FACULTY OF ENGINEERING
CHULALONGKORN UNIVERSITY
2110327 ALGORITHM DESIGN
Year II, Second Semester, Final Examination, May 12, 2023, 13:00-16:00

ชื่อ-นามสกุล.....เลขประจำตัว..... เลขที่ใน CR58.....

<u>หมายเหตุ</u>

1. ข้อสอบมีทั้งหมด 10 ข้อ ในกระดาษคำถามคำตอบ 8 หน้า
2. **ไม่อนุญาตให้นำตำราและเอกสารใดๆ เข้าในห้องสอบ**
3. **ไม่อนุญาตให้ใช้เครื่องคำนวณใดๆ**
4. ห้ามการหยิบยืมสิ่งใดๆ ทั้งสิ้น จากผู้สอบอื่นๆ เว้นแต่เจ้าหน้าที่ควบคุมการสอบจะหยิบยืมให้
5. ห้ามนำส่วนใดส่วนหนึ่งของข้อสอบและสมุดคำตอบออกจากห้องสอบ
6. ผู้เข้าสอบสามารถออกจากห้องสอบได้ หลังจากผ่านการสอบไปแล้ว 45 นาที
7. เมื่อหมดเวลาสอบ ผู้เข้าสอบต้องหยุดการเขียนใดๆ ทั้งสิ้น
8. **นิสิตกระทำผิดเกี่ยวกับการสอบ ตามข้อบังคับจุฬาลงกรณ์มหาวิทยาลัย มีโทษ คือ พ้นสภาพการเป็นนิสิต หรือ ได้รับสัญลักษณ์ F ในรายวิชาที่กระทำผิด และอาจพิจารณาให้ถอนรายวิชาอื่นทั้งหมดที่ลงทะเบียนไว้ในภาคการศึกษานี้**

*** ร่วมรณรงค์การไม่กระทำผิดและไม่ทุจริตการสอบที่คณะวิศวกรรมศาสตร์ ***

ข้าพเจ้ายอมรับในข้อกำหนดที่กล่าวมานี้ ข้าพเจ้าเป็นผู้ทำข้อสอบนี้ด้วยตนเองโดยมิได้รับการช่วยเหลือ หรือให้ความช่วยเหลือ ในการทำข้อสอบนี้

ลงชื่อนิสิต.....
วันที่.....

- **ใช้ดินสอทำข้อสอบได้**
- **ให้เขียนคำตอบลงในพื้นที่ที่กำหนดให้ หากที่ไม่พอ ให้เขียนในด้านหลังของกระดาษของข้อนั้นเท่านั้น ห้ามเขียนในหน้าอื่น**
- **นิสิตสามารถเรียกใช้ algorithm หรือ data structure ใด ๆ ที่อยู่ในเนื้อหาตามที่เรียนมาได้โดยไม่จำเป็นต้องเขียน code หรือ pseudocode ดังกล่าว แต่จำเป็นที่จะต้องบอกให้ชัดเจนว่า input และ output ของ algorithm ดังกล่าวคืออะไร**
  - **ตัวอย่างเช่น นิสิตสามารถเขียนว่า "เรียกใช้ Dijkstra's algorithm โดยให้ input คือ กราฟ G ที่โจทย์กำหนดให้และใช้น้ำหนักของเส้นเชื่อมคืออาเรย์สองมิติ w และให้ผลลัพธ์ของ algorithm คืออาเรย์ D[]" เป็นต้น (หรือเขียนแบบอื่นก็ได้ขอให้เข้าใจได้ตามนี้)**
  - **อย่างไรก็ตาม หากมีการแก้ไข algorithm ให้ไม่เป็นไปตามที่เรียนในชั้นเรียน จะต้องระบุให้ชัดเจน (ด้วยการเขียน pseudocode หรือคำอธิบายที่ชัดเจน)**

1. (20 คะแนน) ในข้อย่อยต่อไปนี้ให้ตอบโดยเลือกคำตอบที่ถูกต้องที่สุด แต่ละข้อย่อยมีคะแนน 1 คะแนน หากไม่ตอบในข้อใด จะได้คะแนน 0 แต่ถ้าหากตอบผิดในข้อใด จะได้คะแนน -0.5 ต่อข้อย่อย อย่างไรก็ตาม ถึงแม้จะตอบผิดจนได้คะแนนรวมติดลบ จะถือว่าข้อนี้ได้คะแนนเป็น 0 ****ให้เขียนคำตอบลงในตารางข้างล่างนี้เท่านั้น****

ข้อ 1	ข้อ 2	ข้อ 3	ข้อ 4	ข้อ 5	ข้อ 6	ข้อ 7	ข้อ 8	ข้อ 9	ข้อ 10
ข้อ 11	ข้อ 12	ข้อ 13	ข้อ 14	ข้อ 15	ข้อ 16	ข้อ 17	ข้อ 18	ข้อ 19	ข้อ 20

1.1. การทำงานของ Greedy Algorithm มีขั้นตอนหลักคืออะไร?
- ก. ค้นหาข้อมูล
- ข. หาคำตอบที่ดีที่สุดเชิงลึก
- ✗ ค. เลือกคำตอบที่ดีที่สุดแบบเฉพาะหน้า
- ง. แก้ปัญหาแบบกลุ่ม

1.2. ข้อใดเป็นตัวอย่างของปัญหาที่ใช้ Greedy Algorithm ในการแก้ปัญหา?
- ก. ปัญหาทอนเงิน : coin change : dp ?
- ข. ปัญหาการหาเส้นทางที่สั้นที่สุด : shortest path
- ค. ปัญหา N ควีน : SSS
- ✗ ง. ทั้ง ก, ข และ ค ไม่ใช่ทั้ง 3

