Discrete Mathematics #Final

2021/12/13

Final Project - Maximum k-core

- A k-core of a graph G is a maximal subgraph of G in which all vertices have degree at least k.

- Find the maximum k-core in the simple graph G.

- Existing source codes are forbidden.
  - Packages for graph or network are also forbidden (Ex. NetworkX)

- No plagiarism.

Format

- Input (Graph):

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

- Output (Maximum k-core):

```
3-core
1 2
1 4
1 5
2 4
2 5
4 5
```

Example-1

- Input (Graph):

0 1

0 2

1 2

1 4

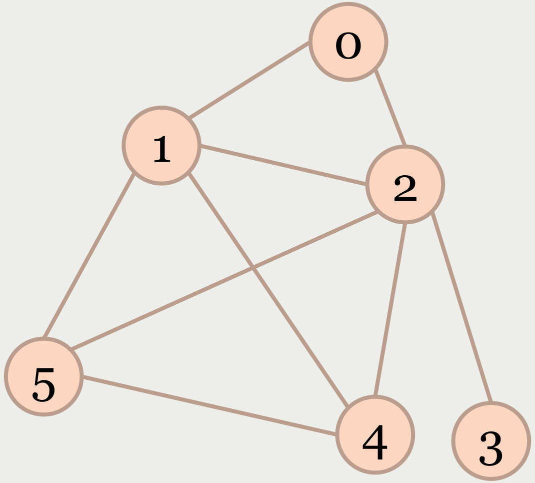
1 5

2 3

2 4

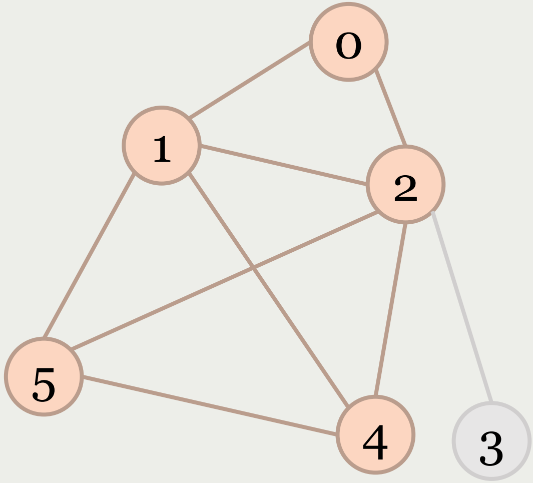
2 5

4 5



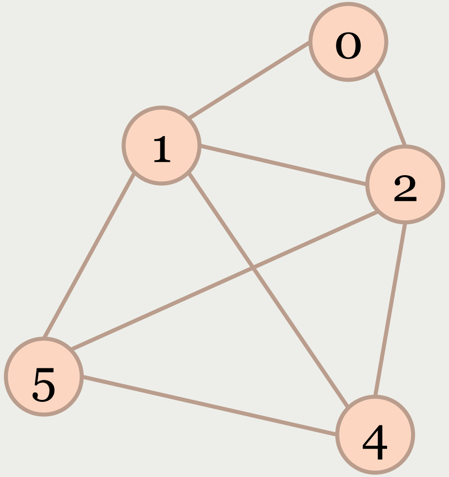
Example-1

- **Create 2-core graph**: Remove the vertex with degree 1.



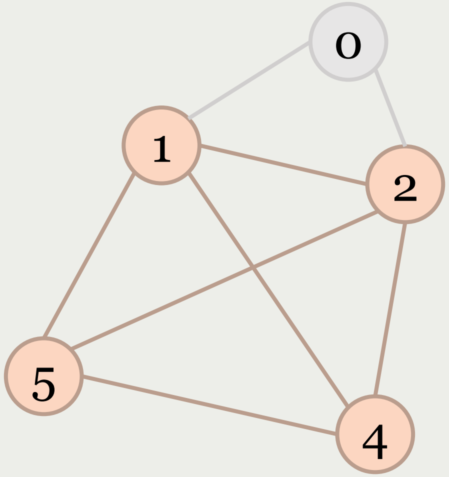
Example-1

- All vertices in 2-core graph have degree at least 2.



