이산수학

휴먼지능정보공학전공

학습 내용

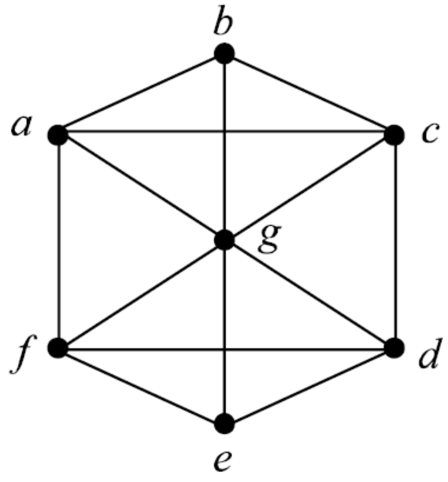
- 그래프의 기본 개념
- 그래프 용어
- 그래프 종류
- 그래프 표현
- 그래프 탐색
- 그래프 응용

기본개념

문제

- 다음 그래프(G)를 만족하는 집합V, 집합 E를 구하세요

- $G=(V,E)$
- $V=\{a,b,c,d,e,f,g\}$
- $E=\{(a,b),(a,c),(a,f),(a,g),(b,c),(b,g),(c,d),(c,g),(d,e),(d,f$
- $(d,g),(e,f),(e,g),(f,g)\}$



기본개념

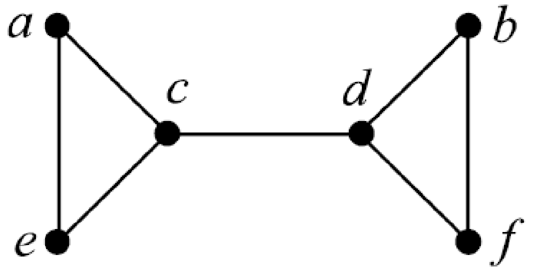
- 그래프 종류
  - 방향 그래프(directed graph 또는 digraph)
    - 방향이 있는 그래프임
    - 연결선을 화살표로 표시하여 방향을 나타내는 그래프임
    - G= 〈V,E〉
    - V={1, 2, 3, 4}
    - E={〈1, 2〉, 〈2, 3〉, 〈2, 4〉, 〈3, 4〉, 〈4,2〉}
  - 방향이 없는 그래프(undirected graph)
    - 방향이 없는 그래프임
    - 그래프의 특수한 형태이므로 특별한 언급이 없는 한 그래프는 방향이 없는 그래프를 의미함
    - G=(V,E)
    - V={1, 2, 3, 4}
    - E={(1, 2), (2, 3), (2, 4),(3,4),(4,2)}

기본개념

- 그래프
  - 길
    - 그래프에서 꼭짓점 v_i와 v_{i+1}을 연결하는 변을 e_i라고 했을 때, v_1, e_1, v_2, e_2, …, e_{k-1}, v_k, e_k, v_{k+1} 또는 $v_1 \rightarrow v_2 \rightarrow \ldots \rightarrow v_{k-1} \rightarrow v_k$
    - v_1 에서 시작해서 v_k에 도착하는 꼭짓점과 변의 나열
  - 경로
    - 모든 1≤i <k에 대해 연결선 (vi , vi + 1)이 존재할 때, 정점들의 열(sequence) v1, v2, v3, ···, vk라고 함
    - k ≥ 1이며 이 경로의 길이는 k - 1임
    - 같은 연결선(변)을 두 번 이상 포함하지 않는 길
  - 순환(사이클)
    - V1 = 파 (k ≠ 1)이면 이러한 경로를 사이클이라고 함
  - 길이
    - 경로 또는 순환을 구성하는 연결선(변)의 수

기본개념

문제



- 다음 그래프(G)를 보고 물음에 답하세요
- 1) a에서f까지의 경로를 모두 찾아라.
- (2) a 에서 시작하는 길이가 5인 경로를 2개 찾아라.
- (3) a 에서 시작하는 회로를 모두 찾아라.
- (4) a 에서 시작하여 b 로 끝나는 경로 중 길이가 가장 짧은 경로는 무엇인가?

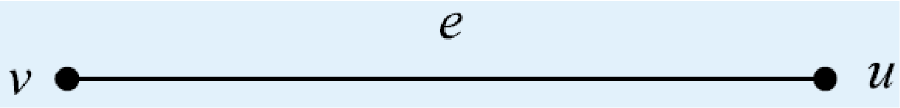
- (1) $a-c-d-f$ $a-c-d-b-f$
- $a-e-c-d-f$ $a-e-c-d-b-f$
- (2) $a-e-c-d-b-f$ $a-e-c-d-f-b$
- (3) $a-c-e-a$ $a-e-c-a$
- (4) $a-c-d-b$(길이:3) $a-c-d-f-b$(길이 :4)
- $a-e-c-d-b$(길이:4) $a-e-c-d-f-b$(길이 :5)
- $\therefore$ 길이가 3인 경로가 가장 짧으므로, $a-c-d-b$

기본개념

- 트리
  - 사이클(cycle)이 존재하지 않는 그래프임
  - 루트(root)라 불리는 특별한 노드가 한 개 존재하고, 루트로부터 다른 모든 노드로 가는 경로가 항상 유일하게 존재함
  - 루트로 들어오는 연결선이 없으므로 루트는 모든 트리의 출발점이 됨

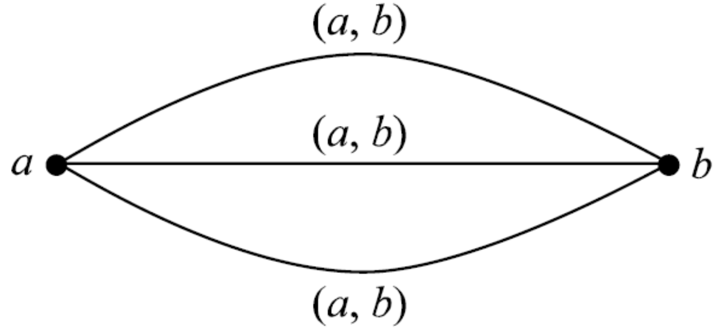
그래프 용어

- 단순그래프
  - 한 쌍의 정점 사이에 많아도 하나의 연결선으로 이루어진 그래프
  - 자기 자신으로 연결선이 없는 그래프



그래프 용어

- 다중(멀티)그래프
  - 단순 그래프의 확장형
  - 한 쌍의 꼭지점 사이에 연결선의 개수의 제한이 없는 일반적인 그래프



그래프 용어

- 그래프 연결선
  - 그래프 G=(V,E)에서 순서화 된 쌍 E를 그래프의 연결선
  - (u,v) ∈ E 일 때 u와 v를 연결하는 연결선 e는 u와 v에 접했다
  - u와 v가 서로 인접했다

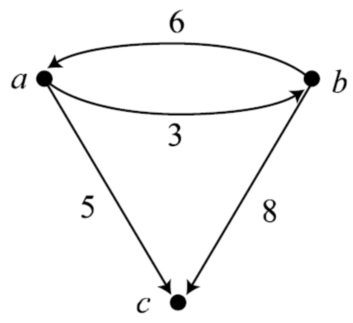
그래프 용어

- 가중치 그래프
  - 그래프 $G = (V, E)$ 에서 각 변에 가중치가 정의되어 있는 그래프
    - W[u,v] = n

그래프 용어

문제

- 다음 그래프(G)를 가중치를 구하세요



- $W[a,b]=3$ $W[b,a]=6$
- $W[a,c]=5$ $W[b,c]=8$

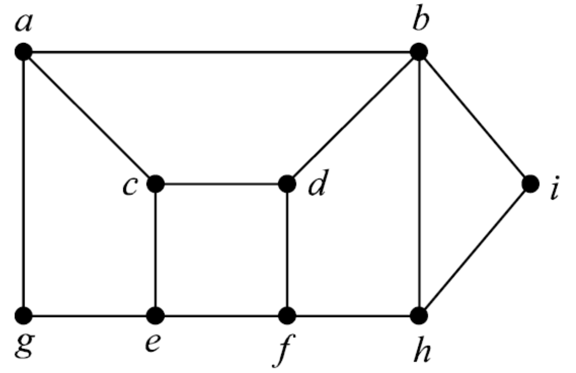
그래프 용어

- 그래프 차수
  - 그래프 G=(V,E)에서 v 가 꼭지점이라고 할 때 v의 차수는 d(v)
  - d(v)는 v에 인접하는 연결선들의 개수
    - 홀수점(Odd Vertex) : 차수가 홀수인 꼭짓점
    - 짝수점(Even Vertex) : 차수가 짝수인 꼭짓점
  - 외차수(Out-degree) $\boldsymbol{out-d(v)}$
    - 방향 그래프에서 꼭짓점 v를 시작으로 하는 화살표의 수
  - 내차수(In-degree) $\boldsymbol{in-d(v)}$
    - 방향 그래프에서 꼭짓점 v를 끝으로 하는 화살표의 수

그래프 용어

문제

- 다음 그래프에서 각 꼭짓점의 차수를 구하고 홀수점과 짝수점을 구하세요



- $d(a)$=3 $d(b)$=4 $d(c)$=3
- $d(d)$=3 $d(e)$=3 $d(f)$=3
- $d(g)$=2 $d(\hbar)$=3 $d(i)$=2
- 짝수점 : $b,\ g,i$ · 홀수점 : $a,c,d,e,f,\hbar$

그래프 용어

- 그래프 차수
  - 그래프 $G=(V,E)$에서 모든 꼭짓점의 차수의 합은 변 수의 두 배다.

$$\sum_{v \in V} d(v) = 2|E|$$

  - 그래프 $G=(V,E)$에서 차수가 홀수인 꼭짓점의 수는 짝수다.