1.3. มีหินจำนวน 4 ชิ้นที่ต้องการนำไปขาย แต่ลำบากต้องการใส่ลงในกระเป๋าที่มีความจุไม่เกิน 50 กิโลกรัม ข้อมูลของหินแต่ละชิ้นคือ น้ำหนัก (กิโลกรัม) และมูลค่า (บาท) ดังนี้

หิน A: น้ำหนัก 10 กิโลกรัม มูลค่า 60 บาท
หิน B: น้ำหนัก 20 กิโลกรัม มูลค่า 100 บาท
หิน C: น้ำหนัก 30 กิโลกรัม มูลค่า 120 บาท
หิน D: น้ำหนัก 40 กิโลกรัม มูลค่า 160 บาท

มูลค่าสูงสุดที่นักสำรวจหินสามารถนำไปขายได้คือเท่าไหร่? โดยเราไม่สามารถหั่นหินให้เป็นชิ้นเล็กลงได้
- ก. 180 0/1 knapsack
- ข. 200
- ✗ ค. 220 : AD / BC
- ง. 240 : ไม่รู้มูลค่ารวมให้เป็นเท่านี้

1.4. ข้อใดเป็นอัลกอริทึมที่ใช้ในการหา Minimum Spanning Tree ของกราฟที่มีน้ำหนักบนเส้นเชื่อม?
- ก. Dijkstra's Algorithm : shortest path
- ✗ ข. Kruskal's Algorithm : MST
- ค. Floyd-Warshall Algorithm : shortest path
- ง. Ford-Fulkerson Algorithm : Unknown

1.5. ในกรณีที่น้ำหนักบนเส้นเชื่อมในกราฟเป็นบวกทั้งหมด ข้อใดเป็นคุณสมบัติของ Minimum Spanning Tree?
- ก. การเพิ่มเส้นเชื่อมใด ๆ จะสร้าง cycle ใหม่ ✓
- ข. การลดเส้นเชื่อมใด ๆ จะสร้างกราฟไม่เชื่อมต่อ ✓
- ค. มีจำนวนเส้นเชื่อมเท่ากับจำนวนปม – 1 ✓
- ✗ ง. ทั้ง ก ข และ ค

1.6. ข้อใดเป็นข้อจำกัดหลักของ Prim's Algorithm และ Kruskal's Algorithm ในการหา Minimum Spanning Tree?
- ก. ไม่สามารถหา Minimum Spanning Tree ที่ไม่ซ้ำกันได้มากกว่า 1 MST
- ข. ไม่สามารถทำงานกับกราฟที่มีน้ำหนักเป็นลบ
- ค. ไม่สามารถทำงานกับกราฟที่ไม่เชื่อมต่อ shortest path
- ✗ ง. ไม่สามารถหาเส้นทางที่สั้นที่สุดระหว่างคู่ยอดใด ๆ

1.7. ข้อใดไม่เป็นอัลกอริทึมที่ใช้ในการหาเส้นทางที่สั้นที่สุด (Shortest Path)?
- ก. Breadth First Search Algorithm
- ✗ ข. Kruskal's Algorithm : MST
- ค. Floyd-Warshall's Algorithm
- ง. Bellman-Ford Algorithm

1.8. ข้อใดคือความแตกต่างหลัก ๆ ระหว่าง Dijkstra's Algorithm และ Bellman-Ford Algorithm สำหรับการหาเส้นทางที่สั้นที่สุด?
- ก. Bellman-Ford Algorithm ใช้เวลาน้อยกว่า Dijkstra's Algorithm (VE vs V²)
- ข. Bellman-Ford หาเส้นทางสั้นที่สุดสำหรับทุกคู่ปม (Single source)
- ค. Dijkstra's Algorithm สามารถใช้กับกราฟที่มีน้ำหนักเป็นจำนวนจริงได้ แต่ Bellman-Ford Algorithm ไม่สามารถทำได้ (ผิด)
- ✗ Dijkstra's Algorithm ไม่สามารถใช้กับกราฟที่มีน้ำหนักเป็นลบได้ แต่ Bellman-Ford Algorithm สามารถทำได้

1.9. ข้อใดเป็นข้อบ่งชี้ว่าปัญหาการหาเส้นทางที่สั้นที่สุดไม่สามารถหาคำตอบได้?
- ก. กราฟมี cycle ที่มีน้ำหนักเป็นบวก
- ข. กราฟไม่มี cycle ที่มีน้ำหนักเป็นบวก
- ✗ ค. กราฟมี cycle ที่มีน้ำหนักเป็นลบ Neg Cycle → dist = -∞
- ง. กราฟไม่มี cycle ที่มีน้ำหนักเป็นลบ

1.10. ข้อใดเป็นอัลกอริทึมที่ใช้ในการหาเส้นทางที่สั้นที่สุดระหว่างทุกคู่ปมในกราฟที่มีน้ำหนักบนเส้นเชื่อม?
- ✗ ก. Floyd-Warshall Algorithm
- ข. Dijkstra's Algorithm
- ค. Bellman-Ford Algorithm
- ง. Kruskal's Algorithm

1.11. ข้อใดไม่ใช่ข้อดีของ Floyd-Warshall Algorithm?
- ก. สามารถทำงานกับกราฟที่มีน้ำหนักเป็นลบ
- ข. สามารถหาเส้นทางที่สั้นที่สุดระหว่างทุกคู่ของปม
- ค. สามารถทำงานกับกราฟที่มี cycle ที่มีน้ำหนักลบ
- ง. ความซับซ้อนของเวลาเป็น $O(n^3)$

