Research talk

Classification and Visualization of Polyhedral Graphs via Degree Sequences

Takumi Shiota

University of Hyogo, Japan

takumi_shiota@gsis.u-hyogo.ac.jp

June 11, 2025

@Maastricht university

Polyhedral graph

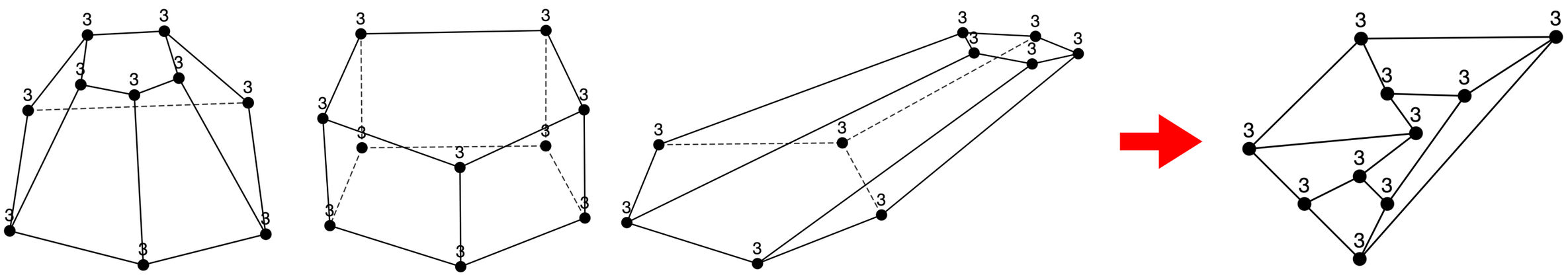
Definition 1

A polyhedral graph G_P is defined as a simple graph that is both planar and 3-connected.

A planar graph is a graph that can be drawn on a plane without any of its edges crossing each other. A 3-connected graph is a graph that remains connected after the removal of any two vertices.

Theorem 2 (Steinitz's Theorem) [B. Grünbaum, 2003]

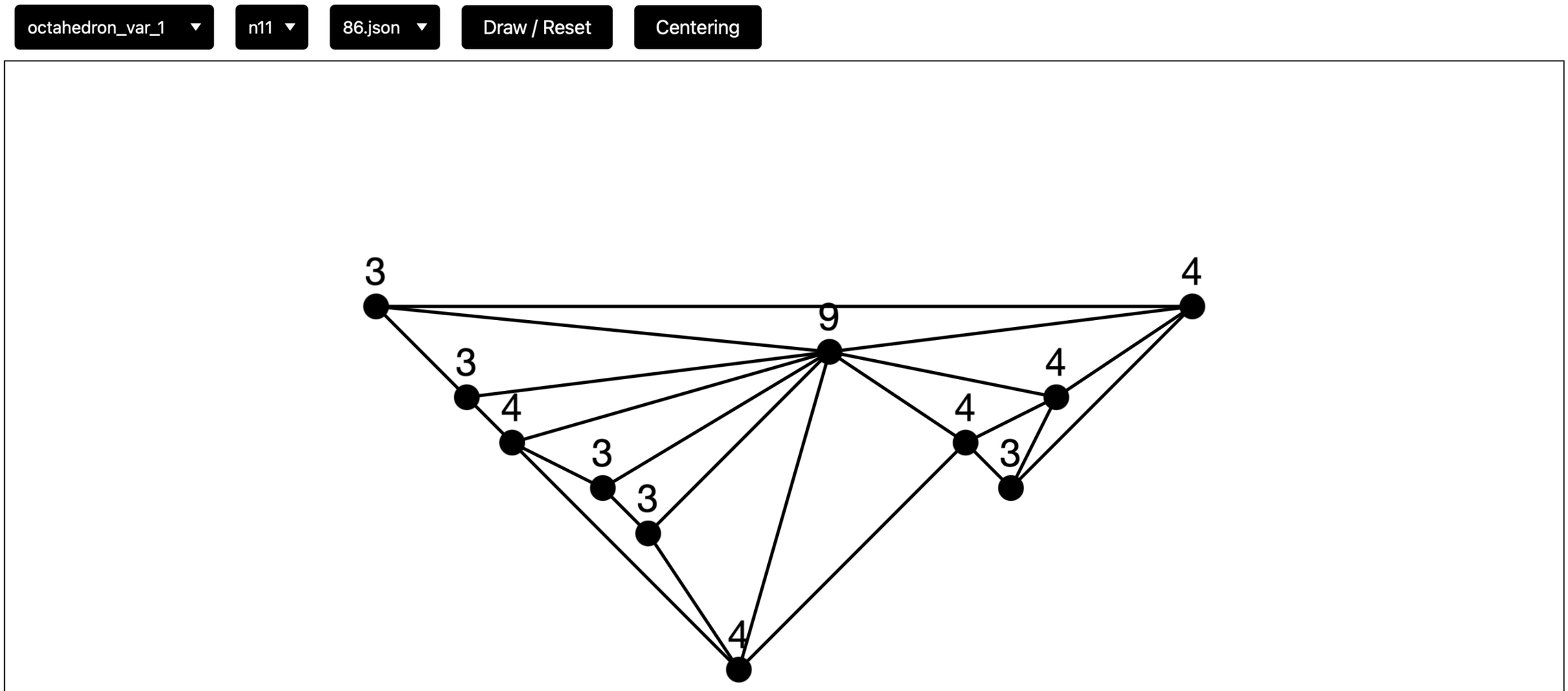
Every convex polyhedron can be represented by G_P.



Research Results During My Stay

Result 1

We made a simple web application that allows free manipulation of edges and vertices in a polyhedral graph.

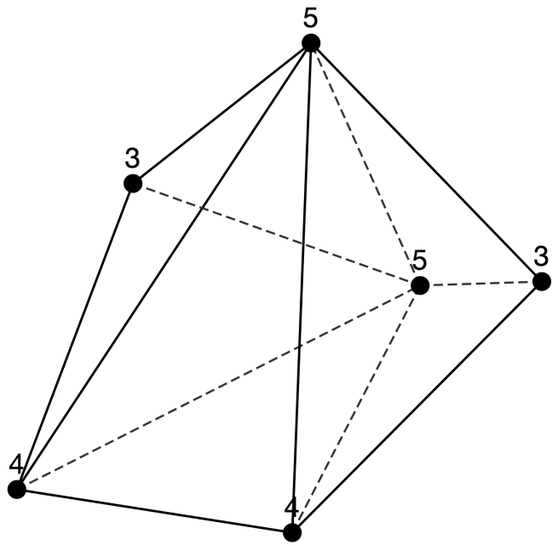


Research Results During My Stay

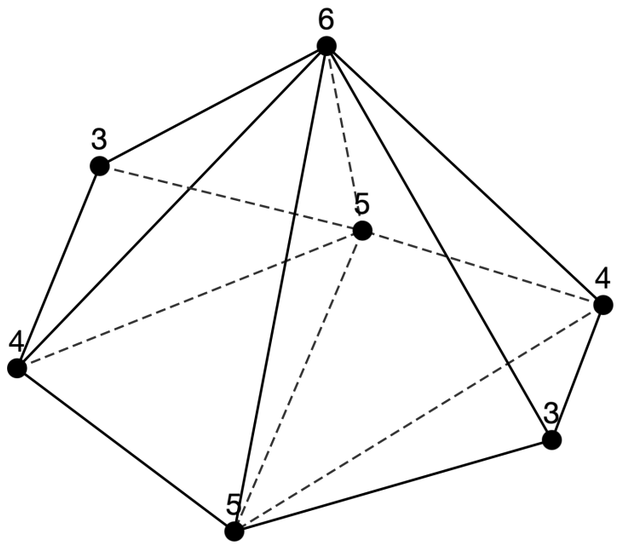
Result 2

We categorized polyhedral graphs by extracting sequences with consecutive numbers of vertices n.

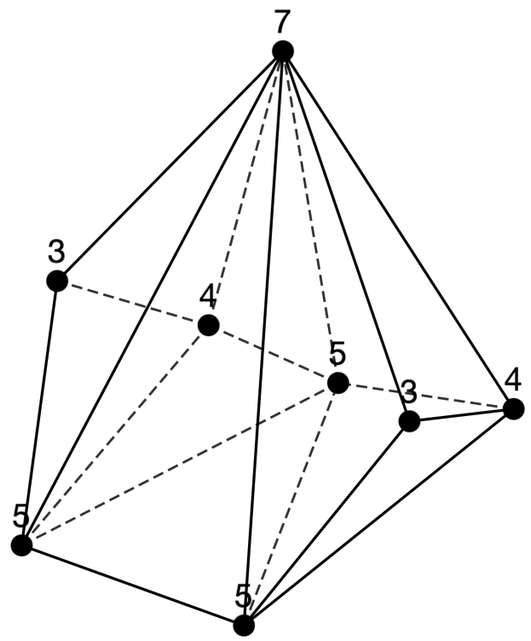
n = 6



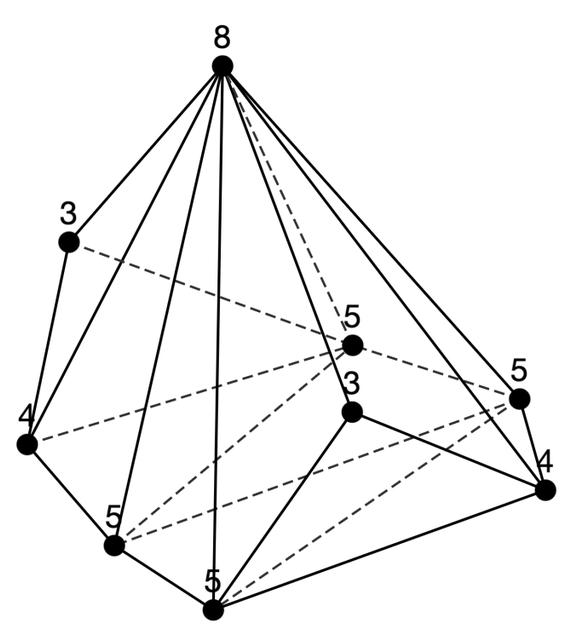
n = 7



n = 8



n = 9



Why did we want sequences of polyhedral graphs?

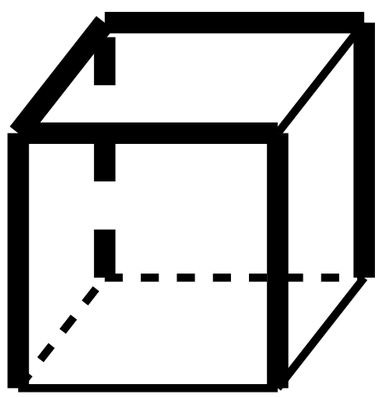
➡ From the following slides explain the background.

Edge Unfoldings

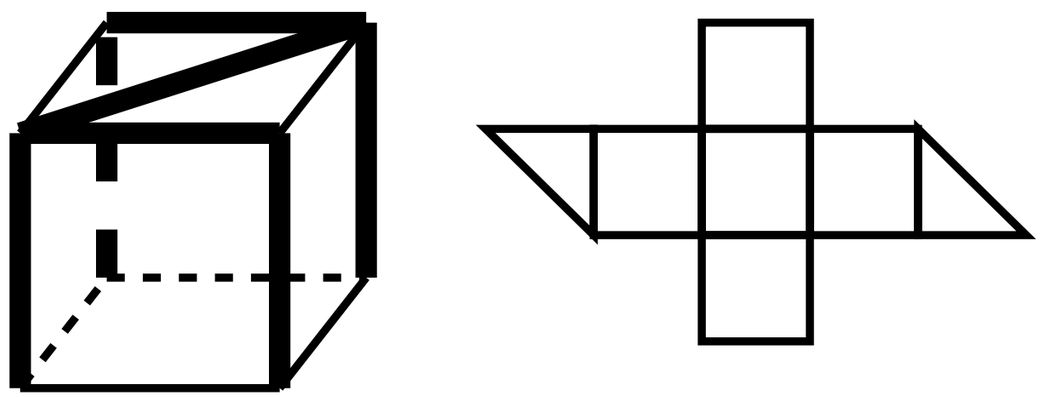
Definition 1 [E. D. Demaine and J. O'Rourke, 2007]

An edge unfolding of the polyhedron is a flat polygon formed by cutting its edges and unfolding it into a plane.

Cutting along the thick line of each left cube …



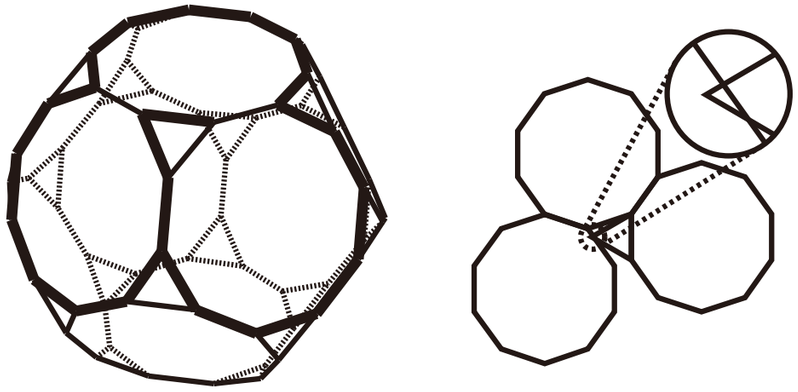
(a) Edge unfolding



(b) General unfolding

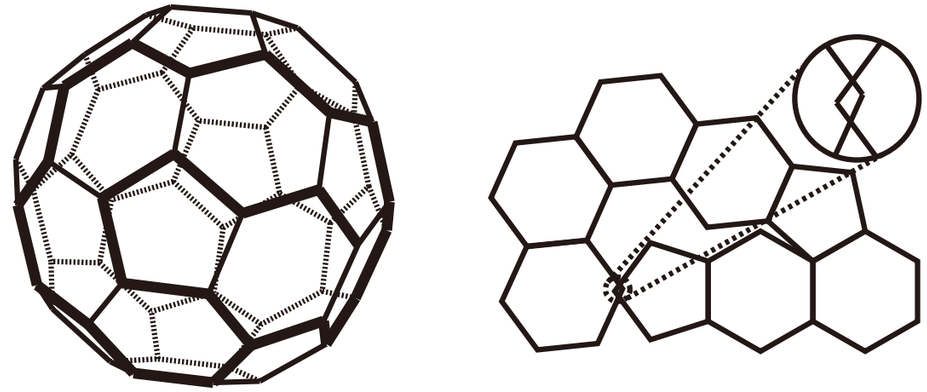
Overlapping Edge Unfoldings

Overlapping edge unfoldings exist in some convex polyhedra.



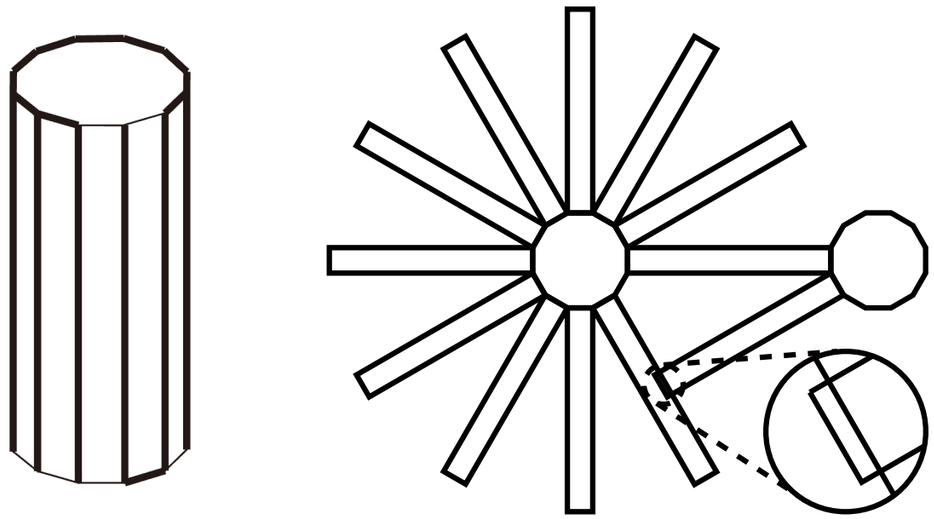
Truncated dodecahedron

[T. Horiyama and W. Shoji, 2011]



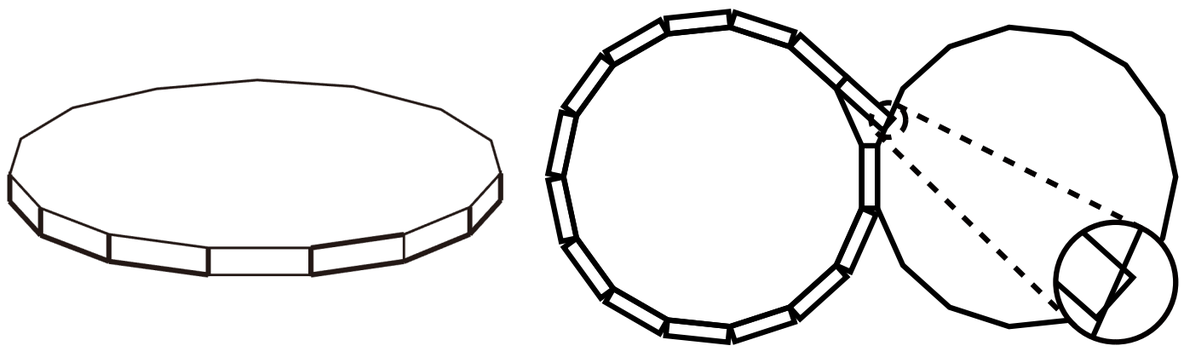
Truncated icosahedron

[T. Horiyama and W. Shoji, 2011]



12-gonal prism

[Schlickenrieder, 1997]



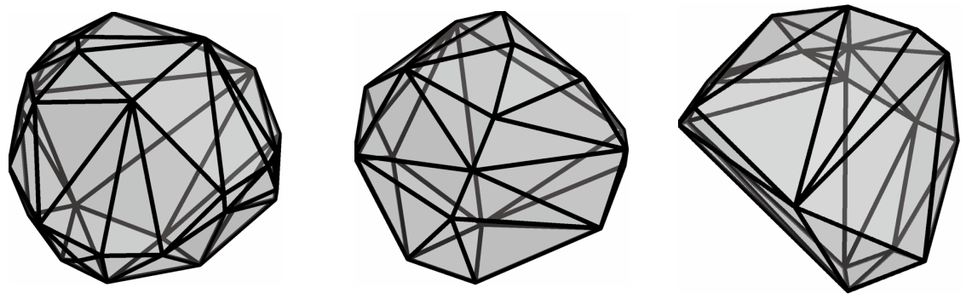
15-gonal prism

[Schlickenrieder, 1997]