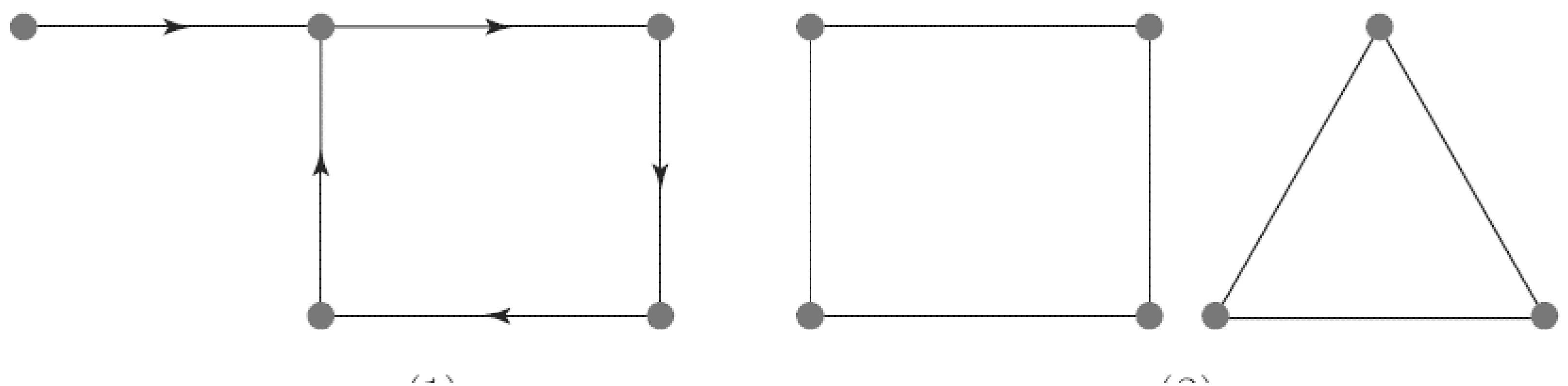
그래프 용어

- 연결 그래프
  - 그래프의 모든 정점들이 연결되어 있는 그래프
  - 그래프 *G=(V,E)* 내에 있는 임의의 꼭짓점 *u,v* 간에 경로가 있는 그래프
- 강한 연결 그래프
  - 그래프에서 모든 두 정점 a와 b에 대해서 a에서 b로의 경로와 b에서 a로의 경로들이 존재하는 그래프를 말하는데, 특히 방향 그래프에서만 의미를 가짐
- 연결 요소
  - 그래프에서 모든 정점들이 연결되어 있는 부분을 말하며, 연결 수(connectivity number)란 그래프 G에서의 연결 요소의 개수를 말함

그래프 용어

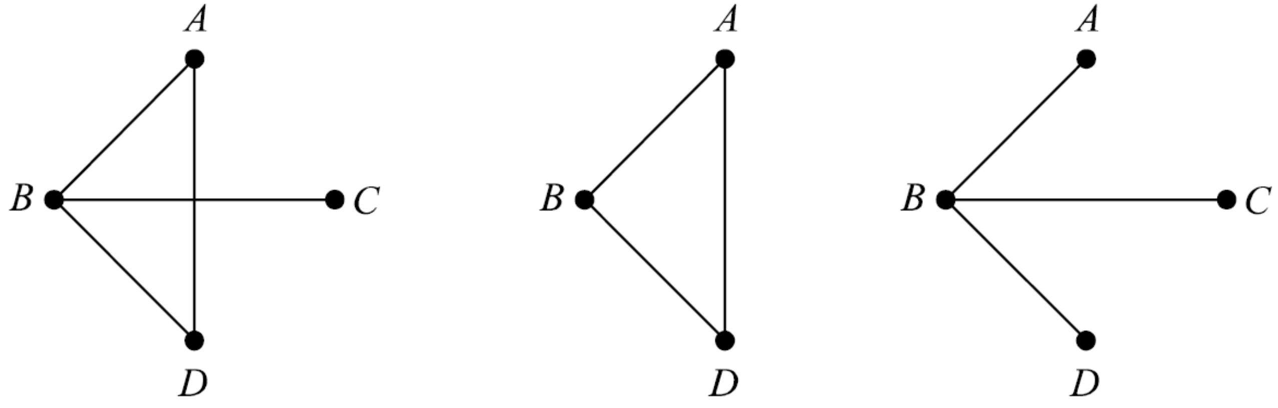
문제

- 다음 그래프에서 연결 그래프를 판단하세요



그래프 종류

- 부분 그래프
  - 두 개의 그래프 G=(V,E), G′ = (V′, E′) 에서 V′ ⊆V, E′ ⊆E 일 때 그래프 G′= (V′,E′)를 G의 부분그래프
- 부분 생성 그래프
  - 두 개의 그래프 G=(V,E), G′ = (V′, E′) 에서 V′=V 이고 E′ ⊂ E 일 때 그래프 G′ = (V′, E′)를 G의 생성 부분 그래프



G 그래프 G의 부분 그래프 G의 부분신장 그래프

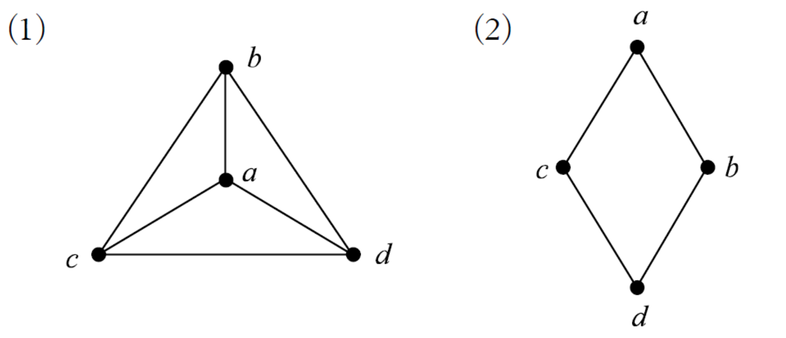
그래프 종류

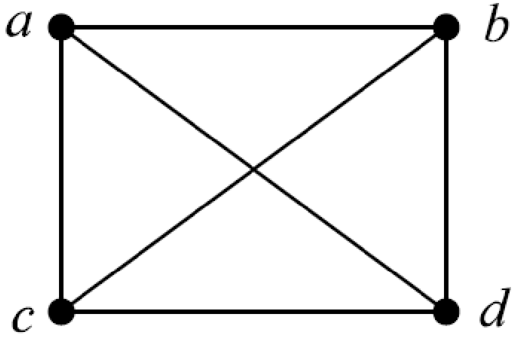
- 동형 그래프
  - 그래프 $G = (V, E)$와 $G' = (V', E')$ 에 대해 함수 $f: V \rightarrow V'$가 $u, v \in V$에 대해 $(u, v) \in E \ \Leftrightarrow (f(u), f(v)) \in E'$인 전단사함수 일때 그래프 $G = (V, E)$와 $G' = (V', E')$ 는 동형 그래프

그래프 종류

문제

- 다음 그래프에서 동형 그래프를 판단하세요





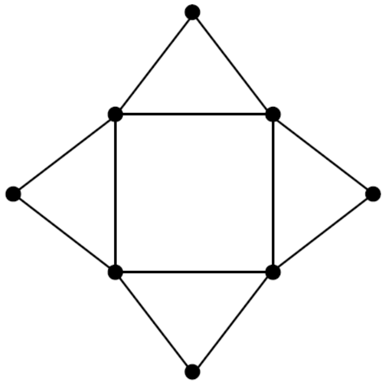
- $G_{(1)}=(V_{(1)},E_{(1)})$
- $V_{(1)}=\{a,b,c,d\}$
- $E_{(1)}=\{(a,b),(a,c),(a,d),(b,c),(b,d),(c,d)\}$
- $V=V_{(1)}$ 이고 $E=E_{(1)}$다.
- $\therefore$ $G=(V,E)$와 $G_{(1)}=(V_{(1)},E_{(1)})$ 은 동형 그래프다.

- (2) $G_{(2)}=(V_{(2)},E_{(2)})$
- $V_{(2)}=\{a,b,c,d\}$
- $E_{(2)}=\{(a,b),(a,c), (b,d),(c,d)\}$
- $V=V_{(2)}$ 지만 $E\neq E_{(2)}$다.
- $\therefore$ $G=(V,E)$와 $G_{(2)}=(V_{(2)},E_{(2)})$ 은 동형 그래프가 아니다

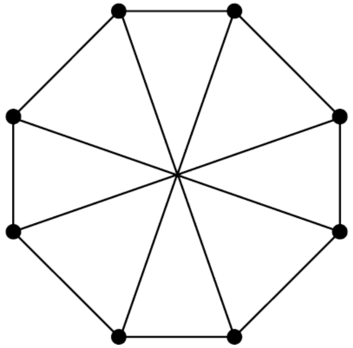
그래프 종류

- 평면 그래프
  - 그래프 $G=(V,E)$를 평면에 그릴 때, 꼭짓점이 아닌 곳에서는 어떤 변도 교차하지 않는 그래프

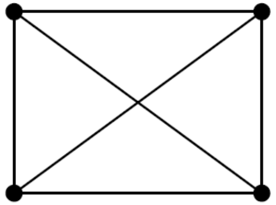
(1)



(2)



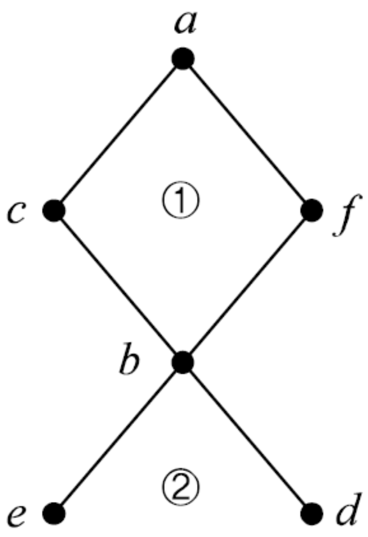
(3)



(4)

그래프 종류

- 면
  - 평면그래프에서만 존재
  - 평면 그래프는 변을 경계로 하여 하나 이상의 면으로 구성된다.