1.12. ถ้าต้องการใช้อัลกอริทึมเส้นทางที่สั้นที่สุดแบบเดี่ยว (Single-Source Shortest Path) เพื่อหาเส้นทางที่สั้นที่สุดระหว่างทุกคู่ของปมสำหรับกราฟที่น้ำหนักเส้นเชื่อมทุกเส้นเป็นบวกควรเลือกอัลกอริทึมใด? (ให้ n คือจำนวนปม)
- ก. ใช้ Dijkstra's Algorithm ทำงาน n ครั้ง
- ข. ใช้ Bellman-Ford Algorithm ทำงาน n ครั้ง
- ค. ใช้ Kruskal's Algorithm ทำงาน n ครั้ง
- ง. ใช้ Prim's Algorithm ทำงาน n ครั้ง

1.13. ข้อใดเป็นข้อจำกัดหลักของการค้นหาแบบ Breadth-First Search (BFS) ใน State Space Search?
- ก. ไม่สามารถหาเส้นทางที่สั้นที่สุดได้
- ข. ไม่สามารถทำงานกับกราฟที่มี cycle
- ค. ใช้หน่วยความจำมากในการค้นหา
- ง. ความซับซ้อนของเวลาเป็นกำลังสอง

1.14. สำหรับ State Space Search ข้อใดเป็นข้อดีของการใช้ Heuristic ในการค้นหา?
- ก. เพิ่มความเร็วในการค้นหา
- ข. สามารถหาคำตอบที่ดีที่สุดได้แน่นอน
- ค. สามารถทำงานในเชิงลึกได้ DFS
- ง. ใช้หน่วยความจำน้อยที่สุดในการค้นหา

1.15. ปัญหาใดเป็นตัวอย่างของปัญหา NP-Complete?
- ก. Minimum Spanning Tree
- ข. All-Pair Shortest Path
- ค. Traveling Salesman Problem
- ง. Single-Source Shortest Path

1.16. ปัญหาที่ใช้เวลาในการทำงานเป็น Polynomial เรียกว่าอะไร
- ก. Intractable
- ข. Tractable
- ค. Decidable
- ง. Turing Complete

1.17. ข้อใดคือกลุ่มของปัญหาที่สามารถแก้ไขได้โดย Non-Deterministic Polynomial Time Algorithm
- ก. NP
- ข. P
- ค. Hard
- ง. Complete

1.18. กราฟแบบมีทิศทางขนาด 4 ปมที่มี strongly connected component เพียง component เดียว สามารถมีเส้นเชื่อมได้น้อยที่สุดเท่าใด
- ก. 0
- ข. 2
- ค. 4
- ง. 6

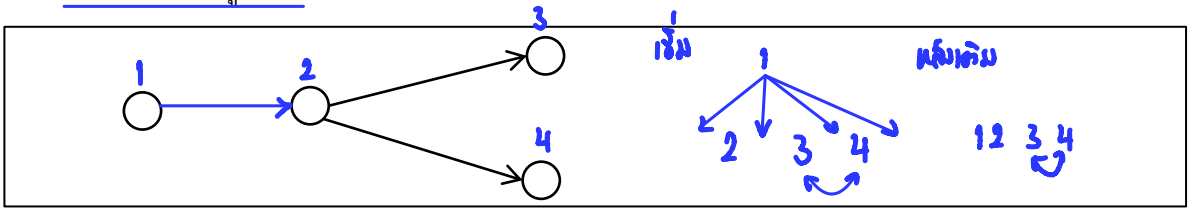
Not in exam

1.19. กำหนดให้ปัญหา A เป็นปัญหาในกลุ่ม P และ B เป็นปัญหาในกลุ่ม NP-C หากเราสามารถลดรูปปัญหา A เป็นปัญหา B ได้ในเวลา polynomial แล้ว ข้อใดถูกต้องที่สุด
- ก. ปัญหา A เป็นปัญหา NP-Complete
- ข. ปัญหา A เป็นปัญหา NP เพราะ P อยู่ใน NP
- ค. ปัญหา B เป็นปัญหา P
- ง. ปัญหา B ไม่เป็นปัญหา NP

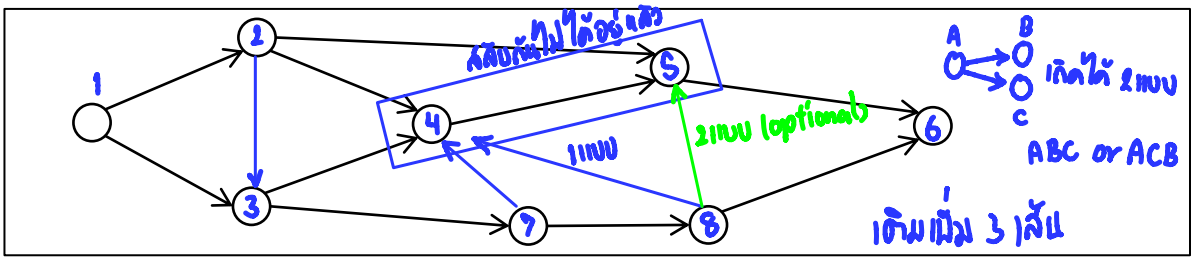
1.20. หากเราทราบว่าปัญหา A, B, C, D สามารถลดรูปได้ดังต่อไปนี้ A→B, B→C, C→A, A→D ข้อใดถูกต้องที่สุด เมื่อให้เครื่องหมาย → คือการลดรูปได้ในเวลา polynomial
- ก. ปัญหา A ยากกว่า D แน่นอน
- ข. ปัญหา D และ B ยากเท่ากันแน่นอน
- ค. ปัญหา C และ A ยากเท่ากันแน่นอน
- ง. ปัญหา B และ D ยากกว่า A แน่นอน

2. (3 คะแนน) สำหรับกราฟแต่ละกราฟต่อไปนี้ จงเพิ่มเส้นเชื่อมในกราฟเป็นจำนวนน้อยที่สุดที่ทำให้ topological sort ที่เป็นไปได้ของกราฟดังกล่าวมีจำนวนไม่เกิน 2 รูปแบบที่แตกต่างกัน

2.1.