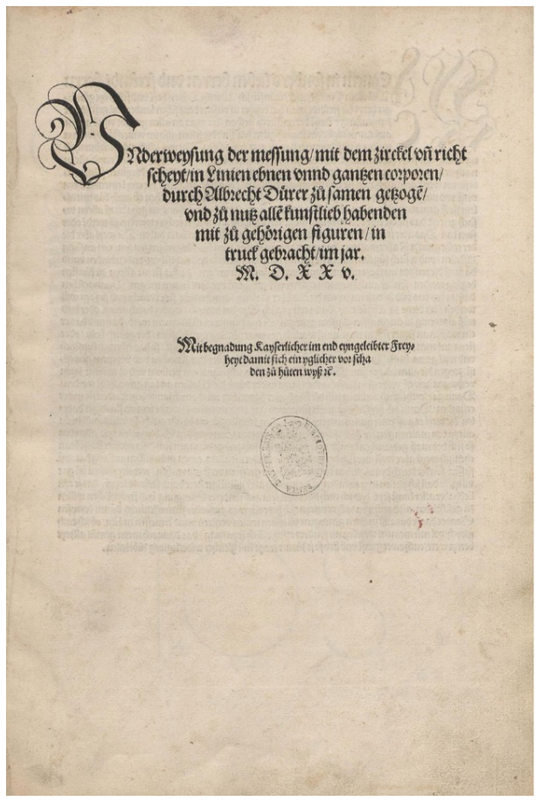
Research Motivation

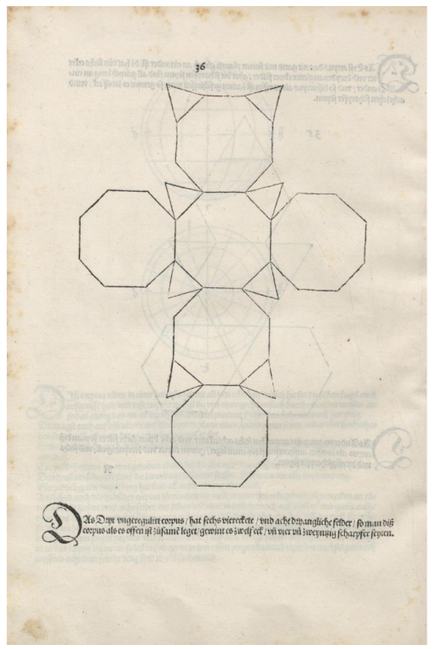
Conjecture 2 [G. C. Shephard, 1975]

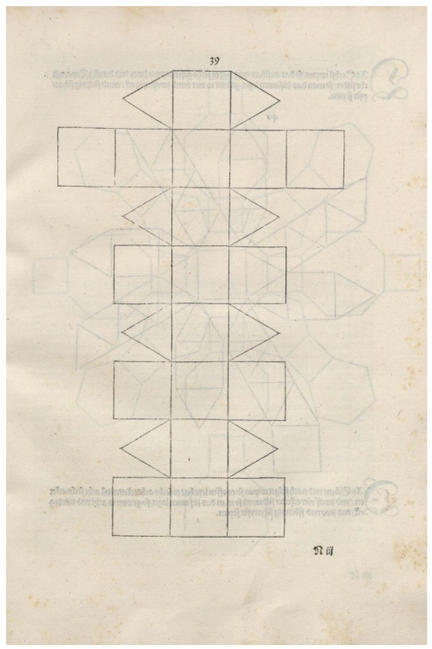
Can every convex polyhedron be edge-unfolded without overlaps?

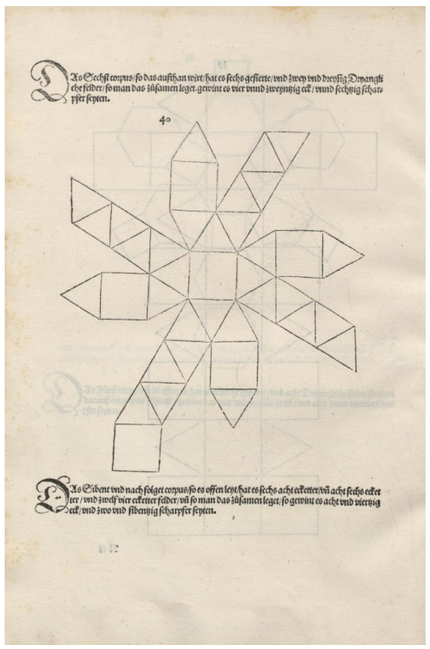


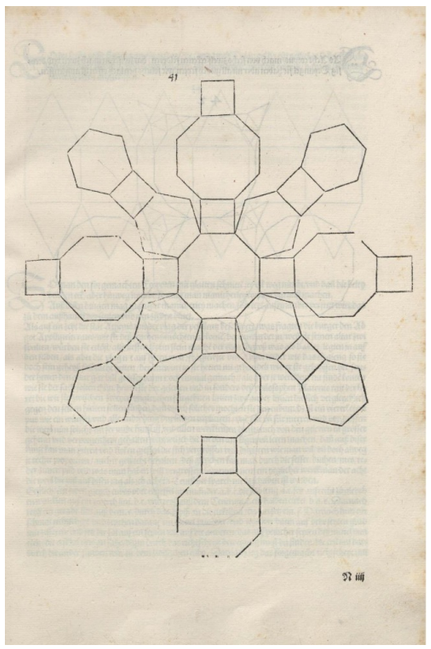
[Albrecht Dürer, 1525] “Underweysung der Messung, mit dem Zirckel und Richtscheyt, in Linien Ebenen unnd gantzen corporen”







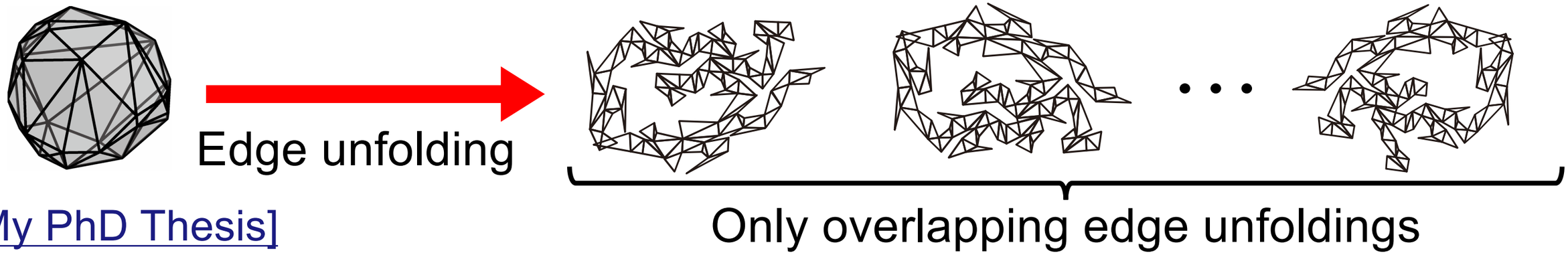




Conjecture 2 is called “Dürer’s Problem”

Approaches to Solving Dürer's Problem

Method 1 (Negative)	Find a convex polyhedron such that all its edge unfoldings result in overlaps.

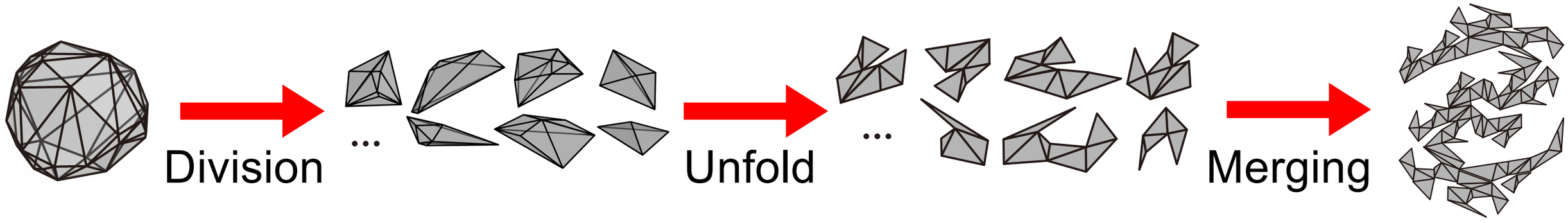


[My PhD Thesis]

Method 2 (Positive)	Develop an edge-unfolding algorithm that avoids overlaps for all convex polyhedra.

Planned Approach for Method 2

Edge-Unfolding Algorithm Based on Division and Merging



Current Phase	Next Phase	Long-Term Phase
Starting soon…		After 5–6 years

Key point of this algorithm

There is a trade-off between division and edge unfolding.

- If we divide too much → Merging becomes complex.
- If we divide too little → Need to design algorithms for a huge number of polyhedra.

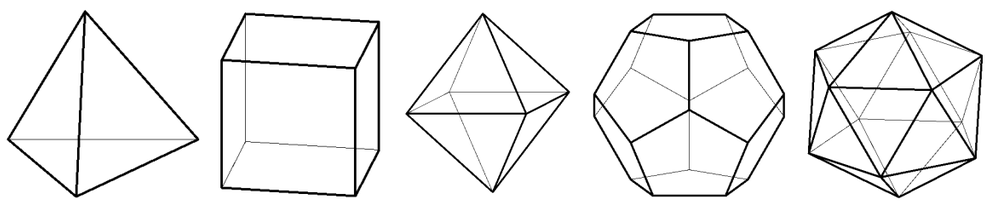
What Are Good Division Strategies?

Strategy 1	Dividing edge-overlap-free polyhedra

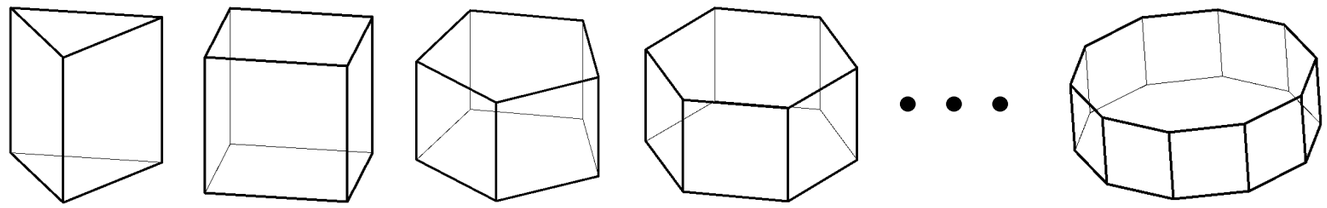
Definition 3 [T. Kamata, T. Shiota and R. Uehara, 2024]

A polyhedron is called edge-overlap-free if all of its edge unfoldings are free of overlaps.

[Ex.] All of the following polyhedra are edge-overlap-free.



Platonic solid
[T. Horiyama and W. Shoji, 2011]



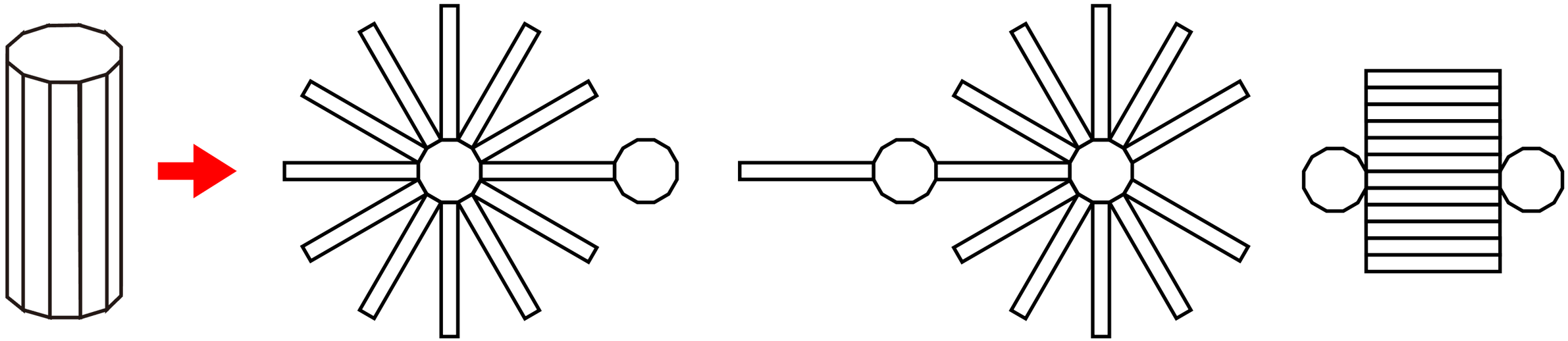
Regular 3- to 10-gonal prisms (h > 0)
[T. Kamata et al., 2025 (in Japanese only)]

Edge-overlap-free polyhedra are easy to handle because their faces can be rearranged freely at the merging phase.

What Are Good Division Strategies?

Strategy 2	Dividing polyhedra with various unfolding methods

[Ex.] Following unfoldings of a prism results in no overlaps.



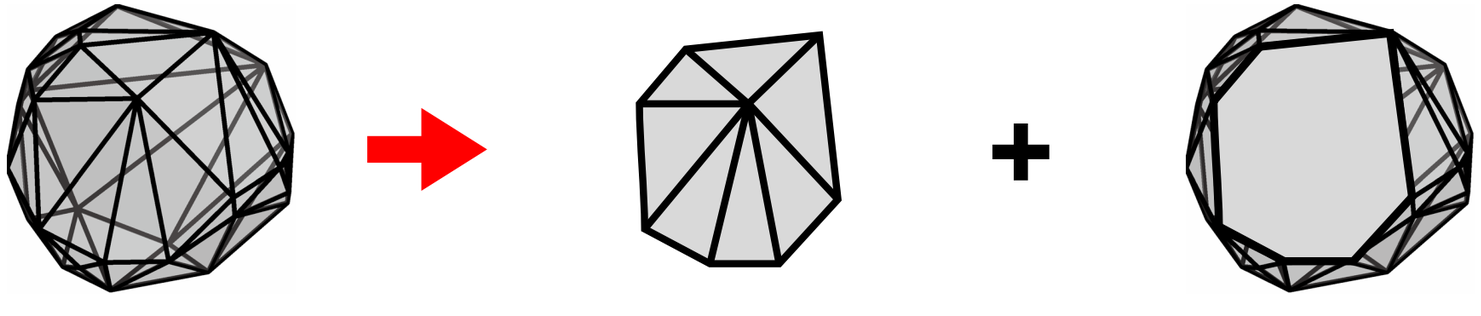
Polyhedra with various unfolding methods are relatively easy to handle because their faces can be rearranged flexibly.

We aim to characterize which polyhedra are edge-overlap-free and which ones have various unfolding methods.

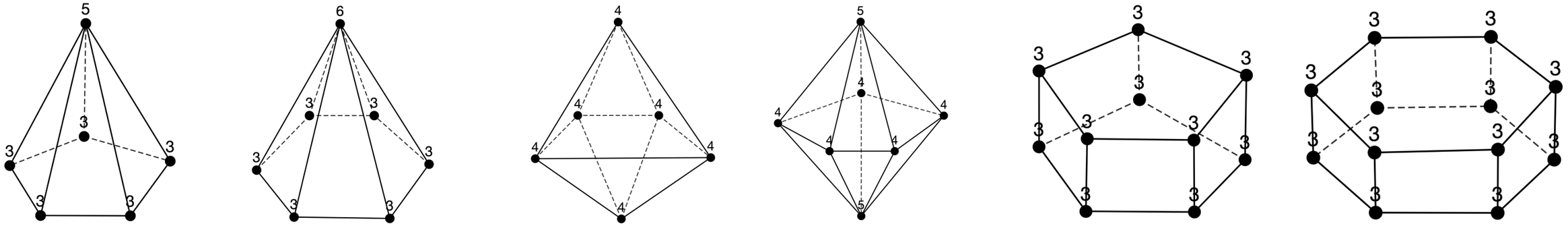
➡ As a first step, we conducted a categorization of polyhedra.

Focused Points in the Categorization

When a polyhedron is divided into a single part, the resulting cutting plane forms an n-gon.



Well-known polyhedra show vertex counts increasing by 1 or 2.



Pyramid **Bipyramid** **Prism**

We extracted series of polyhedral graphs for each number of vertices, based on shared structural characteristics.

Focused Structural Characteristics

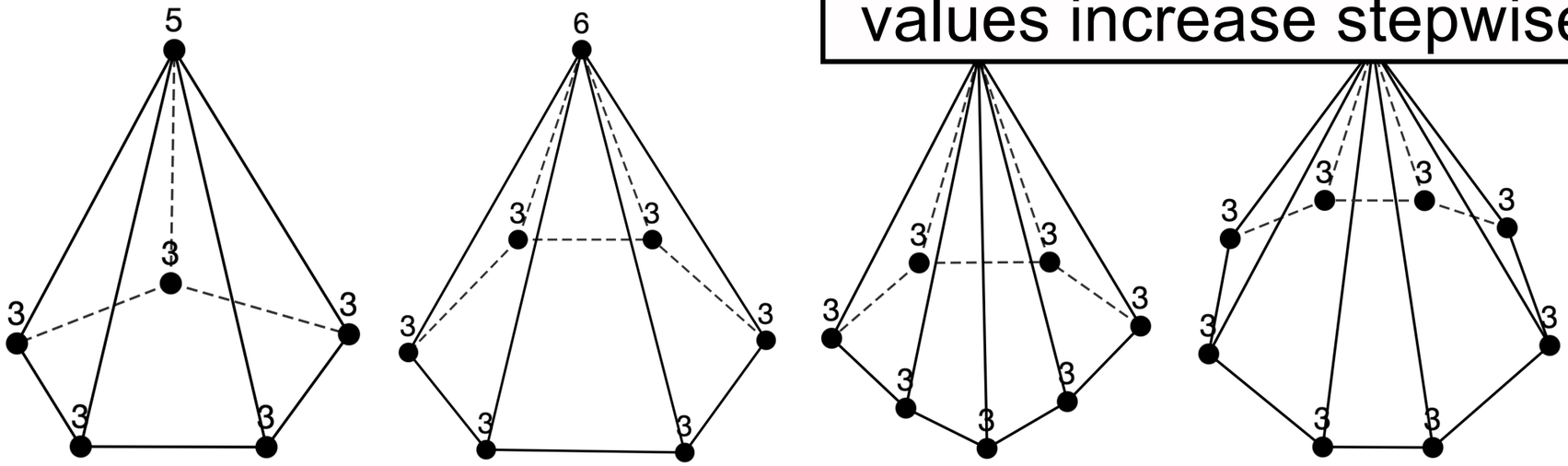
Observation

The degree distributions of well-known polyhedra follow this structure: $[d_1, i\,]\,[\,j, c_2]$

Each pair $[d,c]$ represents:

d : vertex degree c : #(vertices with degree d)

Pyramid



When n increases by 1, the values increase stepwise by 1.

```
[3,3][3,1]
[3,4][4,1]
n=6:  [3,5][5,1]
n=7:  [3,6][6,1]
n=8:  [3,7][7,1]
n=9:  [3,8][8,1]
n=10: [3,9][9,1]
```