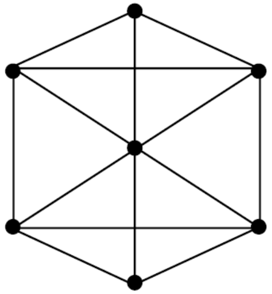


(a)

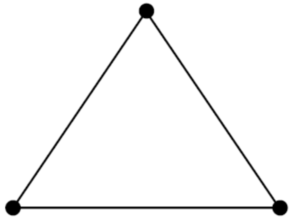
그래프 종류

- 완전 그래프(Complete Graph)
  - 그래프 $G = (V, E)$ 내에 있는 모든 꼭짓점 u, v간에 변이 있는 그래프로, n개의 꼭짓점을 가진 그래프는 K_n 으로 표기

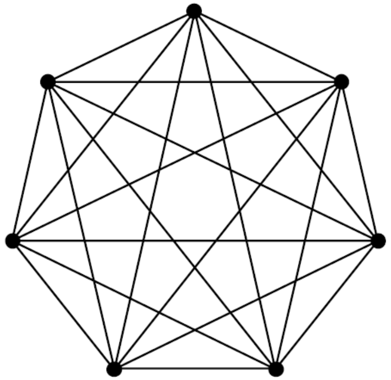
(1)



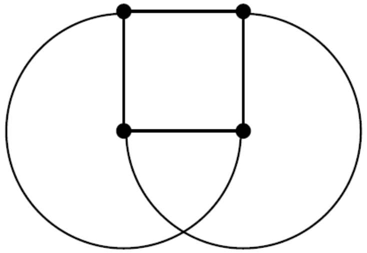
(2)



(3)

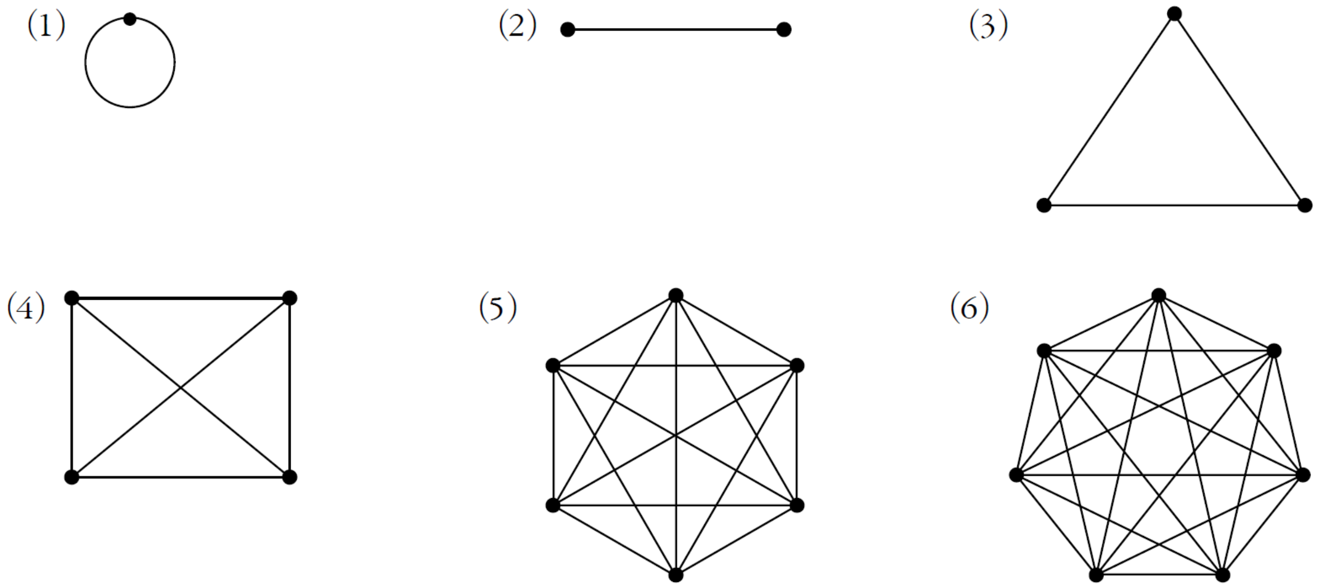


(4)



그래프 종류

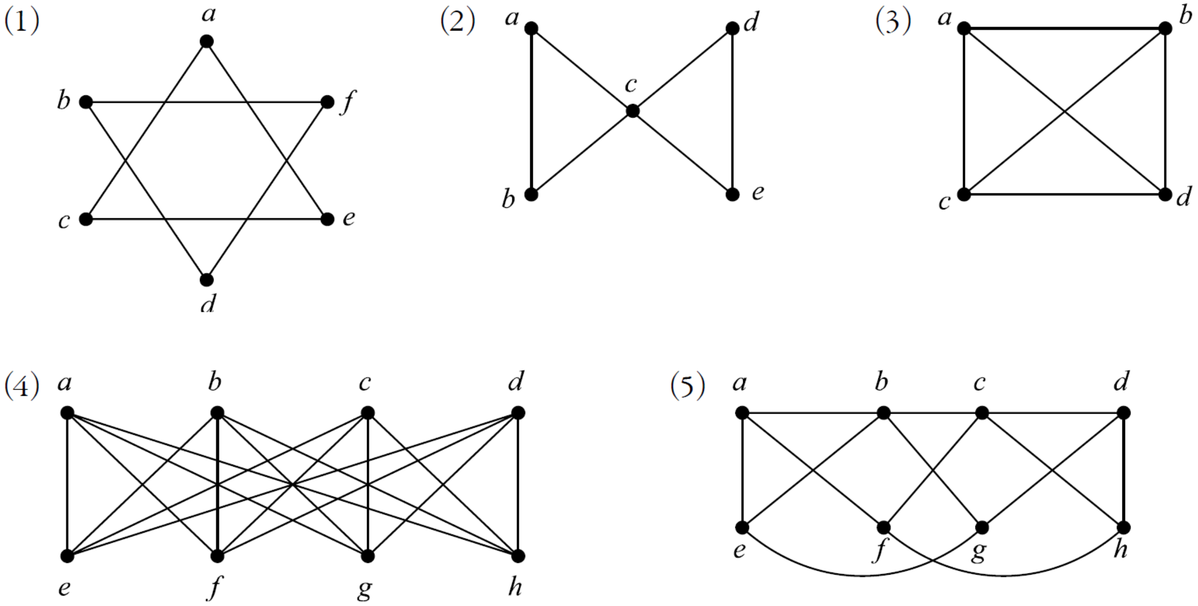
- 완전 그래프(Complete Graph)
  - 그래프 $G = (V, E)$ 내에 있는 모든 꼭짓점 u, v간에 변이 있는 그래프로, n개의 꼭짓점을 가진 그래프는 K_n 으로 표기



(1) K_1　(2) K_2　(3) K_3　(4) K_4　(5) K_6　(6) K_7

그래프 종류

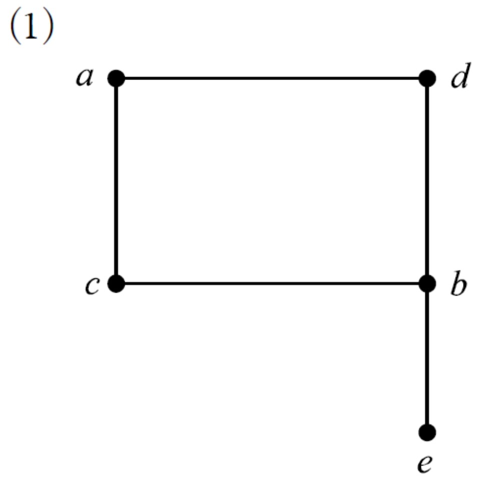
- 정규 그래프(Regular Graph)
  - 그래프 $G = (V, E)$ 내에 있는 모든 꼭짓점의 차수가 같은 그래프, 각 꼭짓점의 차수가 모두 k인 경우 $k-$정규 그래프로 표기

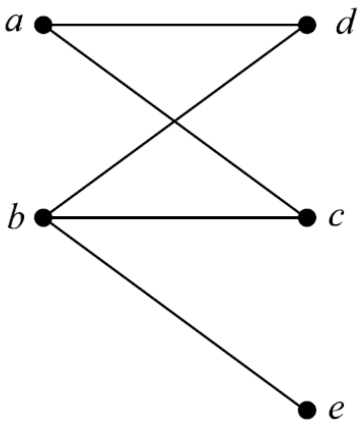


(1) 2 – 정규 (2) 정규 그래프가 아니다. (3) 3 – 정규 그래프 (4) 4 – 정규 그래프 (5) 정규 그래프가 아니다.

그래프 종류

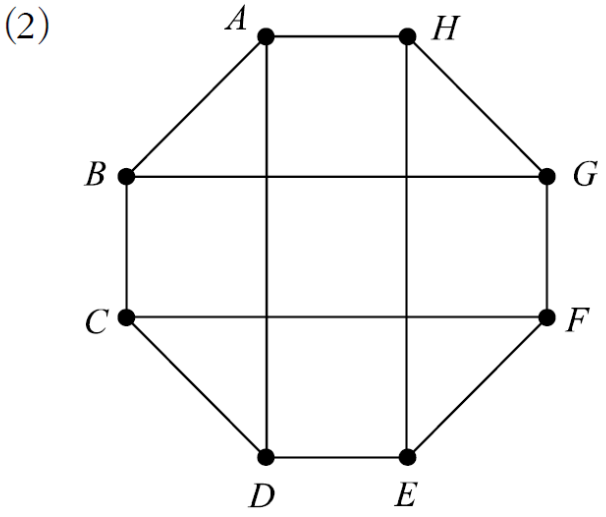
- 이분 그래프(Bipartite Graph)
  - 그래프 $G = (V, E)$ 에서 꼭짓점 집합 V 가 $V = V_1 \cup V_2$, $V_1 \cap V_2 = \emptyset$을 만족하는 두 집합 V_1과 V_2로 분리되고, 그래프의 모든 변이 V_1의 한 꼭짓점에서 V_2의 한 꼭짓점으로 연결되는 그래프

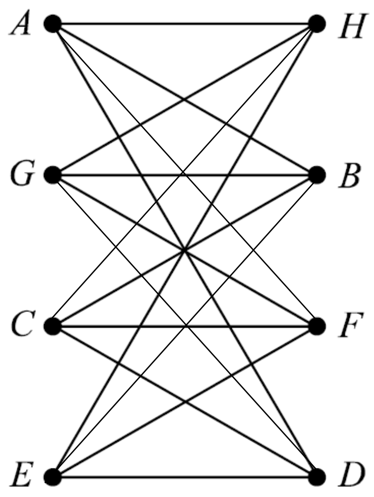




그래프 종류

- 완전 이분 그래프(Complete Bipartite Graph)
  - 이분 그래프 $G = (V, E)$ 에서 V_1의 모든 꼭짓점과 V_2의 모든 꼭짓점 사이에 변이 있는 그래프 $|V_1| = m$, $|V_2| = n$ 일때 $K_{m,n}$으로 표기





그래프 종류

- 오일러 공식에 대한 정리
  - 꼭짓점, 변, 면과의 관계 정리
  - 연결된 평면 그래프 G에서 꼭짓점의 수를v, 변의 수를e, 면의 수s를 라고 할 때 다음 오일러 공식이 성립
    - $v - e + s = 2$

그래프 종류

- 오일러 경로(Eulerian Path)
  - 그래프 $G=(V,E)$ 의 모든 변을 꼭 한 번씩 지나는 경로

- 오일러 회로 / 오일러 순환(Eulerian Circuit / Eulerian Cycle)
  - 그래프 $G=(V,E)$ 의 꼭짓점 v 에서 시작해 모든 변을 꼭 한 번씩 지나 v 로 돌아오는 회로

- 오일러 그래프(Eulerian Graph)
  - 오일러 회로를 포함하는 그래프 $G=(V,E)$

그래프 종류

- 오일러 그래프에 대한 정리
  - 연결 그래프 $G=(V,E)$ 의 모든 꼭짓점의 차수가 짝수일 때, 오일러 그래프의 필요충분조건이 된다.
  - 연결 그래프 $G=(V,E)$ 가 오일러 경로를 갖기 위한 필요충분조건은 그래프 G를 구성하는 꼭짓점 중 차수가 홀수인 꼭짓점의 수가 0 또는 2개인 것이다.