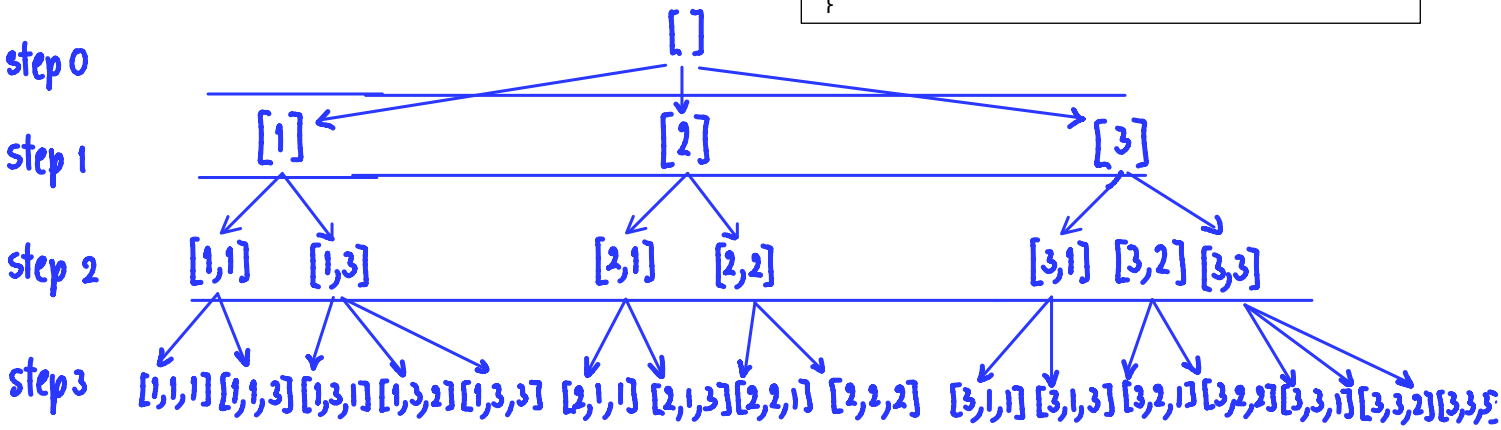


2.2.



3. (3 คะแนน) จากส่วนของโปรแกรมด้านขวานี้ จงวาด State Space Tree ที่เกิดขึ้นเมื่อเรียก gen(0, 3, 3, []) โดยที่ a คืออาเรย์ และ a + [x] หมายถึงการสร้างอาเรย์ใหม่ที่เกิดจากการนำ x ไปต่อท้าย a ในการวาด State Space Tree นั้น ให้ระบุค่าของ a ในแต่ละปม

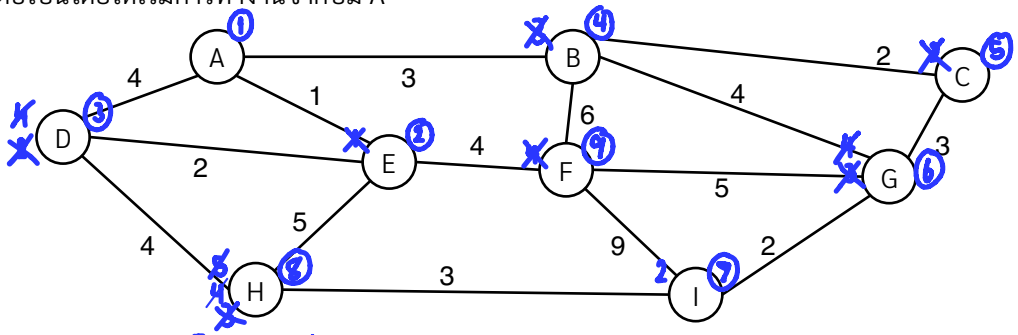
```
void gen(step, k, n, a) {
  if (step < n) {
    for (i in 1..k) {
      if (a.size() == 0 || a.back() != i-1)
        gen(step + 1, k, n, a + [i]);
    }
  } else
    print_array(a);
}
```



4. (2 คะแนน) จงเติมคำในช่องว่างต่อไปนี้
ในการหา Strongly Connected Component ของกราฟ G ด้วยวิธีที่เรียนในวิชานี้ เราจะใช้

_____Search บนกราฟ_____ เพื่อหาปมที่เป็น_____ทางของกราฟ G

5. (3 คะแนน) จงระบุลำดับของปมที่ Prim's Algorithm นำออกจาก พิจารณาในการหา Minimum Spanning Tree ของ Graph ดังต่อไปนี้โดยให้เริ่มการทำงานจากปม A



ตอบ: ลำดับคือ A E D B C G I H F

- **สำหรับข้อที่ 6 เป็นต้นไป เป็นการออกแบบอัลกอริทึม ในแต่ละข้อสามารถตอบโดยการอธิบายอัลกอริทึม โดยใช้รหัสเทียม (Pseudocode) หรือ programming language ภาษาใดที่เคยเรียนมาก็ได้ และต้องวิเคราะห์ประสิทธิภาพในการทำงานของอัลกอริทึมด้วย**
- **ให้อธิบายแนวคิดโดยสังเขปของอัลกอริทึมที่ออกแบบด้วย**
- **คะแนนที่ได้จะแปรตามประสิทธิภาพในการทำงาน**