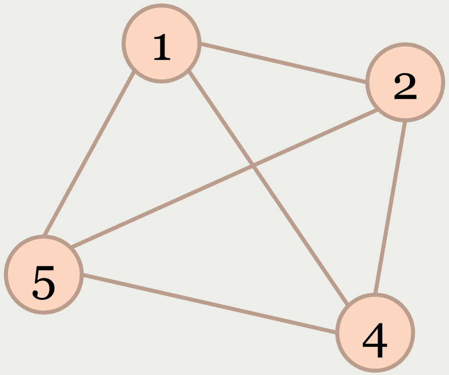
Example-1

- <u>**Create 3-core graph**</u>: Remove the vertex with degree 2.



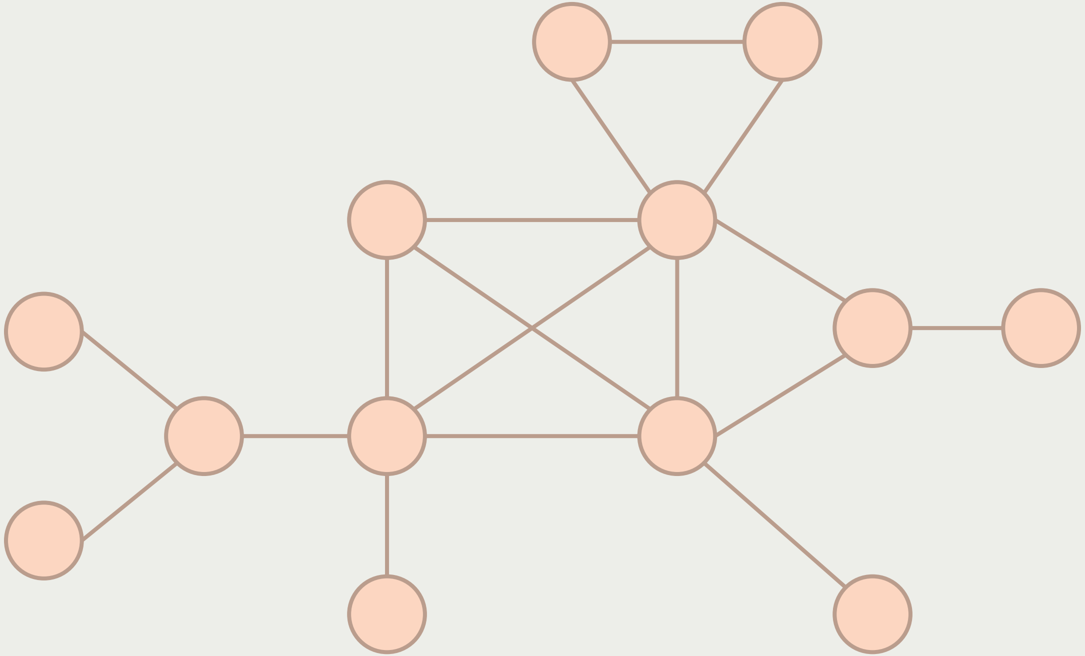
Example-1

- All vertices in 3-core graph have degree at least 3.
- The maximum k-core is 3-core.