그래프 종류

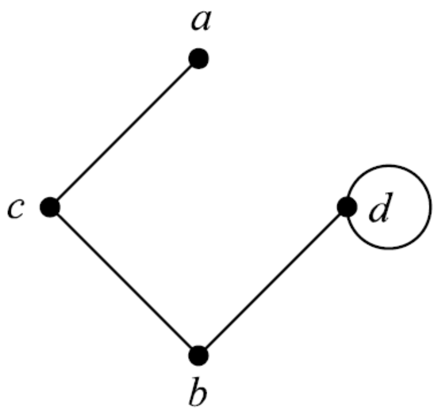
- 해밀턴 경로(Hamiltonian Path)
  - 그래프 $G=(V,E)$ 의 모든 꼭짓점을 꼭 한 번씩 지나는 경로

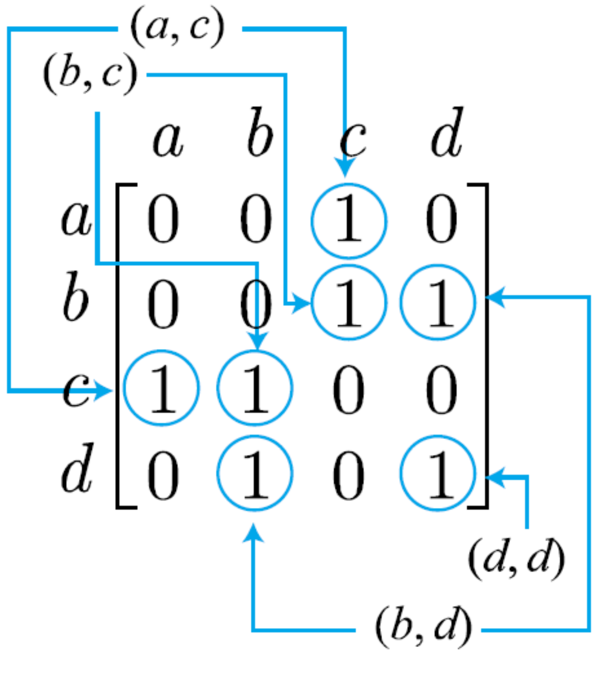
- 해밀턴 회로 / 해밀턴 순환(Hamiltonian Circuit / Hamiltonian Cycle)
  - 그래프 $G=(V,E)$ 의 꼭짓점 v 에서 시작해 모든 꼭짓점을 한 번씩만 지나 v 로 돌아오는 회로

- 해밀턴 그래프(Hamiltonian Graph)
  - 해밀턴 회로를 포함하는 그래프 $G=(V,E)$

그래프 표현

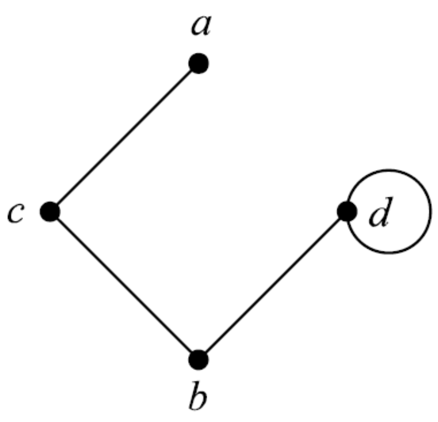
- 인접행렬(Adjacency Matrix)
  - 그래프 $G=(V,E)$에서 $|V|=n$ 일때 $n\times n$ 행렬로 나타내는 방법
  - 그래프 G에 대한 인접행렬 $A=[a_{ij}]$ 의 각 원소 $a_{ij}=\begin{cases}1 & (v_i,v_j)\in E\\ 0 & otherwise\end{cases}$

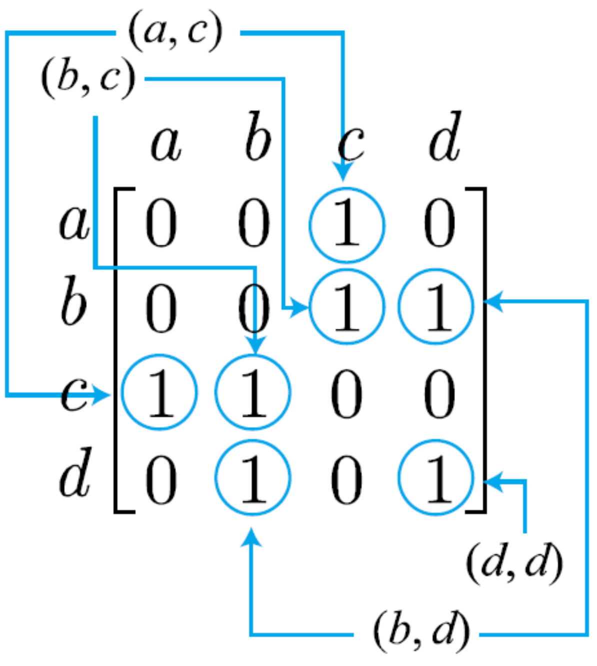


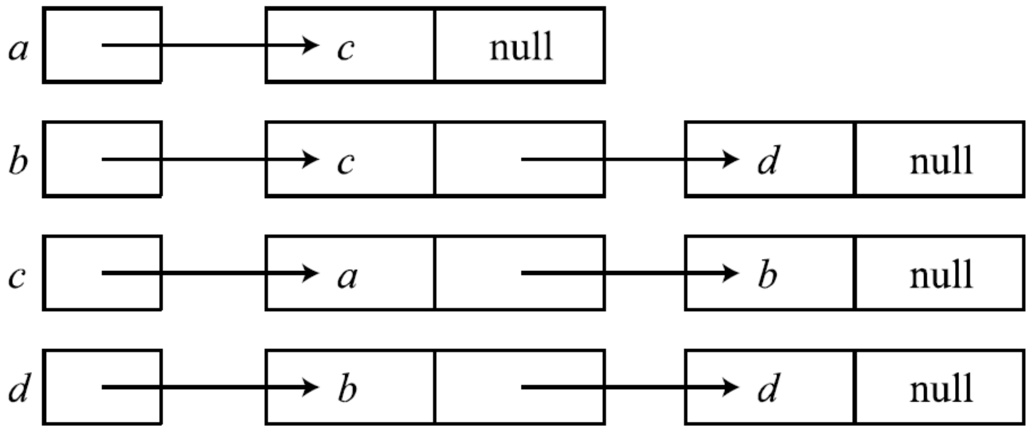


그래프 표현

- 인접리스트(Adjacency List)
  - 그래프 $G = (V, E)$를 구성하는 각 꼭짓점에 인접하는 꼭짓점들을 연결 리스트(Linked List)로 표현한 것







그래프 탐색

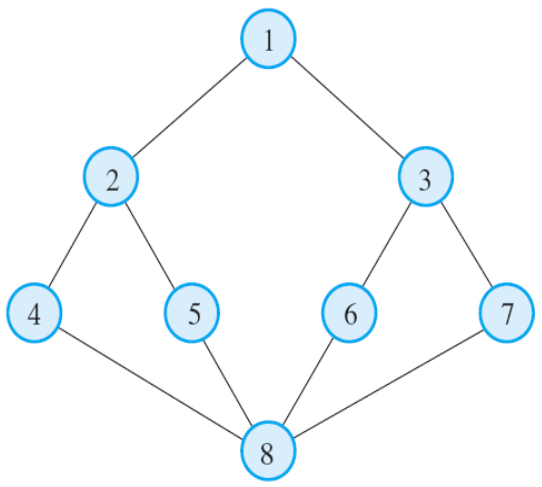
- 최단경로 문제(Shortest Path Problem)
  - $|E|$>0 인 그래프 G=(V,E)에서 꼭짓점 v_1, v_2∈V간의 가장 짧은 거리의 경로를 찾는 문제
    - 출발점(Source) : 경로의 시작점
    - 도착점(Destination) : 경로의 목적지

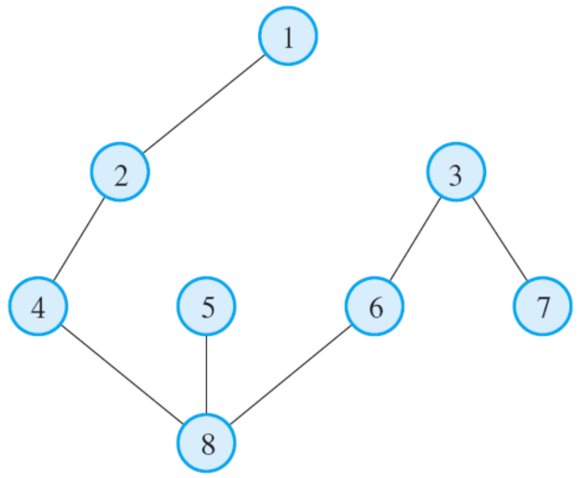
그래프 탐색

- 깊이 우선 탐색(DFS : Depth First Search)
  - 시작점 v_1에서 인접해 있는 꼭짓점 중 아직 탐색하지 않은 꼭짓점 v_2를 방문하고, 꼭짓점 v_2에 인접해 있는 꼭짓점 중 아직 탐색하지 않은 꼭짓점 v_3을 방문하는 것을 반복
    - (1) 시작점 v를 탐색한다.
    - (2) 꼭짓점 v 에 인접한 꼭짓점들 중 탐색되지 않은 꼭짓점 v_(s u b)를 탐색한다.
    - (3) 꼭짓점 v_(s u b)를 v 로 하여 (2)를 반복한다.
    - (4) 더 이상 탐색되지 않은 꼭짓점이 없으면 이전에 탐색한 꼭짓점을 v 로 하여
    - (2)와 (3)을 반복한다.
    - (5) 그래프의 모든 꼭짓점을 탐색할 때까지 반복한다.