6. (10 คะแนน) คุณกำลังจะขับรถบนถนนเส้นตรงเส้นหนึ่ง บนถนนเส้นนี้มีสถานีเติมน้ำมันอยู่ n สถานี โดยสถานีที่ i ตั้งอยู่ห่างจากจุดเริ่มต้นเป็นระยะทาง D[i] กม. รถของคุณมีความจุน้ำมันเพียงพอในการเดินทางได้ 100 กิโลเมตร (กล่าวคือ หากเติมน้ำมันเต็มถังที่ กม. ที่ x รถจะสามารถวิ่งไปถึง กม.ที่ x+100 ได้พอดี) สมมติว่าคุณเริ่มต้นด้วยถังน้ำมันว่าง แต่มีสถานีเติมน้ำมันตรงจุดเริ่มต้นของคุณ (นั่นคือ D[0] = 0) สมมติเช่นเดียวกันว่ามีสถานีเติมน้ำมันที่ปลายทาง D[n-1] จงออกแบบและวิเคราะห์อัลกอริทึม ในการคำนวณจำนวนครั้งในการเติมน้ำมันน้อยที่สุดที่คุณต้องทำเพื่อให้ถึงปลายทาง และมีน้ำมันเต็มถังที่ปลายทางพอดี หรือ คืนค่า ∞ ถ้าไม่สามารถทำได้ กำหนดให้ข้อมูลนำเข้าคือ D[0..n-1] และค่า n Greedy

```cpp
#include<bits/stdc++.h>

using namespace std;

const int N = 110;
int D[N],n;

int main()
{
    cin>>n;
    for(int i = 0 ; i != n ; i++)cin>>D[i];
    sort(D,D+n);
    int cnt = 0;
    int idx = 0;
    int dis = 0;
    bool cantgo = false;
    while(idx<n)
    {
        idx = lower_bound(D,D+n,D[idx]+100)-D;
        if(D[idx]>dis+100)idx--;//ไปไม่ถึงย้อนกลับมา
        if(D[idx] == dis)//อยู่ที่เดิม
        {
            cantgo = true;
            break;
        }
        dis = D[idx];
        cnt++;
    }
    if(!cantgo)cout<<cnt;
    else cout<<"INF";
}
```

testcase

```
/*
5
0 101 200 300 400
ans : INF
5
0 100 200 300 400
ans : 5
10
0 99 100 111 200 255 300 322 399 400
ans : 5
5
0 100 201 300 400
ans : INF
*/
```

7. (10 คะแนน) กำหนดให้มี G = (V, E) เป็นกราฟที่มีทิศทางและไม่มีวัฏจักร (DAG) และสำหรับเส้นเชื่อม (u,v) ใด ๆ ในกราฟนี้ กำหนดให้ d(u,v) เป็นฟังก์ชันที่คืนค่าน้ำหนักของเส้นเชื่อมดังกล่าว โดยรับประกันน้ำหนักของเส้นเชื่อมเป็นบวกเสมอ ให้ s เป็นปมของ G ที่มี in-degree เป็น 0 และรับประกันว่า มีเส้นทางจาก s ไปยังปมอื่นใดในกราฟนี้ทั้งหมด DAG + in-degree → toposort

7.1. (5 คะแนน) จงออกแบบอัลกอริทึมที่ใช้เวลา O(|V| + |E|) ในการคำนวณเส้นทางสั้นที่สุดจาก s ไปยังปมอื่น ๆ ทั้งหมดใน G BFS

```cpp
int dist[N];

int main()
{
    memset(dist,0x3f,sizeof dist);
    queue<int> q;
    q.push(s);
    dist[s] = 0;
    while(!q.empty())
    {
        int t = q.front();q.pop();
        for node in adj(t)
        {
            in_degree[node]--;
            dist[node] = min(dist[node],dist[t] + d(t,node));
            if(in_degree(node)==0)q.push(node);
        }
    }
}
```

7.2. (5 คะแนน) จงออกแบบอัลกอริทึมที่มีประสิทธิภาพในการคำนวณเส้นทางที่ยาวที่สุดจาก s ไปยังปมอื่น ๆ

