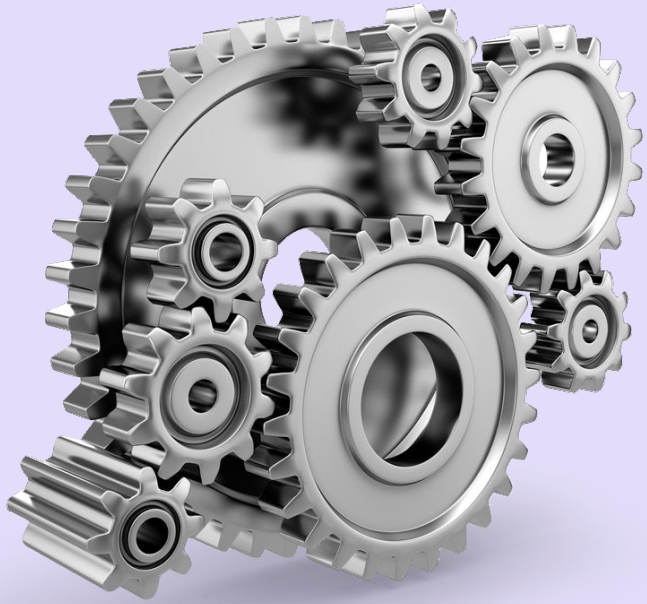
Lecture 31

Software Engineering III

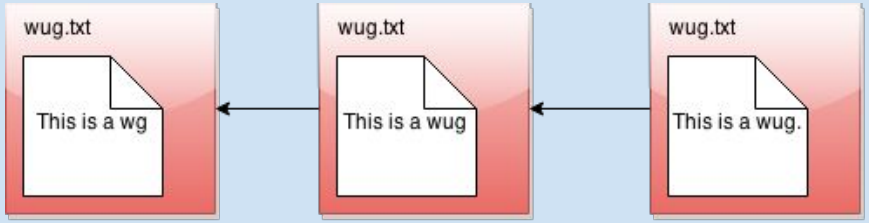


CS61B, Spring 2024 @ UC Berkeley

Background

Ergün Açıköz (he/him)

- 4th year - CS
- 5th Semester on staff
- Istanbul, Turkey
- Cats, geography, travelling



Gitlet

Lecture 34, CS61B, Spring 2024

Gitlet

Whiteboarding

SWE Fundamentals

Leetcode

Q & A

Project 2: Gitlet

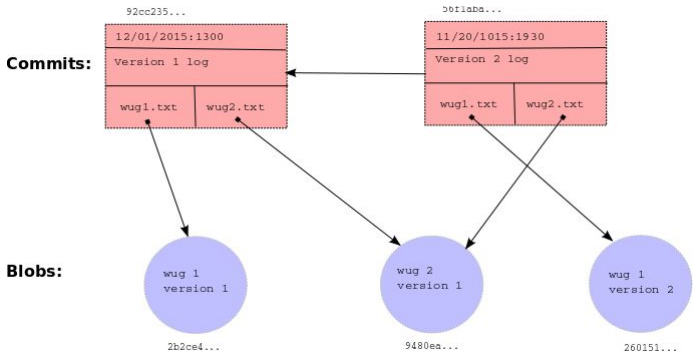
What is Gitlet

[DISCLAIMER]: This is my own personal experience with gitlet and everything I say from here until end of gitlet part should be taken as from my perspective.

TLDR: Implementing your own version control system - like git in real life! As a freshman I was introduced to:

- My first big scale project - none to little skeleton code.
- Creating a design document.
- Comprehensive multiple tests. Randomized tests to prevent hard coding and brute forcing the grader (well kind of…)

… and of course lots of thinking and working a lot 🤠



My Experience

My (bad) plan for Gitlet:

- I was creating version control in my system.
  - I did not pay attention to the subparts of the project, such as merge, add, commit, and took the project as a whole and had no easy way to start designing
- I wrote a couple of things to the design document. I tried to come up with a design that would work for everything way in the beginning. I thought:
  - “My implementation is going to change after all, so I don't need a really detailed design doc.”
  - “If I get stuck, I can just wait for people to ask questions on Ed!”
  - “It is not that important to have a solid foundation now; I’ll get it later.”
- Consequently, I started coding immediately
  - I did not write any tests or think too hard about edge cases.

My Experience

Commits on Apr 3, 2021

did some work on project 2
ergunackz committed 3 years ago

projec2
ergunackz committed 3 years ago

projec2
ergunackz committed 3 years ago

projec2
ergunackz committed 3 years ago

projec2
ergunackz committed 3 years ago

projec2
ergunackz committed 3 years ago

projec2
ergunackz committed 3 years ago

projec2
ergunackz committed 3 years ago

projec2
ergunackz committed 3 years ago

projec2
ergunackz committed 3 years ago

projec2
ergunackz committed 3 years ago

Commits on Apr 2, 2021

projec2
ergunackz committed 3 years ago

Commits on Apr 5, 2021

did some work on project 2
ergunackz committed 3 years ago

did some work on project 2
ergunackz committed 3 years ago

did some work on project 2
ergunackz committed 3 years ago

did some work on project 2
ergunackz committed 3 years ago

did some work on project 2
ergunackz committed 3 years ago

did some work on project 2
ergunackz committed 3 years ago

did some work on project 2
ergunackz committed 3 years ago

did some work on project 2
ergunackz committed 3 years ago

did some work on project 2
ergunackz committed 3 years ago

did some work on project 2
ergunackz committed 3 years ago

did some work on project 2
ergunackz committed 3 years ago

[Project 2 due 04/02]

Showing **1 changed file** with **1 addition** and **1 deletion**.

2 proj2/gitlet/Repository.java

```
@@ -289,7 +289,7 @@ public static void status() {
289 289
290 290
291 291
292     -       System.out.println("=== Untracted Files ===");
    292 +       System.out.println("=== Untracked Files ===");
293 293
294 294       System.out.println("");
295 295
```

My Experience

My gitlet file was removed #3817

Anonymous
3 years ago in **Projects – Project 2: Gitlet**

205
VIEWS

Hi,

I just committed and pushed my project but my whole project has been deleted. How can I recover my gitlet project file? Is it possible to do that?

...

1 Answer

Alex Schedel **INSTRUCTOR**
3 years ago

Use git checkout to get the old version back

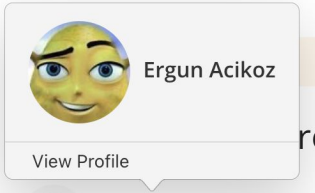
...

Anonymous 3y
the last checkout I have is from lab 8 :)

Samuel Berkun 3y
Push to your snaps repo; on github you can go back through the snaps commits until you find one that has the work you lost. Hopefully you only lose a few minutes of work.

3 ...

My Experience



You are viewing this thread in readonly mode.

removed #3817

Anonymous
3 years ago in Projects – Project 2: Gitlet

205
VIEWS

Hi,

I just committed and pushed my project but my whole project has been deleted. How can I recover my gitlet project file? Is it possible to do that?

...

1 Answer

Alex Schedel INSTRUCTOR
3 years ago

Use git checkout to get the old version back

...

Anonymous 3y
the last checkout I have is from lab 8 :)

Samuel Berkun 3y
Push to your snaps repo; on github you can go back through the snaps commits until you find one that has the work you lost. Hopefully you only lose a few minutes of work.

3 ...

My Experience

#	Submitted On (PDT)	Submitters	Score	Active
62	Apr 5 at 7:12 PM	EA	1600.0	Activate
61	Apr 5 at 6:42 PM	EA	1440.0	Activate
60	Apr 5 at 6:12 PM	EA	1440.0	Activate
59	Apr 5 at 6:08 PM	EA	1400.0	Activate
58	Apr 5 at 5:32 PM	EA	1400.0	Activate

Outcomes

Some of the things were not effective but despite having all these struggles and everything, I still learned a lot! It was a big jump from 61A to 61B.

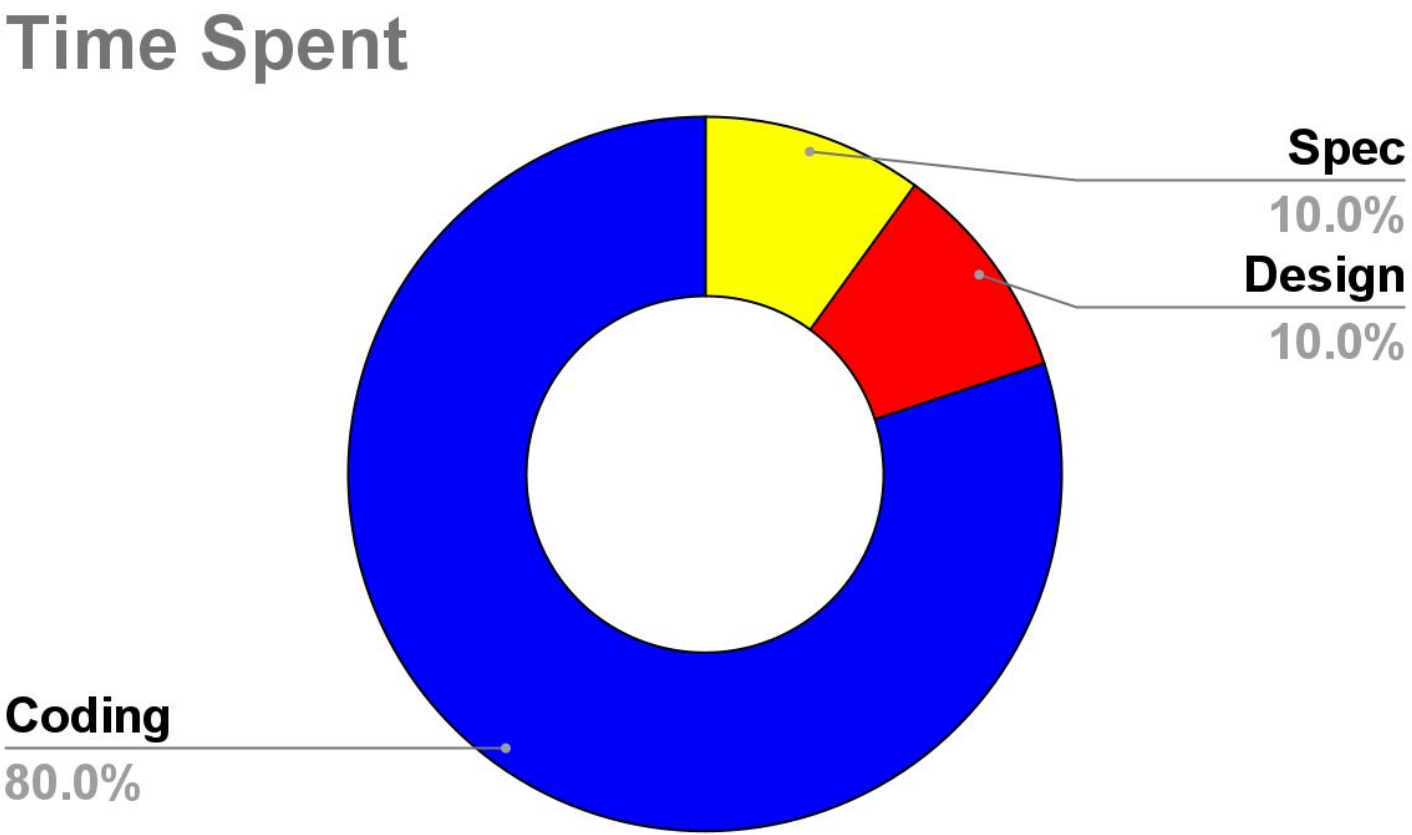
- From skeleton code to nothing at all - I accomplished my first big project!
  - It became a core memory of mine, and is still one of the projects I use for interviews and job applications.
- Learned a lot about the data structures and their implementations!
  - Graph traversals, persistence, data abstraction
- Gained a lot of experience and different perspectives on how to approach problems

Reflection

What went wrong and what could have been done to improve it?

- Mostly ignored the principles from the Software Engineering II Lecture:
  - Time Management: Ended running far behind
  - Design: Didn't design much at the start
  - Testing: No tests.
- Failed many times, and had to start over from scratch repeatedly:
  - Trial → test/error → improvement → trial → dead end → new design
  - Coded a lot
  - While coding, tried to come up with different design
  - Wasted a lot of time

My Experience



Whiteboarding

Lecture 34, CS61B, Spring 2024

Gitlet

Whiteboarding

SWE Fundamentals

How to Practice

Q & A

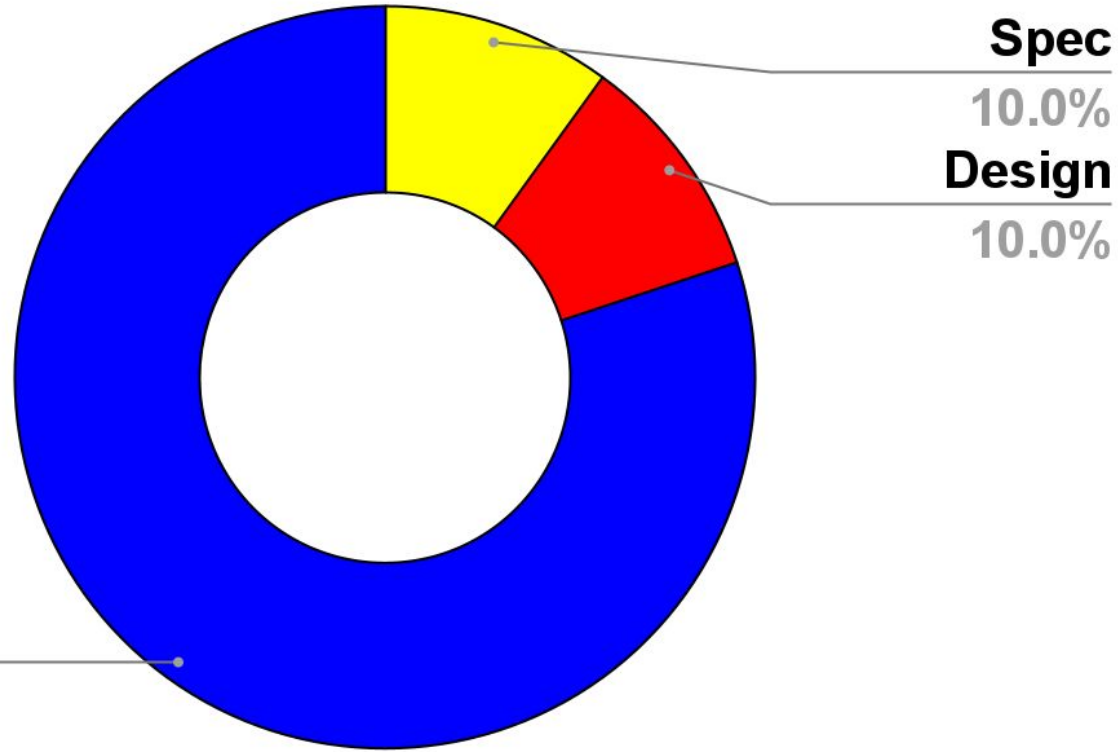
Reflection continued

What went wrong and what could have been done to improve it?

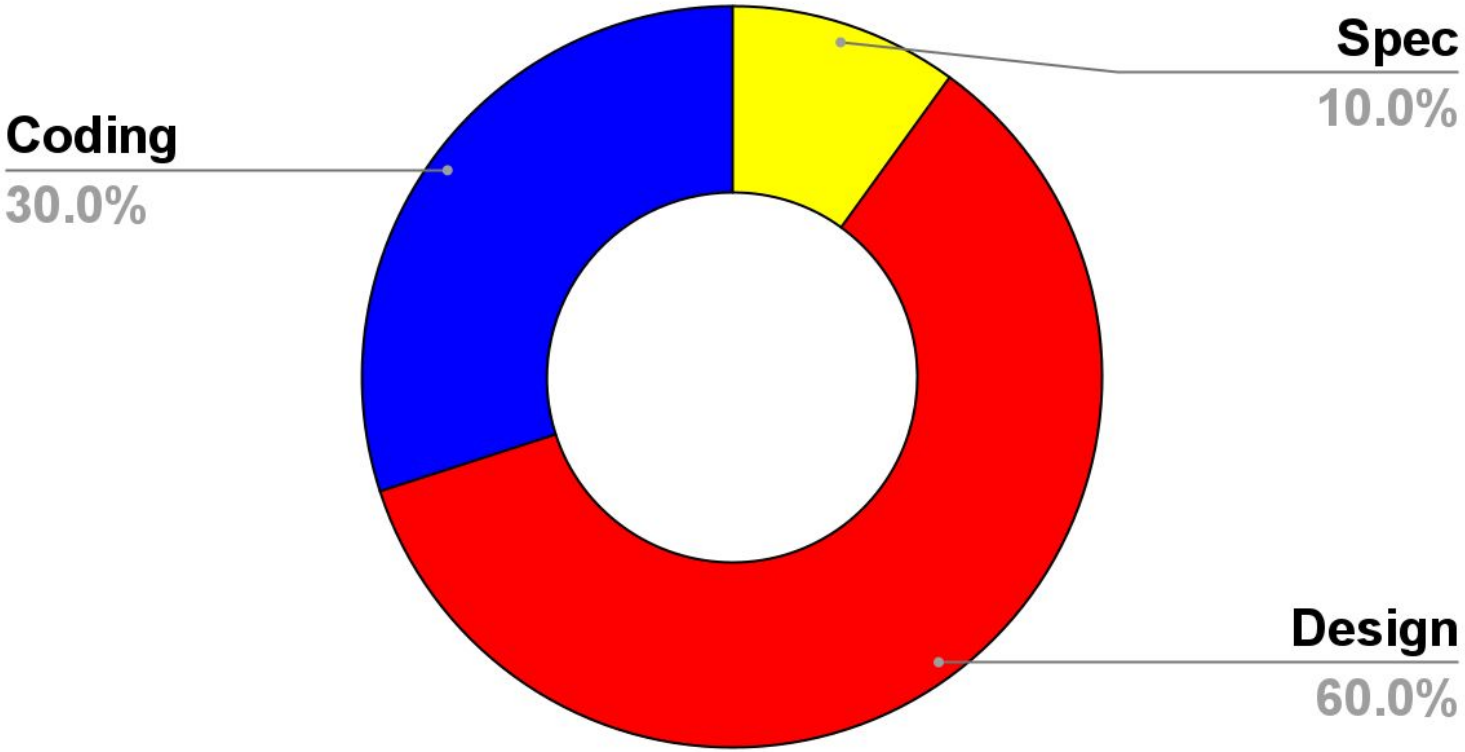
- Mostly ignored the principles from the Software Engineering II Lecture:
  - Time Management: Ended running far behind
  - Design: Didn't design much at the start
  - Testing: No tests.
- Failed many times, and had to start over from scratch repeatedly:
  - Trial → test/error → improvement → trial → dead end → new design
  - Coded a lot
  - While coding, tried to come up with different design
  - Wasted a lot of time