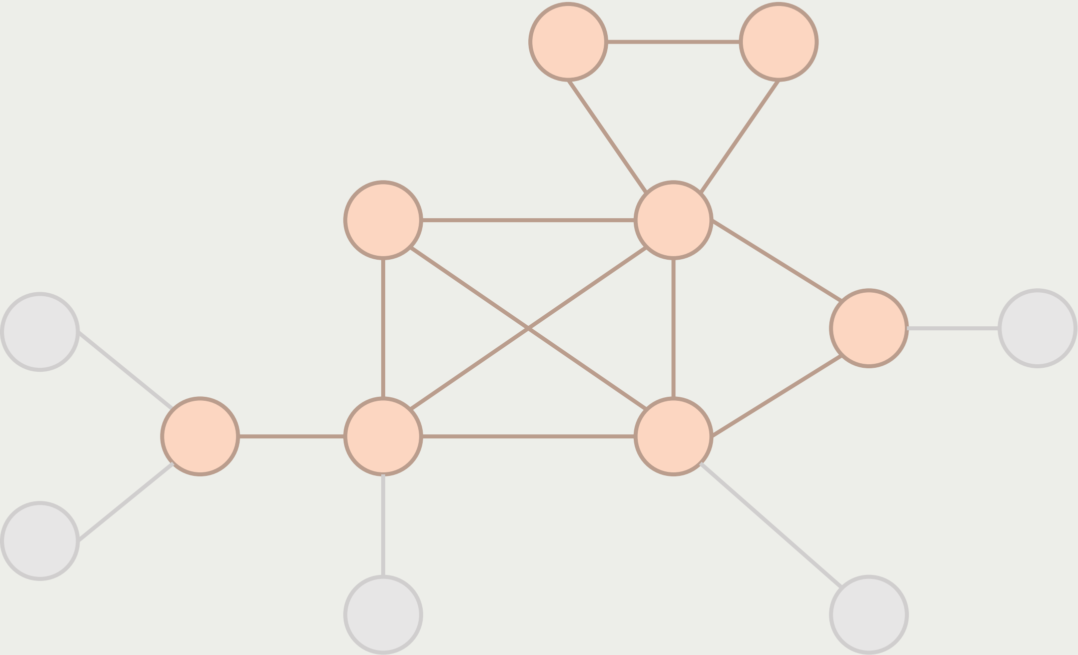
Example-2

- Find the maximum k-core for the following graph.



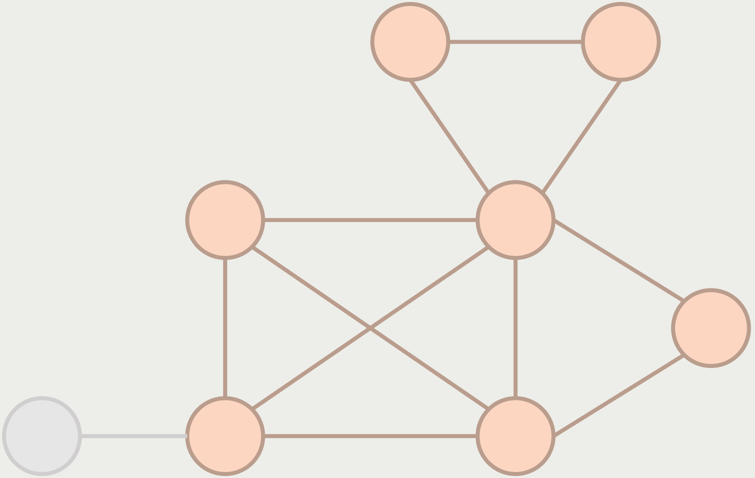
Example-2

- **Create 2-core graph**: Remove the vertex with degree 1.



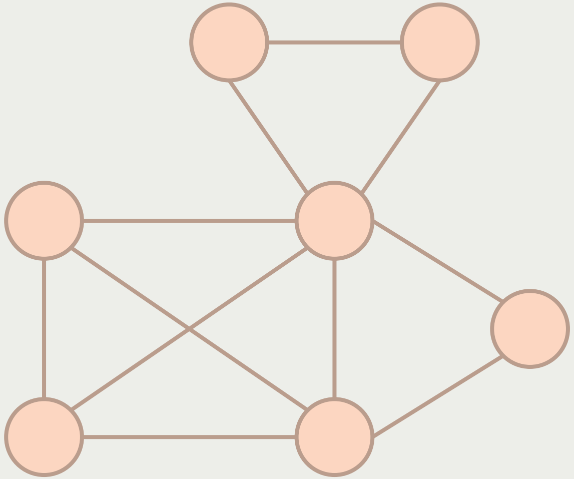
Example-2

- **Create 2-core graph**: Remove the vertex with degree 1.