그래프 탐색

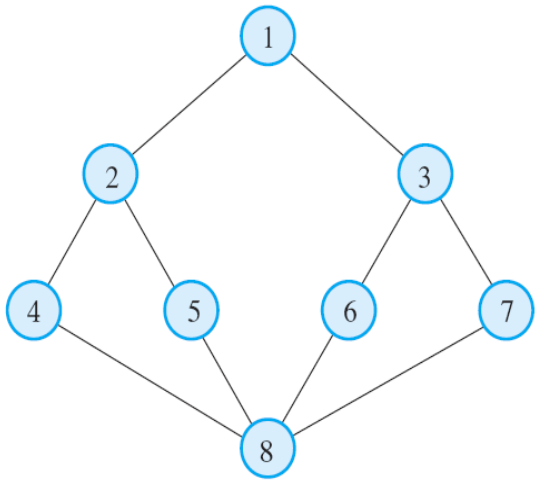
- 깊이 우선 탐색(DFS : Depth First Search)

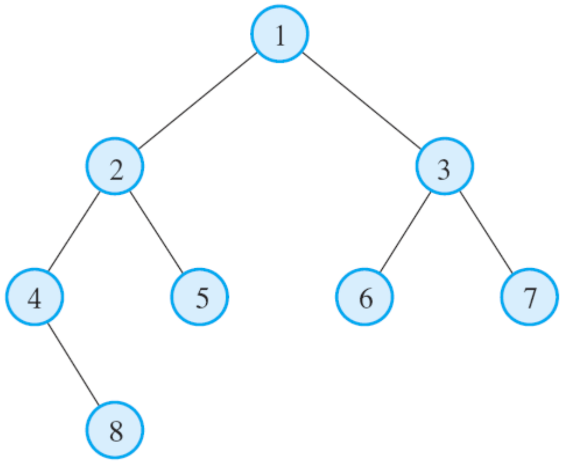




그래프 탐색

- 너비 우선 탐색(Breath First Search)
  - 시작점v_1로부터 인접한 꼭짓점 v_(2_1) ,…, v_(2_n) 을 모두 탐색하고, 다시 꼭짓점 v_(3_1) ,…, v_(3_m)을 시작으로 인접한 꼭짓점을 차례로 탐색하는 방식을 반복



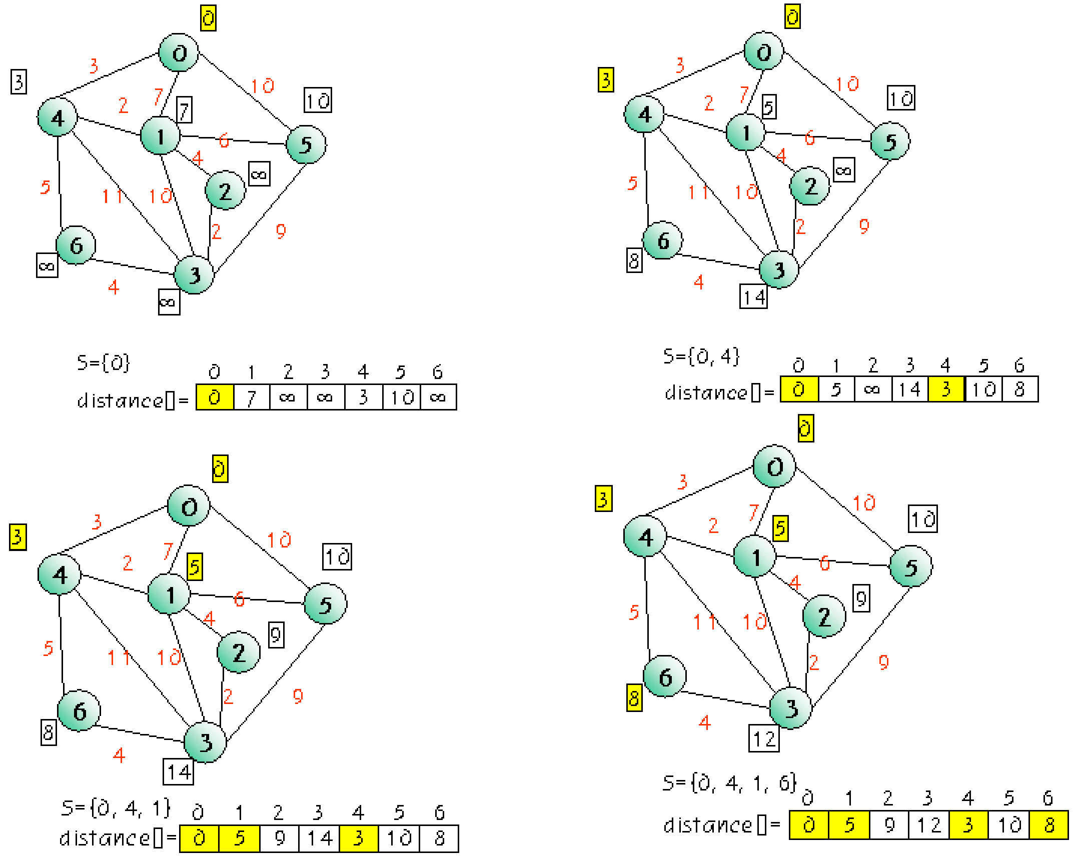


그래프 탐색

- 다익스트라 알고리즘(Dijkstra Algorithm)
  - 시작점으로부터 최단경로를 갖는 점들을 차례로 탐색하는 알고리즘
  - $G=(V,E)$에서 시작점이 v_1일 때 다음과 같이 표기한다.
  - $D[v\_i\ ]$: 시작점 v_1로부터 각 꼭짓점 v_i 의 최단경로
  - $C[v\_i,v\_i\ ]=0$: 꼭짓점 v_i 자신의 거리
  - $C[v\_i,v\_j\ ]=\infty$: 꼭짓점 $〚v〛\_i$ 와 v_j 간에 경로가 존재하지 않는 경우
  - $C[v\_i,v\_j\ ]=C\_(v\_i\ v\_j\ )$: 꼭짓점 v_i와 v_j 간에 경로가 존재는 경우의 거리

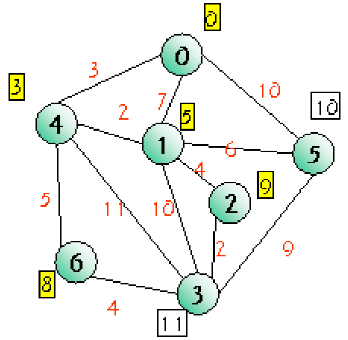
그래프 탐색

- 다익스트라 알고리즘(Dijkstra Algorithm)



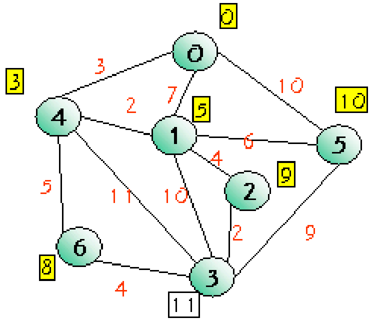
그래프 탐색

- 다익스트라 알고리즘(Dijkstra Algorithm)



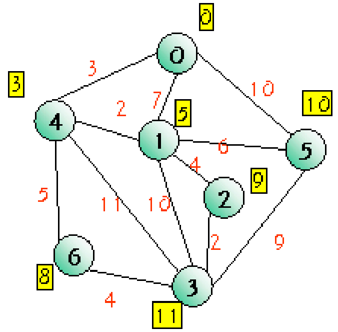
S={0, 4, 1, 6, 2}

	0	1	2	3	4	5	6
distance[]=	0	5	9	11	3	10	8



S={0, 4, 1, 6, 2, 5}

	0	1	2	3	4	5	6
distance[]=	0	5	9	11	3	10	8

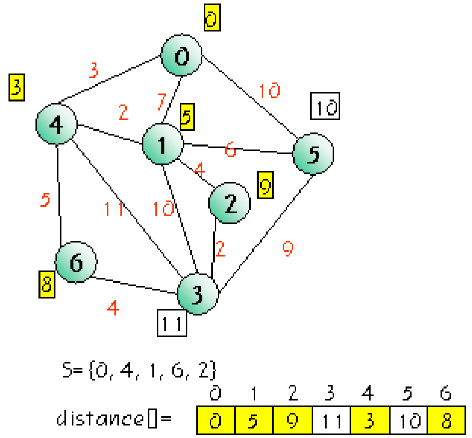


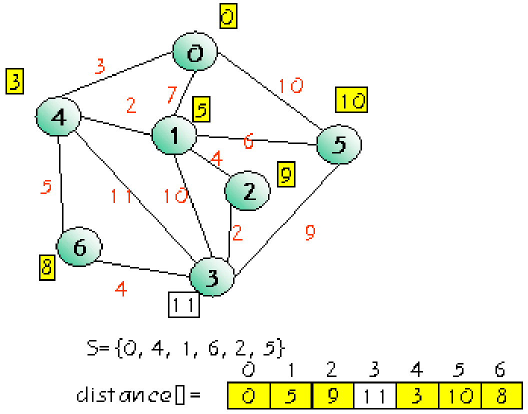
S={0, 4, 1, 6, 2, 5, 3}

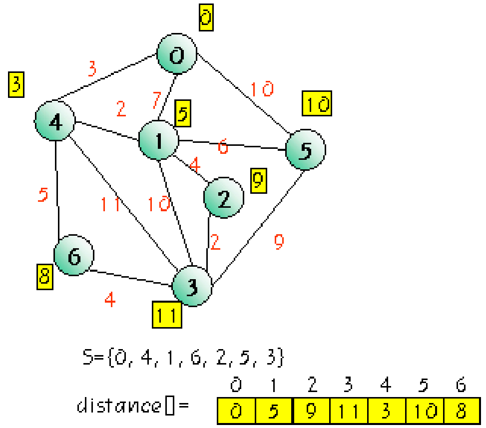
	0	1	2	3	4	5	6
distance[]=	0	5	9	11	3	10	8

그래프 탐색

- 다익스트라 알고리즘(Dijkstra Algorithm)