There are couple of things I would do differently. **Common issue: coding a lot before designing or thinking wastes time on bad implementations.**

Time Spent



Time Spent

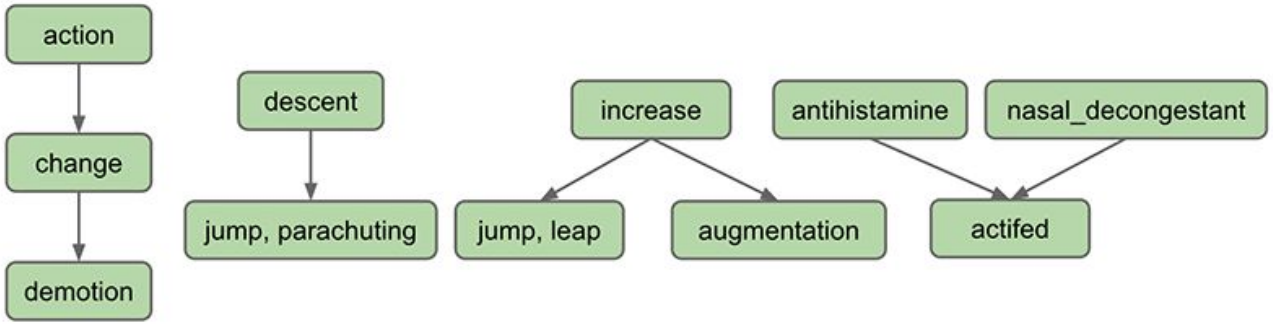


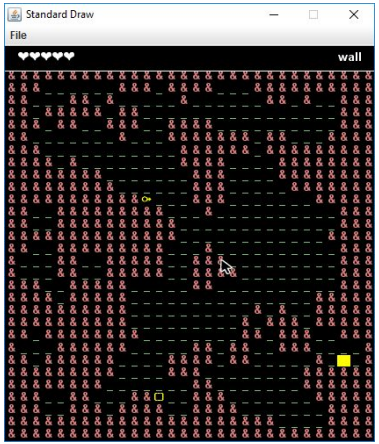
Reflection Continued

In industry, you will be part of projects like Gitlet.

- One big project + multiple objectives.
- Time management and deadlines.
- Comprehensive testing and proof of concept.

Design before you code! But how can you come up with a good design in the first place?





Importance of Thinking Before Coding

Common Issue: Coding a lot before designing or thinking wastes time on bad implementations.

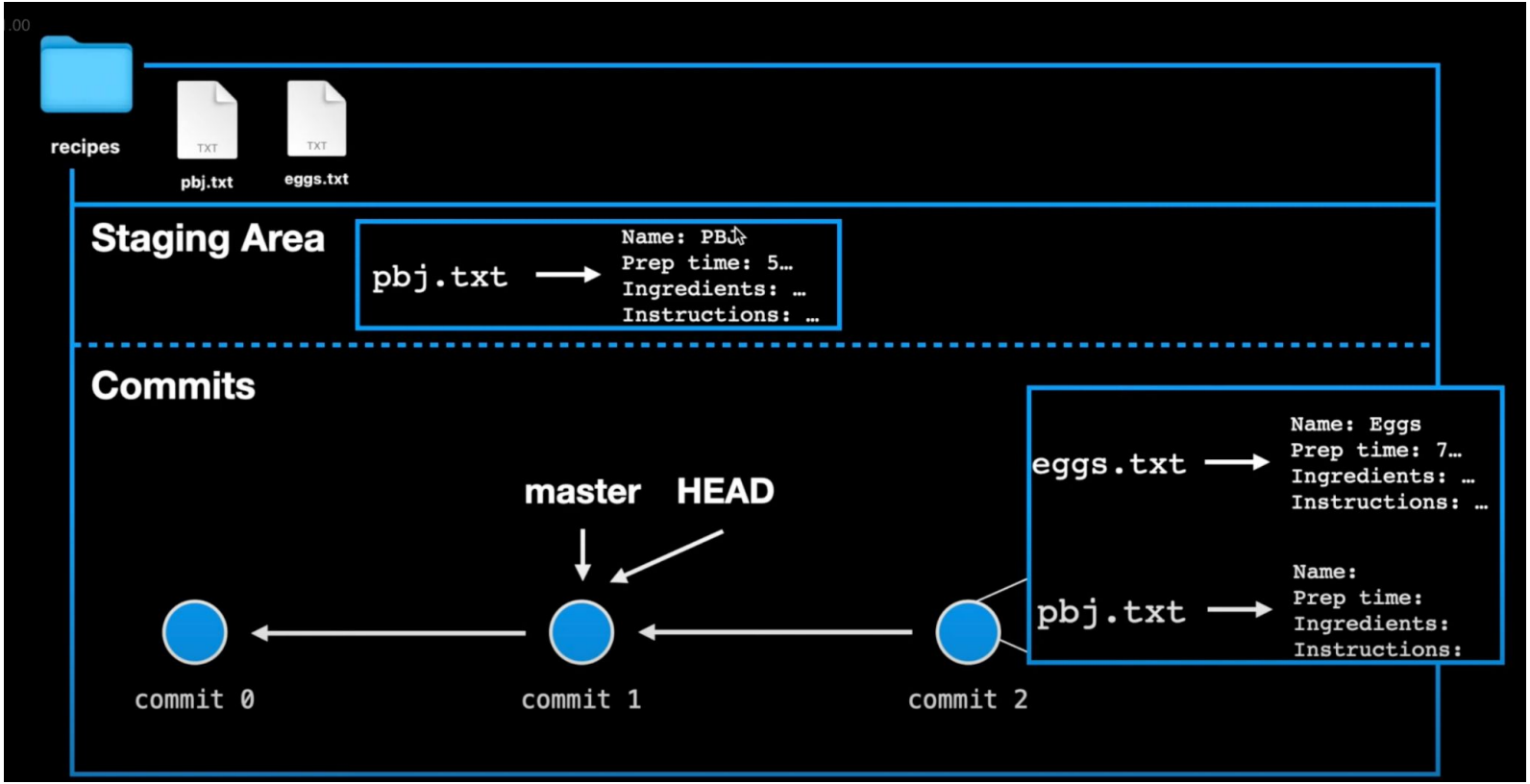
- “My implementation is going to change after all, so I don't need a really detailed design doc.”

Although somewhat true, it is still important to come up with an initial design. However, if the project is too big, how can you start designing in the first place?

- I did not pay attention to the subparts of the project, such as merge, add, commit, and took the project as a whole and had no easy way to start designing!

Solution: Pick up a pen and draw out solutions to small pieces at a time!

Gitlet Continued



Whiteboarding

Two fundamental steps to write a program:

- Try to come up with algorithm that solves the problem.
  - Find a solution in English (or any other language), test yourself, try it, redo it!
  - Document it and convert it to pseudocode.
- Convert this algorithm into a working code.

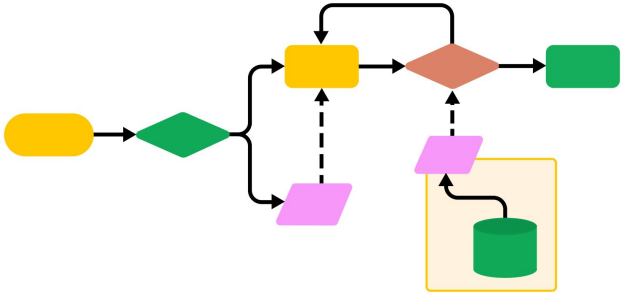
Whiteboarding allows us the start from the first step without a lot of friction since it is freeflow. Pick up pen and paper and visualize what you are thinking!

- Separate the algorithm writing from coding.
  - If the algorithm is flawed, you've spent less time/effort on the flawed algorithm
- Break down the problem even further to little pieces or simplest versions.

Whiteboarding

Whiteboarding:

- Problem A
  - Sub Problem A.1
    - Sub Sub Problem A.1.I
      - Ideate → visualize → test it → modify it → repeat



- Simplify a big problem!
- Convey your ideas and communicate with other people more easily.

Whiteboarding is when you are face to face with someone in the present, while design documents are for the future. Whiteboarding is for you and the people around you to understand in the present time. Design doc is for another person you don't meet.

SWE Fundamentals I

Lecture 34, CS61B, Spring 2024

Gitlet

Whiteboarding

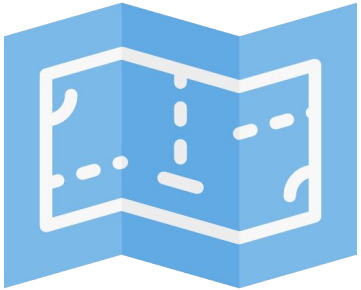
SWE Fundamentals

How to Practice

Q & A

Architect vs Builder

A computer engineer is like an architect:



- The architect is the one who plans out everything:
  - Comes up with initial plan
  - Makes calculations
  - Proof of concept
  - Creates something new in blueprint form

A computer programmer is more like a builder

- The builder is the one who follows the specification:
  - Given blueprint or plan, it provides the end result
  - Good at automating things
  - Cannot come up with a new blueprint by itself

Computer Engineer vs Programmer

Now that LLMs (GPT, Devin, Gemini, Llama 2, etc…) are a thing, the building process is more automated than ever. LLMs are not good at figuring out something new (for now…), so they're not good engineers.

You are the engineer! You are the one who plans out things. You create the blueprint.

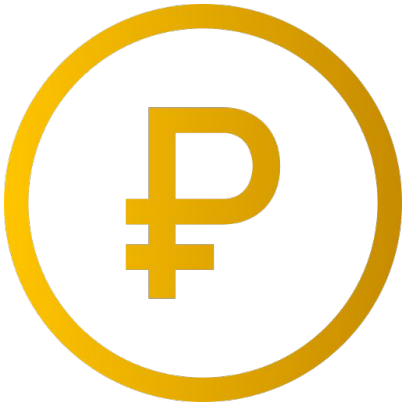
Coming up with blueprint from scratch is hard. Whiteboarding makes things easier to have the engineering mindset. Split up the work one piece at a time!

Whiteboarding: Example (Your Answer)

You are a manager at a fintech company called Perihan. You are going to hire a bunch of interns for the upcoming summer to two teams, one based in San Francisco, and one based in New York. Both teams need a certain number of people, and there are many more candidates that you have slots. You want the best interns but you also want to make interns happy.

Your boss has told you to figure it out, without much further detail.

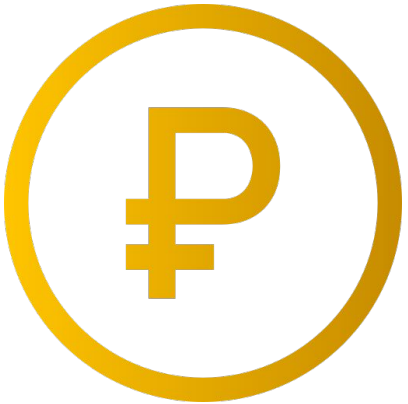
- How do you decide who to hire?
- What aspects do you want to consider while hiring?



Whiteboarding: Example

You are a manager at a fintech company called Perihan. You are going to hire a bunch of interns for the upcoming summer to two teams, one based in San Francisco, and one based in New York. Both teams need a certain number of people, and there are many more candidates that you have slots. You want the best interns but you also want to make interns happy.

Your boss has told you to figure it out, without much further detail.



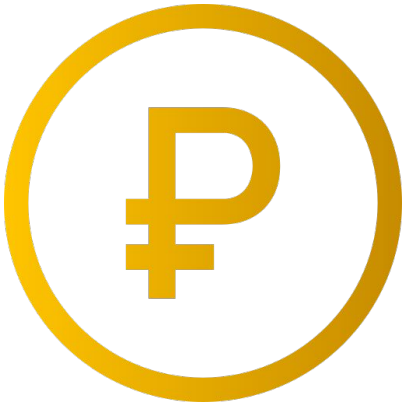
- How do you decide who to hire?
- What aspects do you want to consider while hiring?

For simplicity, model each intern as having two aspects:

- Technical skill: An integer "score" from 0 to 100
  - Can imagine that we collect this data from interviews
- Location preference: Either SF or NY
  - Can imagine that we collect this data from surveys

Whiteboarding: Example

You are a manager at a fintech company called Perihan. You are going to hire a bunch of interns for the upcoming summer to two teams, one based in San Francisco, and one based in New York. Both teams need a certain number of people, and there are many more candidates that you have slots. You want the best interns but you also want to make interns happy.



For simplicity, model each intern as having two aspects:

- Technical skill: An integer "score" from 0 to 100
- Location preference: Either SF or NY

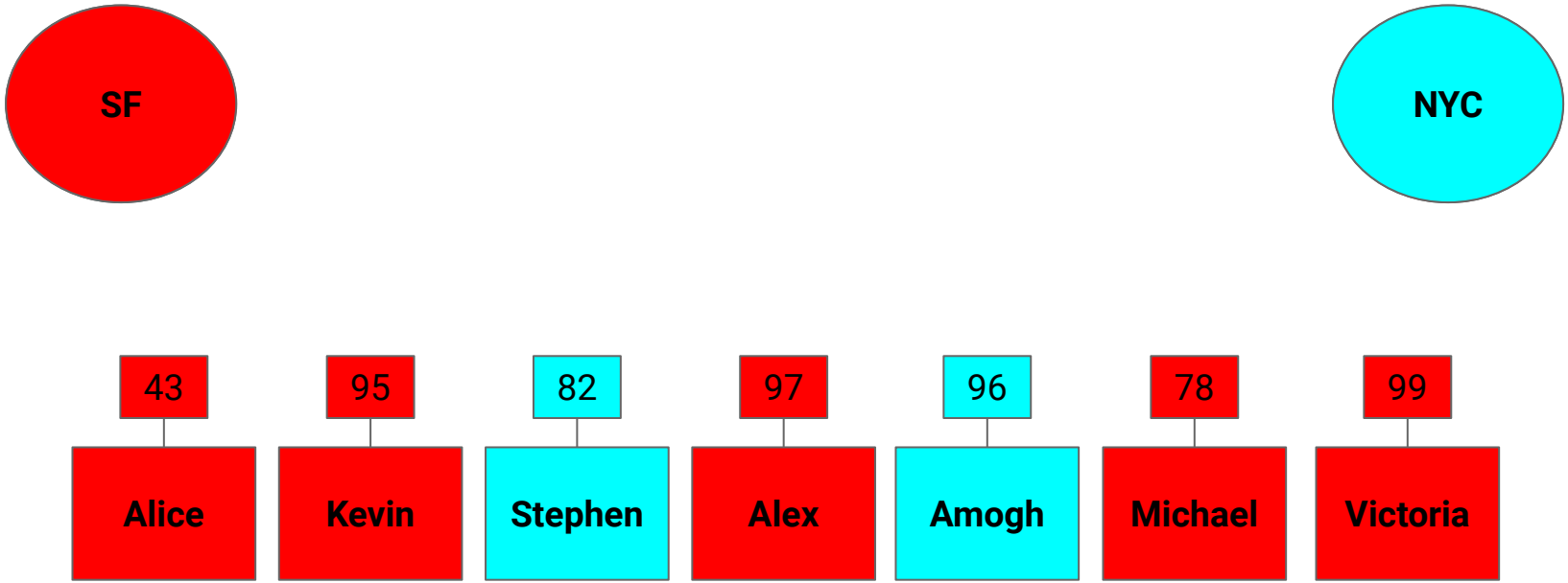
Primary goal: Hire the interns with the highest technical skill