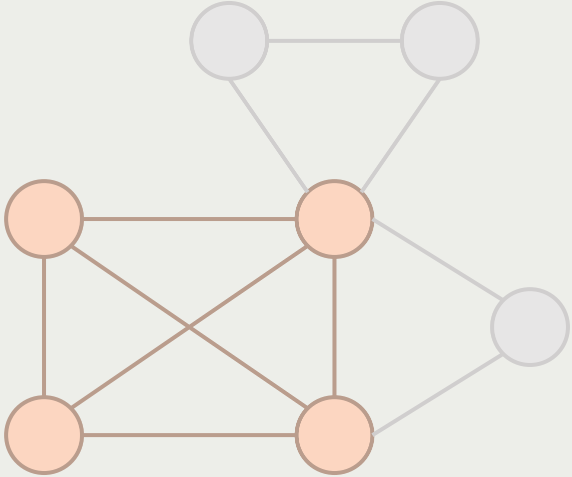
Example-2

- All vertices in 2-core graph have degree at least 2.



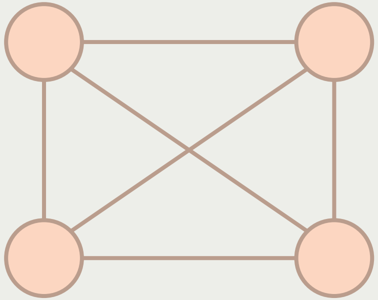
Example-2

- **Create 3-core graph**: Remove the vertex with degree 2.



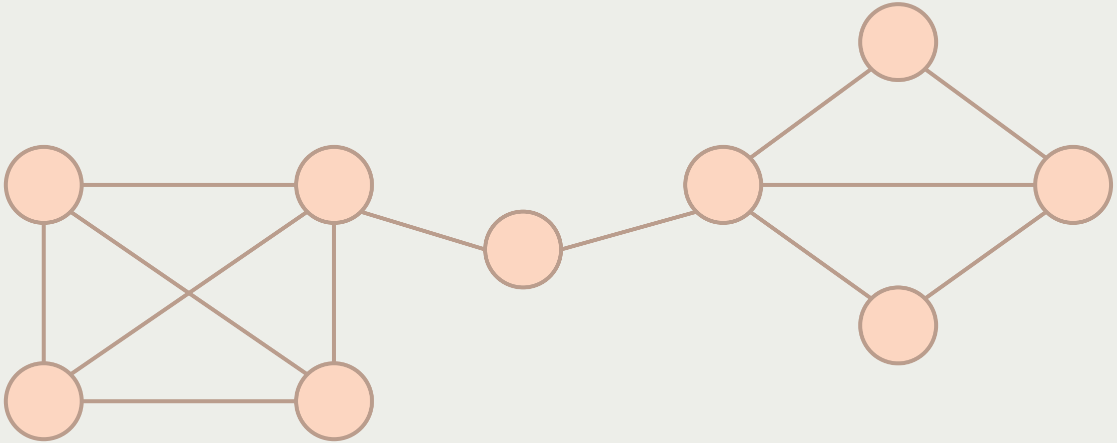
Example-2

- All vertices in 3-core graph have degree at least 3.
- The maximum k-core is 3-core