คำแนะนำ: ทั้งสองข้อย่อยข้างบนนี้ แทบไม่มีความแตกต่างกันในตัวคำตอบเลย

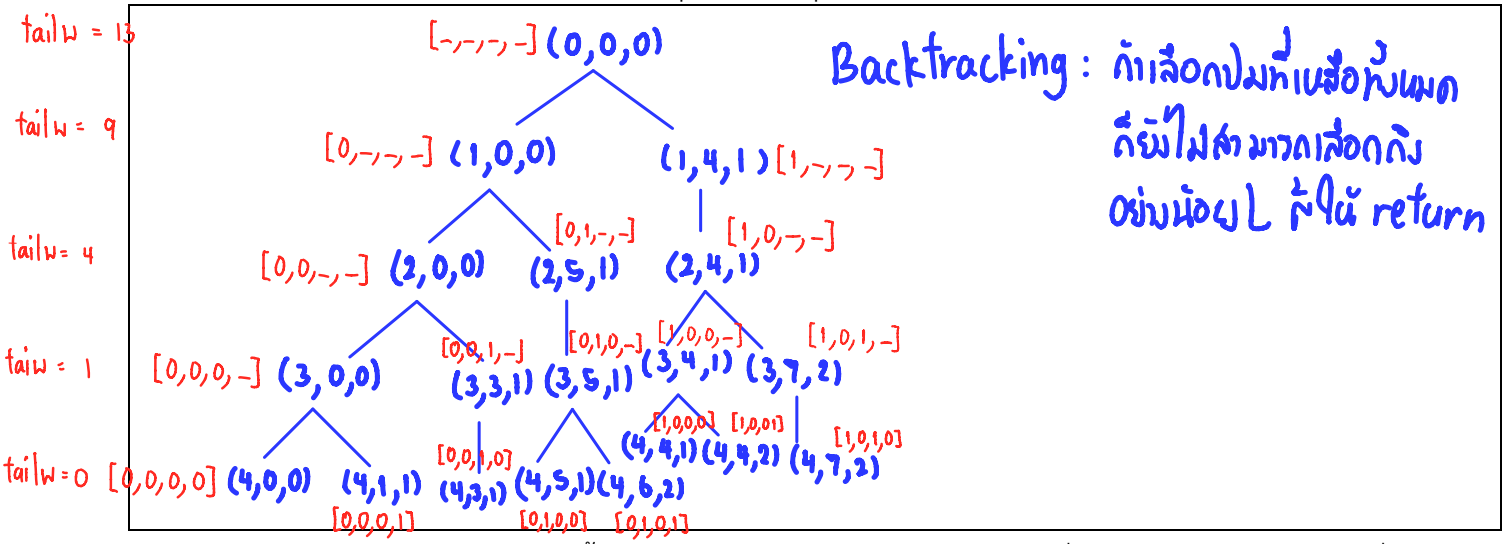
```cpp
int dist[N];

int main()
{
    memset(dist,-1,sizeof dist);
    queue<int> q;
    q.push(s);
    dist[s] = 0;
    while(!q.empty())
    {
        int t = q.front();q.pop();
        for node in adj(t)
        {
            in_degree[node]--;
            dist[node] = max(dist[node],dist[t] + d(t,node));
            if(in_degree(node)==0)q.push(node);
        }
    }
}
```

8. (10 คะแนน) ใกล้เลือกตั้งแล้ว สมชายมีอาชีพเป็นคนติดป้ายหาเสียง ตามกฎหมายเลือกตั้ง สมชายมีเวลา n วันในการติดป้าย ในแต่ละวันจะมีงานจ้างติดตั้งป้ายหาเสียงมาให้สมชาย สมชายสามารถเลือกที่จะทำงานหรือไม่ทำงานในแต่ละวันก็ได้ หากสมชายเลือกทำงานในวันที่ i สมชายจะติดป้ายได้ w[i] ป้าย โดยเสียต้นทุนในการเดินทางทั้งวัน c[i] บาท สมชายมีเป้าหมายว่าภายใน n วันนี้ สมชายจะต้องติดป้ายให้ได้อย่างน้อย L ป้าย โดยที่สมชายต้องการให้เสียต้นทุนรวมให้น้อยที่สุดเท่าที่เป็นไปได้ นอกจากนี้ งานติดป้ายนั้นลำบากมาก หากสมชายออกไปทำงานติดป้ายในวันที่ i แล้ว สมชายจะหมดแรง ไม่สามารถทำงานในวันที่ i+1 ได้ กำหนดให้ข้อมูลนำเข้าของปัญหานี้คือ n, w[1..n], c[1..n] และ L จงตอบคำถามต่อไปนี้

8.1. (2 คะแนน) กำหนดให้ x[i] ∈ {0,1} เป็นการระบุว่าสมชายเลือกทำงานในวันที่ i หรือไม่ โดยที่ x[i] มีค่าเป็น 1 ก็ต่อเมื่อสมชายเลือกทำงานในวันที่ i จงออกแบบอัลกอริทึม ที่แสดงค่า x ทั้งหมดที่เป็นไปได้ที่ทำให้สมชายทำงานได้ตรงตามเงื่อนไข (ซึ่งคือติดป้ายรวมอย่างน้อย L ป้าย และ หากสมชายทำงานในวันที่ i แล้ว สมชายจะต้องไม่ทำงานในวันที่ i+1) ให้แสดงค่า x โดยเรียกใช้ฟังก์ชัน print(x)

```
#include<bits/stdc++.h>
using namespace std;
bool x[N];
void combi(int idx, int sum, int cost)
{
    if(idx >= n && sum >= L)
    {
        min_cost = min(min_cost,cost);
        print(x);
        return;
    }
    if(cost >= min_cost)return;//backtracking
    if(tailw[idx] + sum < L)return;//heuristic
    if(x[idx-1])
    {
        x[idx] = false;
        combi(idx+1,sum,cost);
    }
    else
    {
        x[idx] = true;
        combi(idx+1,sum+w[idx],cost+c[idx]);
        x[idx] = false;
        combi(idx+1,sum,cost);
    }
}
```

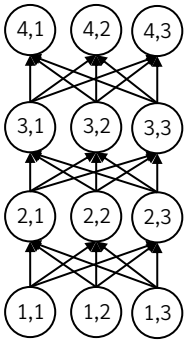
8.2. (4 คะแนน) จากคำตอบในข้อ 8.1 ที่แล้วนั้น จงวาด State Space Tree ของคำตอบในข้อ 8.1 เมื่อกำหนดให้ n = 4, w = [4,5,3,1] , c = [1,1,1,1] และ L = 5 และให้ระบุว่ามีการใช้เทคนิค Backtracking หรือไม่ (หากมีการใช้ Backtracking ให้อธิบายเงื่อนไขของ Backtracking พร้อมทั้งระบุว่าปมใดที่หยุดทำงานเนื่องจากการใช้ Backtracking ดังกล่าว)