Secondary goal: Assign the interns their preferred office if possible

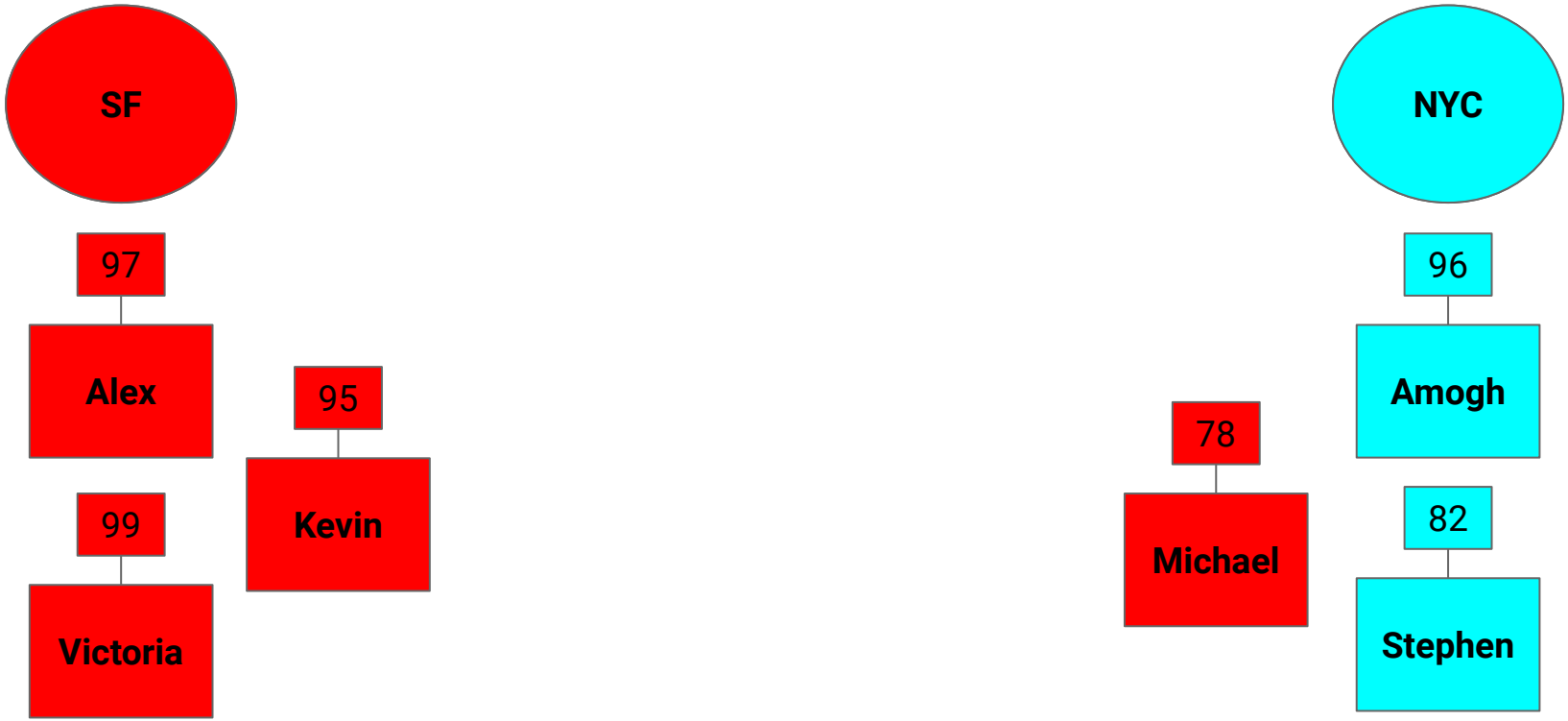
If an intern must be assigned to their unpreferred office, assign the intern with lower technical skill to the other office

Whiteboarding: Example

Whiteboarding: Example



Whiteboarding: Example



SWE Fundamentals II

Lecture 34, CS61B, Spring 2024

Gitlet

Whiteboarding

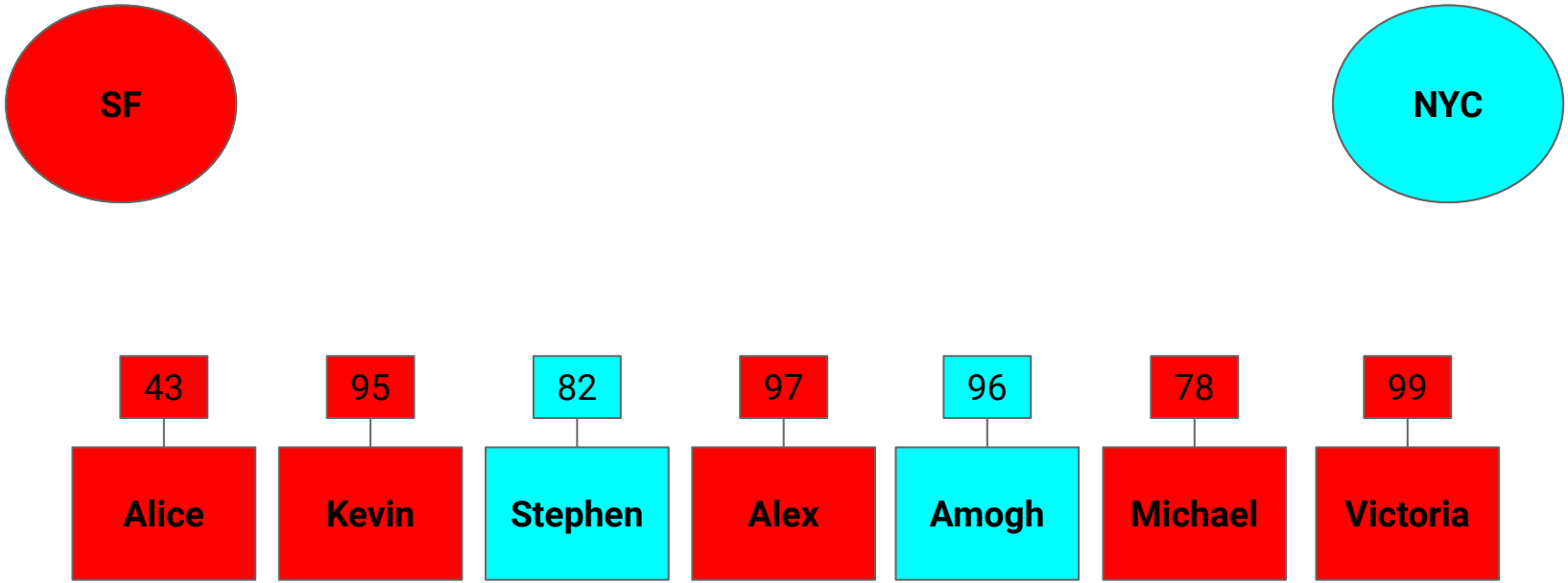
SWE Fundamentals

How to Practice

Q & A

Whiteboarding: Example

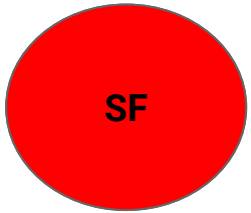
How do you solve this problem?

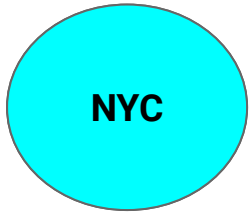


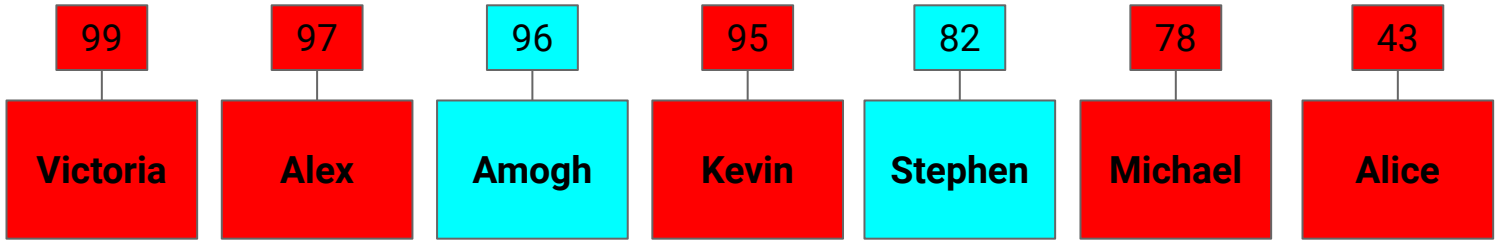
Whiteboarding: Example

Naive solution:
1- Sort the interns
2- Fill up the locations!

ΘNlogN (N: # of interns applied)



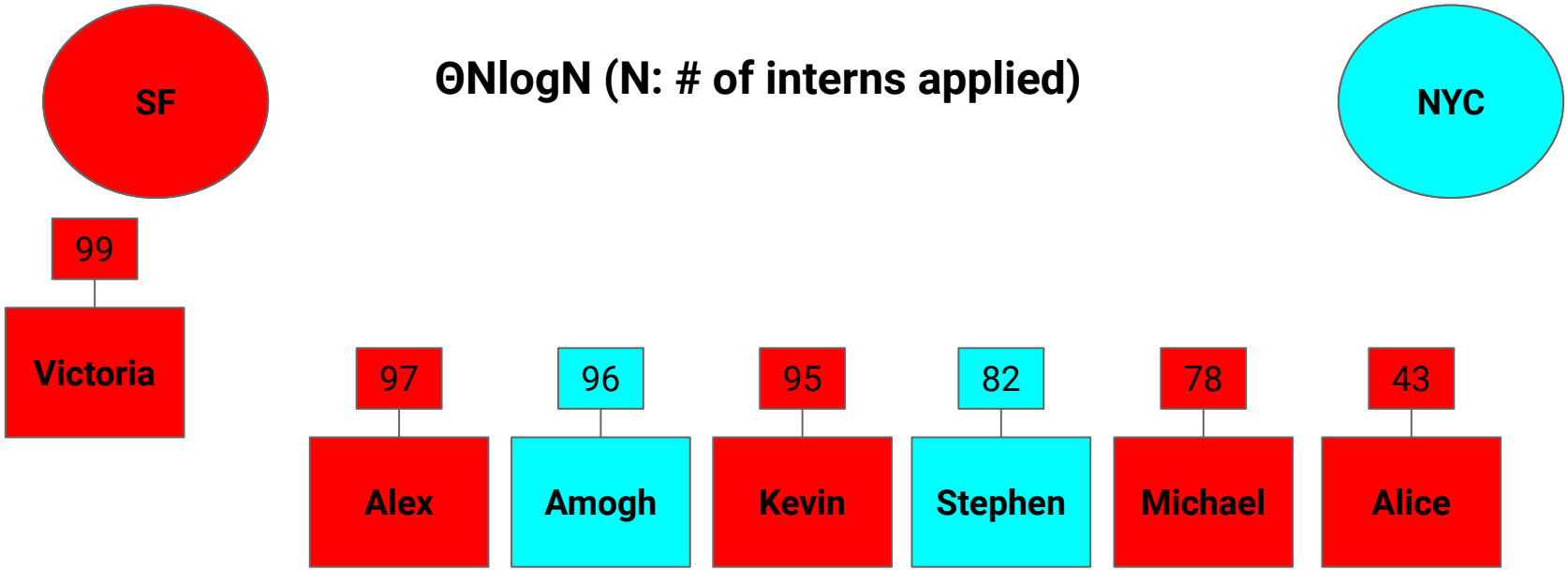




Whiteboarding: Example

Naive solution:
1- Sort the interns
2- Fill up the locations!

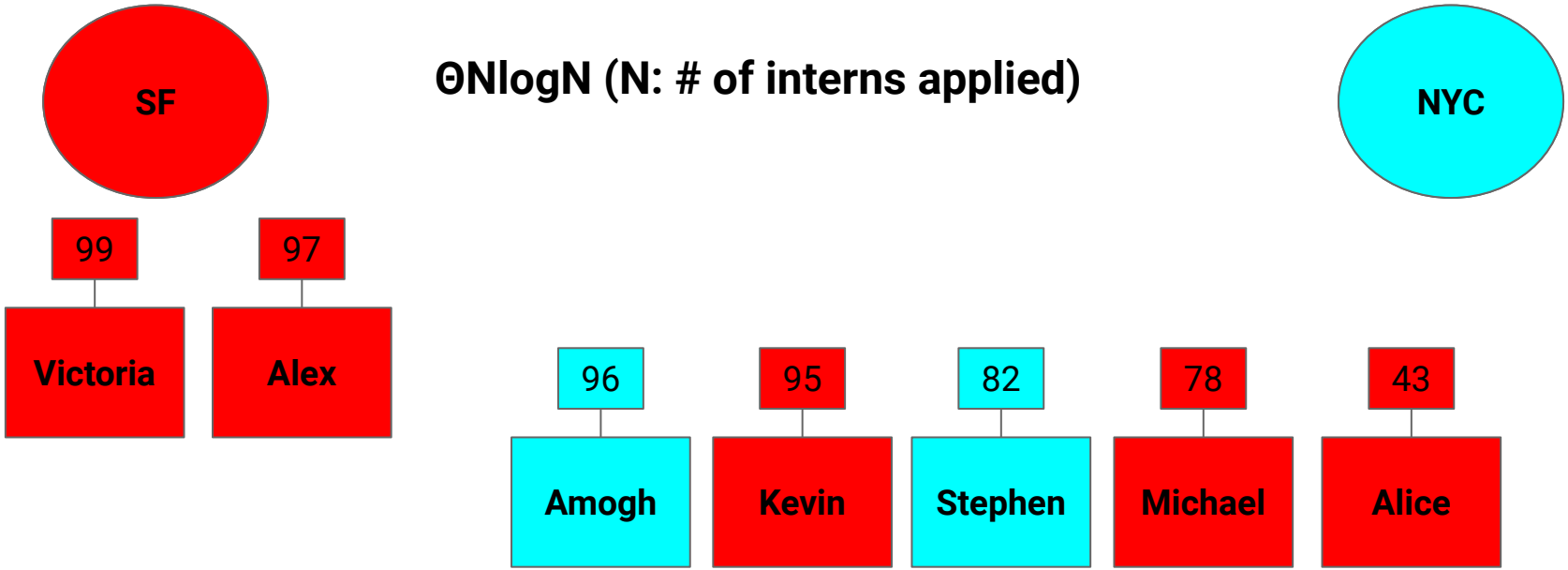
ΘNlogN (N: # of interns applied)



Whiteboarding: Example

Naive solution:
1- Sort the interns
2- Fill up the locations!

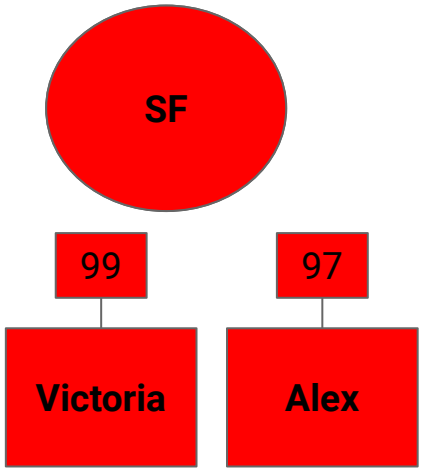
ΘNlogN (N: # of interns applied)

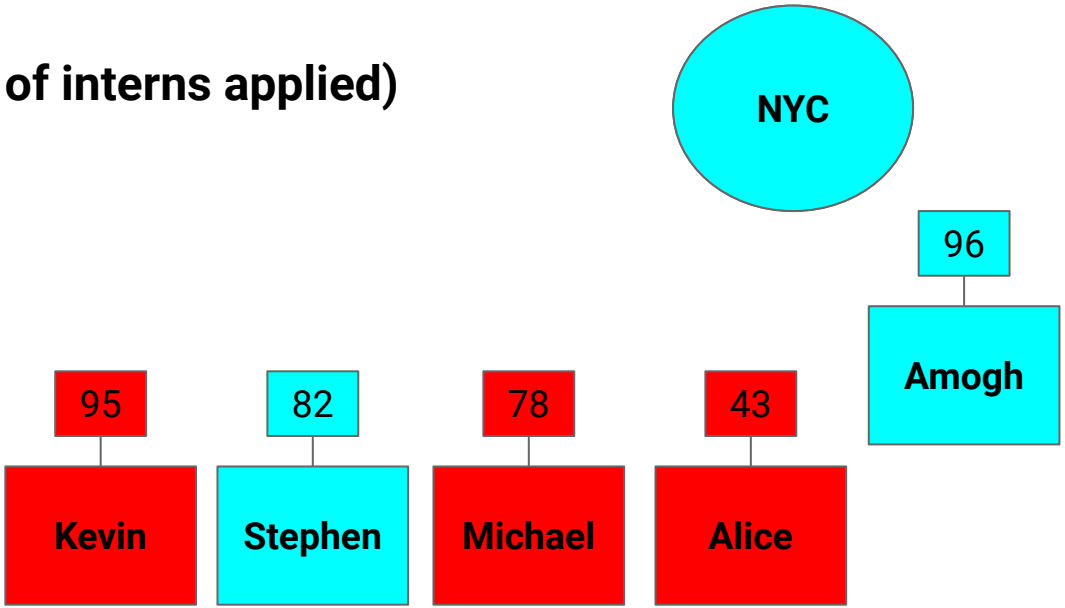


Whiteboarding: Example

Naive solution:
1- Sort the interns
2- Fill up the locations!

ΘNlogN (N: # of interns applied)

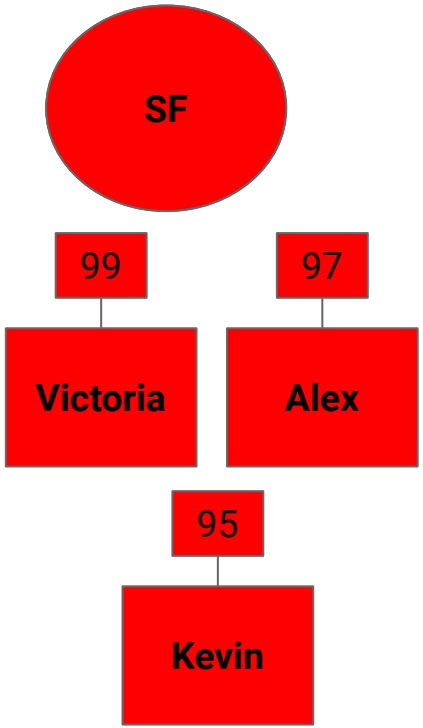


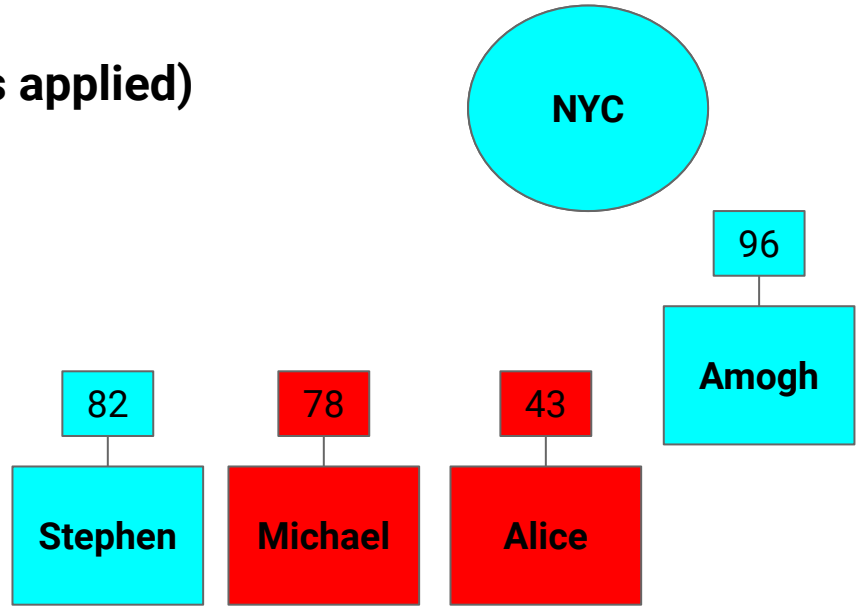


Whiteboarding: Example

Naive solution:
1- Sort the interns
2- Fill up the locations!