Focused Structural Characteristics

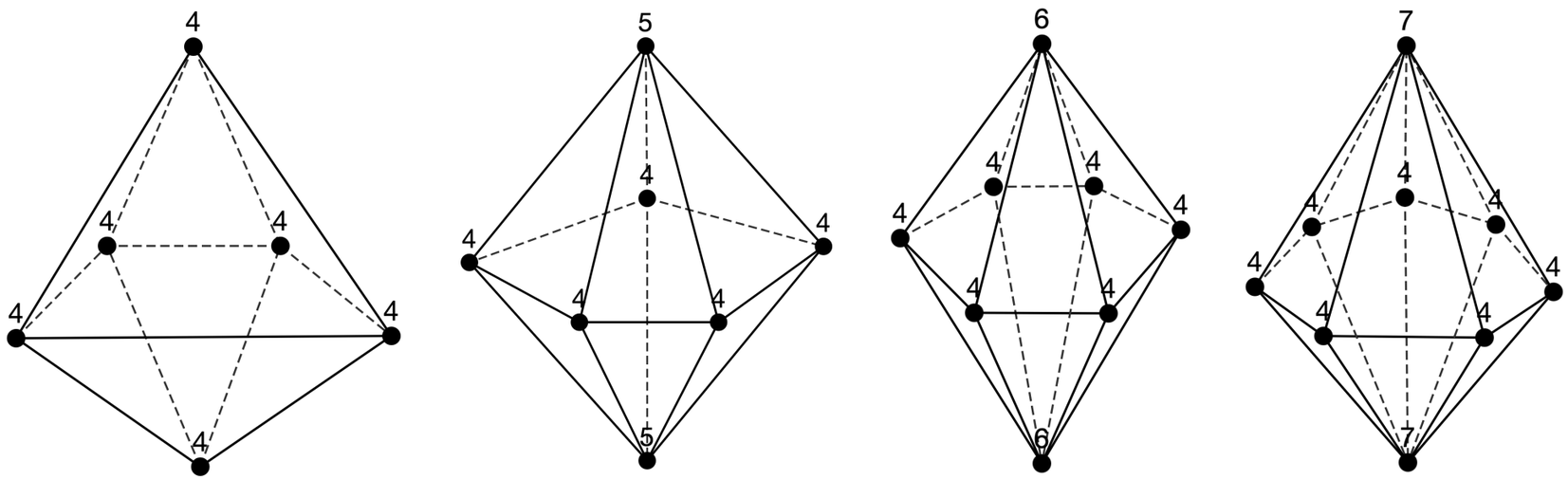
Observation

The degree distributions of well-known polyhedra follow this structure: $[d_1, i\,]\,[\,j, c_2]$

Each pair $[d,c]$ represents:

d : vertex degree c : #(vertices with degree d)

Bipyramid




```
n=5:  [4,3][3,2]
n=6:  [4,4][4,2]
n=7:  [4,5][5,2]
n=8:  [4,6][6,2]
n=9:  [4,7][7,2]
n=10: [4,8][8,2]
n=11: [4,9][9,2]
```

Focused Structural Characteristics

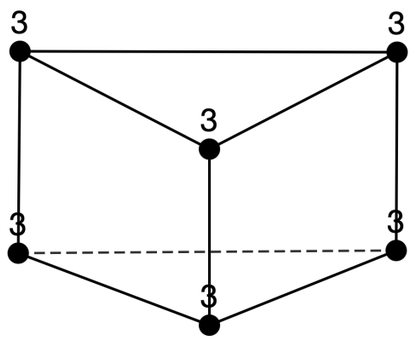
Observation

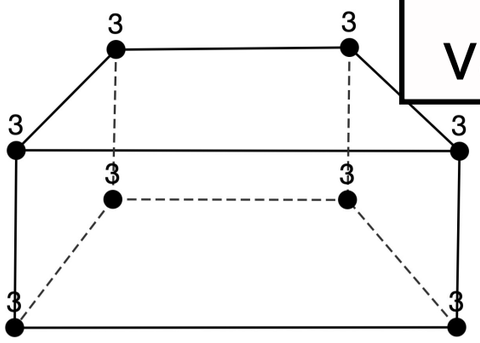
The degree distributions of well-known polyhedra follow this structure: $[d_1, i\,]\,[\,j, c_2]$

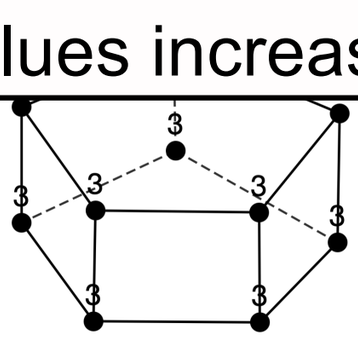
Each pair $[d,c]$ represents:

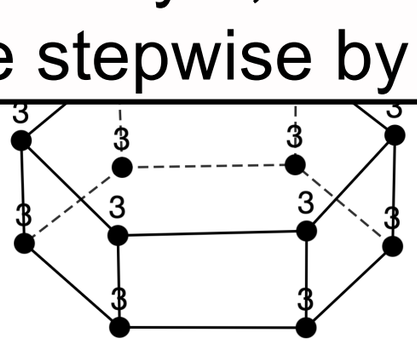
d : vertex degree c : #(vertices with degree d)

Prism









When n increases by 2, the values increase stepwise by 2.

```
n=6: [3,6]
n=8: [3,8]
n=10: [3,10]
n=12: [3,12]
```

From these observations…

We focus on sequences of the form $[d_1, c_1]$ ... $[d_{n-1}, i\,]\,[\,j, c_n]$.

Enumeration of Sequences

Program flow

1. Enumerate non-isomorphic, connected graphs with minimum degree at least 3 [python script: nauty]
2. Remove non-planar graphs [c++ program: Boost]
3. Remove graphs that are not 3-connected [python script]
4. Extract degree sequences of the form

   $[d_1, c_1] \ldots [d_{n-1}, i\,] [\, j, c_n]$ [python script]
5. Apply degree constraints [python script]
6. Export planar drawings as a JSON file [python script]

The entire program is available at:
https://github.com/ShiotaTakumi/UM2025/tree/main/src/geng_boost

List of obtained [deg., cnt.] sequences

Sequence 1

```
n=6:  [3,3][4,2][5,1]
n=7:  [3,4][4,2][6,1]
n=8:  [3,5][4,2][7,1]
n=9:  [3,6][4,2][8,1]
n=10:  [3,7][4,2][9,1]
n=11:  [3,8][4,2][10,1]
```

Sequence 2

```
n=7:  [3,6][4,1]
n=8:  [3,7][5,1]
n=9:  [3,8][6,1]
n=10:  [3,9][7,1]
n=11:  [3,10][8,1]
```

Sequence 3

```
n=7:  [3,3][4,3][5,1]
n=8:  [3,4][4,3][6,1]
n=9:  [3,5][4,3][7,1]
n=10:  [3,6][4,3][8,1]
n=11:  [3,7][4,3][9,1]
```

Sequence 4

```
n=7:  [3,1][4,5][5,1]
n=8:  [3,2][4,5][6,1]
n=9:  [3,3][4,5][7,1]
n=10:  [3,4][4,5][8,1]
n=11:  [3,5][4,5][9,1]
```

List of obtained [deg., cnt.] sequences

Sequence 5

```
n=7:  [3,2][4,2][5,2][6,1]
n=8:  [3,3][4,2][5,2][7,1]
n=9:  [3,4][4,2][5,2][8,1]
n=10:  [3,5][4,2][5,2][9,1]
n=11:  [3,6][4,2][5,2][10,1]
```

Sequence 6

```
n=7:  [3,2][4,4][6,1]
n=8:  [3,3][4,4][7,1]
n=9:  [3,4][4,4][8,1]
n=10:  [3,5][4,4][9,1]
n=11:  [3,6][4,4][10,1]
```

Sequence 7

```
n=7:  [3,3][5,3][6,1]
n=8:  [3,4][5,3][7,1]
n=9:  [3,5][5,3][8,1]
n=10:  [3,6][5,3][9,1]
n=11:  [3,7][5,3][10,1]
```

Sequence 8

```
n=7:  [3,2][4,2][5,2][6,1]
n=8:  [3,2][4,2][5,3][7,1]
n=9:  [3,2][4,2][5,4][8,1]
n=10:  [3,2][4,2][5,5][9,1]
n=11:  [3,2][4,2][5,6][10,1]
```

List of obtained [deg., cnt.] sequences

Sequence 9

```
n=8:  [3,4][4,2][5,2]
n=9:  [3,4][4,3][6,2]
n=10:  [3,4][4,4][7,2]
n=11:  [3,4][4,5][8,2]
```

Sequence 10

```
n=8:  [3,3][4,4][5,1]
n=9:  [3,4][4,4][6,1]
n=10:  [3,5][4,4][7,1]
n=11:  [3,6][4,4][8,1]
```

Sequence 11

```
n=8:  [3,5][4,2][5,1]
n=9:  [3,6][4,2][6,1]
n=10:  [3,7][4,2][7,1]
n=11:  [3,8][4,2][8,1]
```

Sequence 12

```
n=8:  [3,3][4,3][5,1][6,1]
n=9:  [3,3][4,3][5,2][7,1]
n=10:  [3,3][4,3][5,3][8,1]
n=11:  [3,3][4,3][5,4][9,1]
```

List of obtained [deg., cnt.] sequences

Sequence 13

```
n=8:  [3,2][4,3][5,2][6,1]
n=9:  [3,3][4,3][5,2][7,1]
n=10:  [3,4][4,3][5,2][8,1]
n=11:  [3,5][4,3][5,2][9,1]
```

Sequence 14

```
n=8:  [3,2][4,3][5,2][6,1]
n=9:  [3,2][4,3][5,3][7,1]
n=10:  [3,2][4,3][5,4][8,1]
n=11:  [3,2][4,3][5,5][9,1]
```

Sequence 15

```
n=8:  [3,3][4,2][5,1][6,2]
n=9:  [3,3][4,3][5,1][7,2]
n=10:  [3,3][4,4][5,1][8,2]
n=11:  [3,3][4,5][5,1][9,2]
```

Sequence 16

```
n=8:  [3,4][4,2][5,1][7,1]
n=9:  [3,4][4,2][5,2][8,1]
n=10:  [3,4][4,2][5,3][9,1]
n=11:  [3,4][4,2][5,4][10,1]
```

List of obtained [deg., cnt.] sequences

Sequence 17

```
n=8:  [3,4][4,2][6,2]
n=9:  [3,4][4,3][7,2]
n=10:  [3,4][4,4][8,2]
n=11:  [3,4][4,5][9,2]
```

Sequence 18

```
n=8:  [3,3][4,2][5,2][7,1]
n=9:  [3,3][4,2][5,3][8,1]
n=10:  [3,3][4,2][5,4][9,1]
n=11:  [3,3][4,2][5,5][10,1]
```

Sequence 19

```
n=8:  [3,1][4,6][5,1]
n=9:  [3,2][4,6][6,1]
n=10:  [3,3][4,6][7,1]
n=11:  [3,4][4,6][8,1]
```

Sequence 20

```
n=8:  [3,2][4,4][5,1][7,1]
n=9:  [3,2][4,4][5,2][8,1]
n=10:  [3,2][4,4][5,3][9,1]
n=11:  [3,2][4,4][5,4][10,1]
```

List of obtained [deg., cnt.] sequences

Sequence 21

```
n=8:  [3,1][4,3][5,3][6,1]
n=9:  [3,2][4,3][5,3][7,1]
n=10:  [3,3][4,3][5,3][8,1]
n=11:  [3,4][4,3][5,3][9,1]
```

Sequence 22

```
n=8:  [3,1][4,4][5,1][6,2]
n=9:  [3,1][4,5][5,1][7,2]
n=10:  [3,1][4,6][5,1][8,2]
n=11:  [3,1][4,7][5,1][9,2]
```

Sequence 23

```
n=8:  [3,2][4,2][5,3][7,1]
n=9:  [3,3][4,2][5,3][8,1]
n=10:  [3,4][4,2][5,3][9,1]
n=11:  [3,5][4,2][5,3][10,1]
```

List of obtained [deg., cnt.] sequences

Sequence 24

```
n=6:  [3,3][4,2][5,1]
n=8:  [3,3][4,4][7,1]
n=10:  [3,3][4,6][9,1]
```

Sequence 25

```
n=6:  [3,4][4,2]
n=8:  [3,6][5,2]
n=10:  [3,8][6,2]
```

Sequence 26

```
n=7:  [3,3][4,3][5,1]
n=9:  [3,3][4,5][7,1]
n=11:  [3,3][4,7][9,1]
```

Sequence 27

```
n=7:  [3,2][4,3][5,2]
n=9:  [3,4][4,3][6,2]
n=11:  [3,6][4,3][7,2]
```

Sequence 28

```
n=7:  [3,4][4,1][5,2]
n=9:  [3,4][4,3][6,2]
n=11:  [3,4][4,5][7,2]
```

Sequence 29

```
n=7:  [3,4][4,1][5,2]
n=9:  [3,6][4,1][6,2]
n=11:  [3,8][4,1][7,2]
```

List of obtained [deg., cnt.] sequences

Sequence 30

```
n=7:  [3,4][4,2][6,1]
n=9:  [3,4][4,4][8,1]
n=11:  [3,4][4,6][10,1]
```

Sequence 31

```
n=7:  [3,5][4,1][5,1]
n=9:  [3,5][4,3][7,1]
n=11:  [3,5][4,5][9,1]
```

Sequence 32

```
n=7:  [3,2][4,2][5,2][6,1]
n=9:  [3,2][4,4][5,2][8,1]
n=11:  [3,2][4,6][5,2][10,1]
```

Sequence 33

```
n=7:  [3,2][4,3][6,2]
n=9:  [3,4][4,3][7,2]
n=11:  [3,6][4,3][8,2]
```

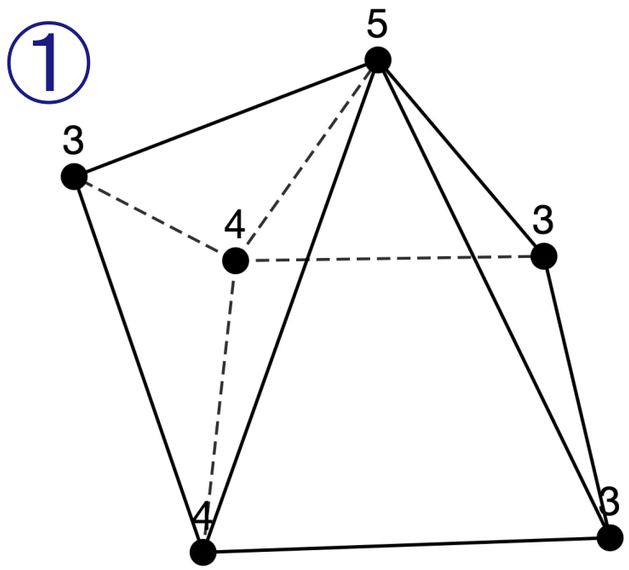
Sequence 34

```
n=7:  [3,2][4,4][6,1]
n=9:  [3,2][4,6][8,1]
n=11:  [3,2][4,8][10,1]
```

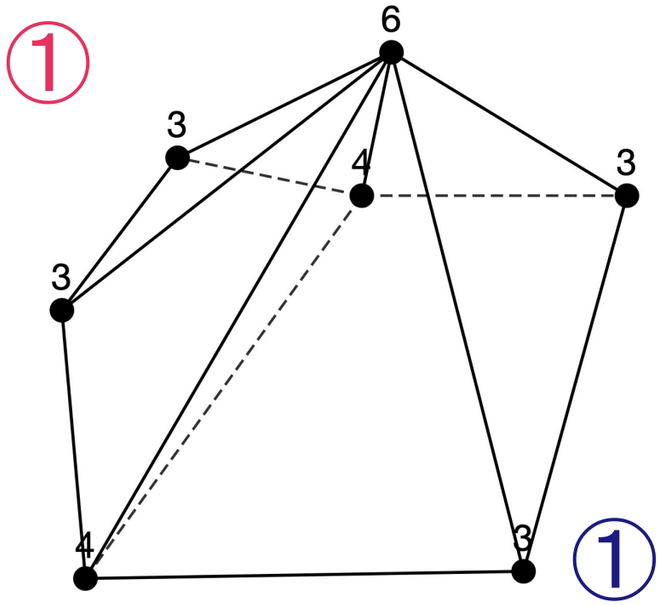
Sequence 1

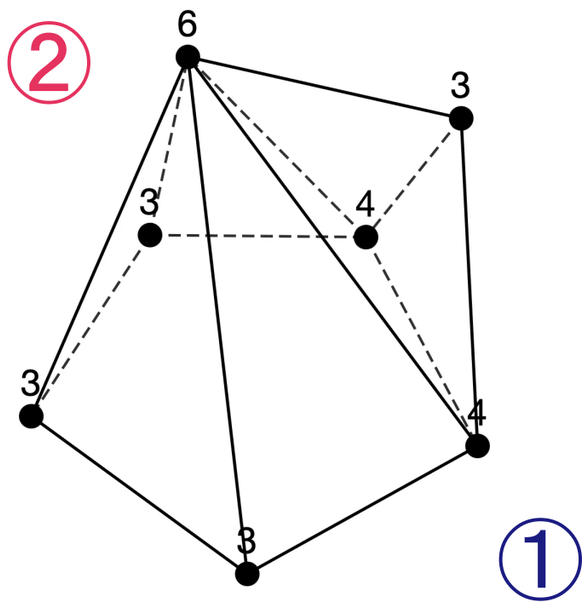
```
n=6:  [3,3][4,2][5,1]
n=7:  [3,4][4,2][6,1]
n=8:  [3,5][4,2][7,1]
n=9:  [3,6][4,2][8,1]
n=10:  [3,7][4,2][9,1]
n=11:  [3,8][4,2][10,1]
```

n = 6 (1 type)



n = 7 (2 types)

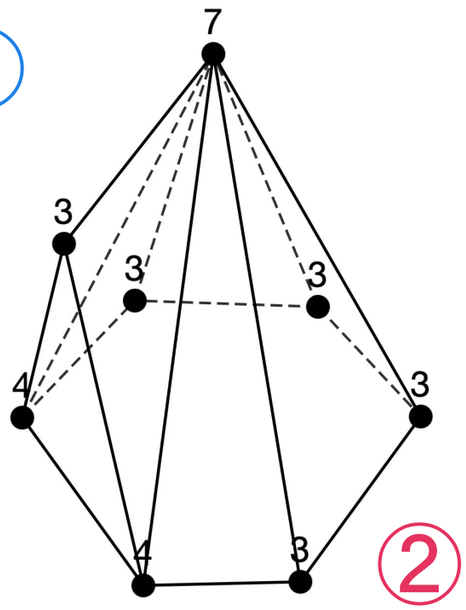




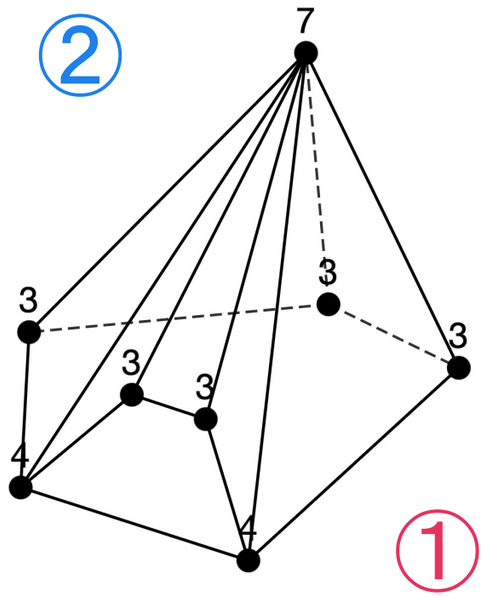
Sequence 1

n = 8 (2 types)