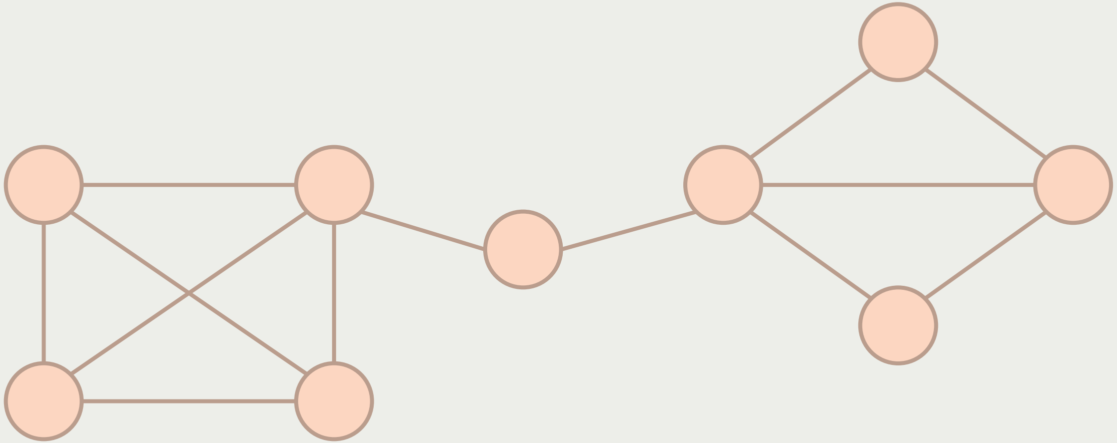
Example-3

- Find the maximum k-core for the following graph.



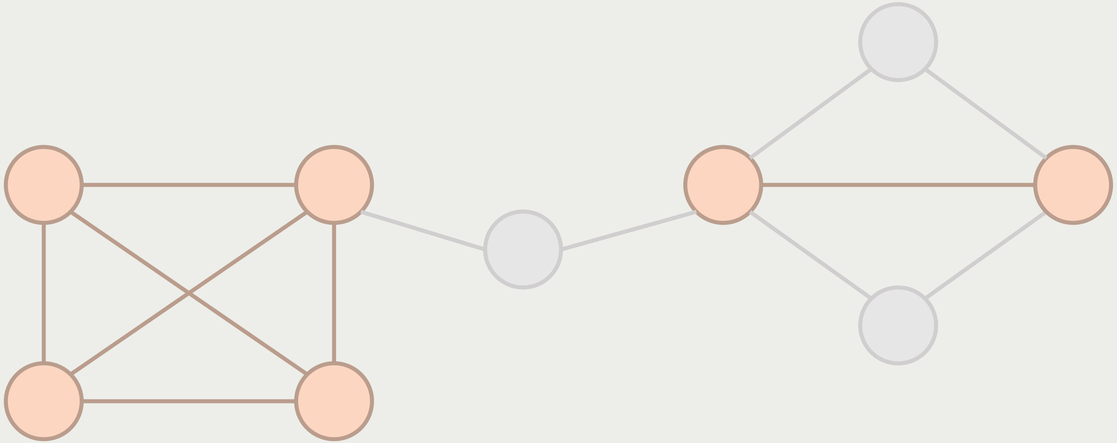
Example-3

- **Create 2-core graph**: Remove the vertex with degree 1.



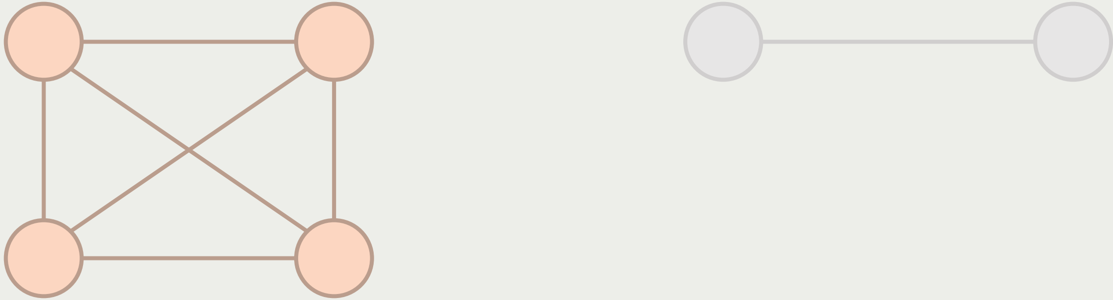
Example-3

- **Create 3-core graph**: Remove the vertex with degree 2.