ΘNlogN (N: # of interns applied)

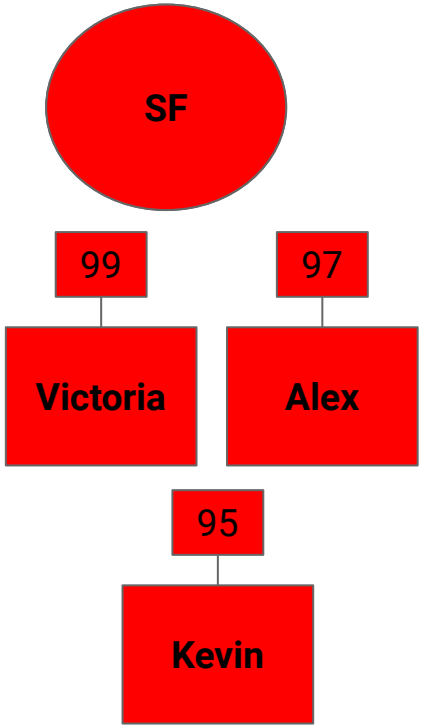


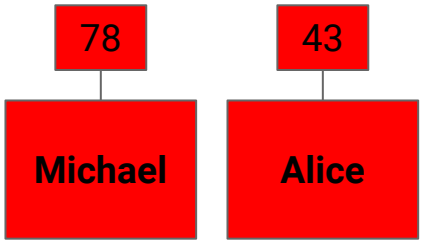


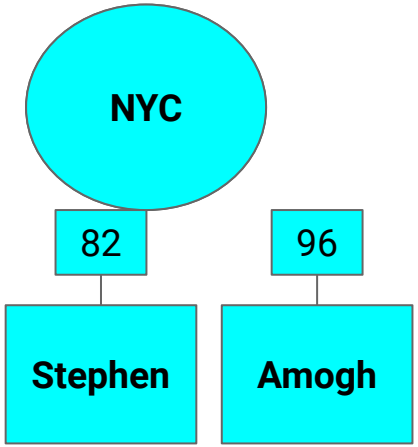
Whiteboarding: Example

Naive solution:
1- Sort the interns
2- Fill up the locations!

ΘNlogN (N: # of interns applied)



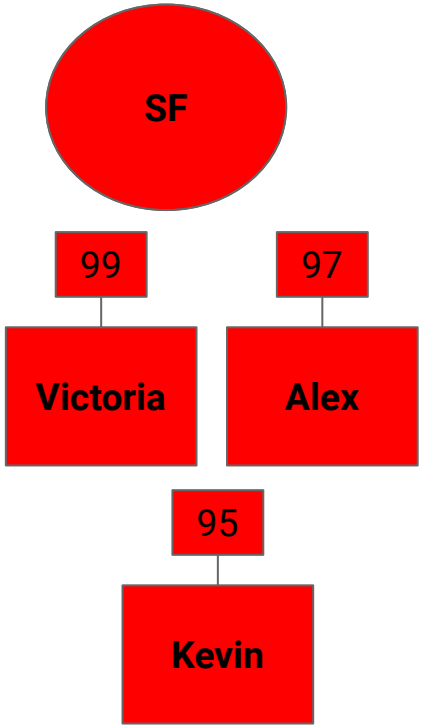


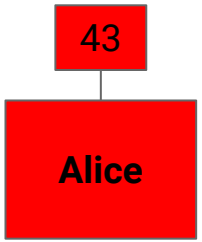


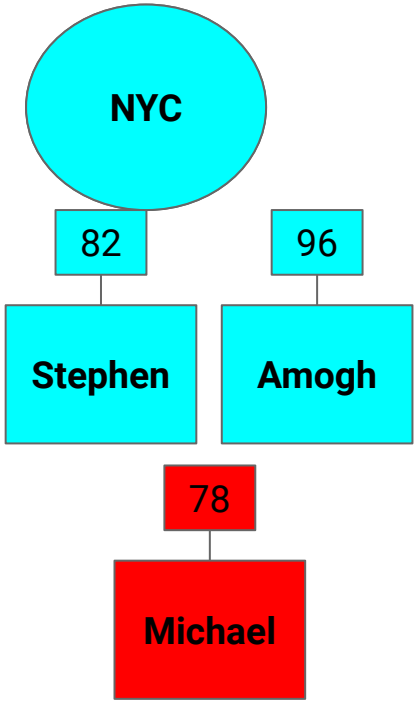
Whiteboarding: Example

Naive solution:
1- Sort the interns
2- Fill up the locations!

ΘNlogN (N: # of interns applied)



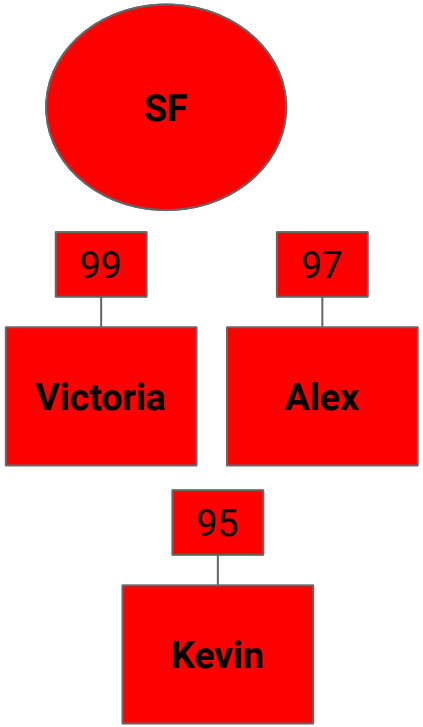


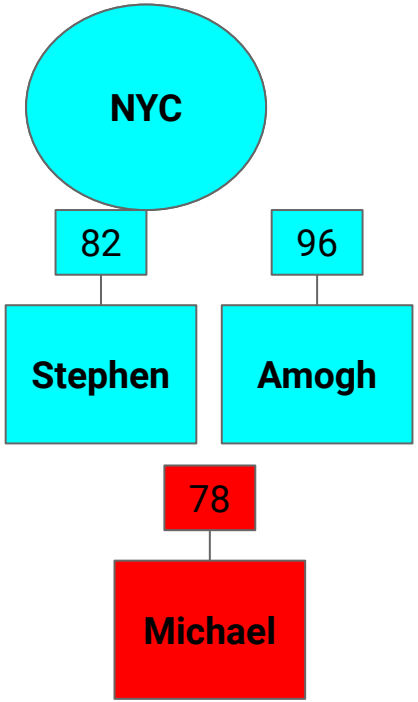


Whiteboarding: Example

Naive solution:
1- Sort the interns
2- Fill up the locations!

ΘNlogN (N: # of interns applied)



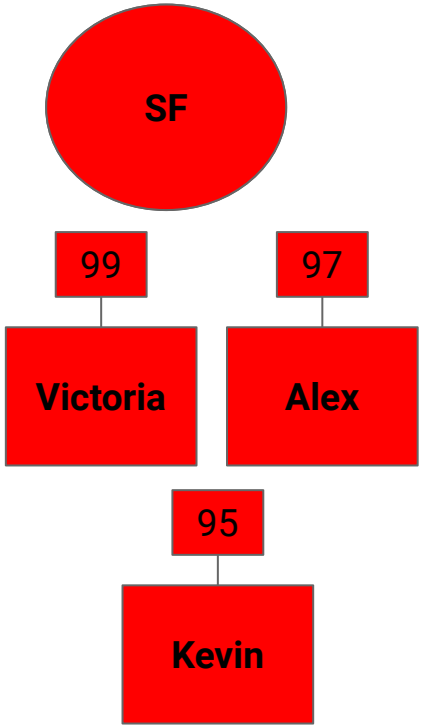


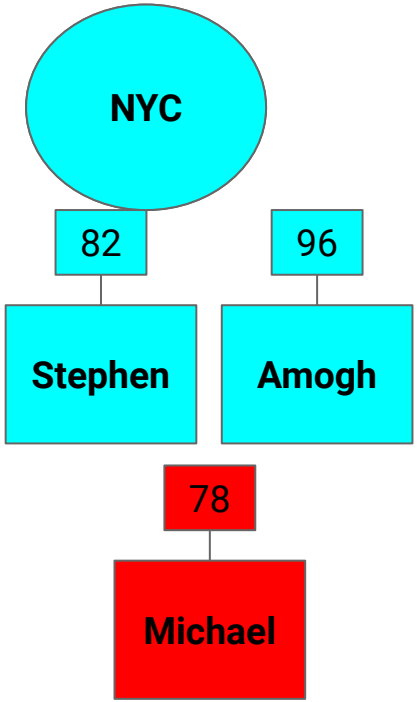
Whiteboarding: Example

Naive solution:
1- Sort the interns
2- Fill up the locations!

ΘNlogN (N: # of interns applied)

Can we do better?





Whiteboarding

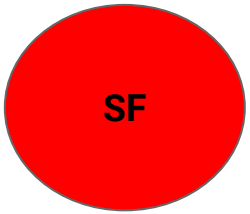
Our naive approach works, but it is slow when N >> the number of slots we have available. We can do better.

Let’s come up with design!

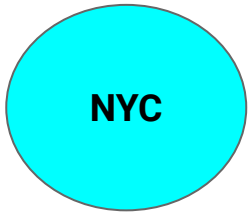
- Pretend like the interns come one at a time.
- Keep track of preferences and technical skills of interns.
- Be aware of the capacity of both locations.
- **Primary goal**: Hire the interns with the highest technical skill
- **Secondary goal**: Assign the interns their preferred office if possible
- If an intern must be assigned to their unpreferred office, assign the intern with lower technical skill to the other office