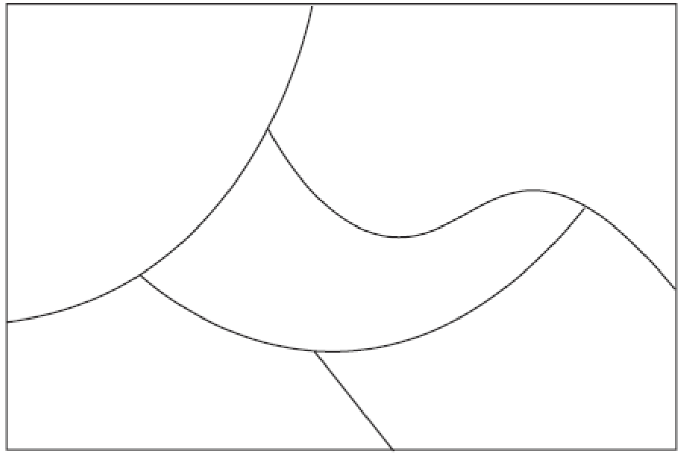


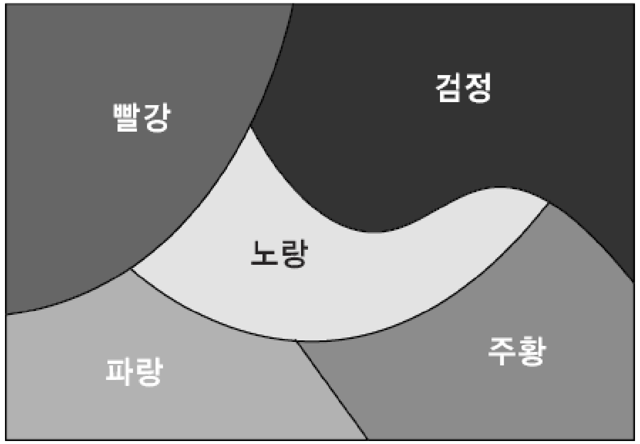


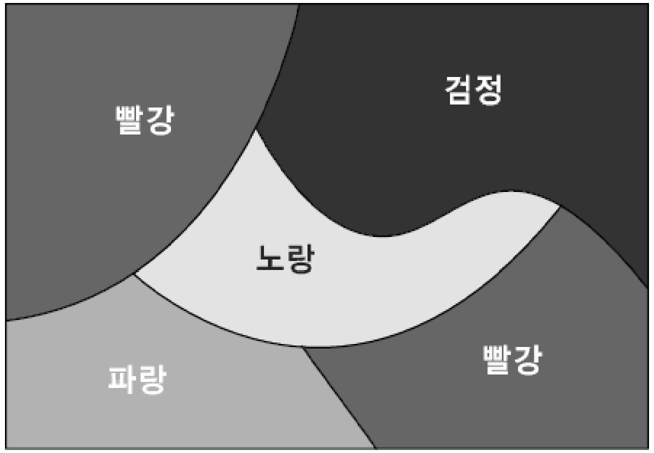


그래프 응용

- 그래프 색칠 문제
  - 어떤 주어진 그래프 G에 대해 인접한 어느 두 영역도 같은 색이 안되도록 각 정점에 색을 칠하는 문제
  - 그래프 G를 색칙하는데 필요한 최소한의 색의 수를 x(G)로 표현하고 색칠 수라고 한다

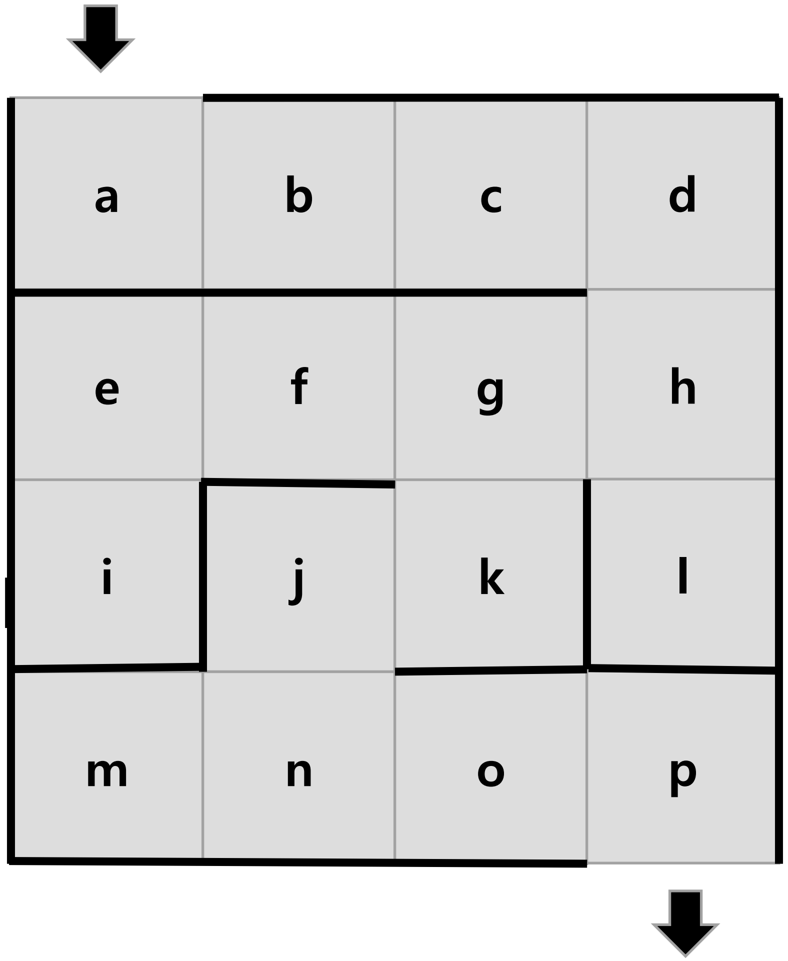






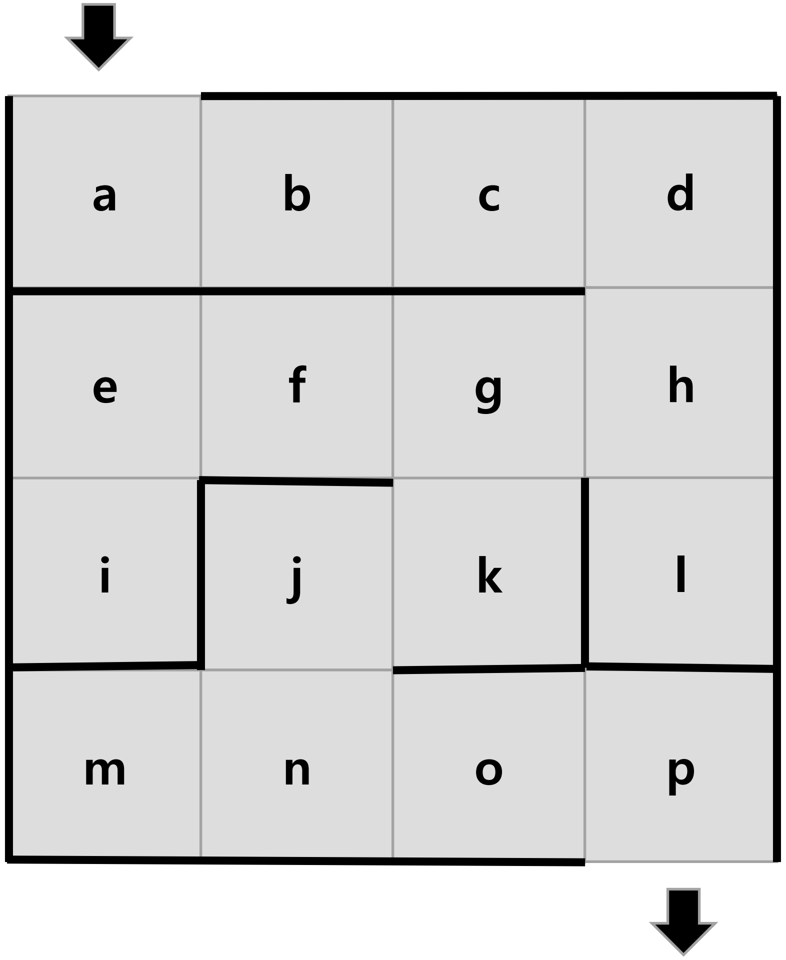
이산수학 – 문제해결

- 이산수학 – 실생활 문제해결 (미로찾기)
  - 문제인식
    - 출발점, 끝점, 경로
  - 문제해결 모델
    - 눈으로 찾기, 직접 그리기, 그래프, 트리
  - 문제해결 방법
    - 직접 붓그리기, 자료구조(리스트, 집합, 딕셔너리
  - 문제해결 시도
    - 붓 그리기, 코딩
  - 문제해결 결과
    - 경로