①



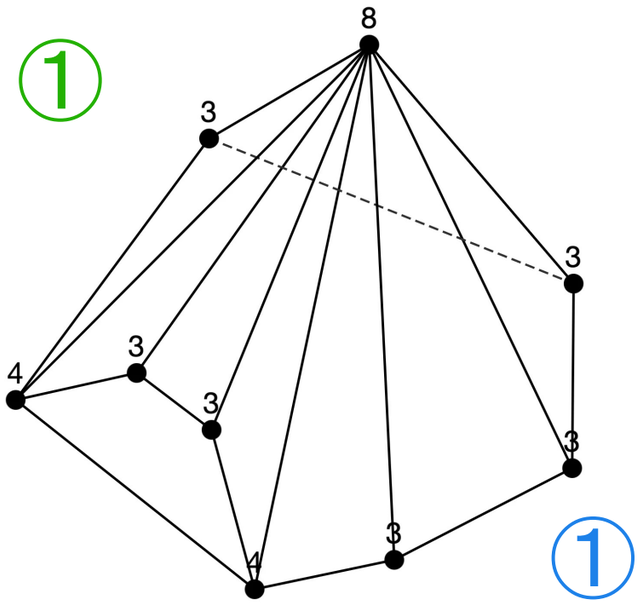
②



① ②

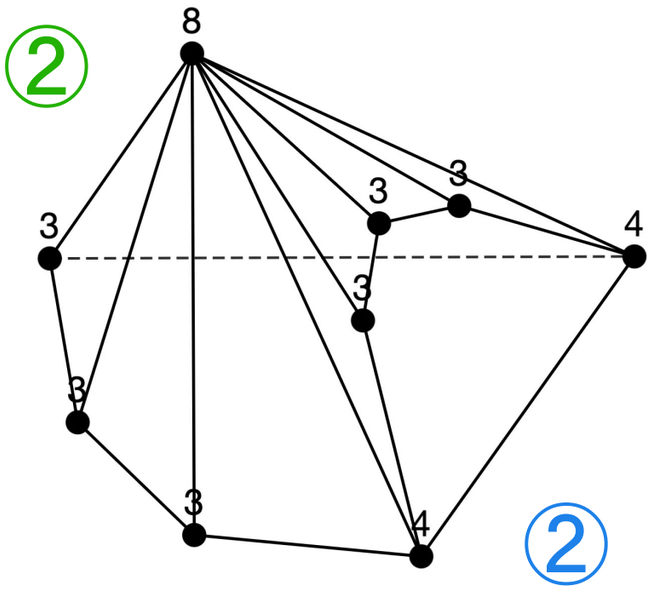
n = 9 (3 types)

①



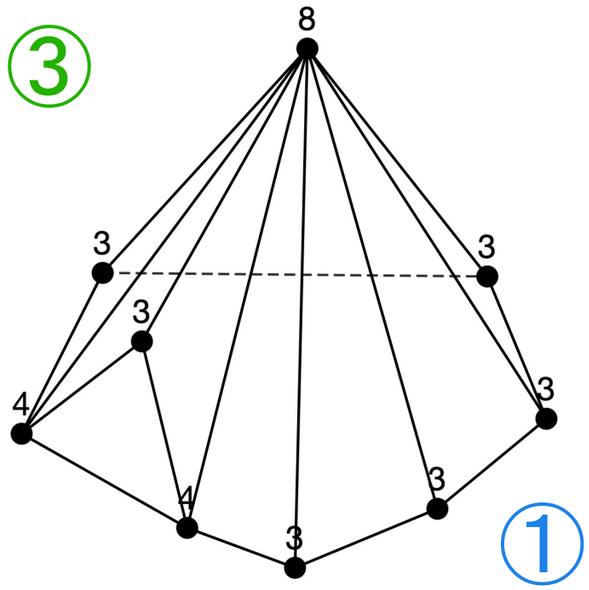
①

②



②

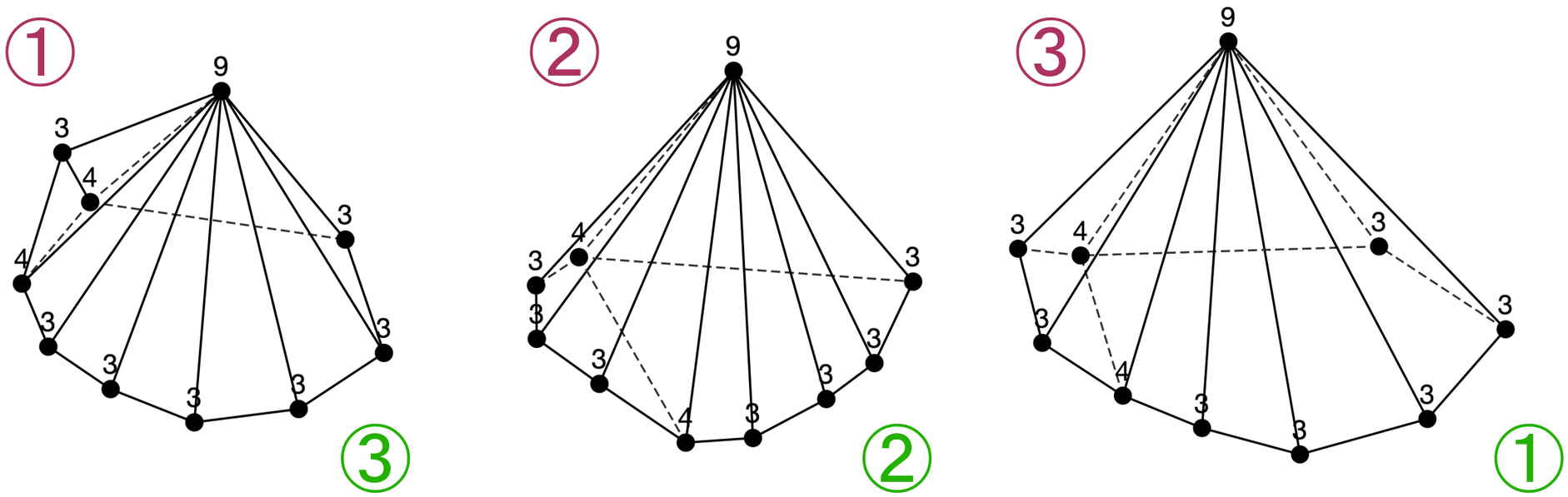
③



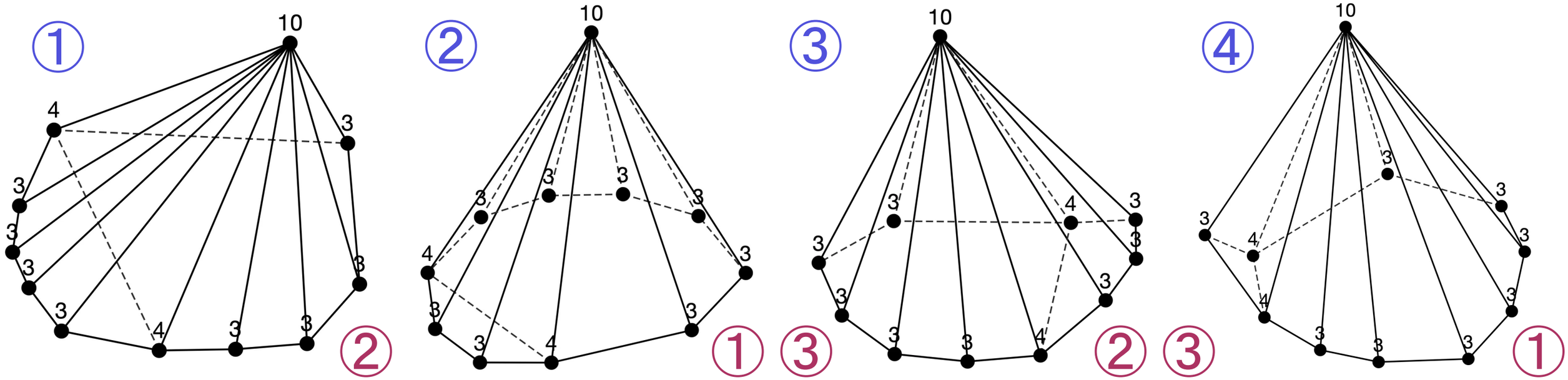
①

Sequence 1

n = 10 (3 types)



n = 11 (4 types)

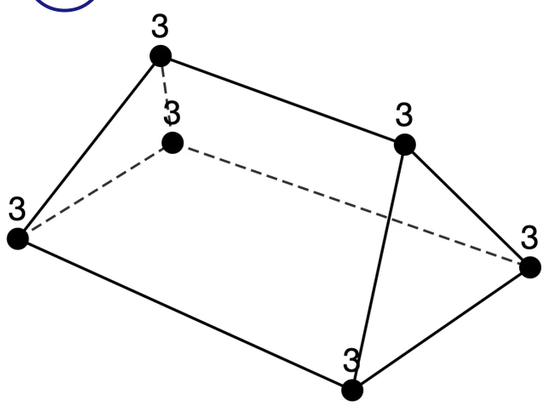


Sequence 2

```
n=6: [3,5][3,1] (=[3,6])
n=7: [3,6][4,1]
n=8: [3,7][5,1]
n=9: [3,8][6,1]
n=10: [3,9][7,1]
n=11: [3,10][8,1]
```

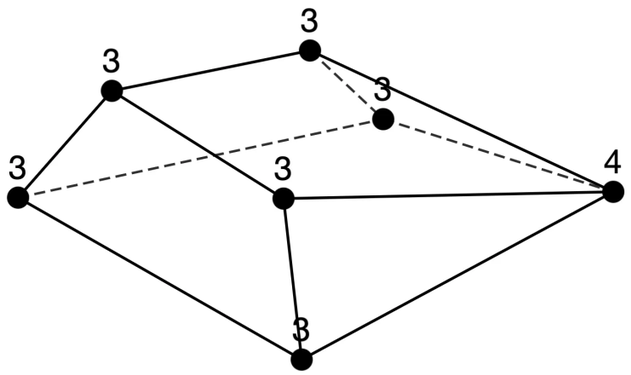
n = 6 (1 type)

①

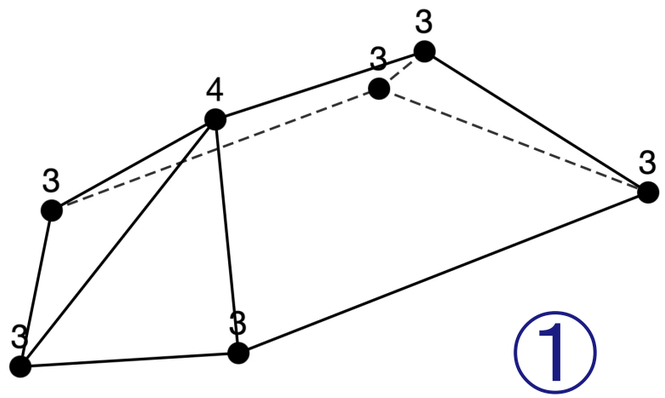


n = 7 (2 types)

①



②

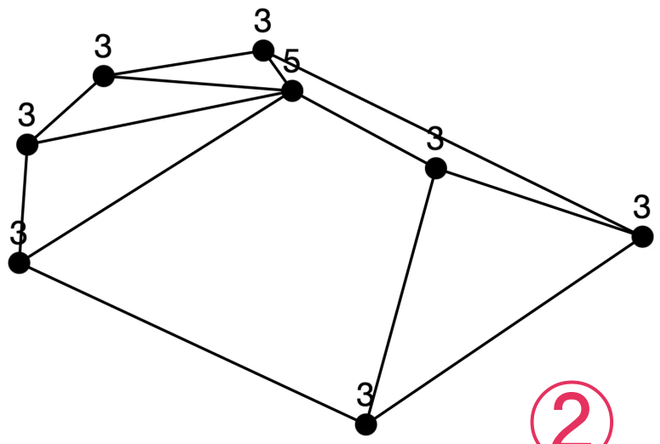


①

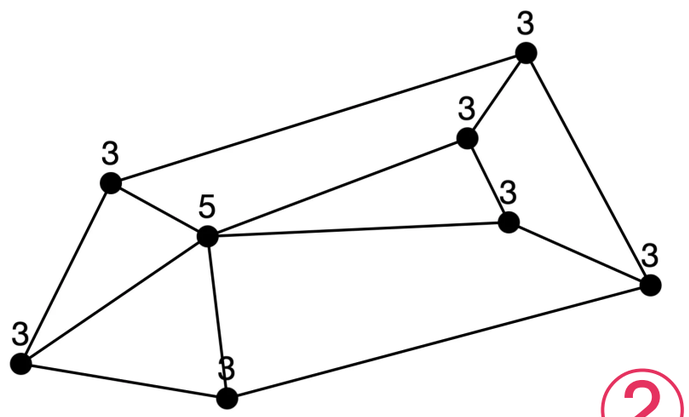
Sequence 2

n = 8 (2 types)

①

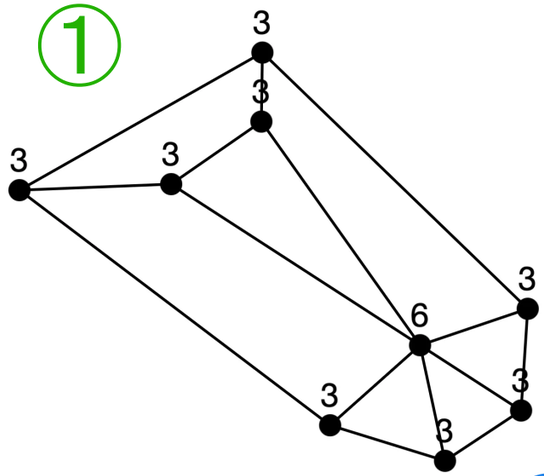


②

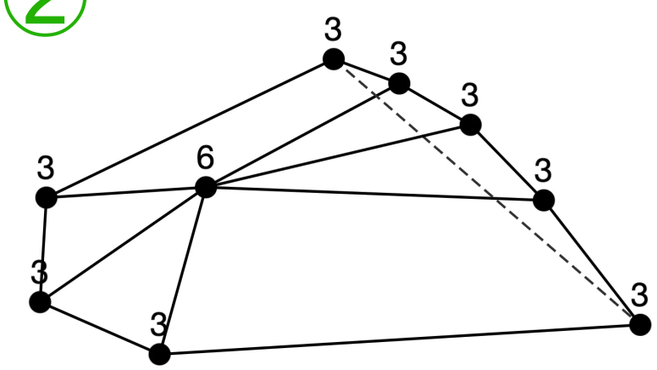


n = 9 (3 types)

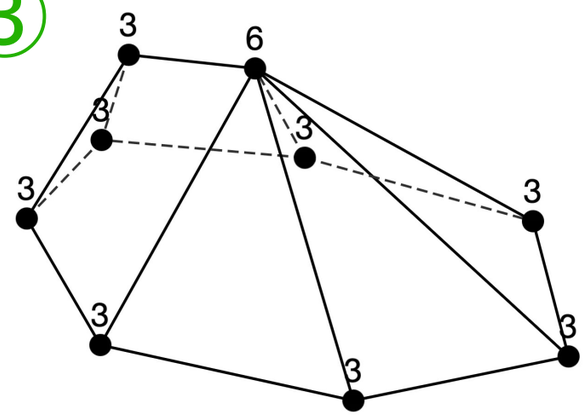
①



②



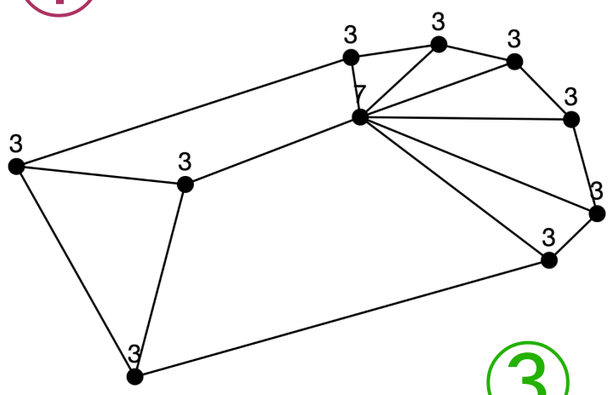
③



Sequence 2

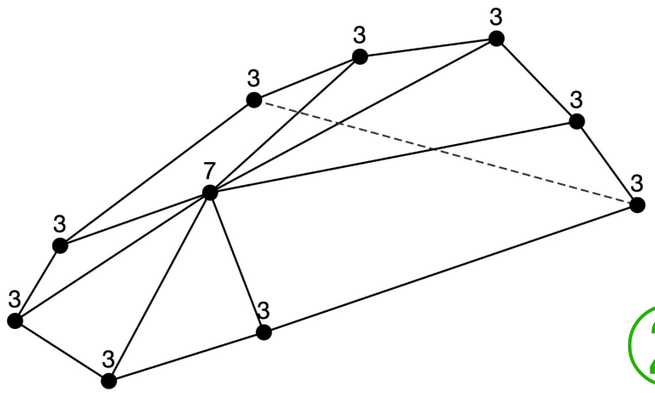
n = 10 (3 types)

①



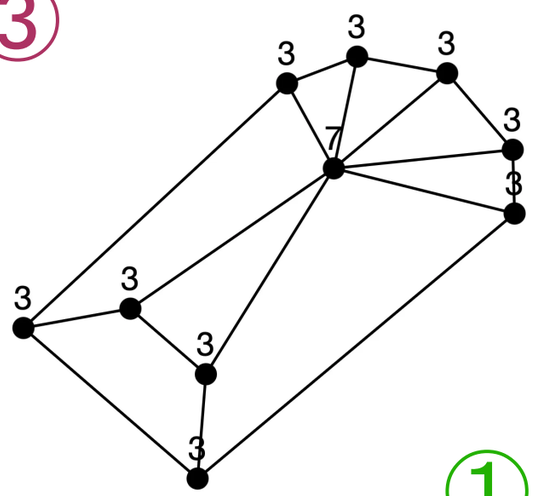
③

②



②

③

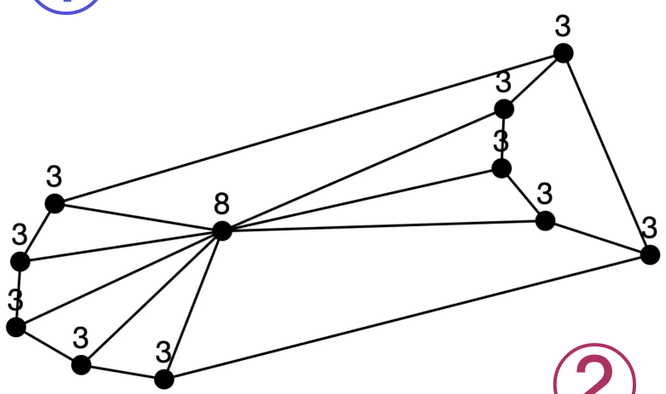


①

Sequence 2

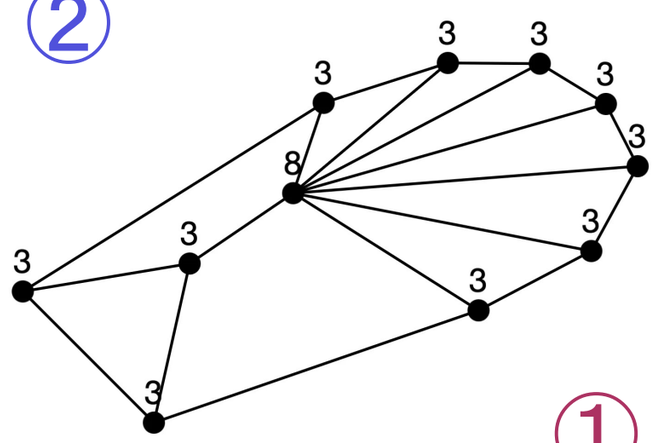
n = 11 (4 types)