Whiteboarding: Example

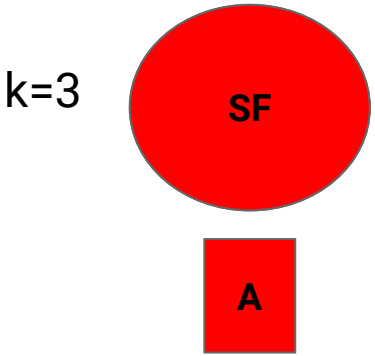
k=3

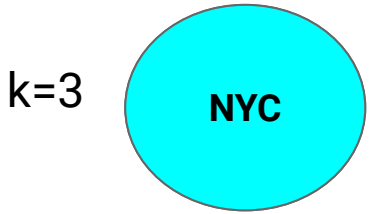


k=3

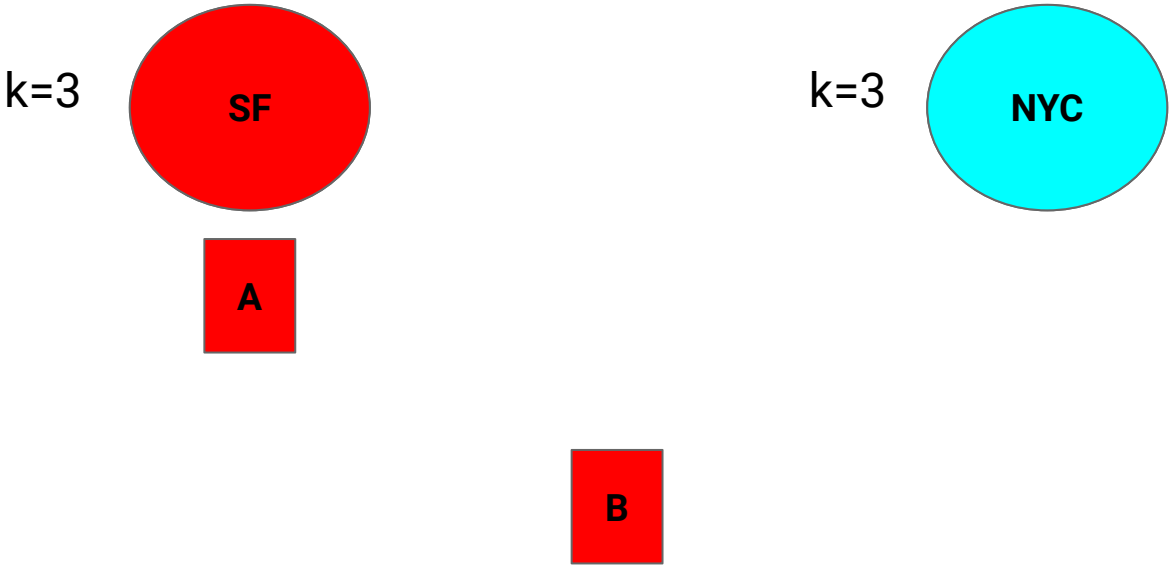


Whiteboarding: Example

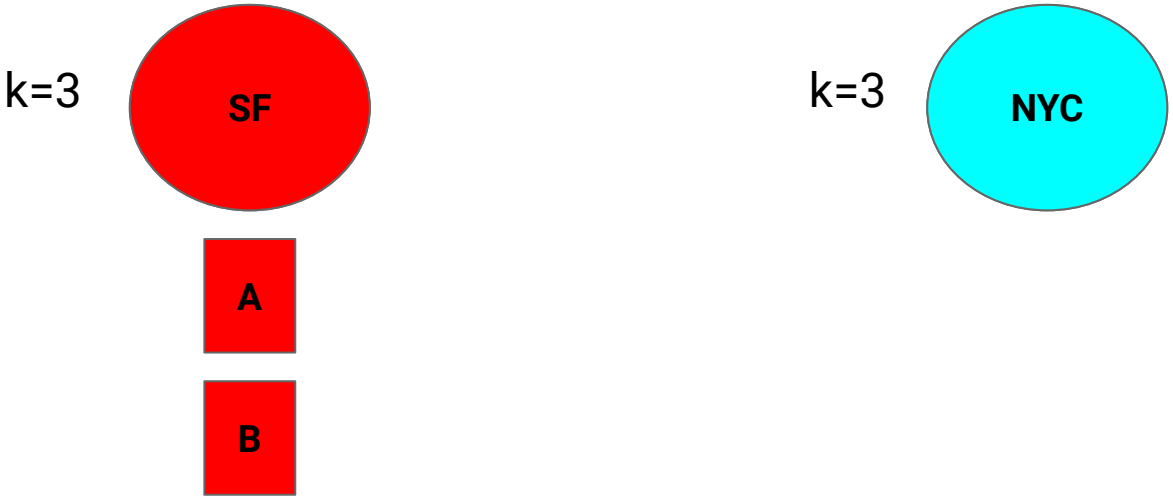




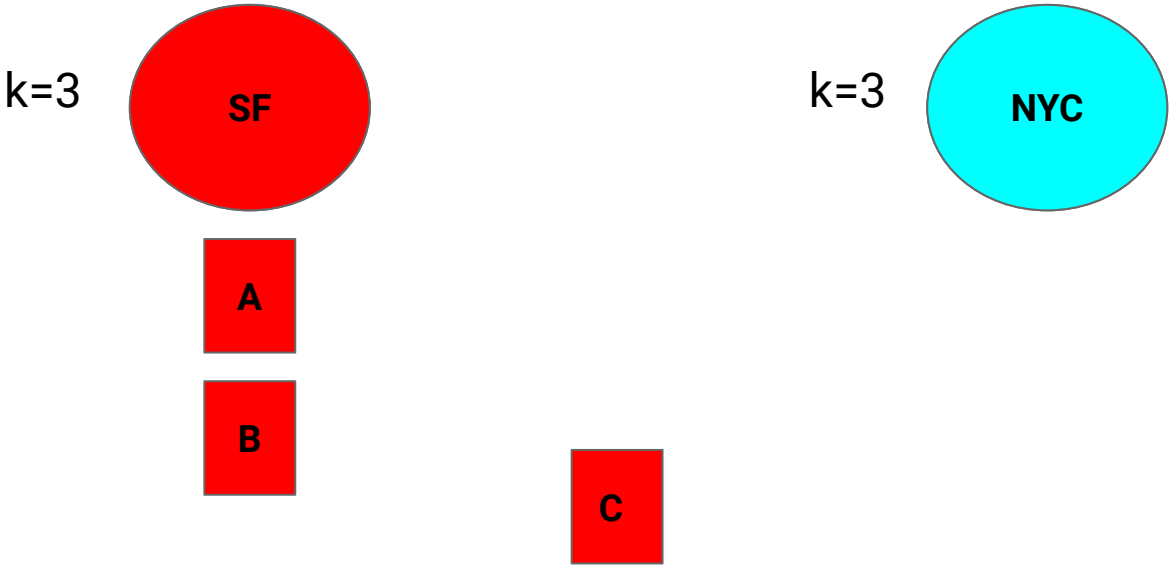
Whiteboarding: Example



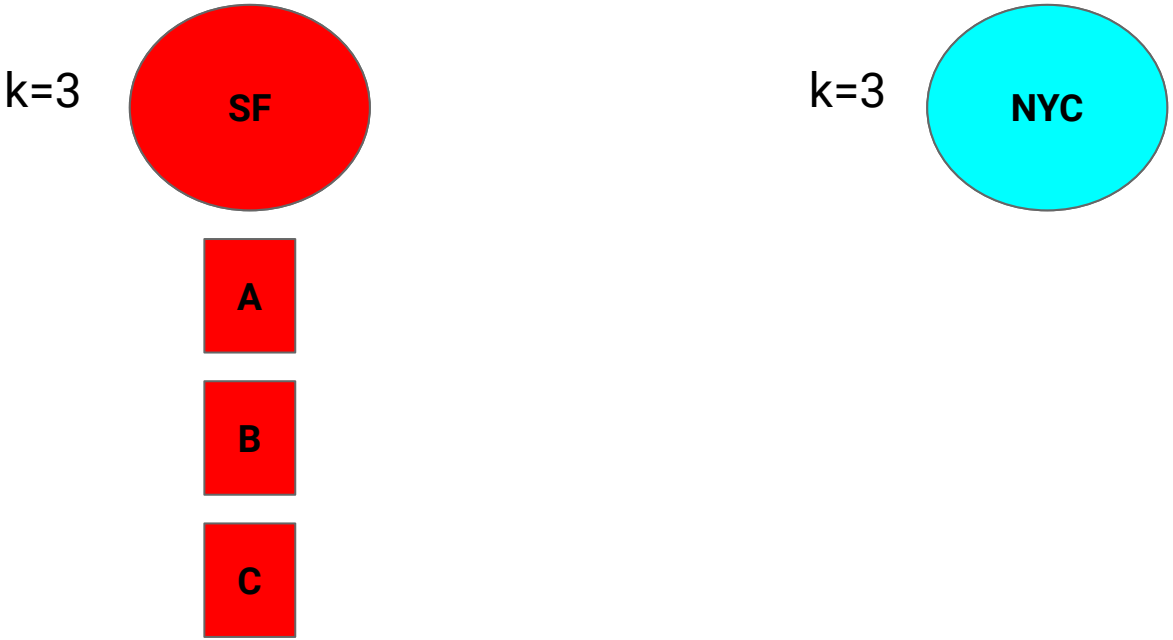
Whiteboarding: Example



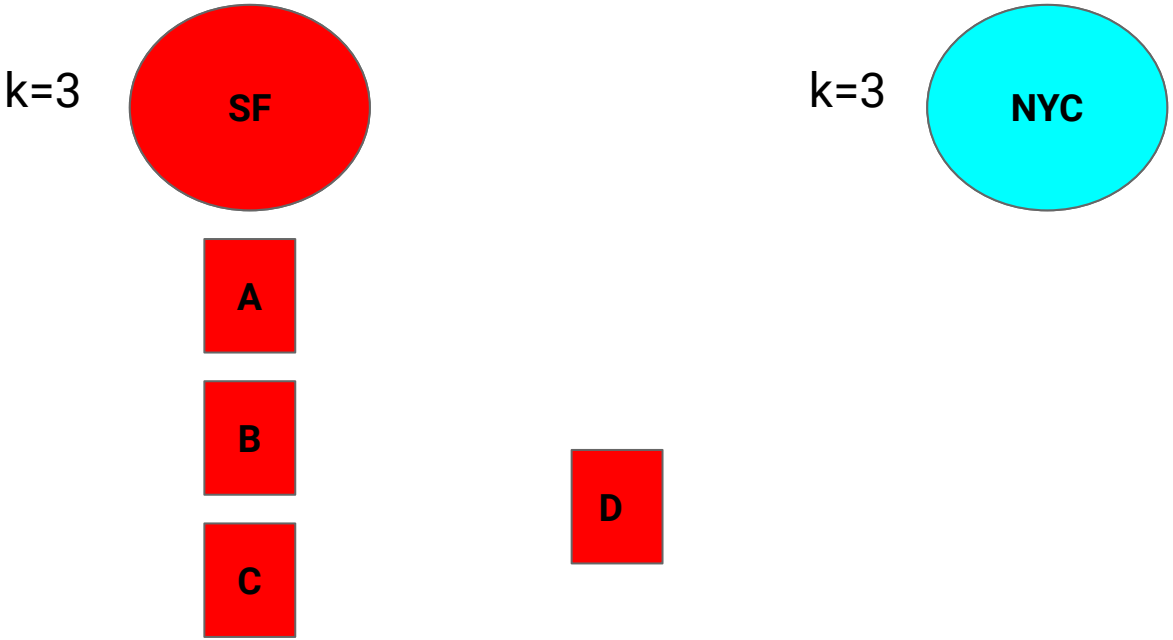
Whiteboarding: Example



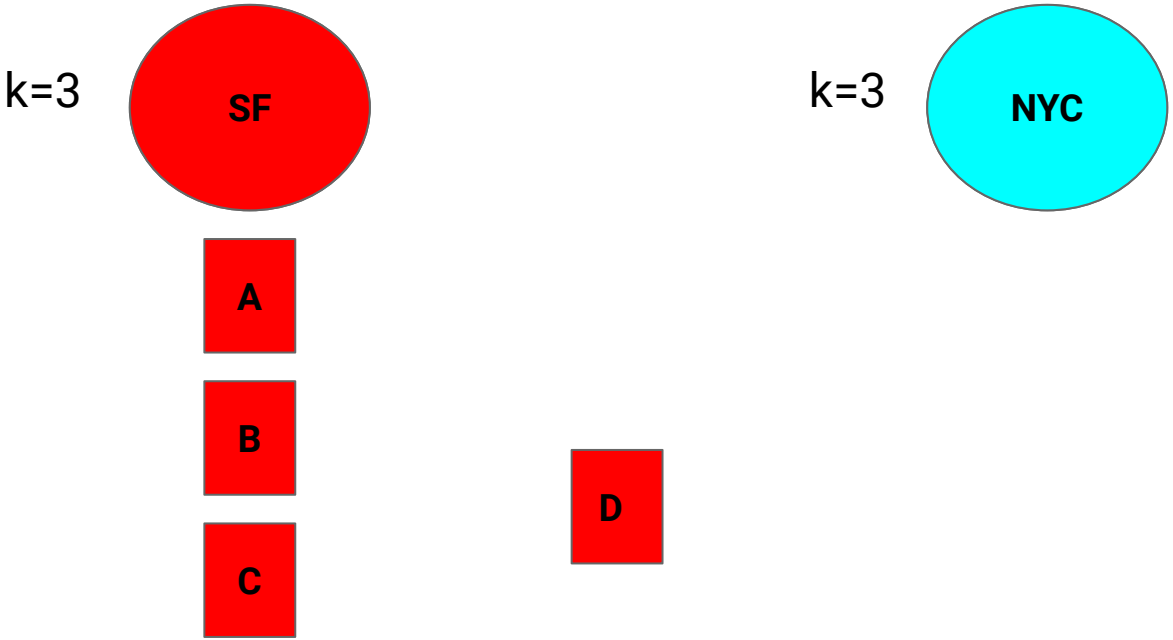
Whiteboarding: Example



Whiteboarding: Example

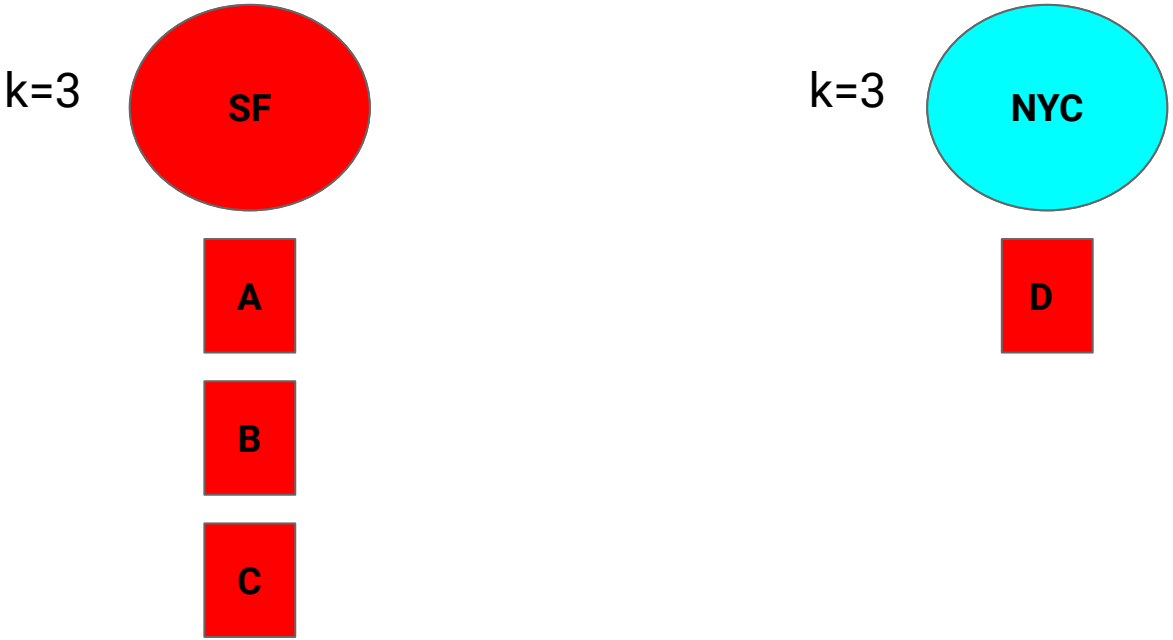


Whiteboarding: Example

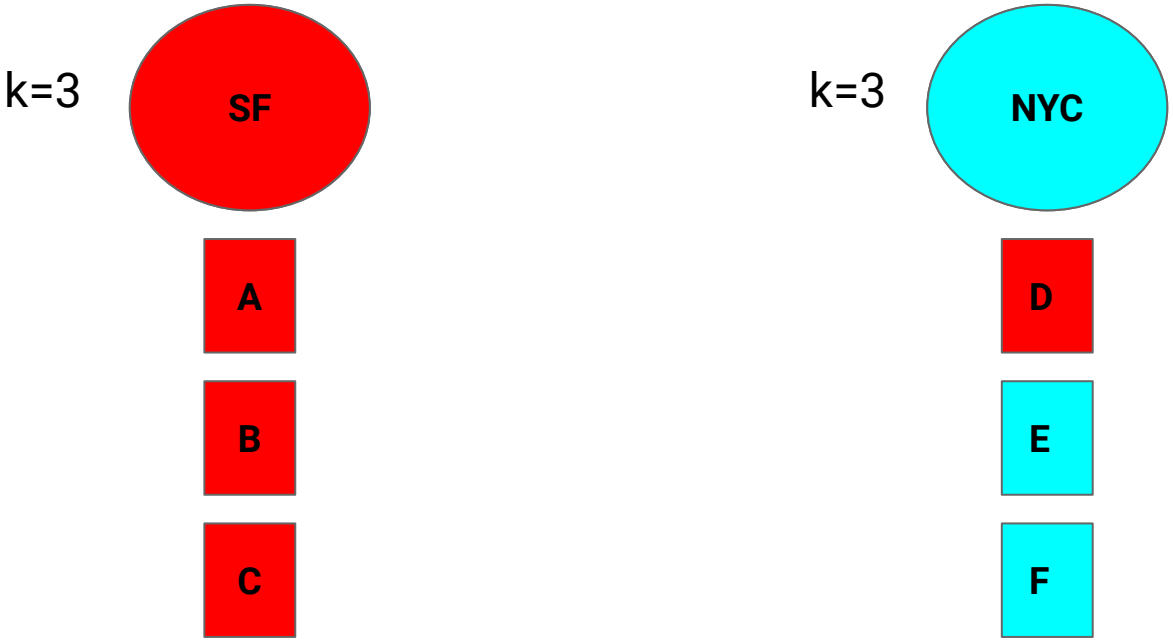


All of interns' technical skills in SF greater than D!