8.3. (4 คะแนน) หากเราต้องการแก้ปัญหานี้ด้วยวิธี Branch & Bound จงเสนอ function ที่คำนวณ Lower Bound (หรือที่เรียกว่า heuristic function) ของ state นี้ โดยกำหนดให้ state ที่เราพิจารณาอยู่มีคือ state ที่เราทราบค่าของ x[1..k] แล้ว โดยให้ตอบคำถามนี้ด้วยการ อธิบายวิธีการคำนวณ heuristic และให้ยกตัวอย่างการคำนวณค่า lower bound ดังกล่าว ของ state อย่างน้อย 1 state โดยใน state ให้ระบุค่าของ w[1..n], c[1..n], L, k, x[1..k] มาด้วย โดยมีข้อบังคับคือ n ต้องเป็น 5

$$\text{if}\left(\text{cost} + \left(\frac{\text{tailc}[\text{idx}]}{\text{tailw}[\text{idx}]} \times (L - \text{sum})\right) >= \text{min\_cost}\right) \text{return}$$

9. (10 คะแนน) กำหนดให้ “กราฟ k ชั้น กว้าง m” คือกราฟมีทิศทางแบบมีน้ำหนัก ปมในกราฟนี้ถูกแบ่งออกเป็นชั้น ๆ ตั้งแต่ชั้นที่ 1 ถึงชั้นที่ k โดยในชั้นที่ i ใด ๆ นั้นมีปมอยู่ m ปม และสำหรับปมใด ๆ ในชั้นที่ i นั้น จะมีเส้นเชื่อม ไปยังทุก ๆ ปมในชั้นที่ i+1 รูปด้านขวานี้แสดงตัวอย่างของ “กราฟ 4 ชั้น กว้าง 3”



กำหนดให้ปมแต่ละปมในกราฟนี้ถูกระบุได้ด้วยคู่อันดับ (i,j) ซึ่งระบุถึงปมในชั้นที่ i ในลำดับที่ j ให้ d((a1,b1), (a2,b2)) คือน้ำหนักของเส้นเชื่อมที่เชื่อมระหว่างปม (a1,b1) กับปม (a2,b2) รับประกันว่าน้ำหนักของเส้นเชื่อมใด ๆ ในกราฟมีค่าเป็นบวกเสมอ

จงออกแบบและวิเคราะห์อัลกอริทึมสำหรับการหาระยะทางสั้นสุดจากทุก ๆ ปมในชั้นที่ 1 ไปยังทุก ๆ ปมในชั้นที่ 2 ถึง k ในกราฟนี้ เงื่อนไขจำเป็นในการที่จะได้คะแนนเต็มในข้อนี้คือเวลาในการทำงานจะต้องดีกว่า O((k*m)^3)

ใช้ Dijkstra m รอบ, $O(m^2k\log(m*k) + m^3k\log(m*k))$

bellman m รอบได้อยู่, $e = (k*m^2), n = (k*m)$ $O(ne) = O(k^2m^4)$ ช้าไปหน่อย

Dp

```
int main()
{
    dist[m][k][m] = INF;
    for(int i = 1 ; i <= m ; i++)
    {
        for(int j = 1 ; j <= m ; j++)dist[i][1][j] = 0;
        for(int r = 2 ; r <= k ; r++)
            for(int c = 1 ; c <= m ; c++)
                for(int p = 1 ; p <= m ; p++)
                    dist[i][r][c] = min(dist[i][r][c], dist[i][r-1][m]+d((r-1,m),(r,c)));
    }
}
```

10. (10 คะแนน) สมชายกำลังเล่นเกมเกมหนึ่ง เกมนี้ประกอบด้วยห้องจำนวน n ห้อง (กำกับด้วยหมายเลข 1 ถึง n) มีประตูมิติอยู่ m ประตู ประตูแต่ละประตูจะเชื่อมห้องสองห้องเข้าด้วยกัน และสามารถใช้เดินทางไปมาระหว่างสองห้องนี้ได้ และไม่มีประตูใดเชื่อมคู่ห้องที่เหมือนกัน ประตูในแต่ละด้านจะมีหมายเลขของห้องปลายทางเขียนไว้อยู่ กล่าวคือ เมื่อเราอยู่ในห้องใด เราจะทราบว่าแต่ละประตูในห้องนั้นเชื่อมไปที่ใด โดยไม่จำเป็นต้องเดินทางผ่านประตูดังกล่าว แต่ในตอนเริ่มต้นเราไม่ทราบเลยว่าในเกมนี้มีประตูอยู่กี่บาน และแต่ละประตูเชื่อมไปยังที่ใดบ้าง แต่เราทราบเพียงว่า สำหรับคู่ห้องใด ๆ ก็ตาม จะมีวิธีการเดินทางระหว่างคู่ห้องนั้นที่เดินผ่านห้องต่าง ๆ ที่ไม่ซ้ำกันเลยเพียงวิธีเดียวเท่านั้น

เนื่องจากสมชายซื้อเกมนี้มาแพงมาก สมชายต้องการสำรวจห้องต่าง ๆ ในเกมอย่างละเอียด สมชายได้กำหนดวิธีการเล่นไว้ดังนี้ สมชายเริ่มต้นที่ห้องหมายเลข 1 และ สมชายมี “รายการห้องที่อยากไป” อยู่ในมือ โดยเริ่มต้น รายการนี้มีเพียงห้องหมายเลข 1 อยู่ภายใน เมื่อสมชายเข้าไปในห้องใด ๆ ก็ตาม สมชายจะดูทุก ๆ ประตูในห้องนั้นว่าเชื่อมไปยังห้องปลายทางใดบ้าง และสมชายจะเพิ่มหมายเลขห้องปลายทาง**ที่ไม่เคยอยู่ในรายการ** ต่อท้ายเข้าไปในรายการของเขาตามลำดับหมายเลขห้องจากน้อยไปมาก หลังจากนั้นสมชายจะดูรายการแล้วเลือกห้องที่สมชายไม่เคยไป ที่อยู่เป็นลำดับก่อนหน้ามากที่สุดในรายการ แล้วเดินทางไปยังห้องนั้น โดยสมชายอาจจะต้องเดินย้อนผ่านห้องต่าง ๆ ที่เคยเดินมาแล้วก็เป็นได้ หากไม่มีห้องใดที่สมชายยังไม่เคยไปแล้ว สมชายจะหยุดเล่น