①



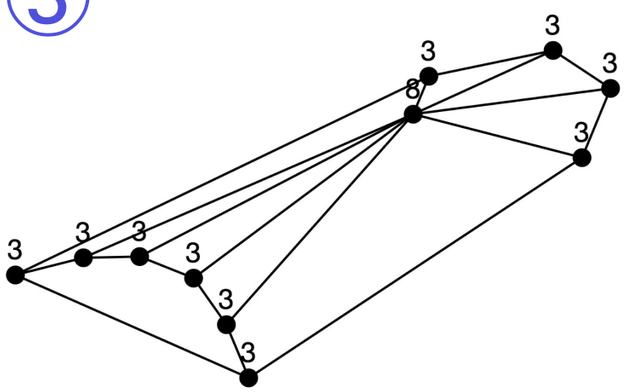
②

②



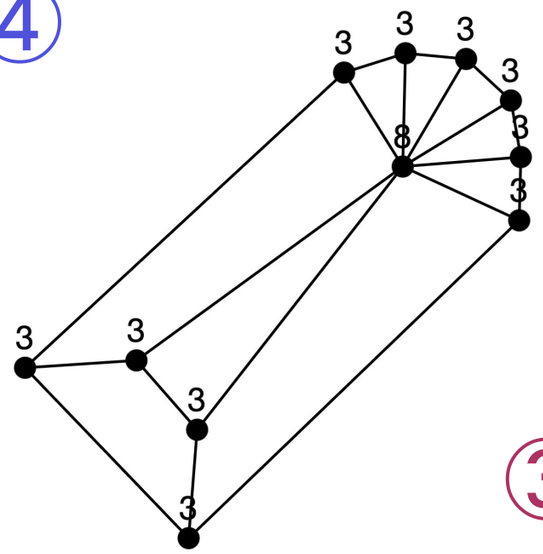
①

③



②

④

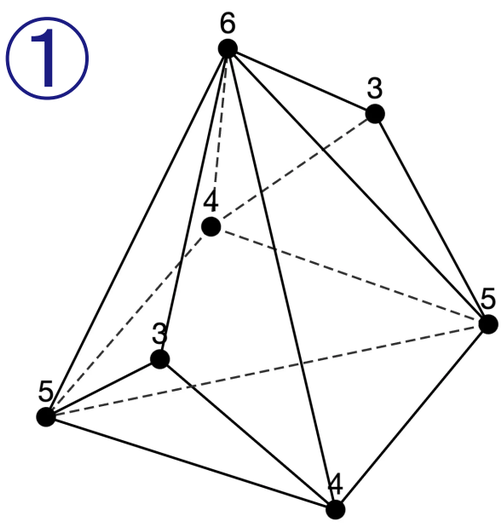


③

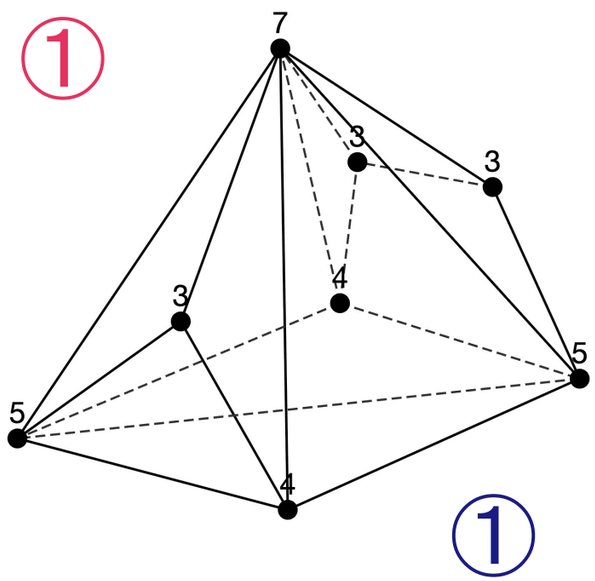
Sequence 5

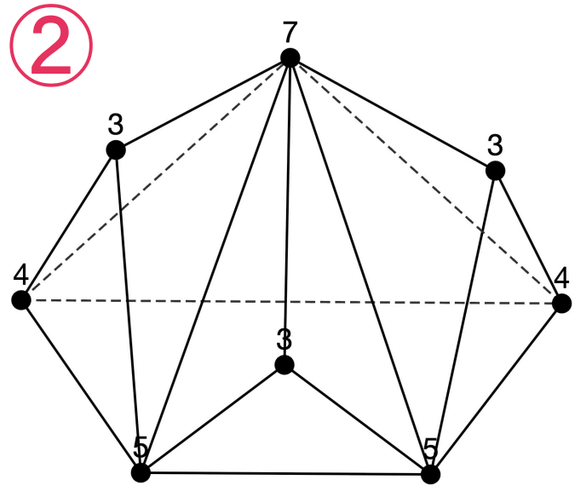
```
n=7:  [3,2][4,2][5,2][6,1]
n=8:  [3,3][4,2][5,2][7,1]
n=9:  [3,4][4,2][5,2][8,1]
n=10:  [3,5][4,2][5,2][9,1]
n=11:  [3,6][4,2][5,2][10,1]
```

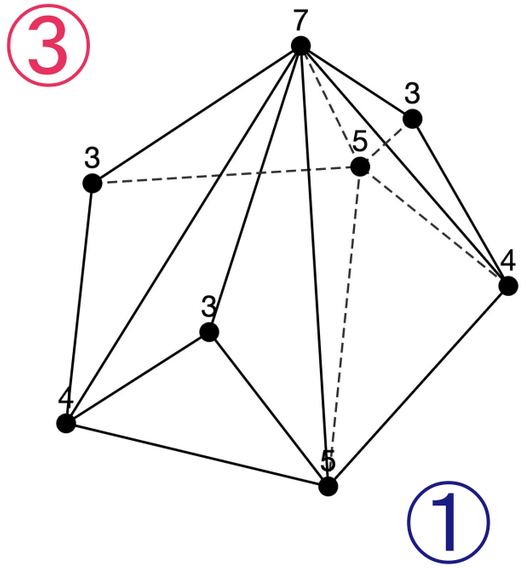
n = 7 (1 type)



n = 8 (3 types)



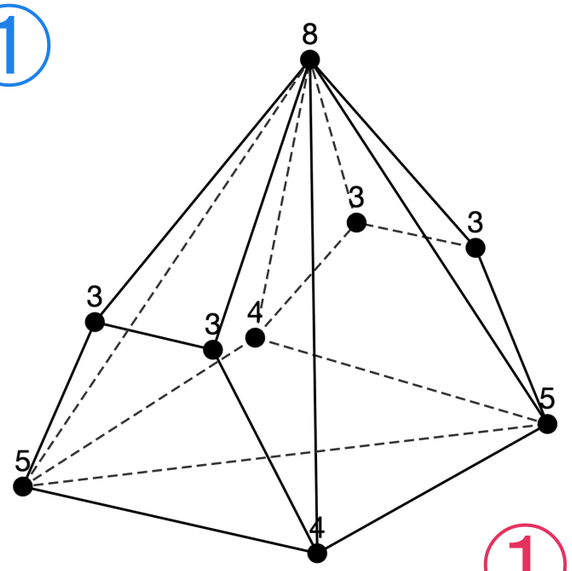




Sequence 5

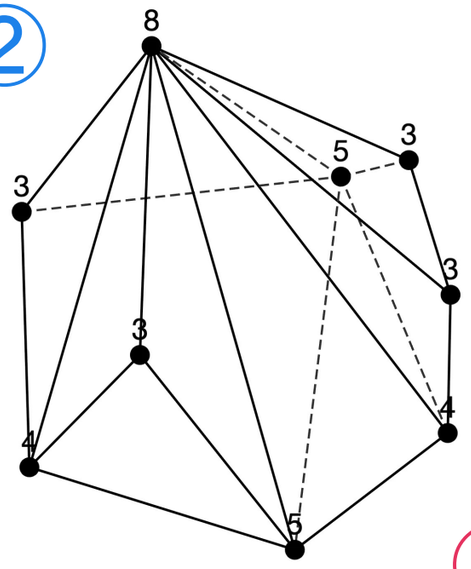
n = 9 (9 types)

①



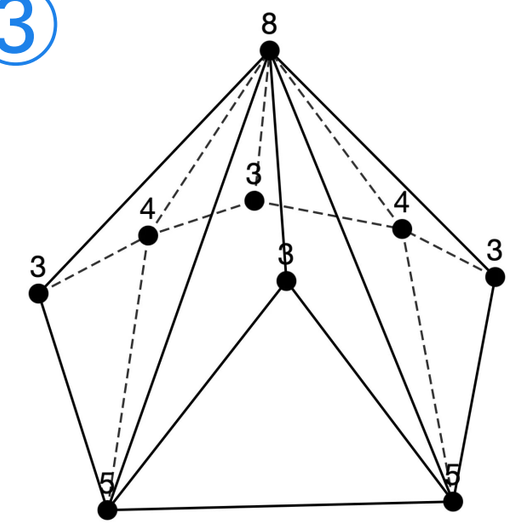
①

②



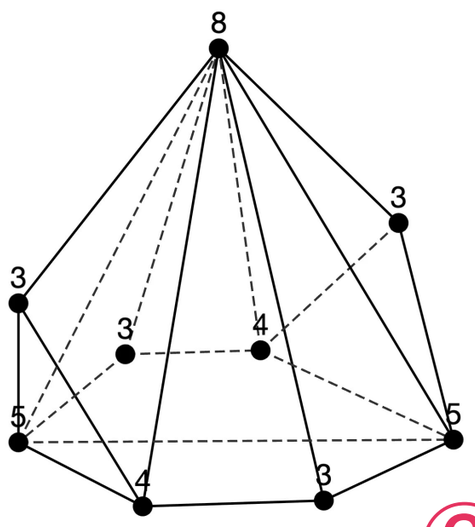
③

③



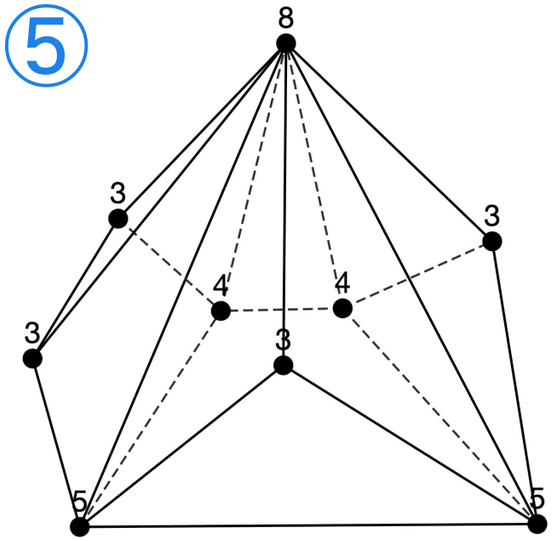
②

④



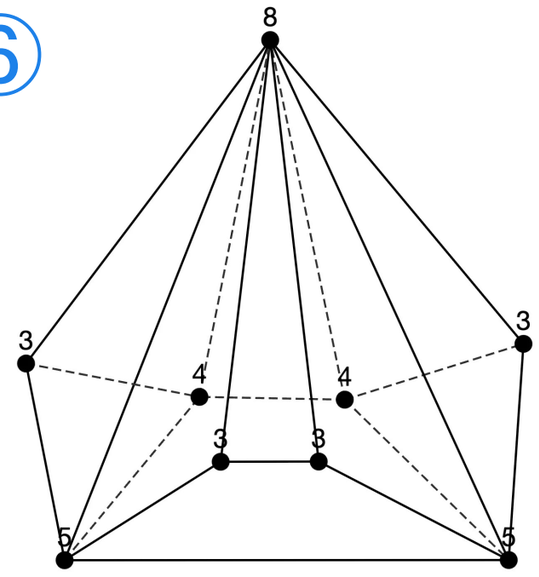
③

⑤



②

⑥

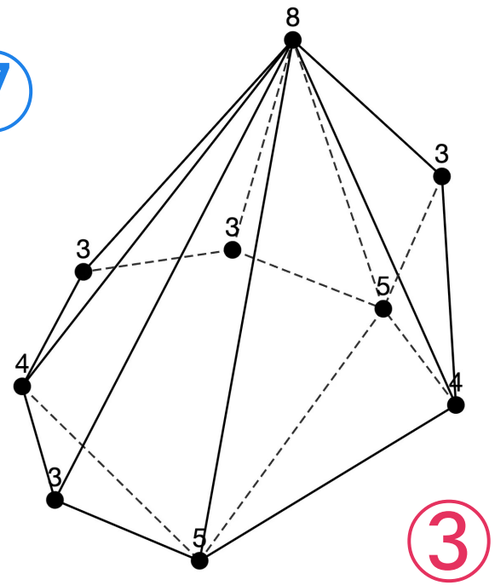


②

Sequence 5

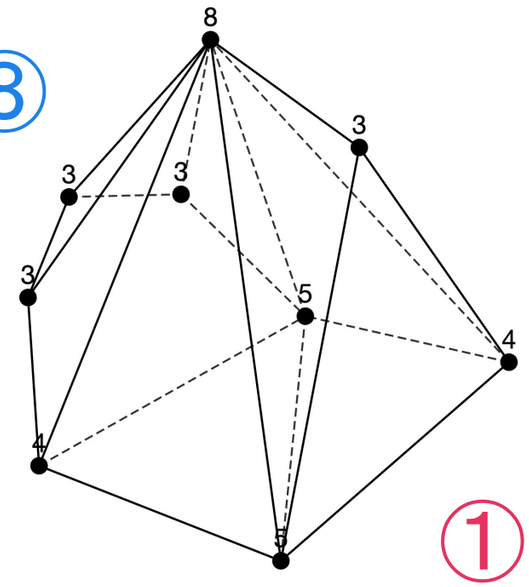
n = 9 (continued)

⑦



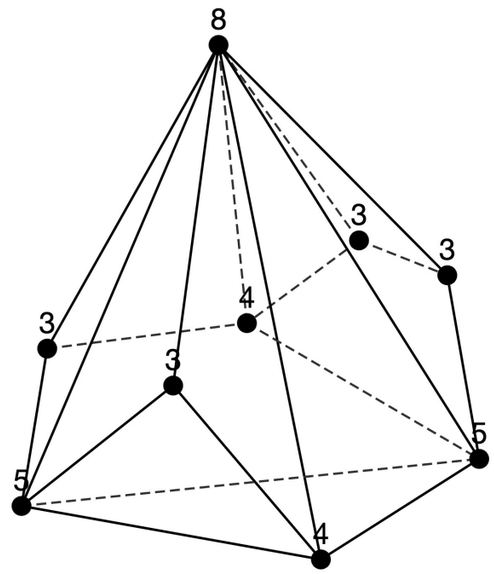
③

⑧



①

⑨

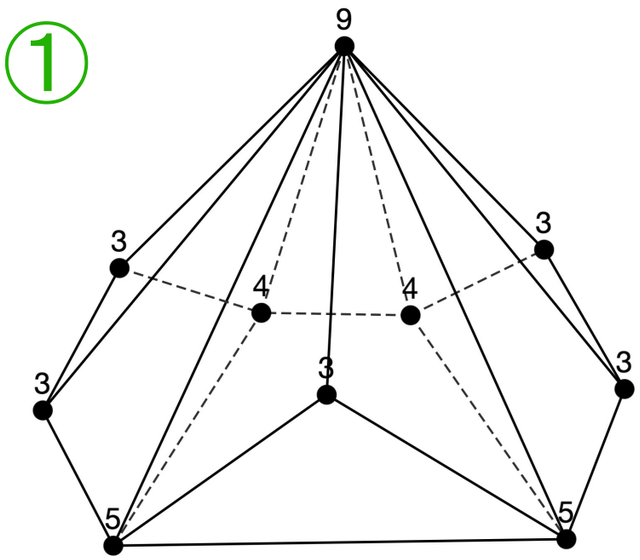


①

Sequence 5

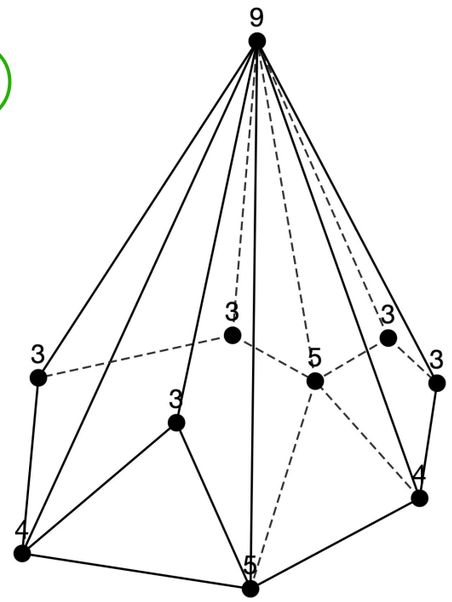
n = 10 (17 types)