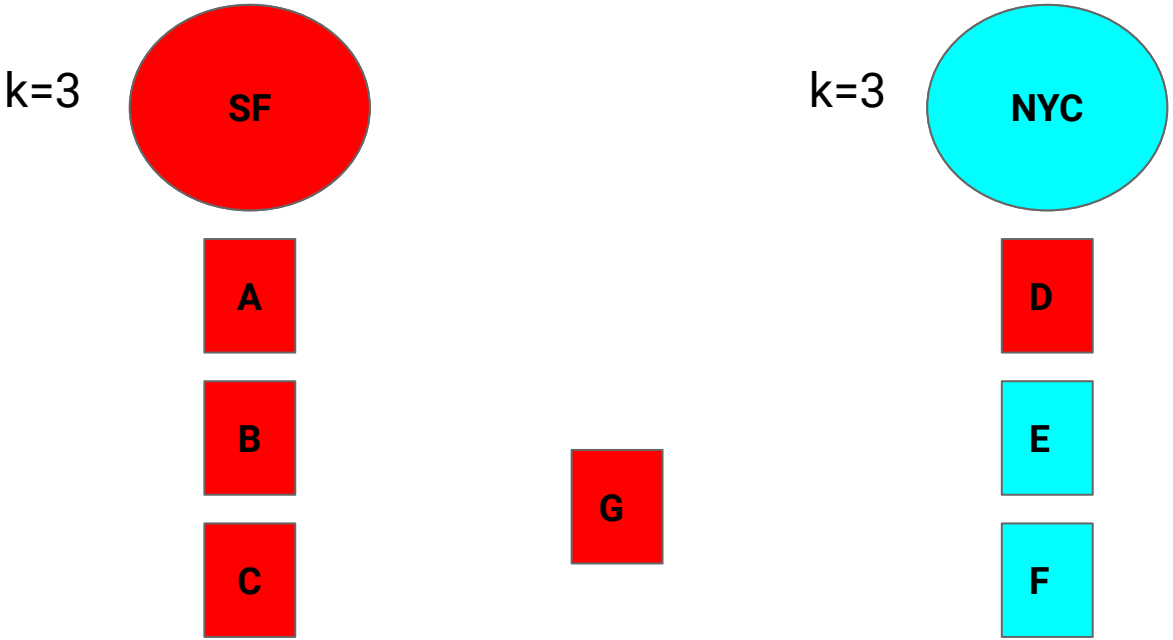
Whiteboarding: Example



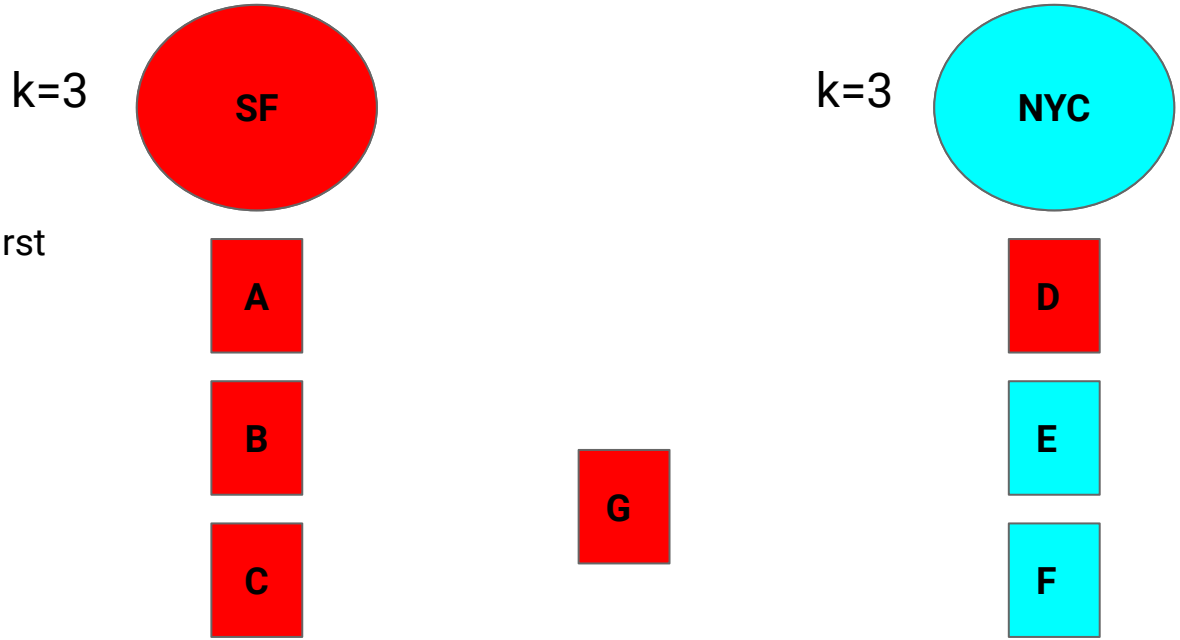
Whiteboarding: Example



Whiteboarding: Example

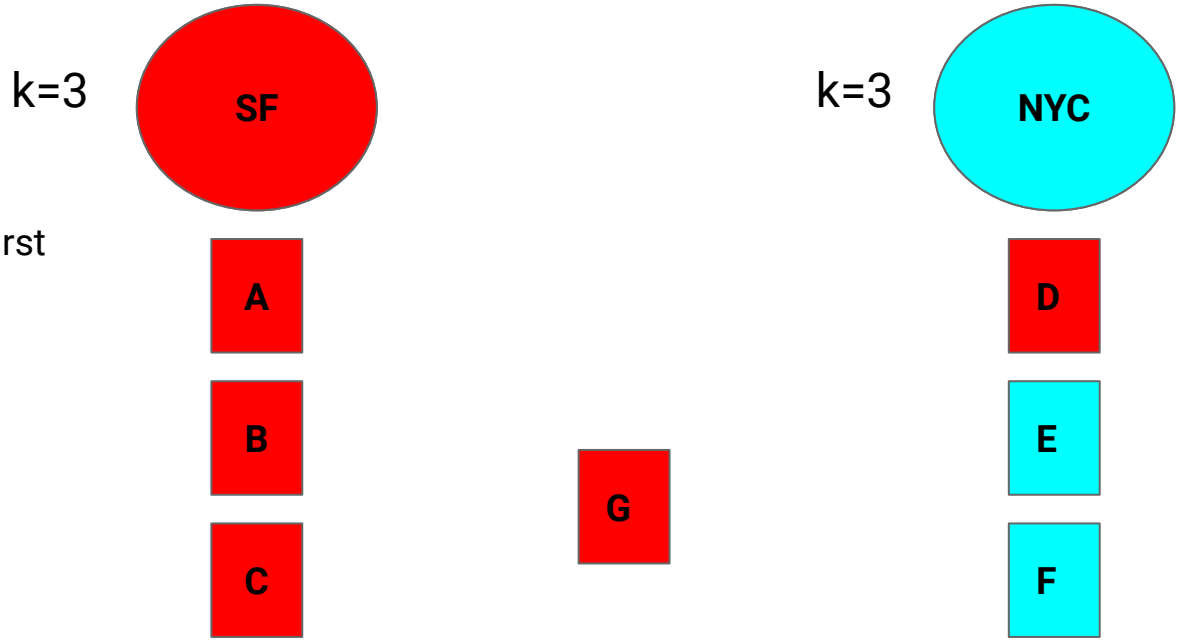


Whiteboarding: Example

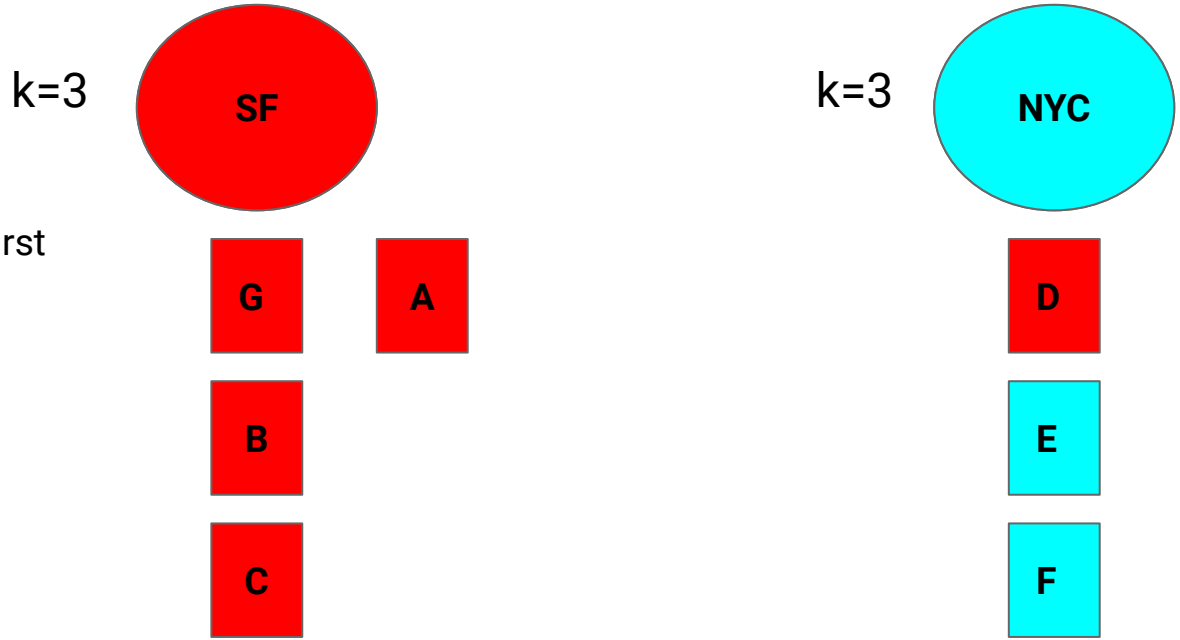


Remember the first objective!

Whiteboarding: Example



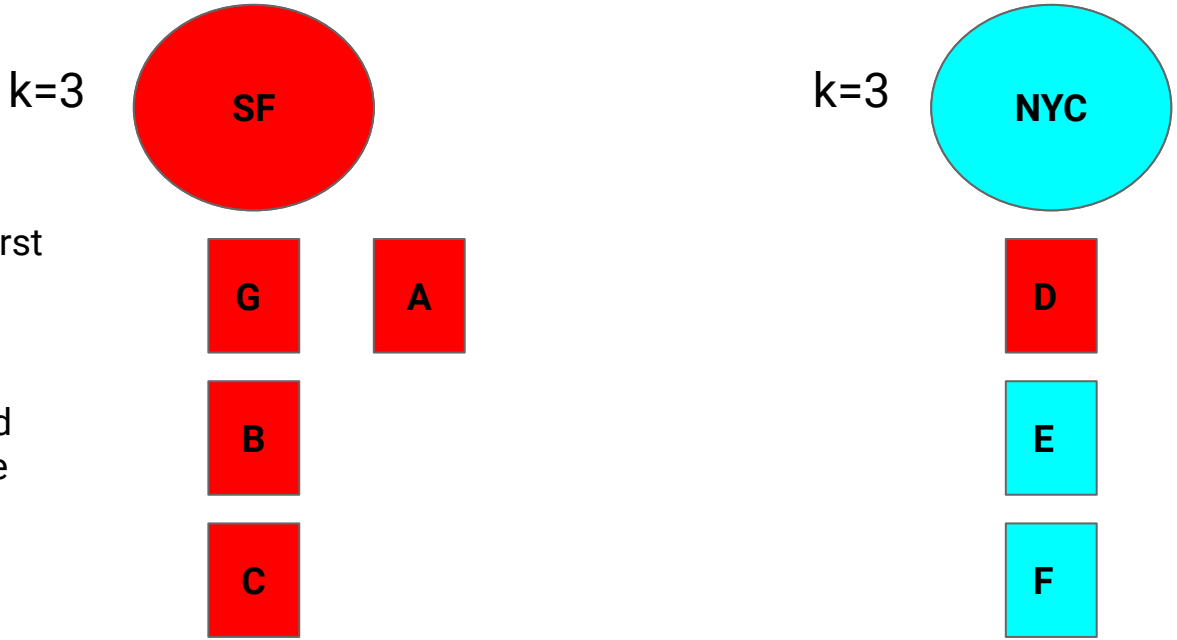
Whiteboarding: Example



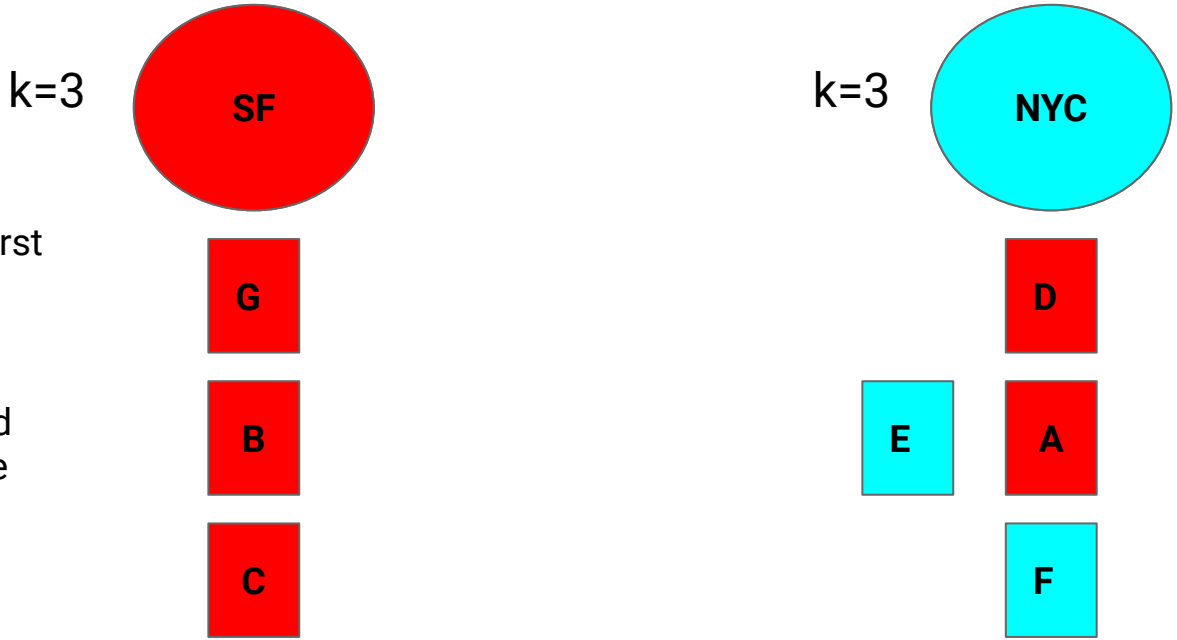
Remember the first objective!

Replace the minimum with incoming one if possible

Whiteboarding: Example



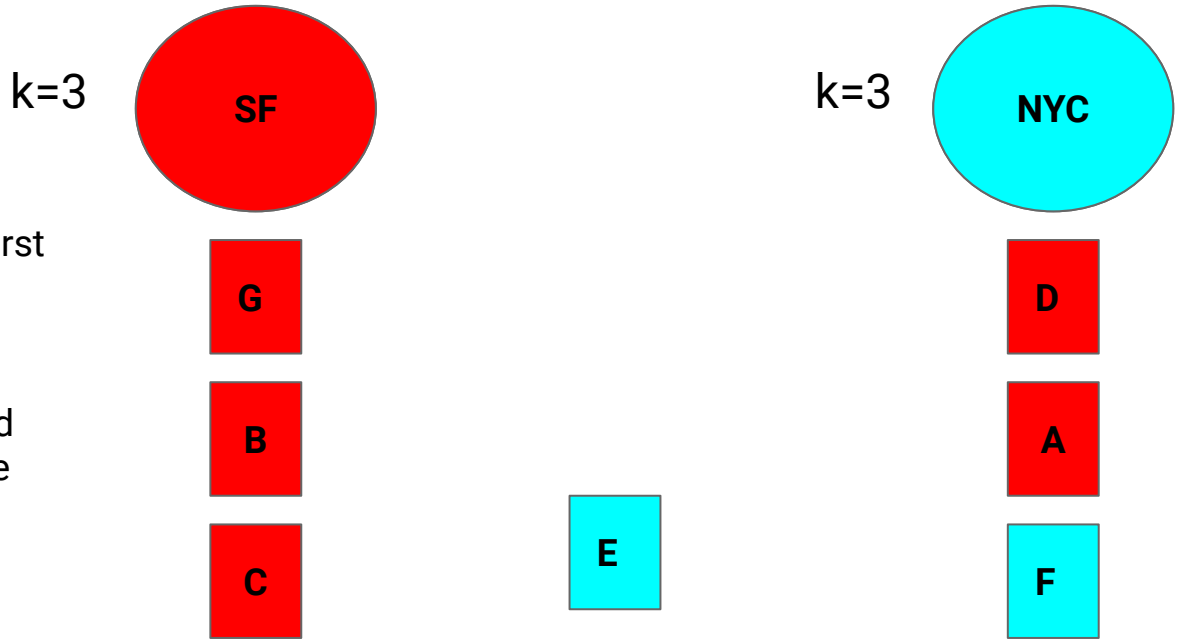
Whiteboarding: Example



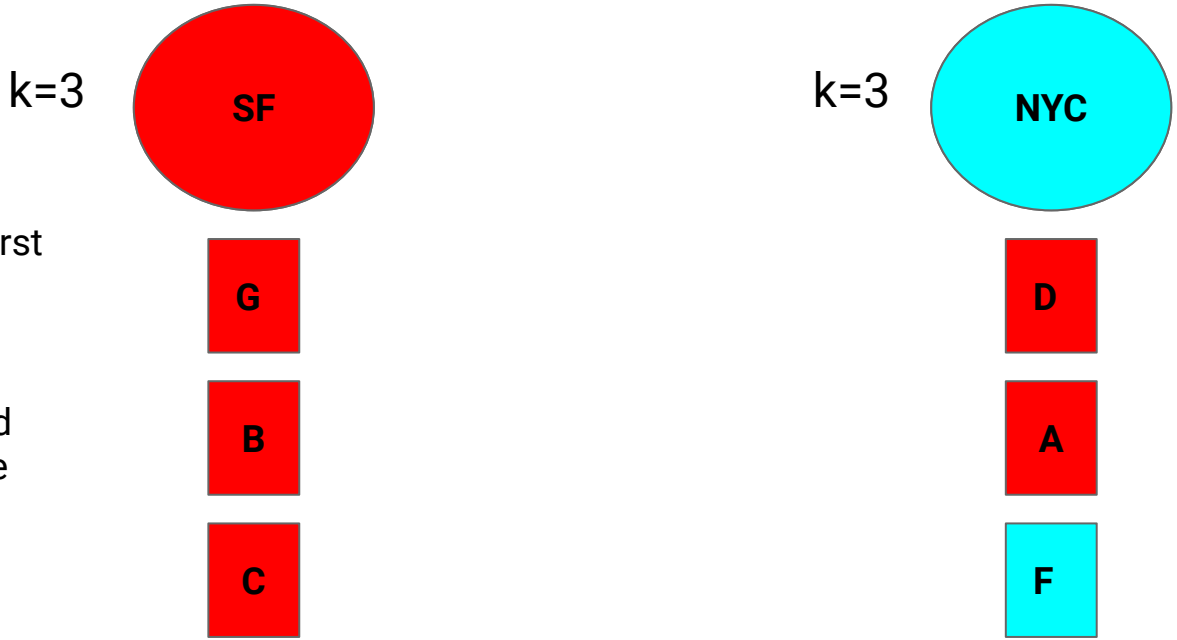
Remember the first objective!

Check if replaced intern still can be placed to other location

Whiteboarding: Example

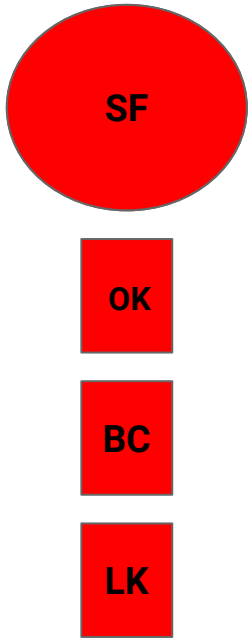


Whiteboarding: Example

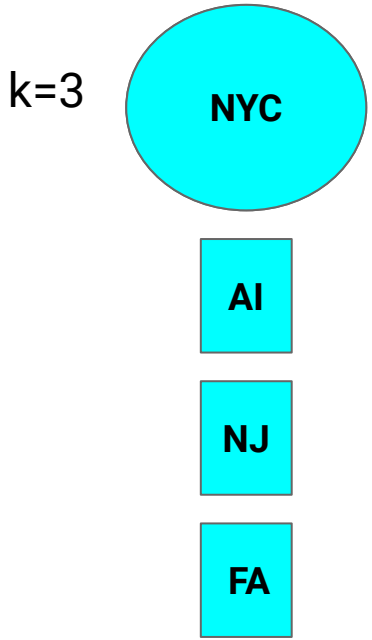


Whiteboarding: Example

k=3



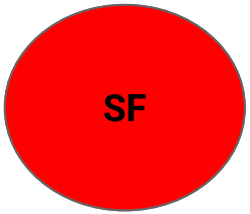
k=3



Repeat and final result.

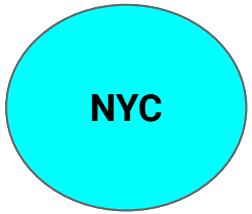
Whiteboarding: Example

k=3



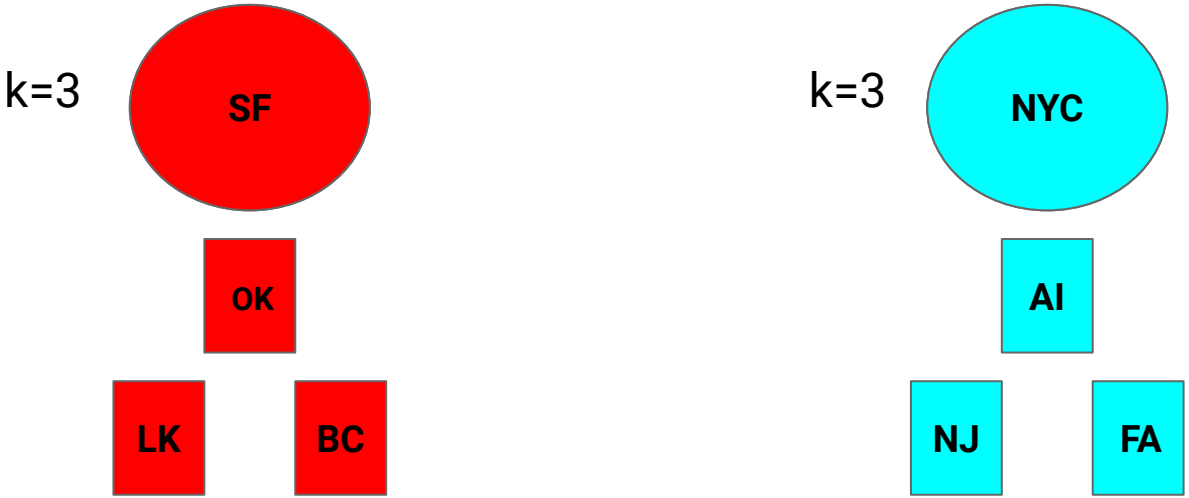
How can I get the minimum element each time?

k=3

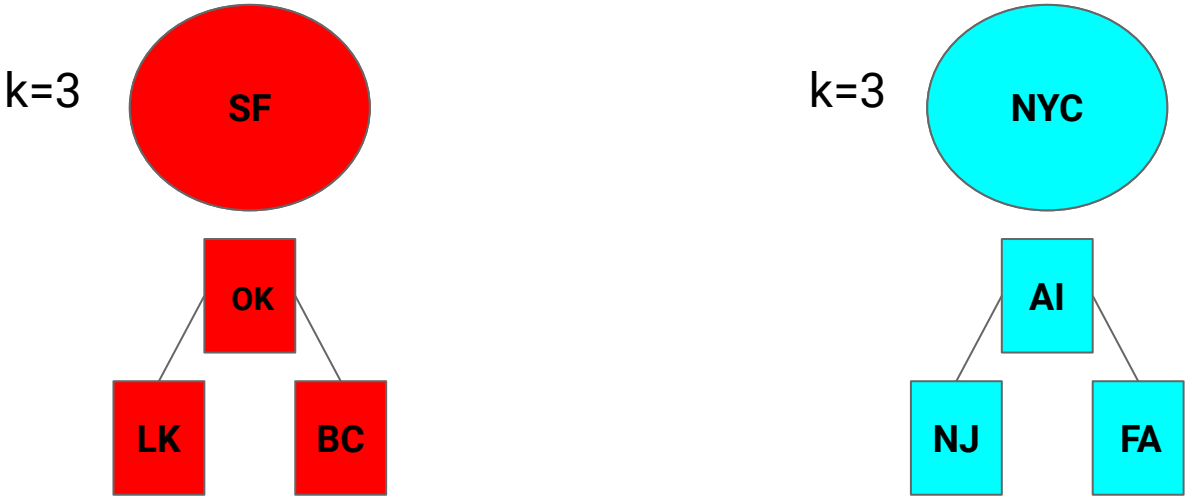


How can I reorganize this data in some type of data structure?