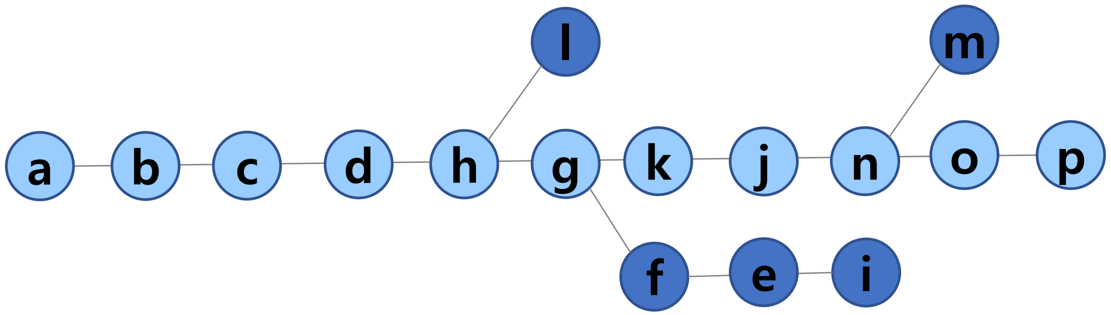
이산수학 – 문제해결

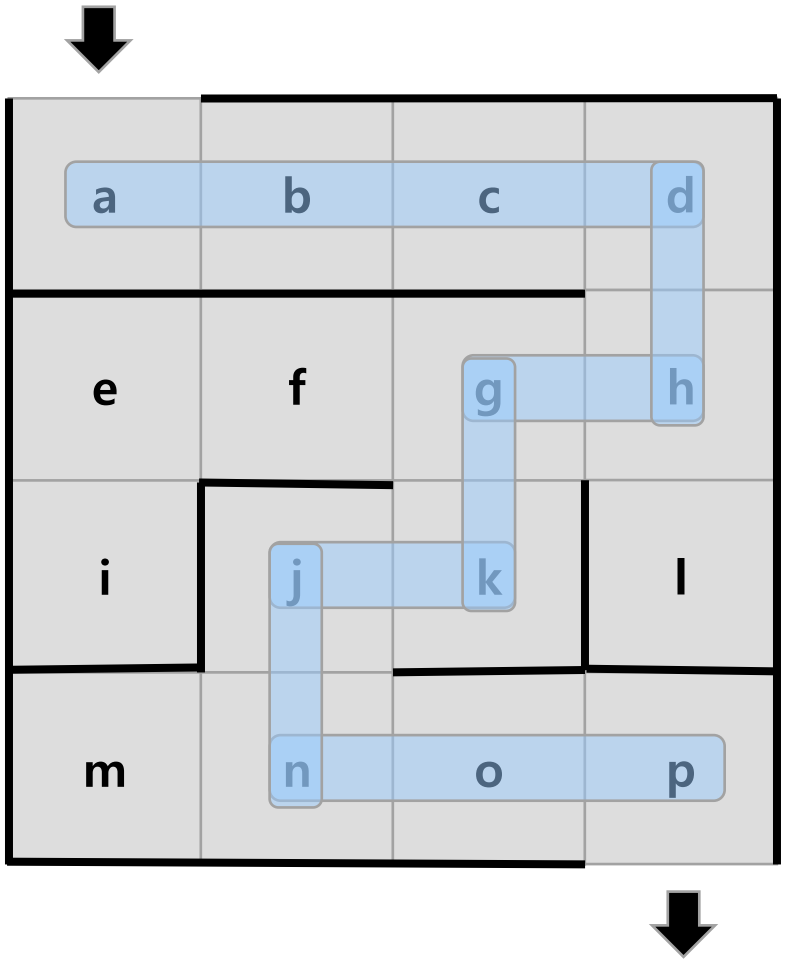
- 이산수학 – 실생활 문제해결 (미로찾기)
  - 문제인식
    - 출발점, 끝점, 경로
  - 문제해결 모델
    - 눈으로 찾기
    - a->b->c->d->h->g->k->j->n->o->p



이산수학 – 문제해결

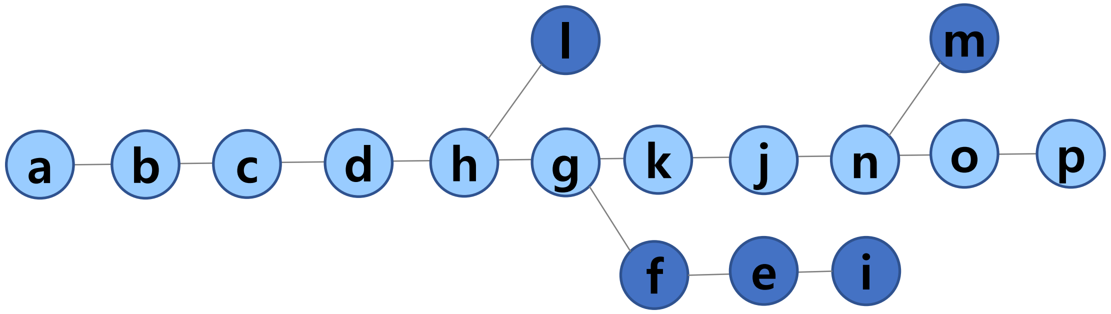
- 이산수학 – 실생활 문제해결 (미로찾기)
  - 문제인식
    - 출발점, 끝점, 경로
  - 문제해결 모델
    - 그래프, 트리 찾기

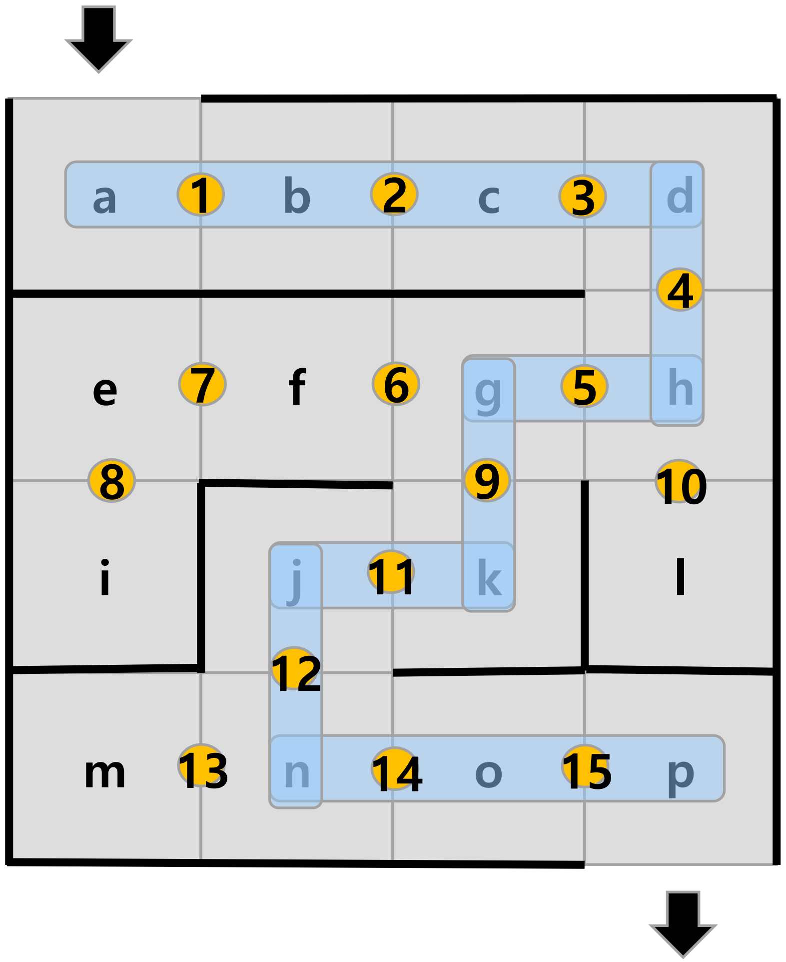




이산수학 – 문제해결

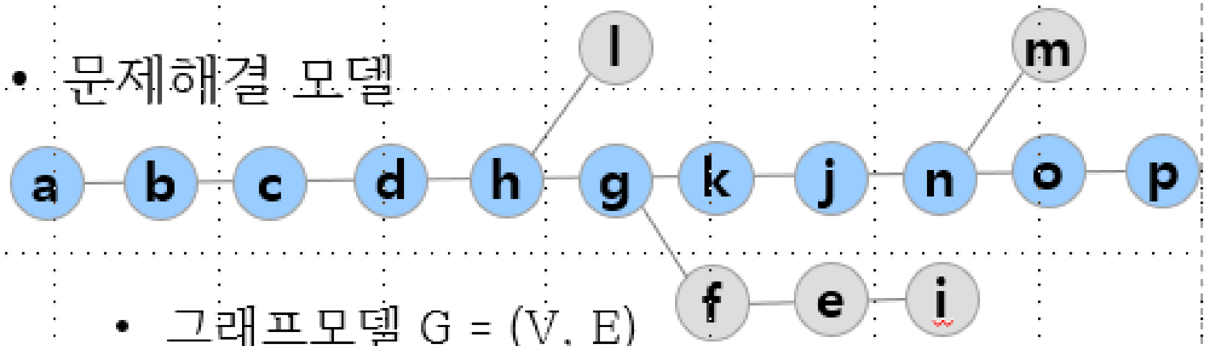
- 이산수학 – 실생활 문제해결 (미로찾기)