①



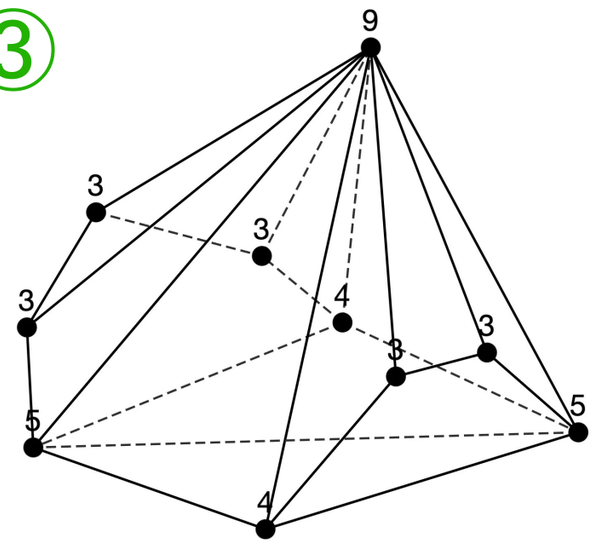
⑤

②



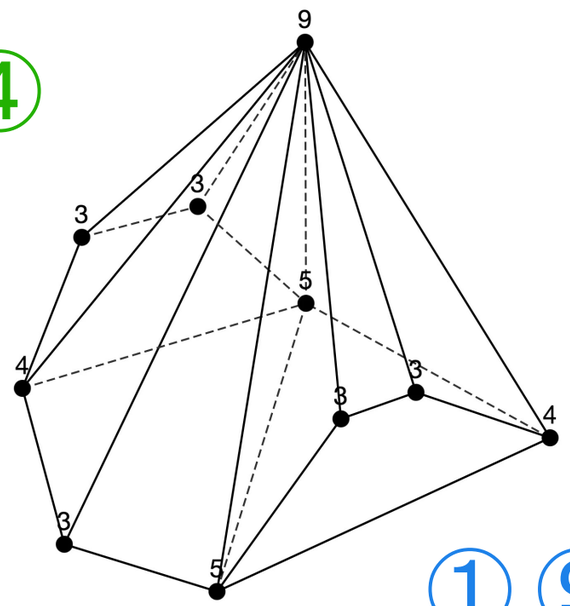
②

③



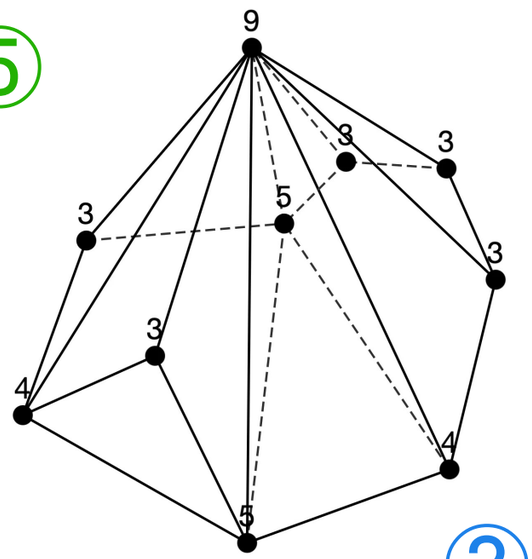
①

④



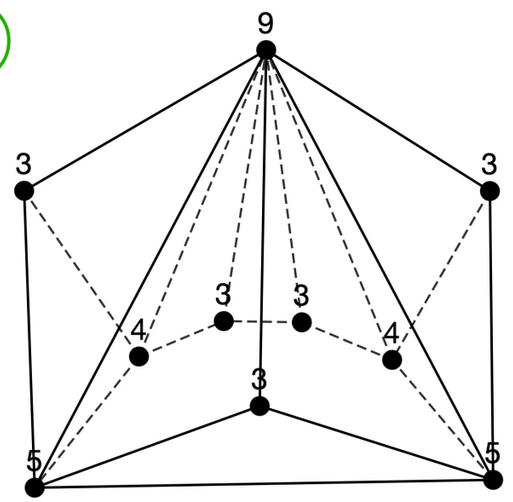
① ⑨

⑤



②

⑥

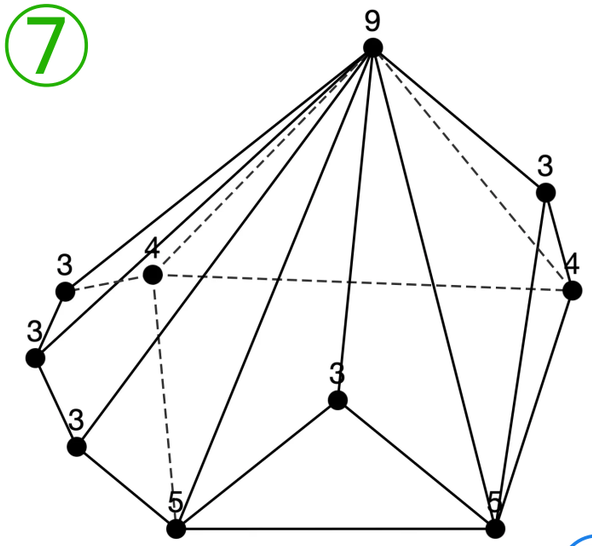


③

Sequence 5

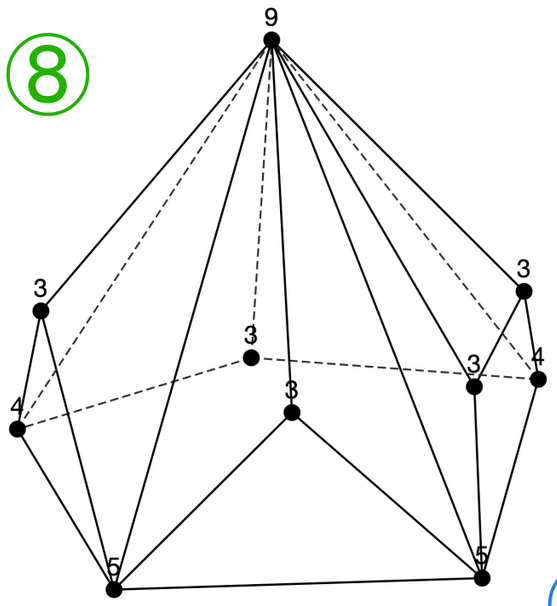
n = 10 (continued)

⑦



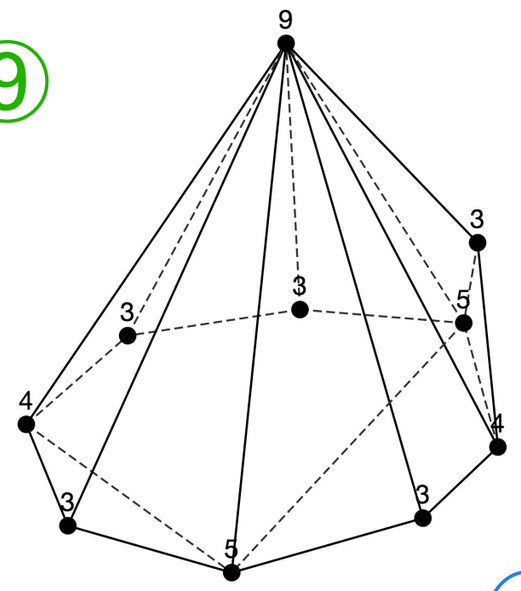
5

⑧



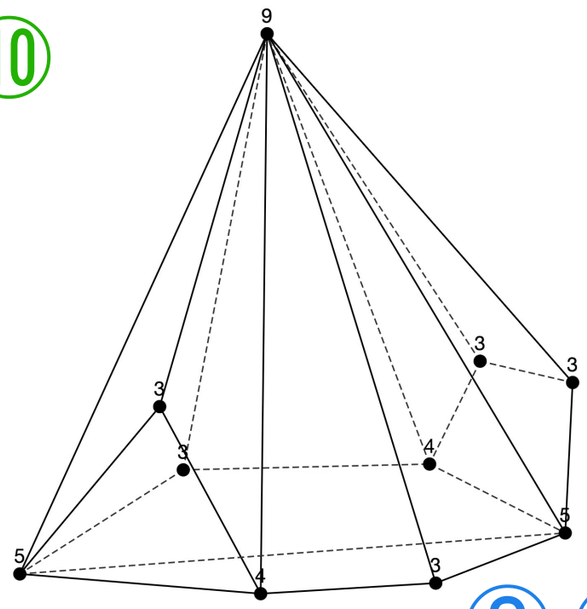
3 5

⑨



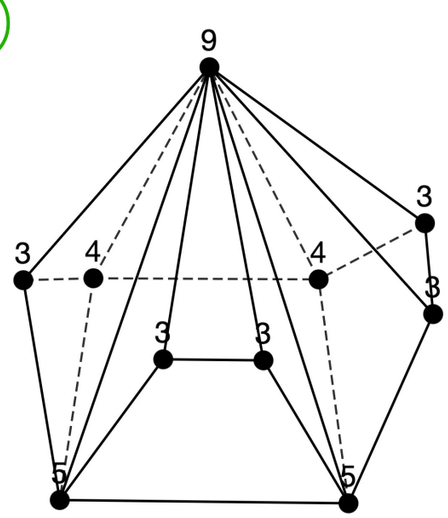
4 7

⑩



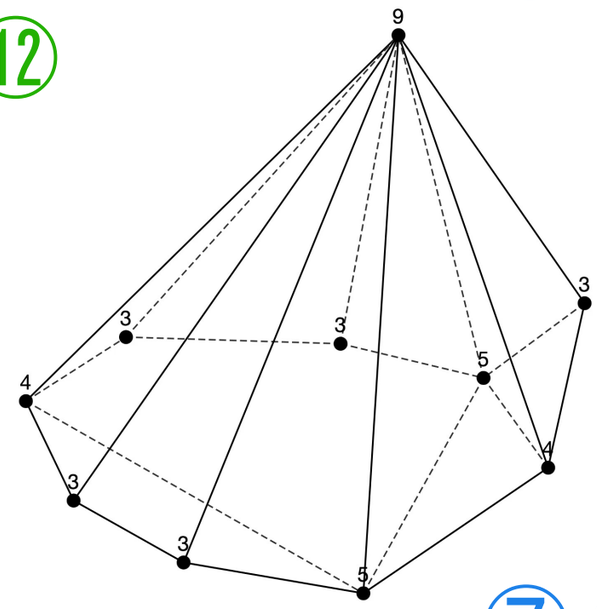
2 4 9

⑪



5 6

⑫

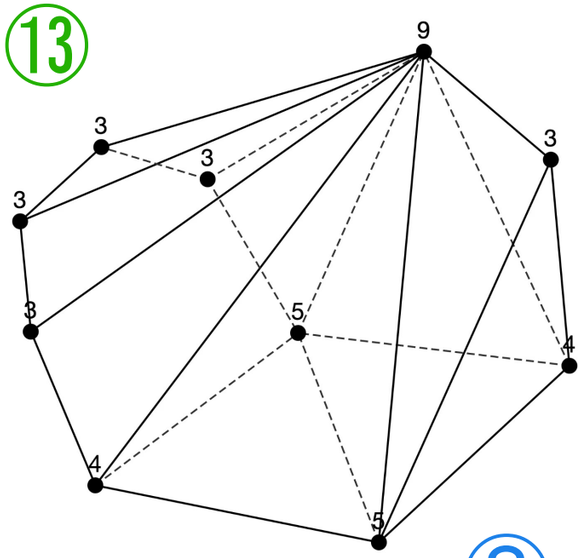


7 9

Sequence 5

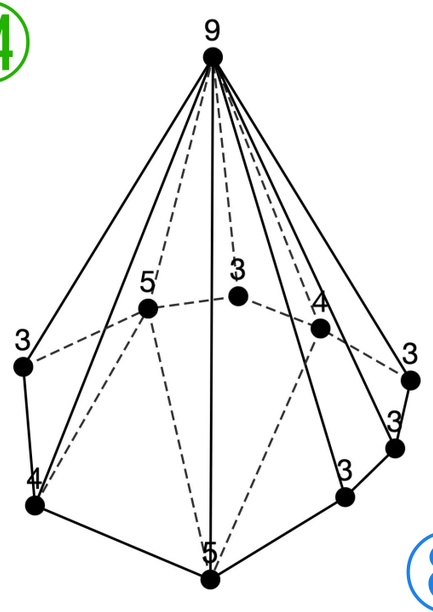
n = 10 (continued)

⑬



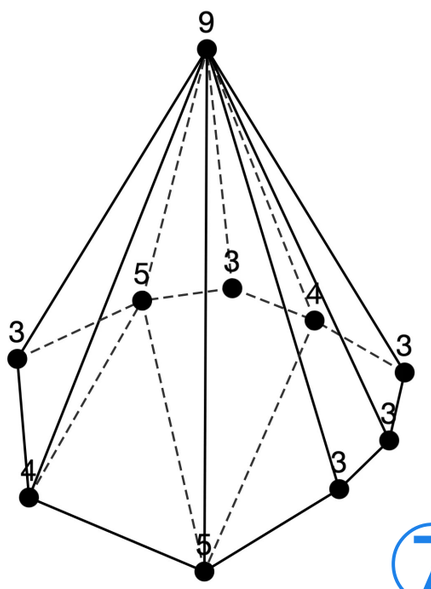
⑧

⑭



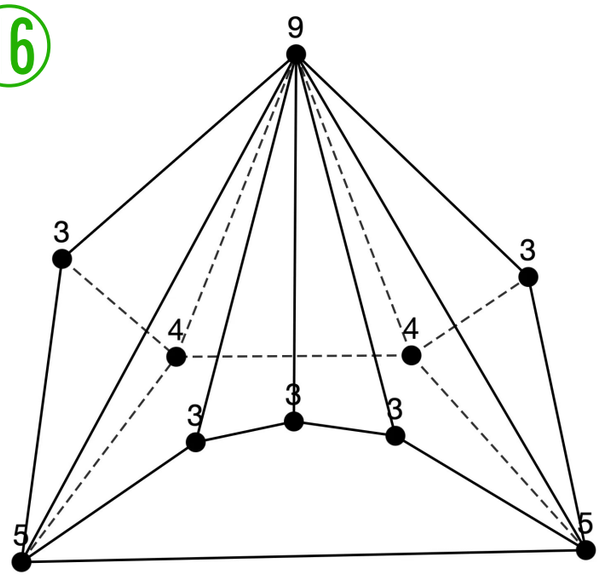
⑧ ⑨

⑮



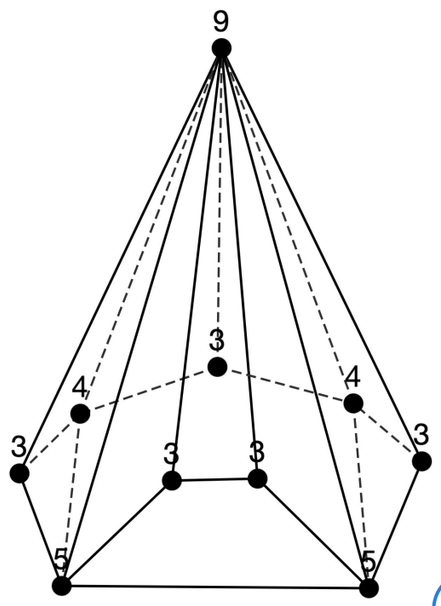
⑦

⑯



⑥

⑰

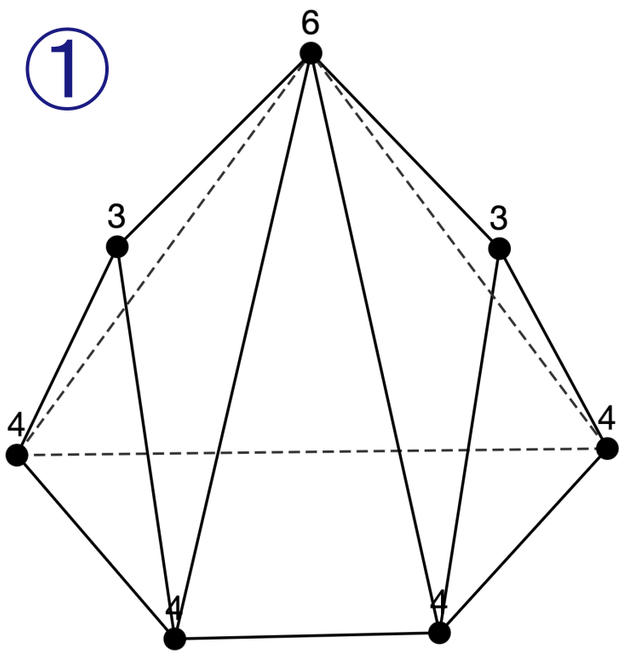


③

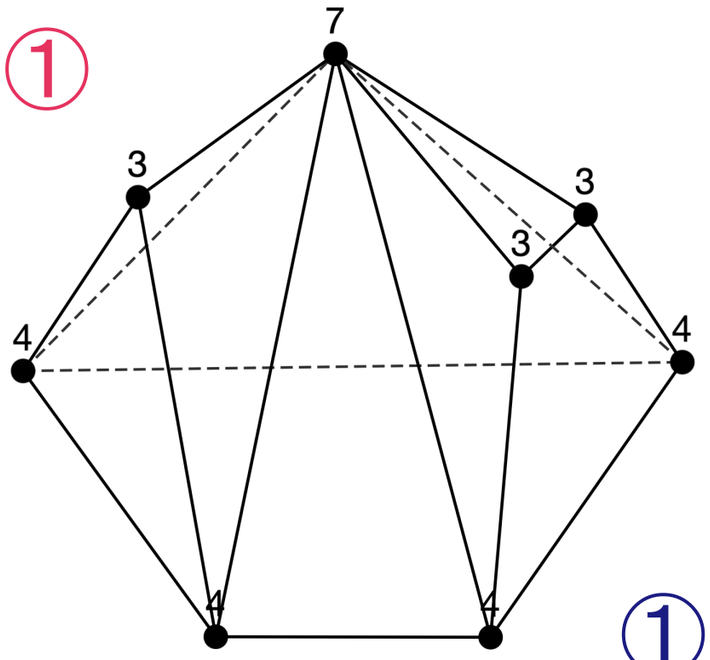
Sequence 6

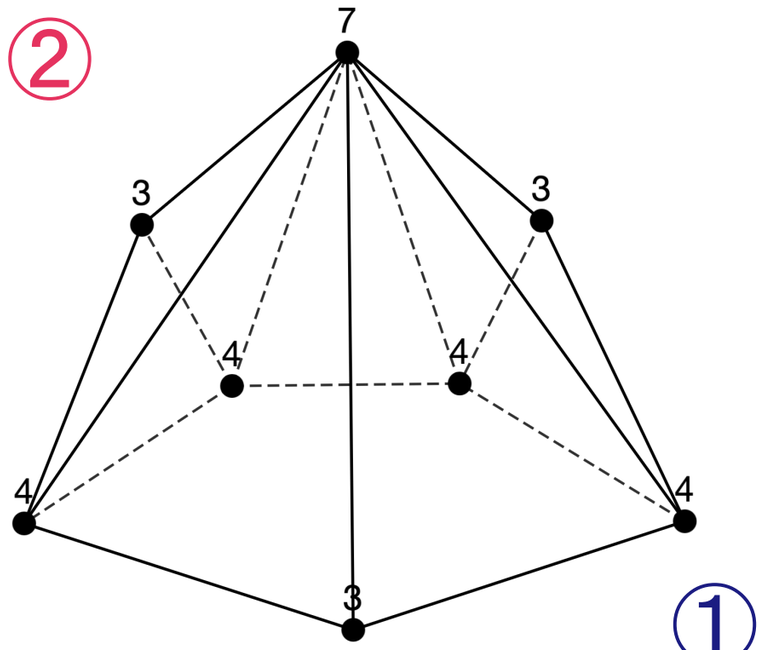
```
n=7:  [3,2][4,4][6,1]
n=8:  [3,3][4,4][7,1]
n=9:  [3,4][4,4][8,1]
n=10:  [3,5][4,4][9,1]
n=11:  [3,6][4,4][10,1]
```

n = 7 (1 type)



n = 8 (2 types)