Whiteboarding: Example



Whiteboarding: Example



Min Heap!

Use custom comparator to compare technical skills of applicants.

Conclusion

Solution 1

- Sort the interns
- Take top interns
- Time Complexity: $\Theta(N \log N)$

Solution 2

- Iterate through interns one by one
- Check preference and location capacities
- Place interns
- Time Complexity: $\Theta(N \log k)$

We did not code directly after Solution 1! We realized that we could do better and without thinking about data structures at first, we came up with a better solution.

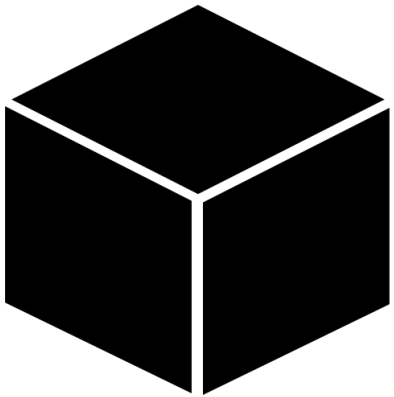
Whiteboarding vs Coding

The idea of a heap is different when you are whiteboarding vs when you are coding.

- At the higher level of algorithm design, you don't even think about a tree or heap - I want to put something and get the minimum value fast! Don't worry about interfaces, underlying variables, API calls.
- While whiteboarding, we broke the problem down into different steps and came up with the min heap structure!
    - Visualization helped along with domain knowledge!

Examples:

- … come up **unique** ids … → Set
- … **match** the partners … → map



How to Practice

Lecture 34, CS61B, Spring 2024

Gitlet

Whiteboarding

SWE Fundamentals

How to Practice

Q & A

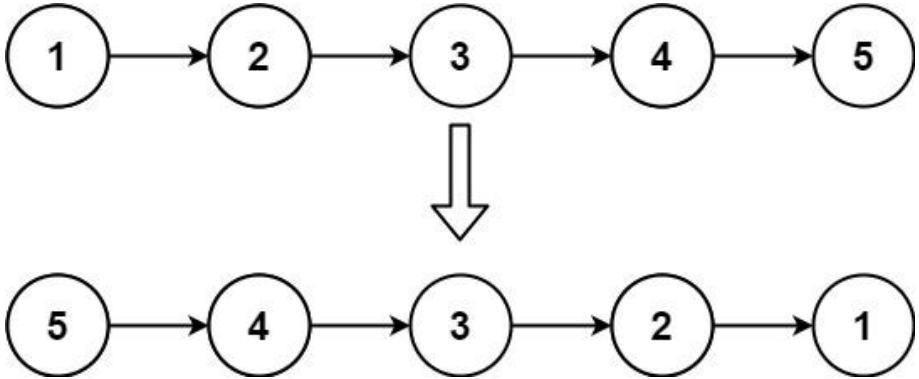
Challenge Problems

How do you get better at these problems? There are tons of resources:

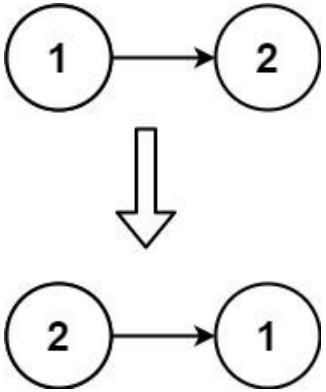
- Leetcode
  - Most popular. Used for job applications and interviews. Easy- to medium-scale programming questions!
- Codeforces
  - Harder than leetcode, CS Olympiad style questions.
- Project Euler
  - More math based/algorithm design-focused: goal is to find the answer without caring too much about your actual code
- Cryptopals
  - Niche problems about computer security
- SpaceChem/TIS-800/A=B/Bombe/7 Billion Humans/etc.
  - Programming-style video games
    - Built-in incentive structure and difficulty curve, so very good for developing skills

Example Leetcode question: Reverse a Linked List

Given the “head” of a singly linked list, reverse the list, and return the reversed list.

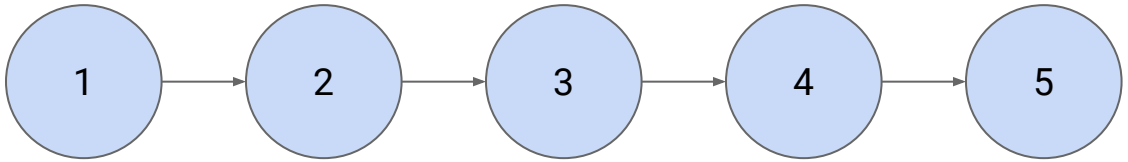


Input: head = [1,2,3,4,5]
Output: [5,4,3,2,1]

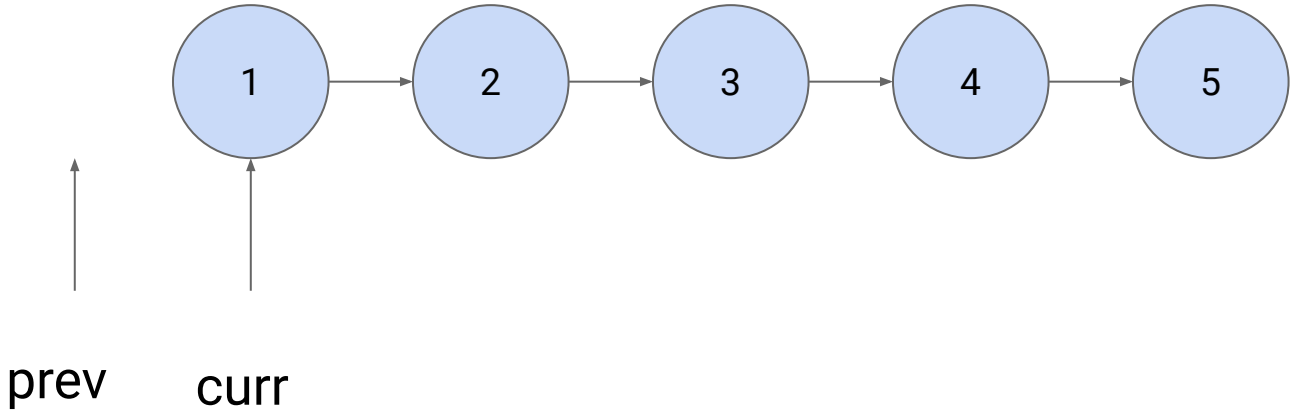


Input: head = [1,2]
Output: [2,1]

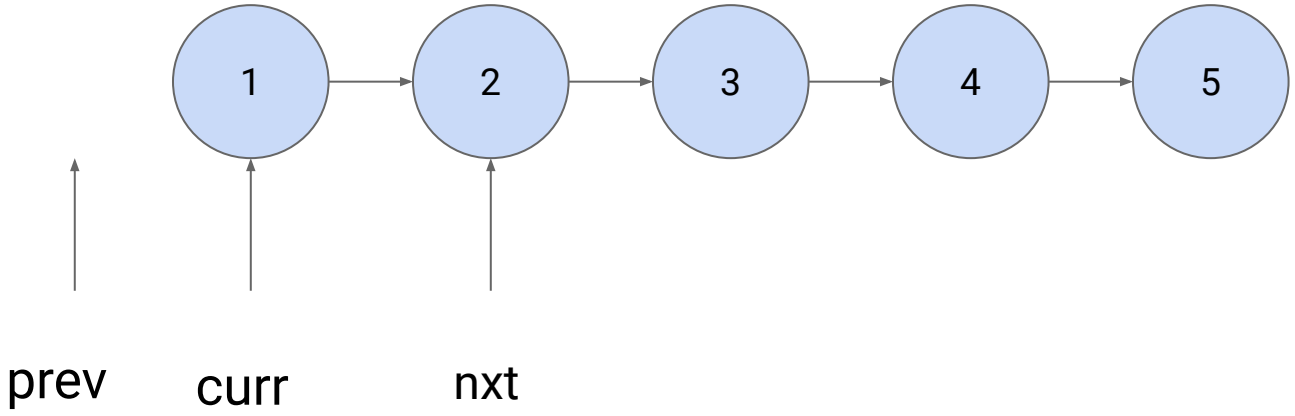
Example Leetcode question: Reverse a Linked List



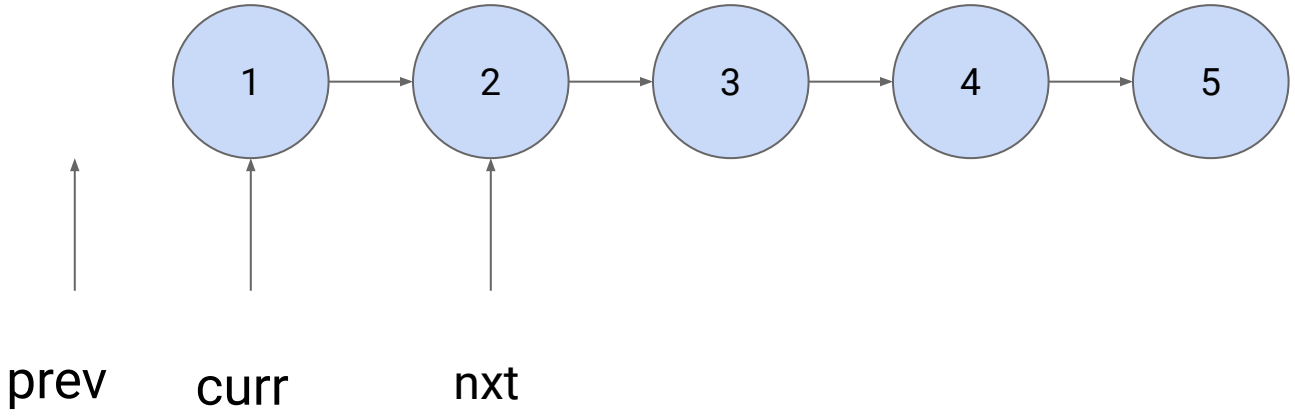
Example Leetcode question: Reverse a Linked List



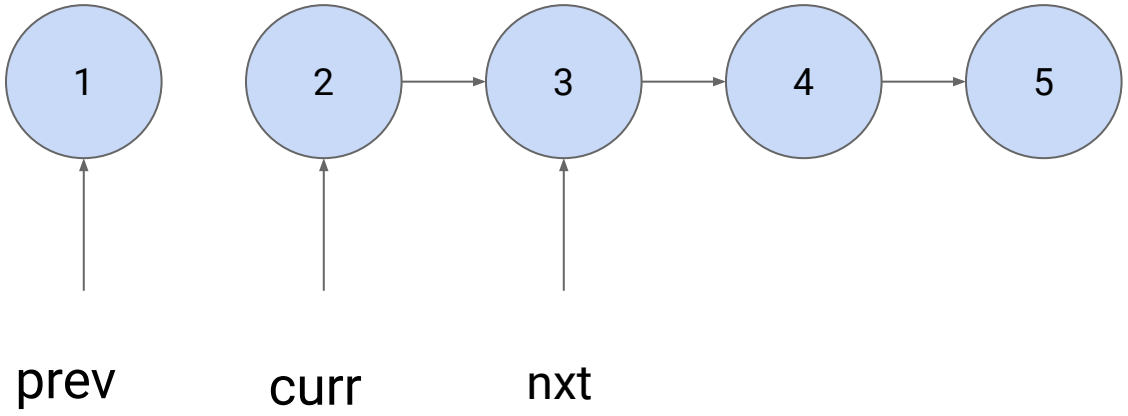
Example Leetcode question: Reverse a Linked List



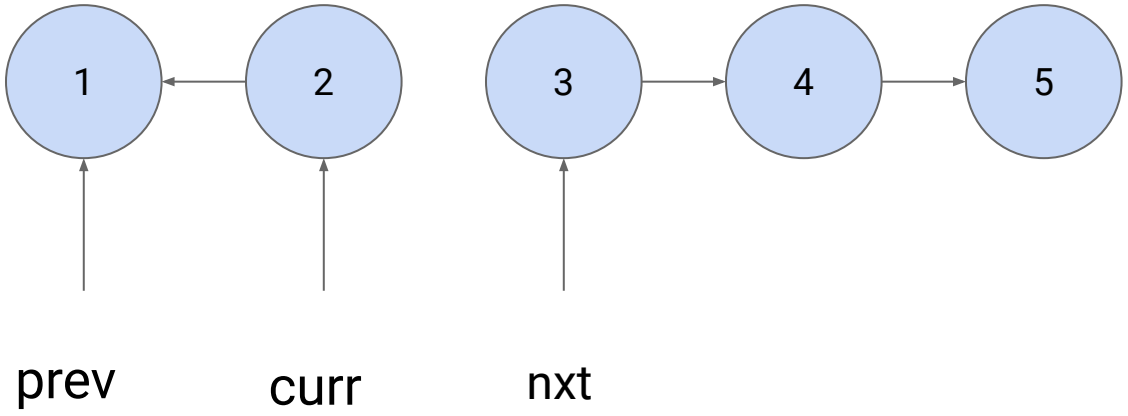
Example Leetcode question: Reverse a Linked List



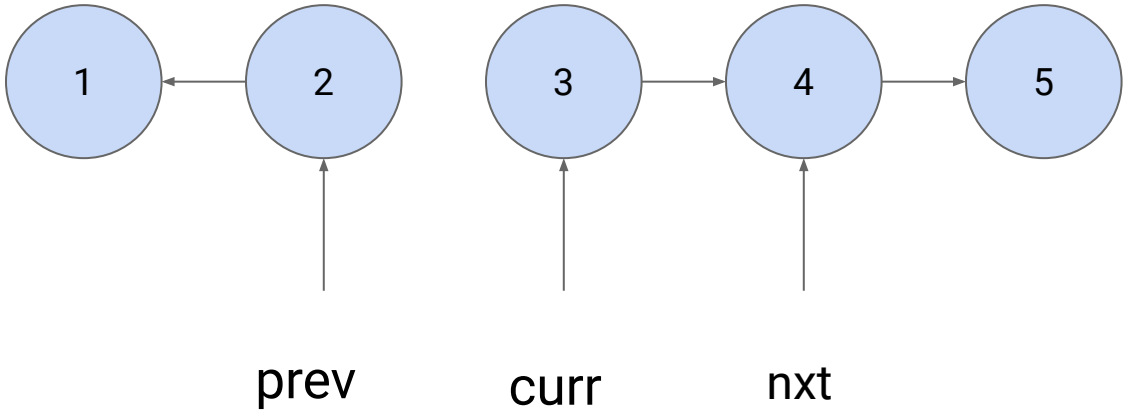
Example Leetcode question: Reverse a Linked List



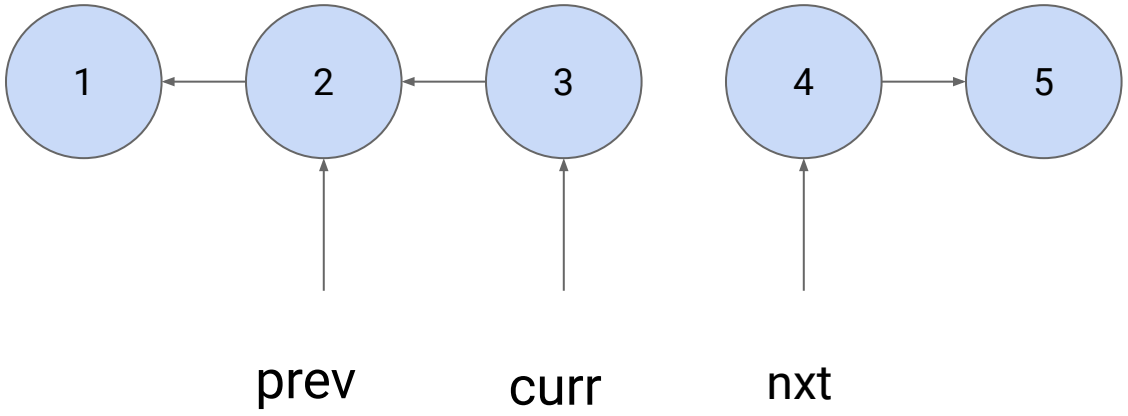
Example Leetcode question: Reverse a Linked List