เราต้องการทราบว่า สมชายเดินทางผ่านประตูเป็นจำนวนทั้งหมดกี่ครั้ง ก่อนที่สมชายจะหยุดเล่น

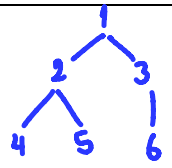
ตัวอย่างเช่น สมมติให้มีห้อง 4 ห้อง และมีประตูเชื่อมห้อง (1-2), (2-3) และ (1-4) สมชายจะดำเนินการดังนี้

1) เริ่มที่ห้อง 1 แล้วเพิ่มห้อง 2 และ 4 เข้าไปในรายการ (ในรายการเป็น [1, 2, 4] โดยห้อง 1 คือห้องที่เคยไปมาแล้ว) ให้สังเกตว่า รายการไม่ได้เป็น [1, 4, 2] เพราะว่าสมชายต้องเพิ่มห้องหมายเลข 2 ก่อน 4
2) สมชายเลือกห้องหมายเลข 2 แล้วเดินเข้าไปจากห้อง 1 **(ผ่าน 1 ประตู)**
3) เมื่ออยู่ที่ห้อง 2 สมชายจะเพิ่มห้อง 3 เข้าไปในรายการ (ให้สังเกตว่า สมชายไม่ได้เพิ่มห้อง 1 เข้าไปเพราะห้อง 1 อยู่ในรายการแล้ว ทำให้รายการเป็น [1, 2, 4, 3] โดยห้อง 1, 2 เคยไปมาแล้ว)
4) สมชายเลือกห้อง 4 จากรายการ (เพราะเป็นห้องที่อยู่เป็นลำดับก่อนหน้ามากสุดในรายการที่ไม่เคยไป) แล้วเดินทางไปยังห้อง 4 จากห้อง 2 โดยเดินผ่านห้อง 1 **(ผ่าน 2 ประตู)**
5) เมื่ออยู่ที่ห้อง 4 สมชายไม่ได้เพิ่มห้องใดเข้าไปในรายการเลย ทำให้รายการเป็น [1, 2, 4, 3]
6) สมชายเลือกห้อง 3 และเดินทางไปยังห้อง 3 จาก 4 โดยเดินผ่านห้อง 1 และ 2 ตามลำดับ **(ผ่าน 3 ประตู)**
7) สมชายหยุดเล่น รวมผ่านทั้งหมด 6 ประตู (ลำดับของหมายเลขห้องที่เดินผ่านตั้งแต่ตอนเริ่มต้นคือ <1,2,1,4,1,2,3>)

จงตอบคำถามต่อไปนี้

10.1.(2 คะแนน) มีห้อง 6 ห้องและมีประตูดังนี้ (1-2), (1-3), (2-4), (2-5) และ (3-6) จงระบุลำดับของหมายเลขห้องที่สมชายเดินผ่านทั้งหมดก่อนที่จะหยุดเล่นตามลำดับ

<1, 2, 1, 3, 1, 2, 4, 2, 5, 2, 1, 3, 6>



10.2.(8 คะแนน) จงออกแบบอัลกอริทึมที่คำนวณจำนวนประตูที่สมชายเดินผ่าน โดยกำหนดให้มีข้อมูลนำเข้าคือ n และมีฟังก์ชัน door(x) ให้เราเรียกใช้ โดยที่ฟังก์ชัน door(x) จะคืนอาเรย์ของหมายเลขห้องปลายทางของประตูต่าง ๆ ของห้องหมายเลข x ตัวอย่างเช่น หากเรียก door(2) จากข้อมูลในข้อ 10.1 ฟังก์ชันนี้จะคืนค่า [1, 4, 5] การที่จะได้คะแนนเต็มในข้อนี้ อัลกอริทึมจะต้องใช้เวลาเป็น $O(n^2)$

```
int dist[N][N];
bitset<N> vst;

void bfs(int s)
{
    vst.reset();
    dist[s][s] = 0;
    queue<int> q;
    q.push(s);
    while(!q.empty())
    {
        int t = q.front();q.pop();
        vst[t] = true;
        for next in door(t)
        {
            if(!vst[next])
            {
                dist[s][next] = dist[s][t] + 1;
                q.push(next);
            }
        }
    }
}
```

Time complexity

$O(n^2+ne+n+e)$

$\Downarrow$

$O(n^2)$

```
int main()
{
    //get dist[a][b] first cuz every edge weight = 1 so bfs will be great O(n**2+ne)
    for(int i = 1 ; i <= n ; i++)bfs(i);
    //bfs get door cnt O(n+e)
    vst.reset();
    int cnt = 0;
    queue<int> gonnavst;
    gonnavst.push(1);
    int curr = 1;
    while(!gonnavst.empty())
    {
        int now = gonnavst.front();gonnavst.pop();
        cnt += dist[curr][u];
        curr = now;
        vst[curr] = true;
        for next in door(now)if(!vst[next])gonnavst.push(next);
    }
}
```