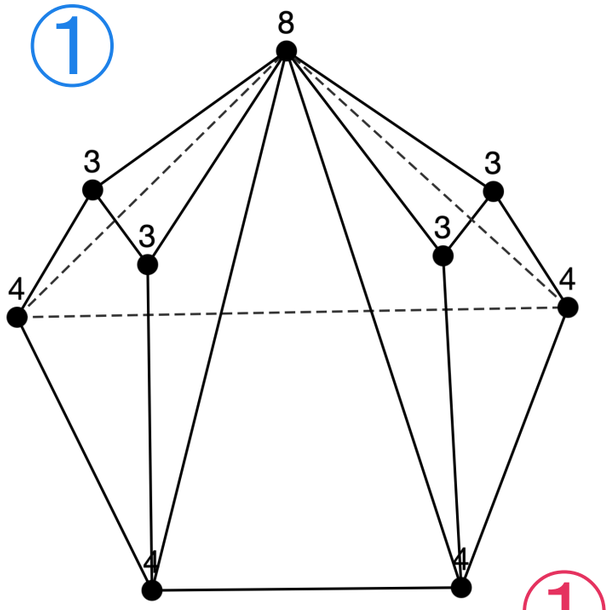




Sequence 6

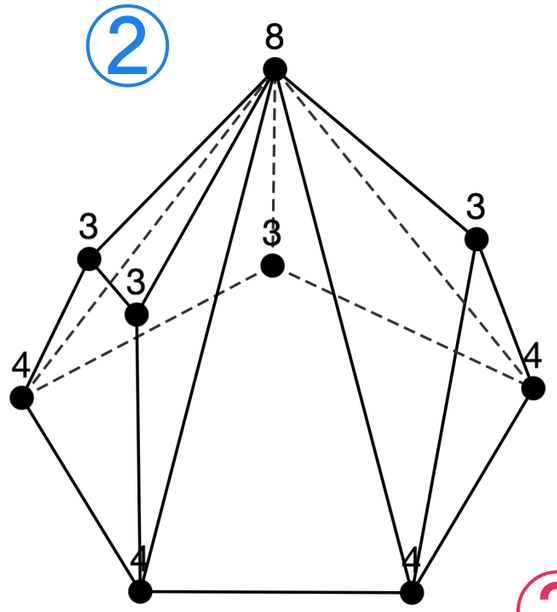
n = 9 (5 types)

①



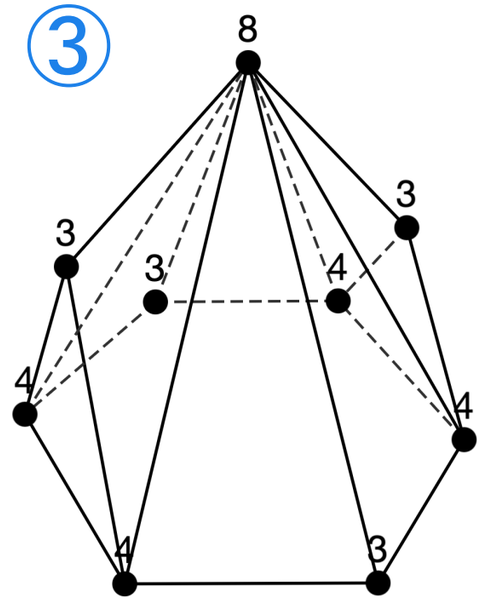
①

②



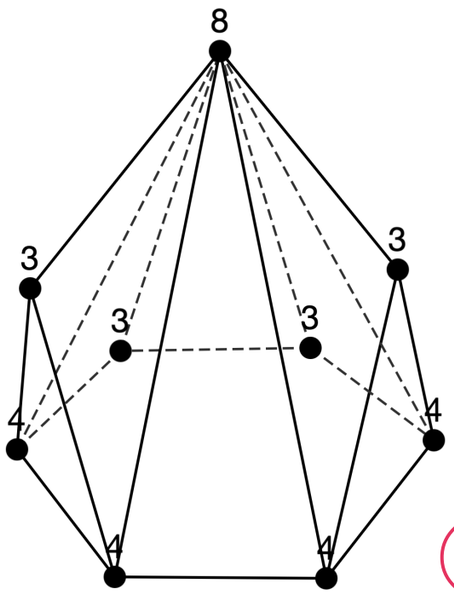
②

③



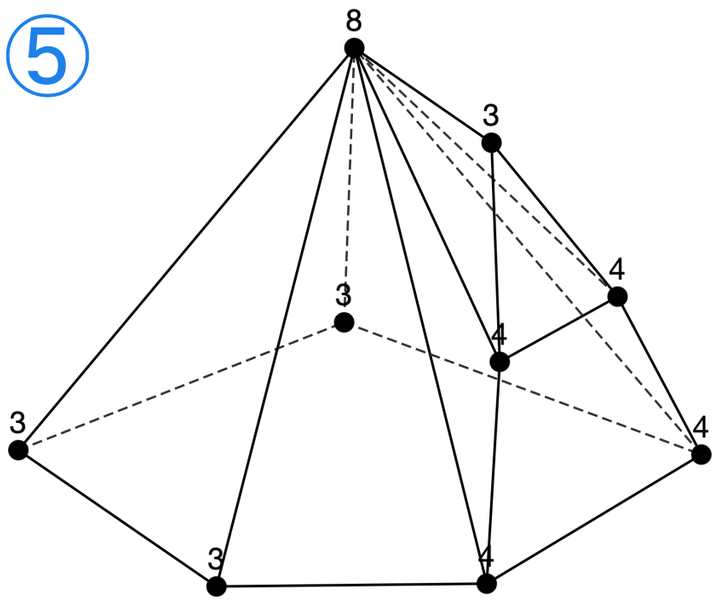
②

④



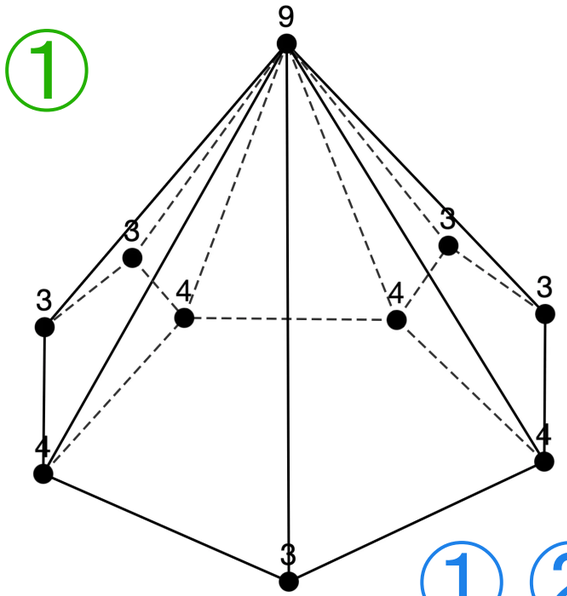
②

⑤

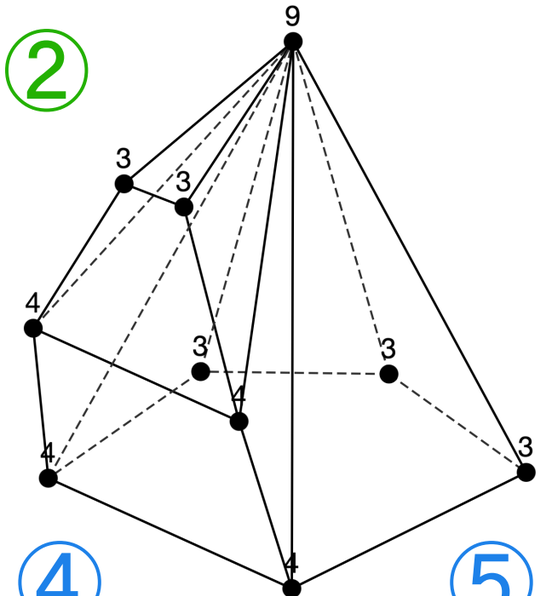


Sequence 6

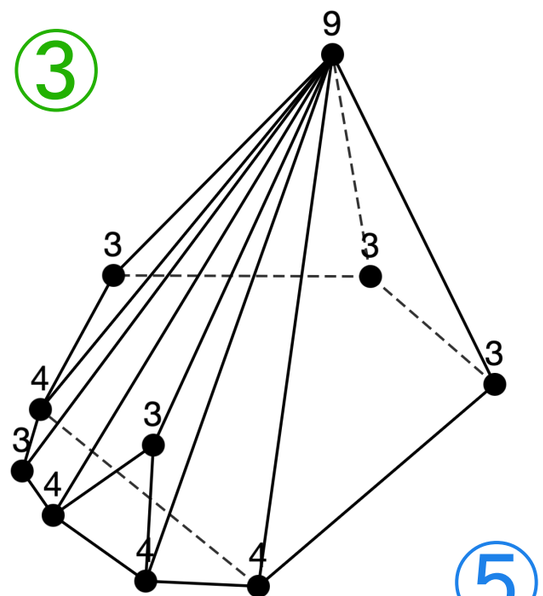
n = 10 (8 types)

① 

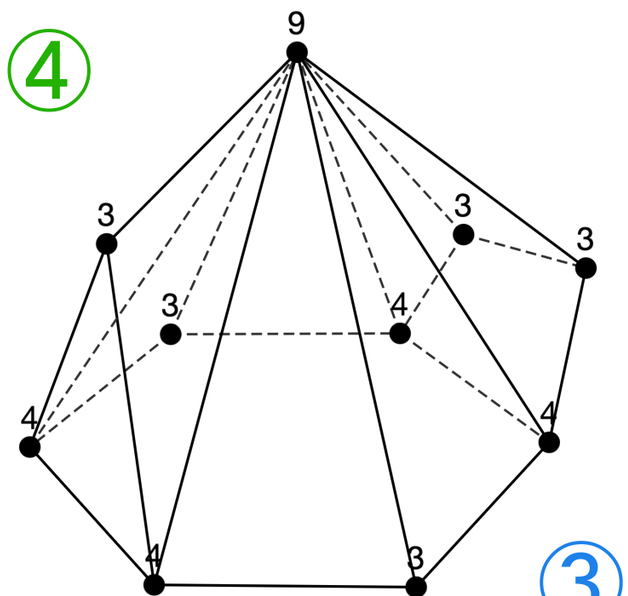
① 2 4

② 

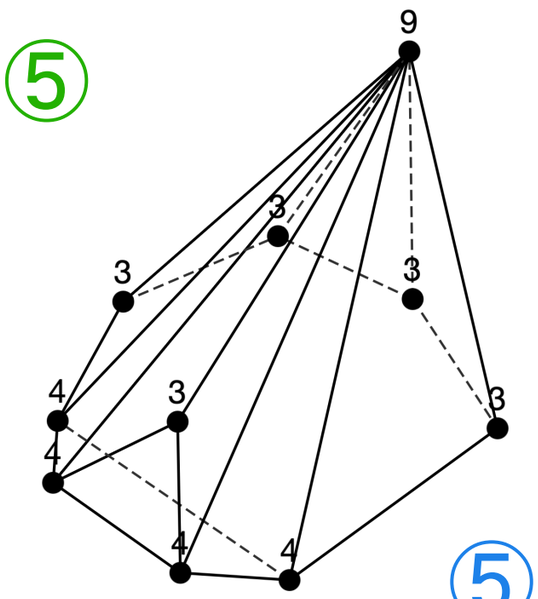
5

③ 

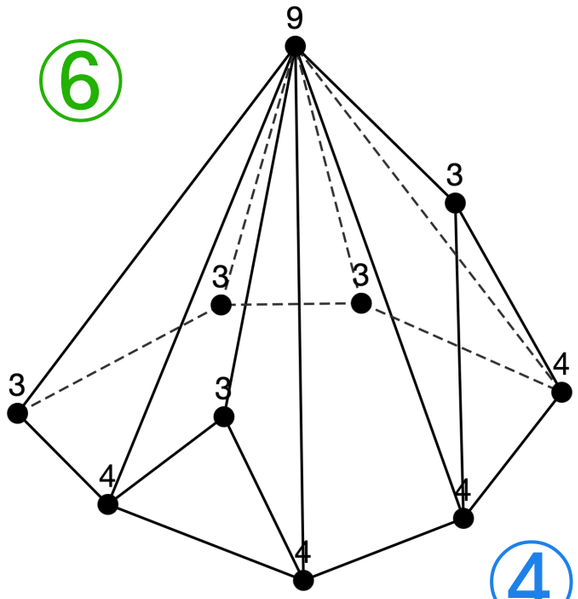
5

④ 

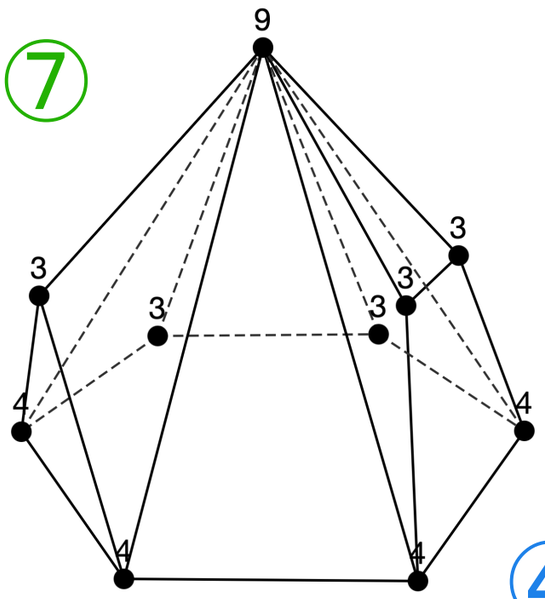
3

⑤ 

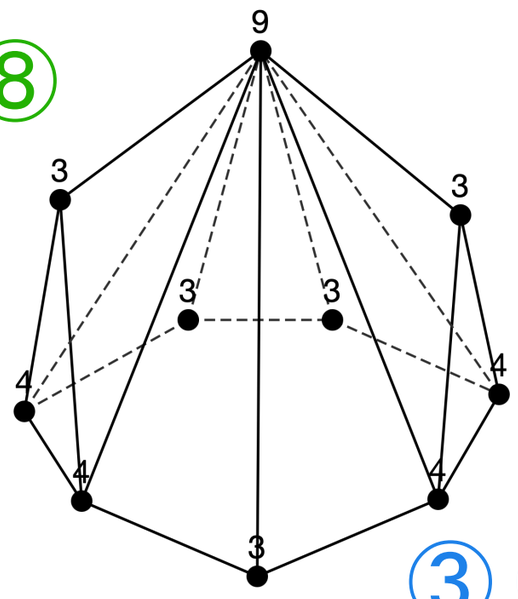
5

⑥ 

4

⑦ 

4

⑧ 

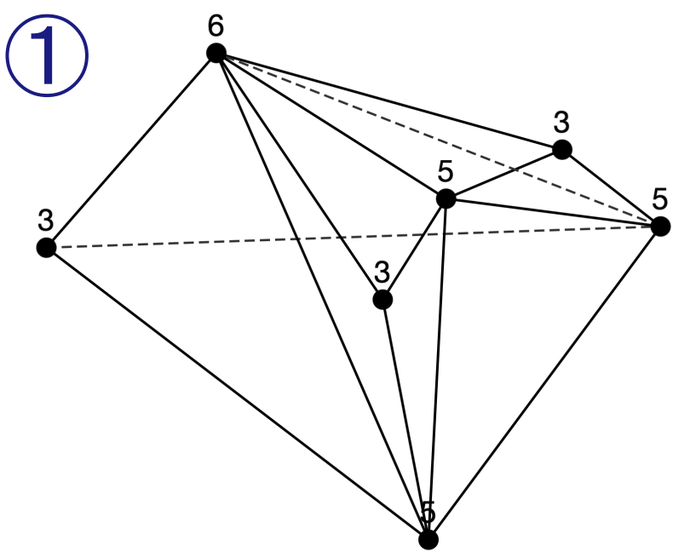
3 4

Sequence 7

```
n=7:  [3,3][5,3][6,1]
n=8:  [3,4][5,3][7,1]
n=9:  [3,5][5,3][8,1]
n=10:  [3,6][5,3][9,1]
n=11:  [3,7][5,3][10,1]
```

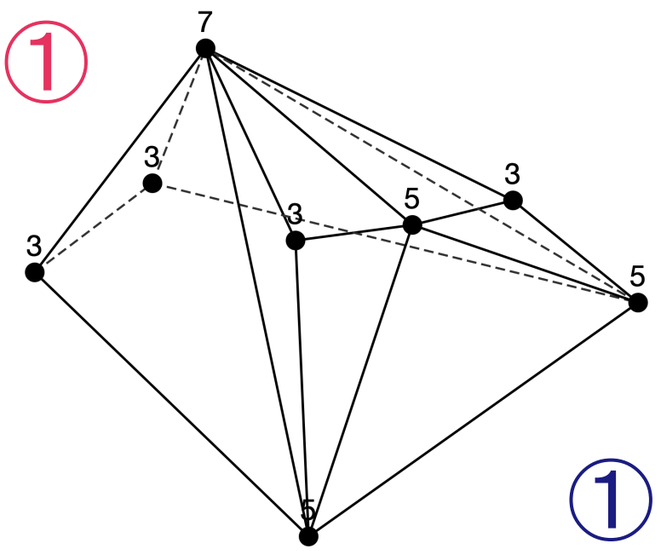
n = 7 (1 type)

①



n = 8 (1 type)

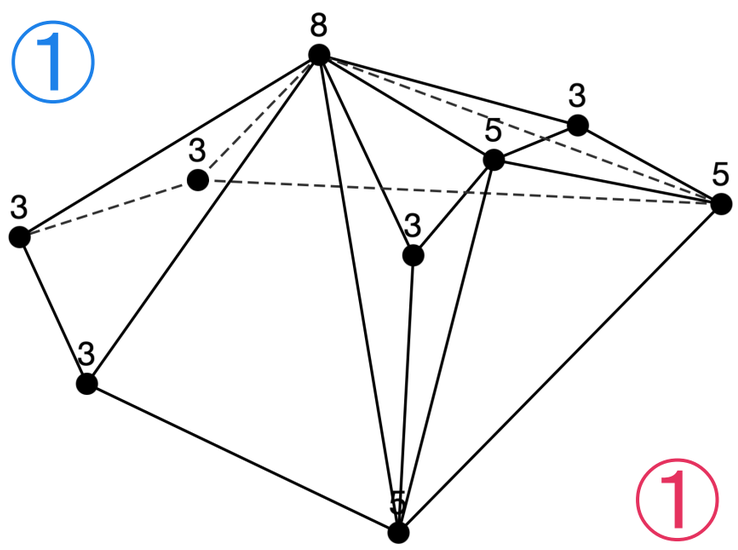
①



①

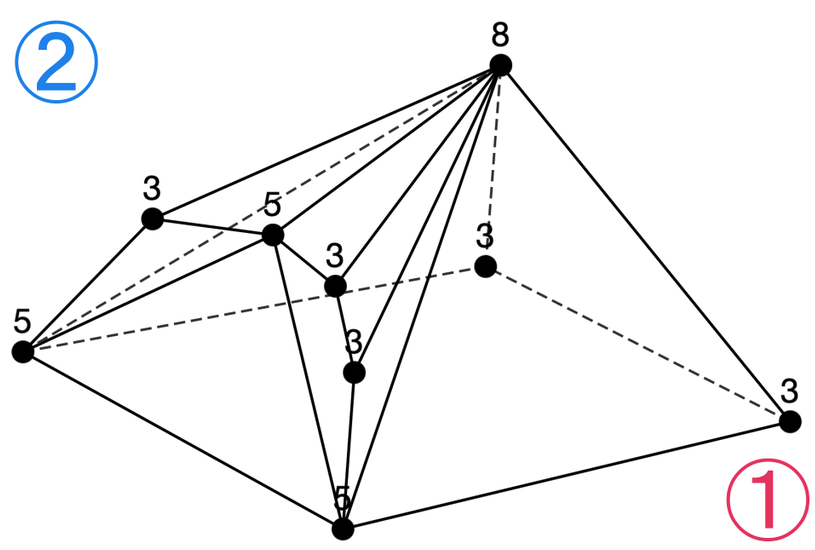
n = 9 (2 types)