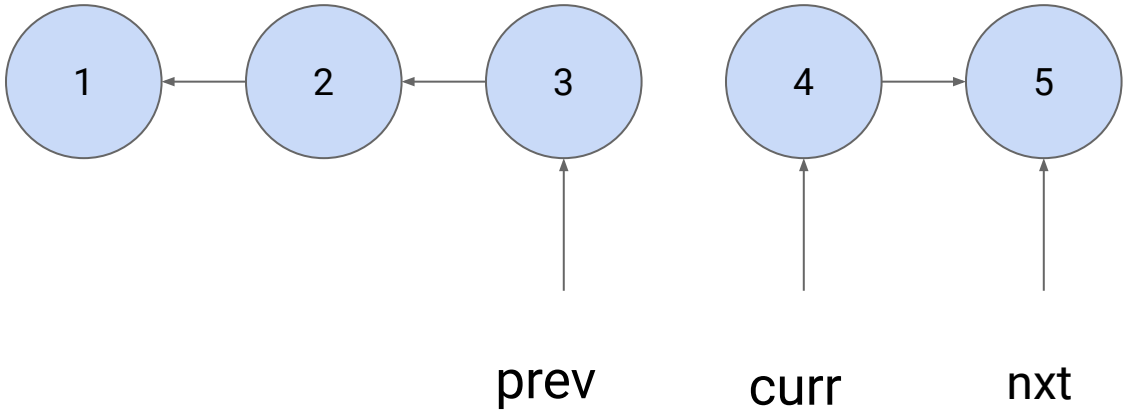
Example Leetcode question: Reverse a Linked List



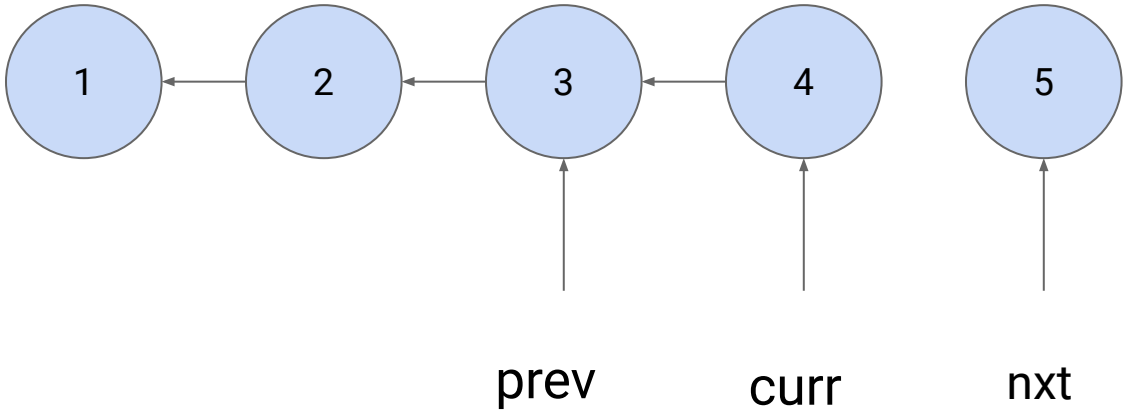
Example Leetcode question: Reverse a Linked List

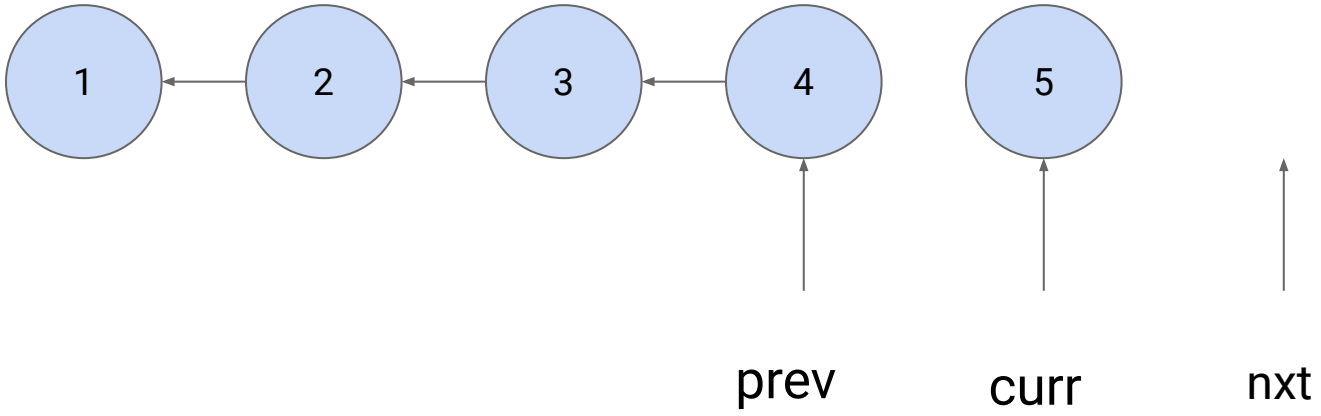


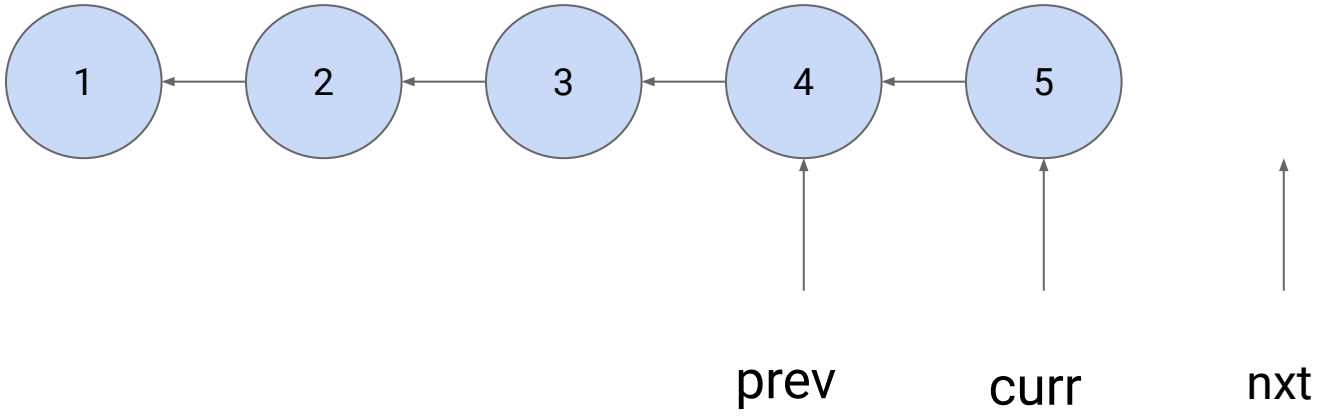
Example Leetcode question: Reverse a Linked List

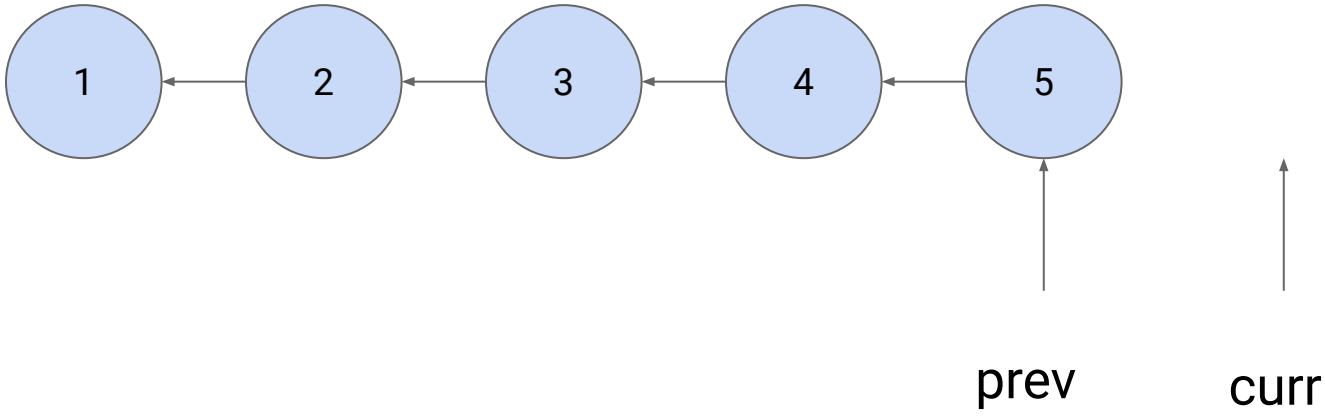


Example Leetcode question: Reverse a Linked List









Example Project Euler Question: Question 81

In the 5 by 5 matrix below, the minimal path sum from the top left to the bottom right, by **only moving to the right and down**, is indicated in bold red and is equal to 2427.

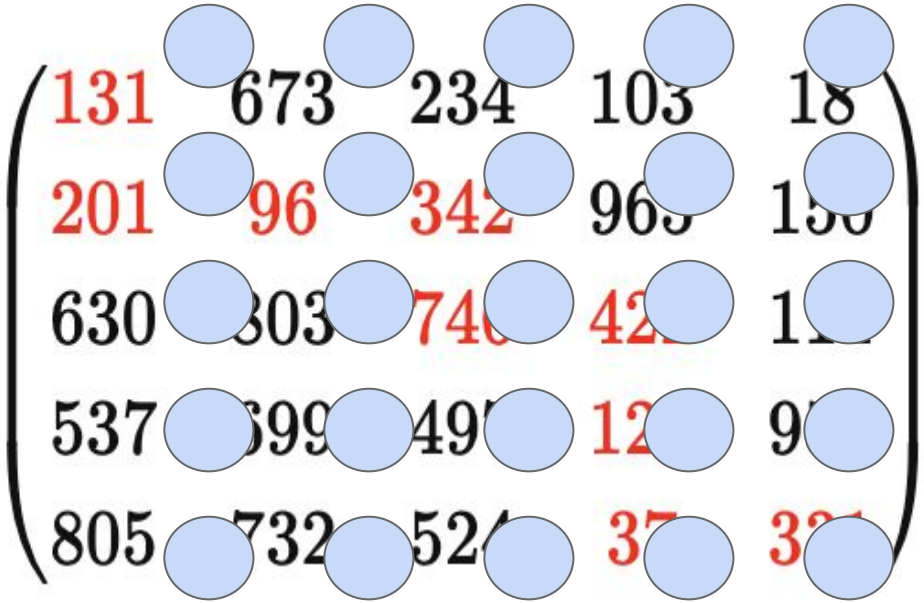
$$\begin{pmatrix} \color{red}{131} & 673 & 234 & 103 & 18 \\ \color{red}{201} & \color{red}{96} & \color{red}{342} & 965 & 150 \\ 630 & 803 & \color{red}{746} & \color{red}{422} & 111 \\ 537 & 699 & 497 & \color{red}{121} & 956 \\ 805 & 732 & 524 & \color{red}{37} & \color{red}{331} \end{pmatrix}$$

Find the minimal path sum from the top left to the bottom right by only moving right and down in **matrix.txt** (right click and "Save Link/Target As..."), a 31K text file containing an 80 by 80 matrix.

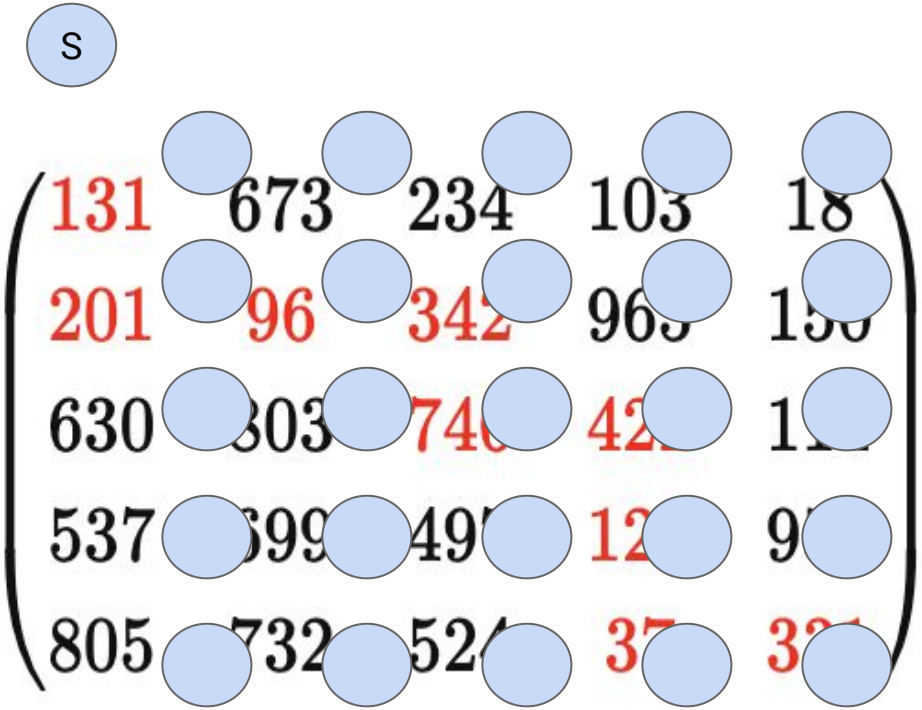
Example Project Euler Question: Question 81

$$\begin{pmatrix} 131 & 673 & 234 & 103 & 18 \\ 201 & 96 & 342 & 965 & 150 \\ 630 & 803 & 746 & 422 & 111 \\ 537 & 699 & 497 & 121 & 956 \\ 805 & 732 & 524 & 37 & 331 \end{pmatrix}$$

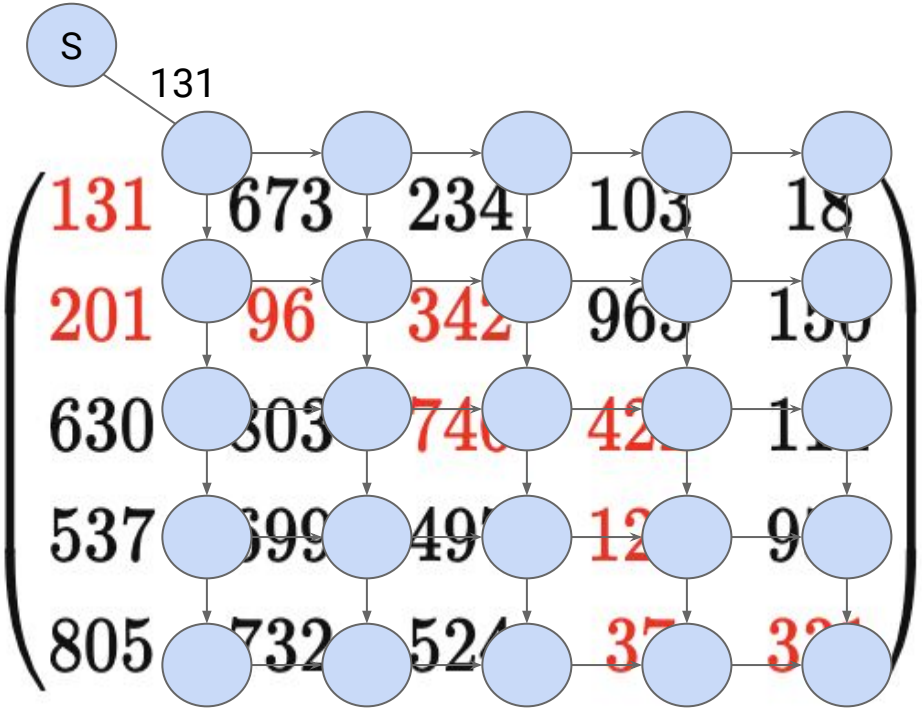
Example Project Euler Question: Question 81



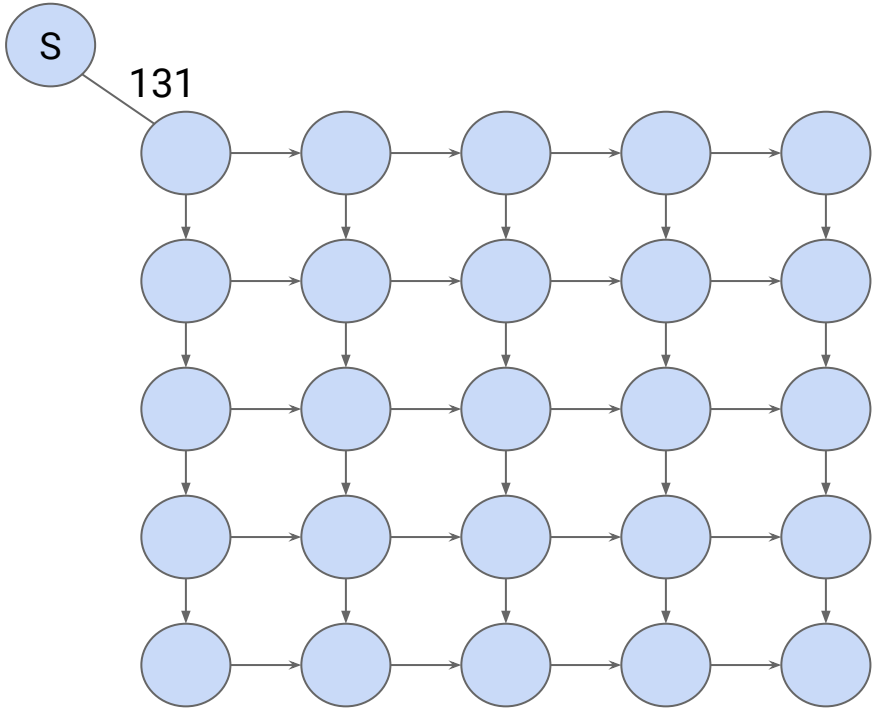
Example Project Euler Question: Question 81



Example Project Euler Question: Question 81



Example Project Euler Question: Question 81



Q&A

Lecture 34, CS61B, Spring 2024

Gitlet

Whiteboarding

SWE Fundamentals

How to Practice

Q&A