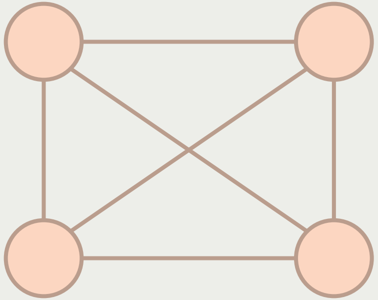
Example-3

- **<u>Create 3-core graph</u>**: Remove the vertex with degree 2.

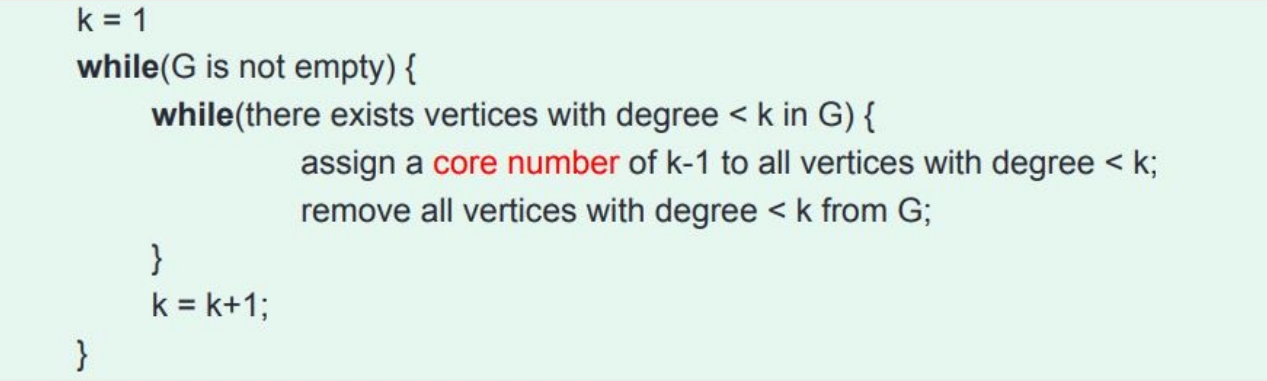


Example-3

- All vertices in 3-core graph have degree at least 3.
- The maximum k-core is 3-core



Pseudo Code



```
k = 1
while(G is not empty) {
    while(there exists vertices with degree < k in G) {
            assign a core number of k-1 to all vertices with degree < k;
            remove all vertices with degree < k from G;
    }
    k = k+1;
}
```

- How to optimize the search of existing vertices?
    - DFS/BFS from the lowest node degree?
    - Any other method?

Test Data

- Number of test cases: 10
- Time limit: 4000ms
- Memory limit: 1000000KiB
- 4% for each test case, all 10 test cases = 40%

Grading Policy

1. Correctness (40%)
2. Speed (20%)
   - Top 25%: 20%
   - Top 50%: 15%
   - Top 75%: 10%
   - The rest: 5%
3. Report (40%)
   - English / Chinese
   - Novelty – Using what kind of method to save more time?
   - Comprehensiveness of experiments – Any comparisons with different searching methods?
   - Theoretical results – Is there any way to describe or prove the complexity of your algorithm?

Important Dates

- 1/13 (Thu) 23:59 - Formosa OJ closed
- 1/16 (Sun) 23:59 - Report & Code Submission deadline

If you have any question…

- We encourage everyone to ask any question in TA hours.
    - 10:00-12:00 every Friday online
    - https://meet.google.com/yuf-bghs-vqk

- If the question is personal or the time slot is not available to you, please send an email to TAs.
    - Ex: TAs miss to approve your Formosa OJ request.

Q & A

Thank you