①



①

②

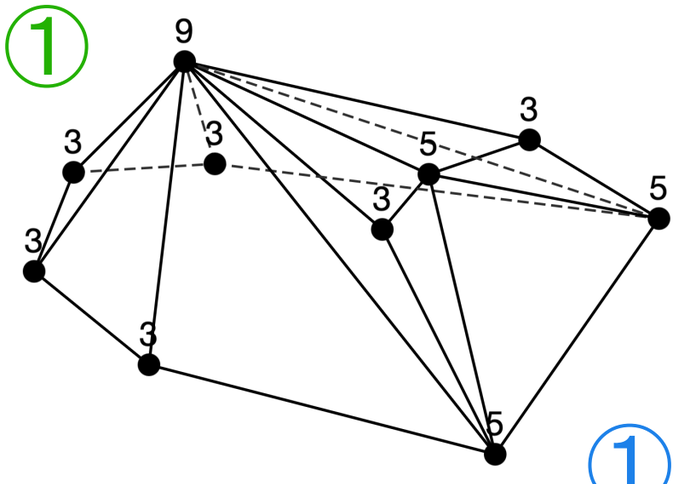


①

Sequence 7

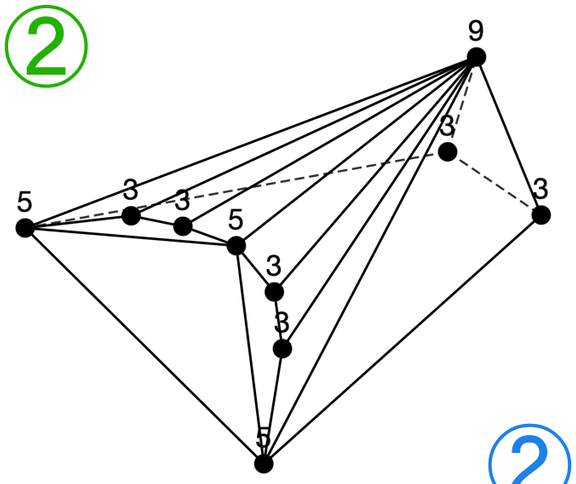
n = 10 (3 types)

①



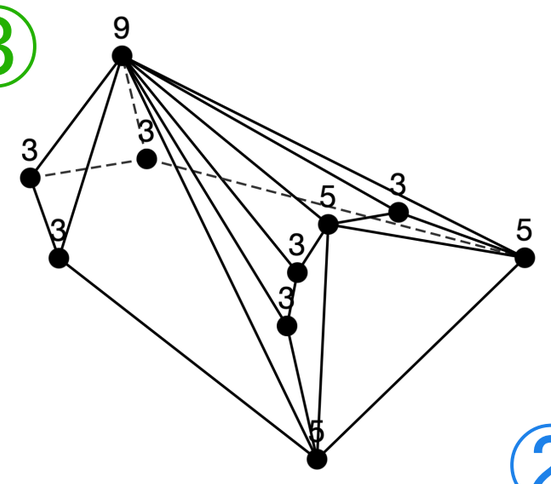
①

②



②

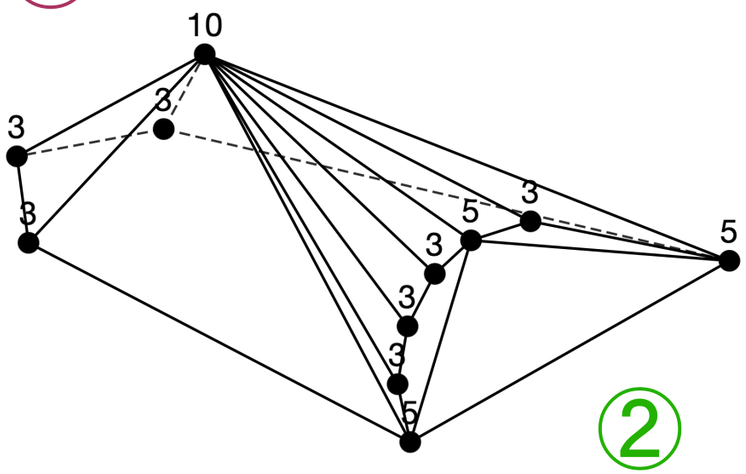
③



②

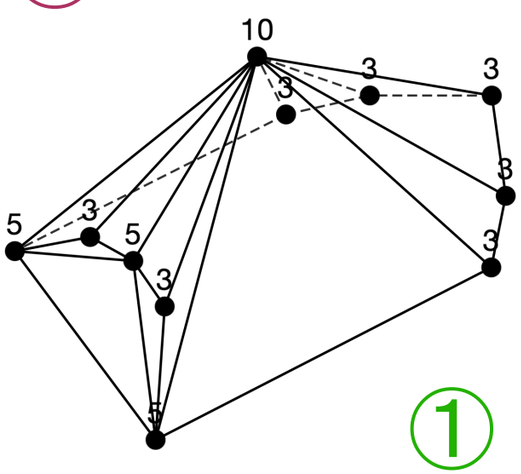
n = 11 (4 types)

①



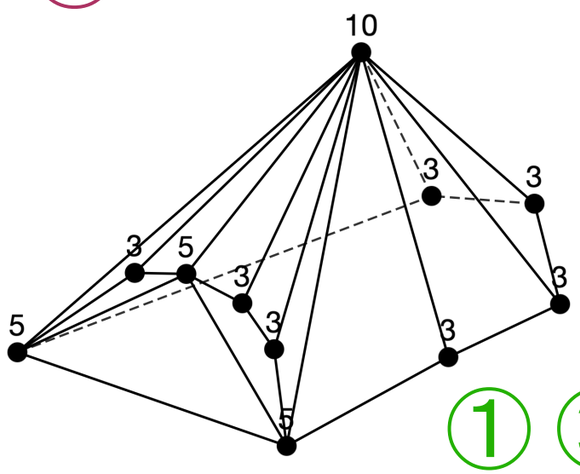
②

②



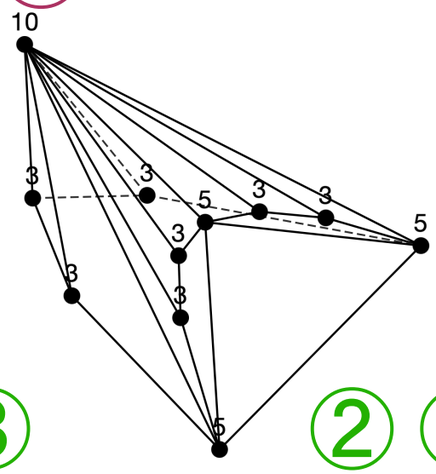
①

③



① ③

④

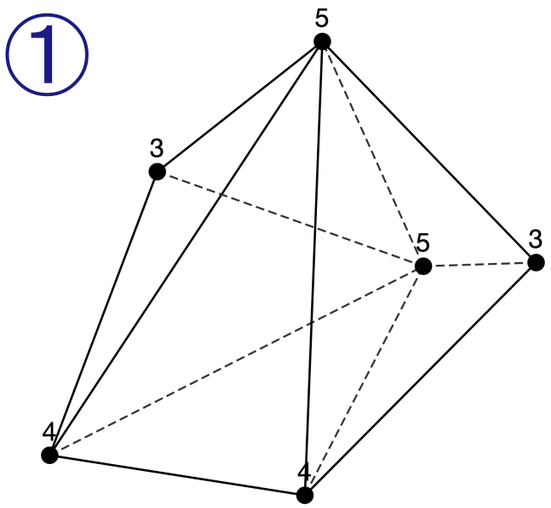


② ③

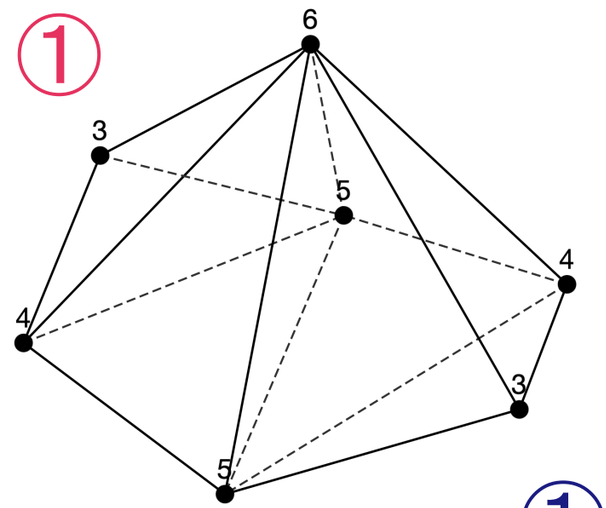
Sequence 8

```
n=6:  [3,2][4,2][5,1][5,1]
n=7:  [3,2][4,2][5,2][6,1]
n=8:  [3,2][4,2][5,3][7,1]
n=9:  [3,2][4,2][5,4][8,1]
n=10:  [3,2][4,2][5,5][9,1]
n=11:  [3,2][4,2][5,6][10,1]
```

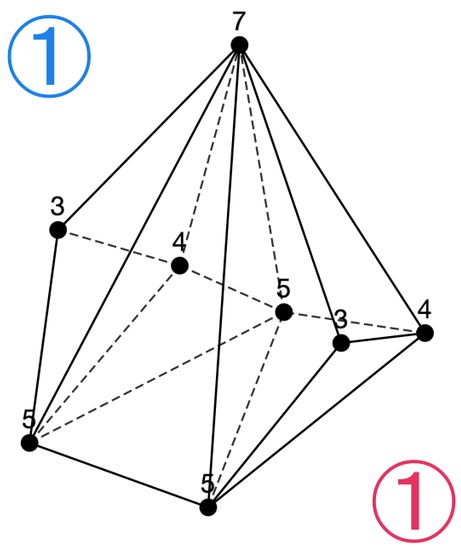
n = 6 (1 type)



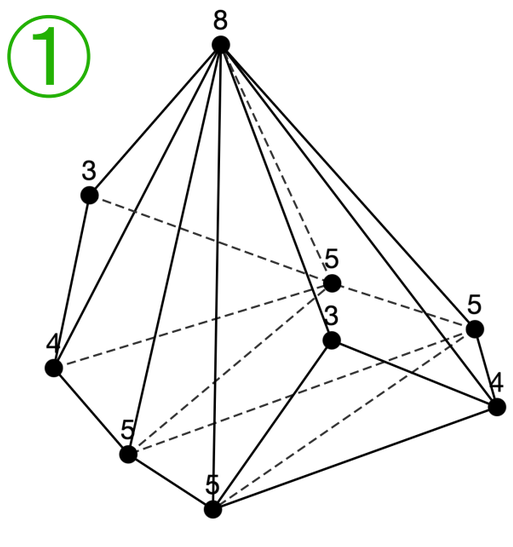
n = 7 (1 type)



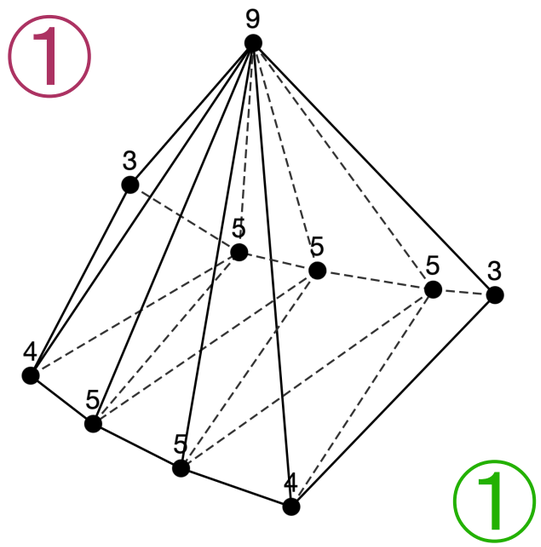
n = 8 (1 type)



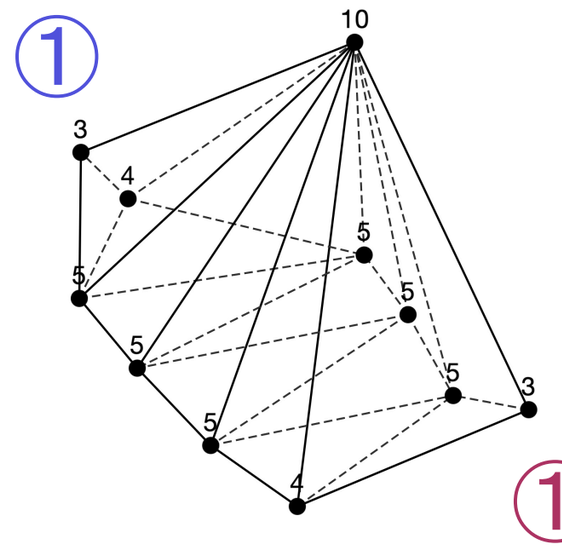
n = 9 (1 type)



n = 10 (1 type)



n = 11 (1 type)



Conclusion

We classified polyhedral graphs based on degree distributions and successfully visualized various series.

Future works

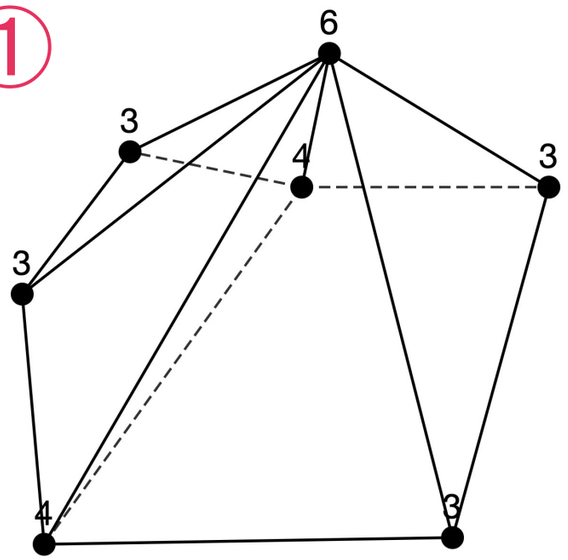
- The visualization of all sequences is not yet complete.
  - ➡ We will continue exploring other interesting sequences.
- Due to hardware limits, exploration is currently up to $n = 11$.
  - ➡ We aim to extend this to $n = 12$ and search for interesting sequences where $n = 8 \rightarrow 10 \rightarrow 12$.
- Sometimes, expected polyhedral graphs do not appear.
  - ➡ We aim to provide explanations for why such polyhedral graphs do not exist.

Missing Expected Polyhedral graphs

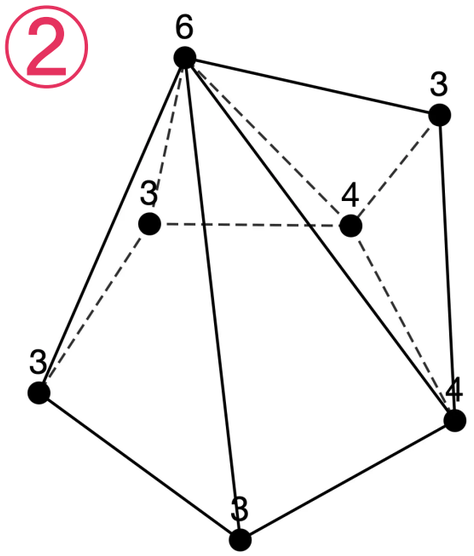
Sequence 1

n = 7 (2 types)

①

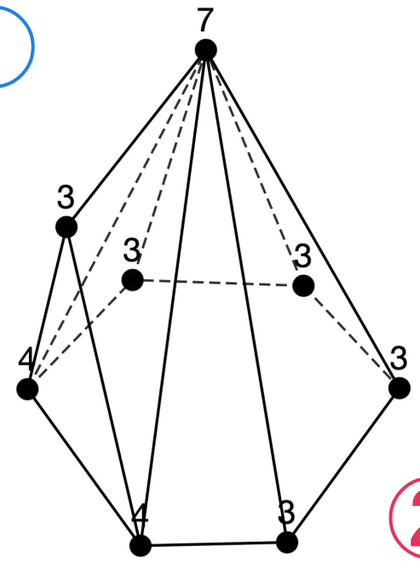


②

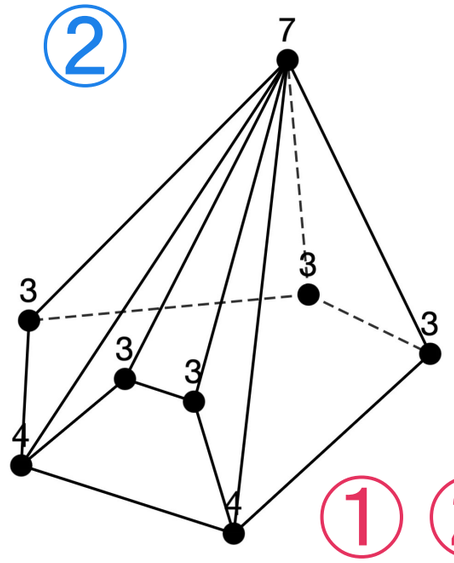


n = 8 (2 types)

①



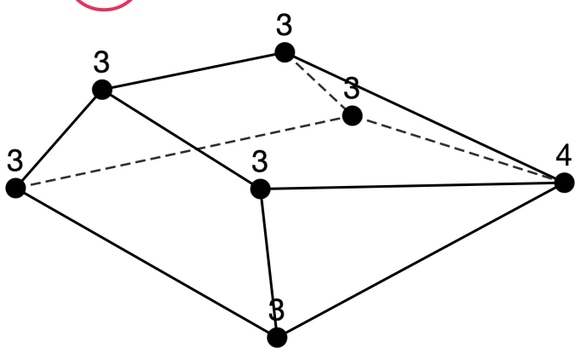
②



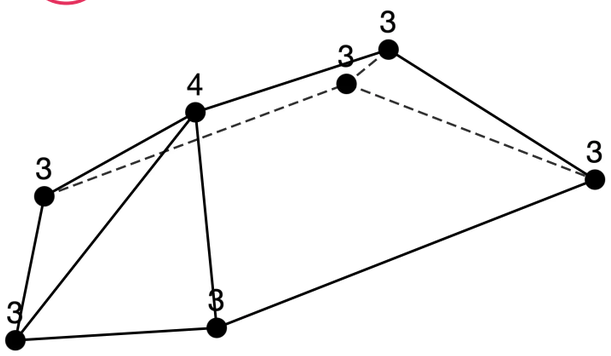
Sequence 2

n = 7 (2 types)

①

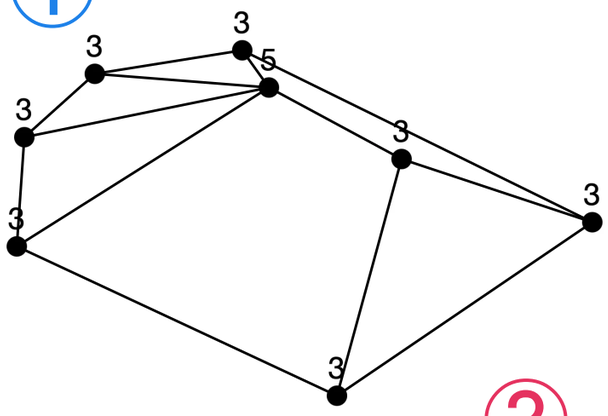


②



n = 8 (2 types)

①



②