>	1	0	1	1	1	1	1	0	1	0	1	0	1	0	1	0	1	1	1	1	1	0	1	1																								
1	0	1	1	1	1																																														
1	0	1	0	1	0																																														
1	0	1	0	1	1																																														
1	1	1	0	1	1																																														

q	<table border="1"> <tr><td>1</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td></tr> </table>	1	0	1	1	1	1
1	0	1	1	1	1		

Step 4 q가 모두 비었을 때까지 Step 3를 반복 한다.

arr	<table border="1"> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> </table>	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	dist	<table border="1"> <tr><td>1</td><td>0</td><td>9</td><td>10</td><td>11</td><td>12</td></tr> <tr><td>2</td><td>0</td><td>8</td><td>0</td><td>12</td><td>0</td></tr> <tr><td>3</td><td>0</td><td>7</td><td>0</td><td>13</td><td>14</td></tr> <tr><td>4</td><td>5</td><td>6</td><td>0</td><td>14</td><td>15</td></tr> </table>	1	0	9	10	11	12	2	0	8	0	12	0	3	0	7	0	13	14	4	5	6	0	14	15
0	0	0	0	0	0																																														
0	0	0	0	0	0																																														
0	0	0	0	0	0																																														
0	0	0	0	0	0																																														
1	0	9	10	11	12																																														
2	0	8	0	12	0																																														
3	0	7	0	13	14																																														
4	5	6	0	14	15																																														

q	<table border="1"> <tr><td> </td><td> </td><td> </td><td> </td><td> </td><td> </td></tr> </table>						

백준 1753, 최단거리

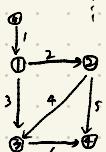
기워드
다익스트라 알고리즘
크스피팅

정의

5(경점 A) 6(경점 B)

1(시작점)

5(경점 A) 1(경점 B) 1(경점 C)



Step 1 최단거리를 나타내는 dist 배열을 무한대로 초기화

INF	INF	INF	INF	INF
1	2	3	4	5

Step 2 dist[시작점, 1] 을 0으로 초기화

0	INF	INF	INF	INF
1	2	3	4	5

Step 3 dist[] 가 1→2 거리를 따른 값을 빼서 dist[2] 와 비교하여 작은 값으로 업데이트
예제, 1을 경유해 5를 때 5(4번선)을 위해 3(2번선)에 1은 1→2를 push 한다.

0	2	INF	INF	INF
1	2	3	4	5

Step 4. 경유 점과 D이 될 때까지 Step 3를 반복

0	2	3	7	INF

백준 11675, 태일마인

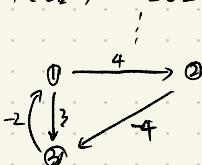
기워드

백만·포도 알고리즘

정의

3(경점 A) 4(경점 B)

1(경점 A) 2(경점 B) 4(경점 C)



Step 1 dist 배열 (1부터 각 경점 까지 걸리는 시간)을 무한대로 초기화

INF	INF	INF
1	2	3

Step 2 dist[시작점, 1] 을 0으로 초기화

0	INF	INF
1	2	3

Step 3 경선속도 × 경점 수 만큼 가중치를 더한 값을 적은 값으로 계속 업데이트
본 dist가 INF는 경우 업데이트 개선되어 처리

0	INF	INF
1	2	3

 →

0	9	INF
1	2	3

 →

0	4	3
1	2	3

Step 4 마지막으로 경점 4 번째를 엘리미트 하는데 기본값보다 적어지면 사이클이 존재

0	4	3

 ⇒

0	0	3

같이 처리되었으므로 사이클을 존재