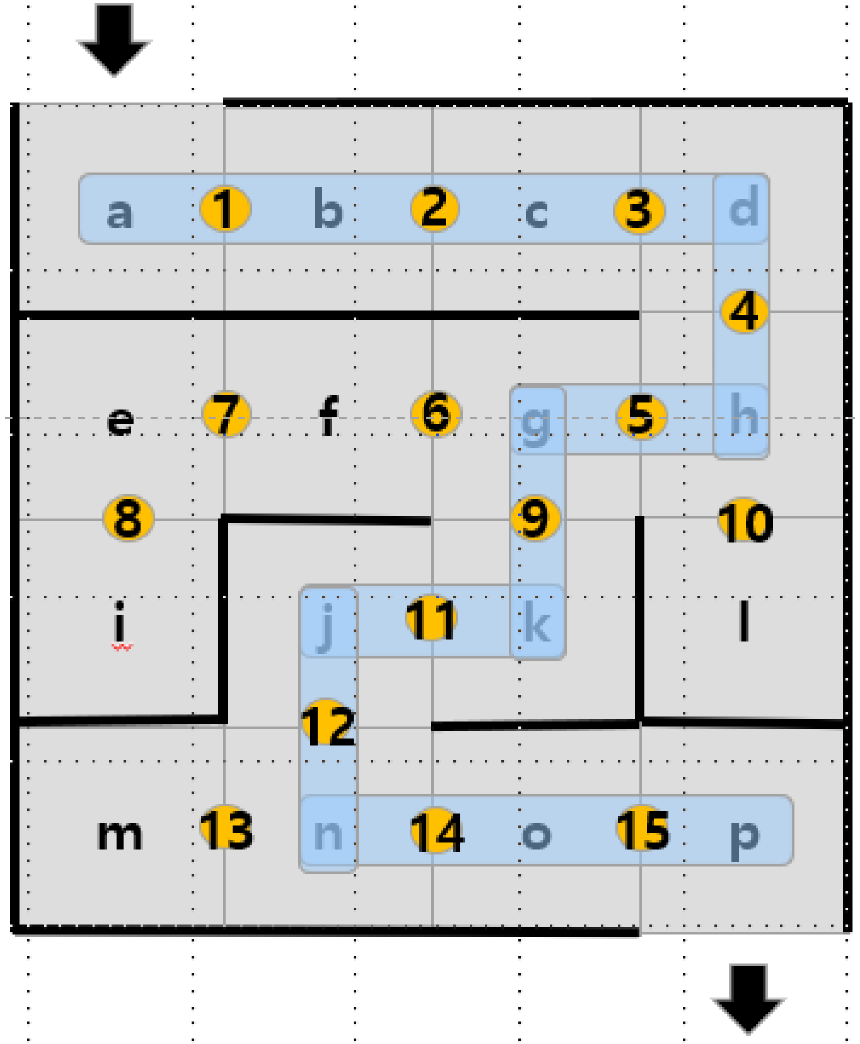


이산수학 – 문제해결

- 문제해결 모델



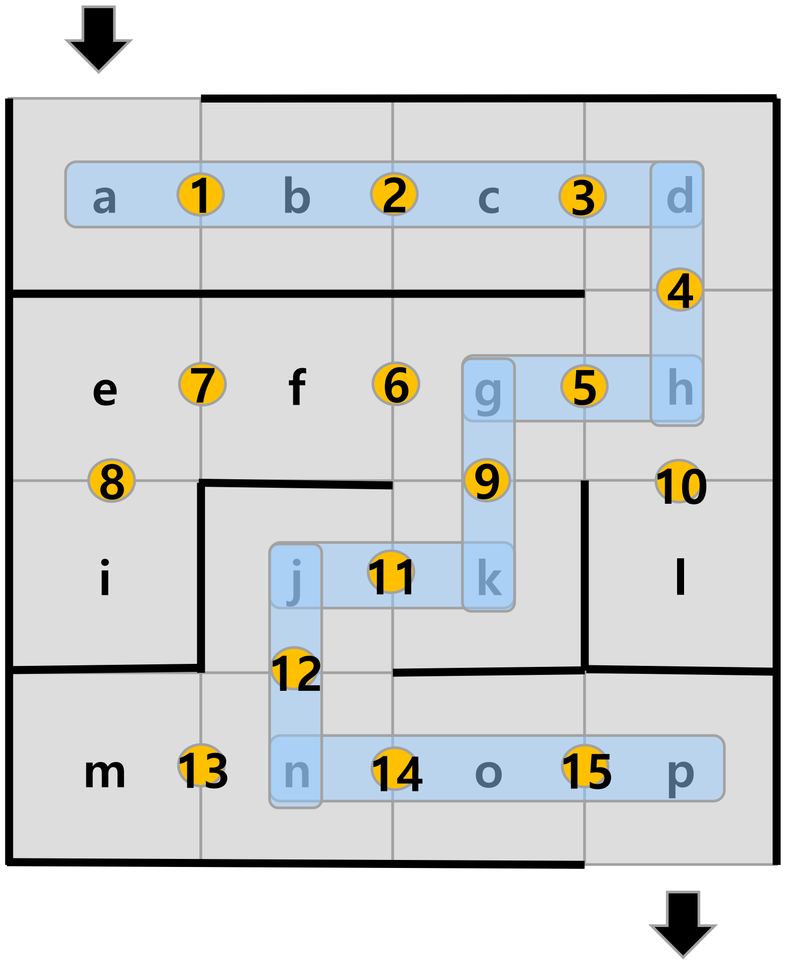
- 그래프모델 G = (V, E)
  - (a, (a-b)) ←→ (a, 1)
  - (b, (b-c)) ←→ (b, 2)
  - (c, (c-d)) ←→ (c, 3)
  - (d, (d-h)) ←→ (d, 4)
  - (h, (h-g), (h-l)) ←→ (h, 5, 10)
  - (g, (g-k), (g-f)) ←→ (g, 9, 6)
  - (f, (f-e)) ←→ (f, 7)
  - (e, (e-i)) ←→ (e, 8)
  - (k, (k-j)) ←→ (k, 11)
  - (j, (j-n)) ←→ (j, 12)
  - (n, (n-o), (n-m)) ←→ (n, 14, 13)
  - (o, (o-p)) ←→ (o, 15)



이산수학 – 문제해결

- 이산수학 – 실생활 문제해결 (미로찾기)
  - 문제해결방법
    - 파이썬 코딩

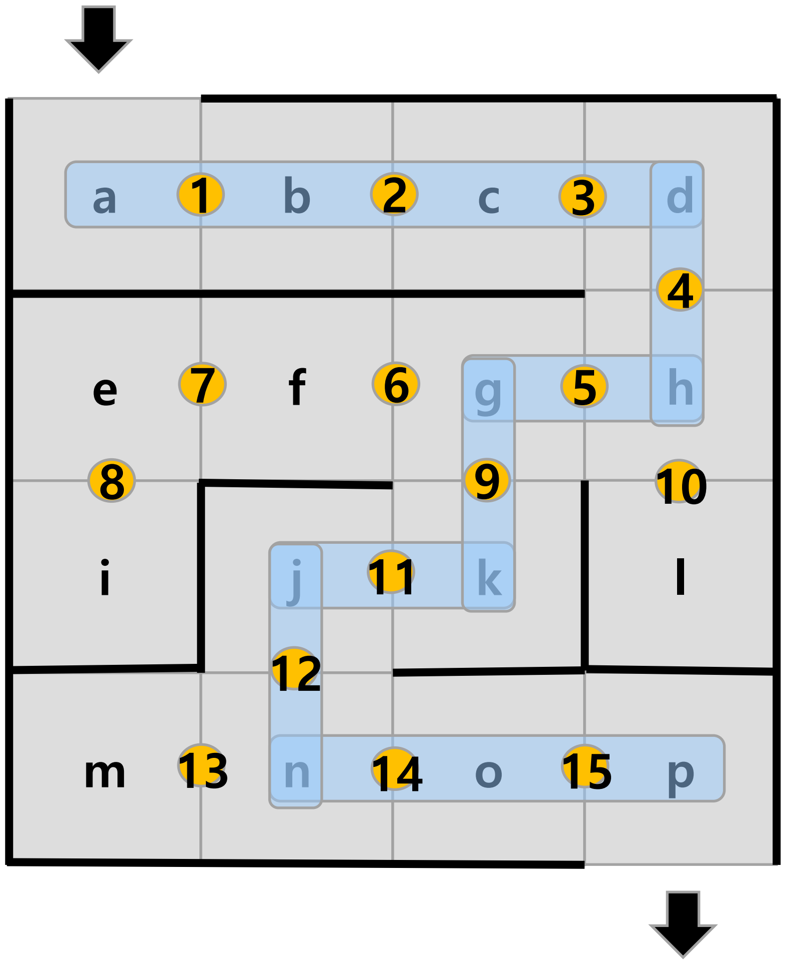
```
maze = {
    'a' : ['b'],
    'b' : ['c'],
    'c' : ['d'],
    'd' : ['h'],
    'e' : ['i'],
    'f' : ['e'],
    'g' : ['k', 'f'],
    'h' : ['g', 'l'],
    'i' : ['i'],
    'j' : ['n'],
    'k' : ['j'],
    'l' : ['l'],
    'm' : ['m'],
    'n' : ['o', 'm'], }
```



이산수학 – 문제해결

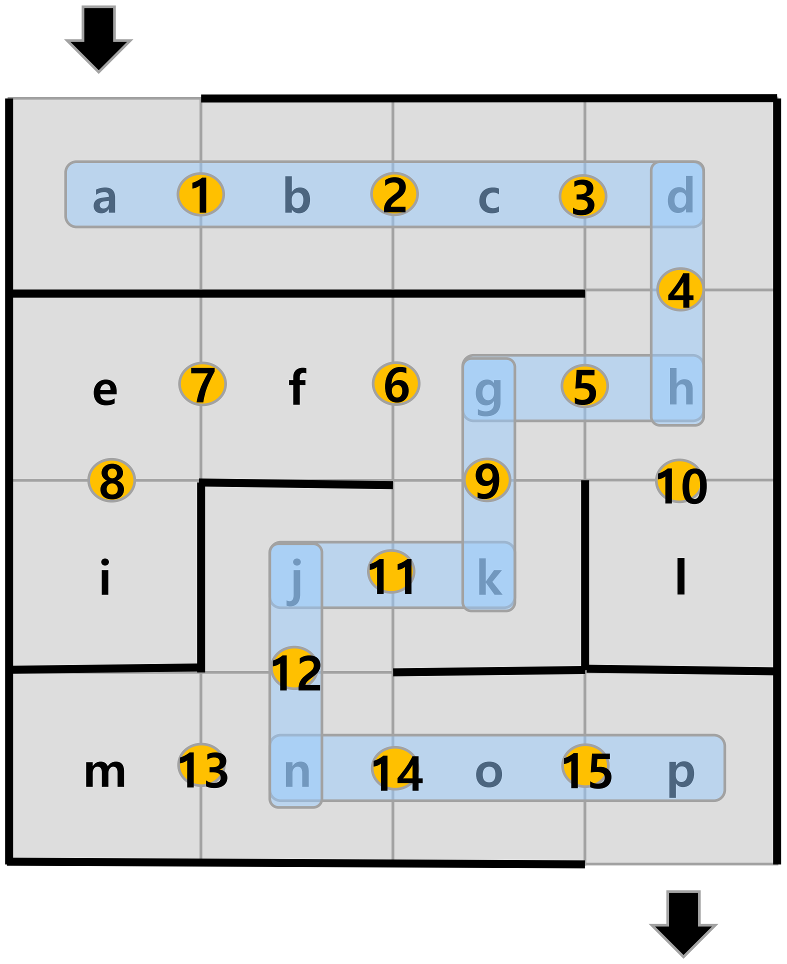
- 이산수학 – 실생활 문제해결 (미로찾기)
  - 문제해결방법

```
def my_maze(g, start, end):
        qu []
        done = set()
        qu.append(start)
        done.add(start)
        while qu:
                p = qu.pop(0)
                v= p[-1]
                if v == end:
                        return p
                for x in g[v]:
                        if x not in done:
                                qu.append(p+x)
                                done.add(x)
                return “?”
        maze = {'a' : ['b'],  'b' : ['c'],  'c' : ['d'],   'd' : ['h'],  'e' : ['i'],   'f' : ['e'],
           'g' : ['k', 'f'],  'h' : ['g', 'l'],   'i' : ['i'],   'j' : ['n'],   'k' : ['j'],
           'l' : ['l'],  'm' : ['m'],  'n' : ['o', 'm'], }
print(my_maze(maze, ‘a’,’p’)
```



이산수학 – 문제해결

- 이산수학 – 실생활 문제해결 (미로찾기)
  - 문제해결 결과
    - a b c d h g k j n o p
    - 1 2 3 4 5 9 11 12 14 15



학습 내용 요약

- 그래프의 기본 개념
- 그래프 용어
- 그래프 종류
- 그래프 표현
- 그래프 탐색
- 그래프 응용