Comments on Lab 1
Winston On Presentations
Algorithms
Vertex Cover

REU

4pm in Park 245

Lab 1

- Average: 3.1
  - marching bands

Formatting

- No black backgrounds
- Use serif fonts for text
- Use monospaced fonts for code
  - Block indent
- Code appears exactly as in IDE

- Label clearly in appendix.

Printing

- Use Latex defaults
  - 12 point font
  - Single space
  - 1.5 inch margins on all sides
    - see http://cs.brynmawr.edu/cs337/Lec04/text.pdf
- Stapler?

Content

- Be careful about assumptions
  - Cite it
  - Prove it
  - State "I assume that ..."
  - Leave it out
- Compiler vs interpreter
- Stick to the prompt

Winston on Presentations

Pick

- Time
- The room
  - Shape matters (Park 227, Park 338)
  - A happy place

Practice

- Pick your location
- AV issues
- Lights on
- Chat up early arrivers

The talk

- Be Happy
- VSN-C
  - Start with Vision
  - Steps
  - News
  - Finish with Contributions

Contributions == Conclusions

- No "thank you"
- No collaborators
  - if needed, do early

"you have too many slides and all of them have too many words"

Winston

Do not read

Avoid bullet lists

Use big fonts

(use even bigger fonts)

Progress bars -- maybe

"page 1 or 12"?

Props

Titles

Shakespeare

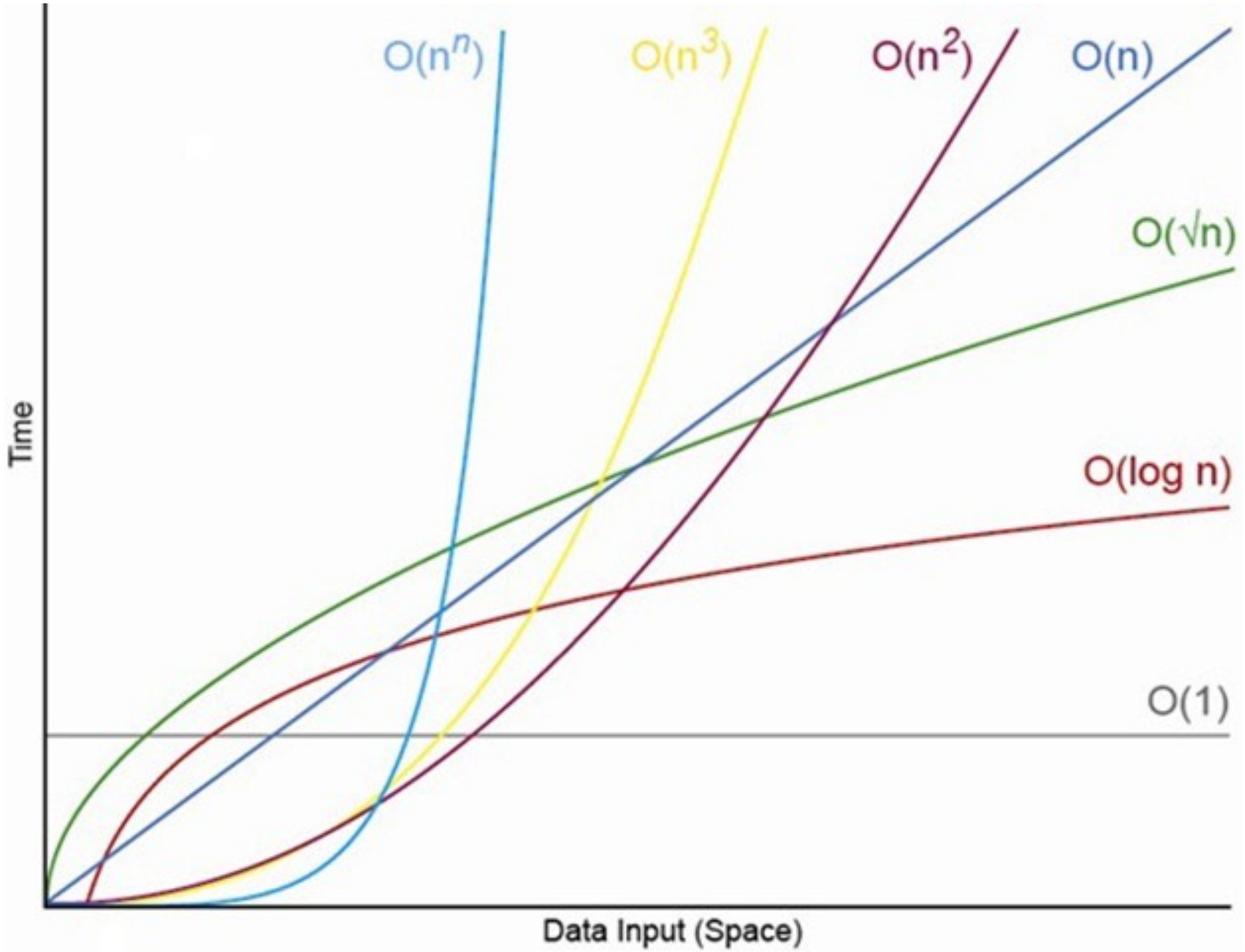
Um
like
er...
you know

Bellow!
(use a mic, practice)

~~Monotone~~

~~Pockets~~

Back to talking about Algorithms



http://apelbaum.wordpress.com/2011/05/

How long to sort 10 million numbers?

Computer A

Speed: 10^{10} instructions/sec
Running $O(n^2)$ sort
Requires $2n^2$ instructions

How long will it take?

Computer B

Speed: 10^7 instructions/sec
Running $O(n \log n)$ sort
Requires $50\, n \log n$ instructions

How long will it take?

How long to sort 10 million numbers?

Computer A

Speed: 10^{10} instructions/sec
Running O(n^2) sort
Requires $2n^2$ instructions

$$\frac{2 * (10^7)^2}{10^{10}} \approx 20{,}000s$$

~5.5 hours

Computer B

Speed: 10^7 instructions/sec
Running O($n \log n$) sort
Requires $50\ n \log n$ instructions

How long will it take?

How long to sort 10 million numbers?

Computer A

Speed: 10^{10} instructions/sec
Running O(n^2) sort
Requires $2n^2$ instructions

$$\frac{2 * (10^7)^2}{10^{10}} \approx 20{,}000s$$

~5.5 hours

Computer B

Speed: 10^7 instructions/sec
Running O($n \log n$) sort
Requires $50\, n \log n$ instructions

$$\frac{50 * 10^7 * log 10^7}{10^7} \approx 1163s$$

under 20 minutes!

How long to sort 10 million numbers?

Computer A

Speed: 10^{10} instructions/sec
Running O(n^2) sort
Requires $2n^2$ instructions

$$\frac{2 * (10^7)^2}{10^{10}} \approx 20{,}000s$$

If running $50\ n \log n$ program: < 2s!!

Computer B

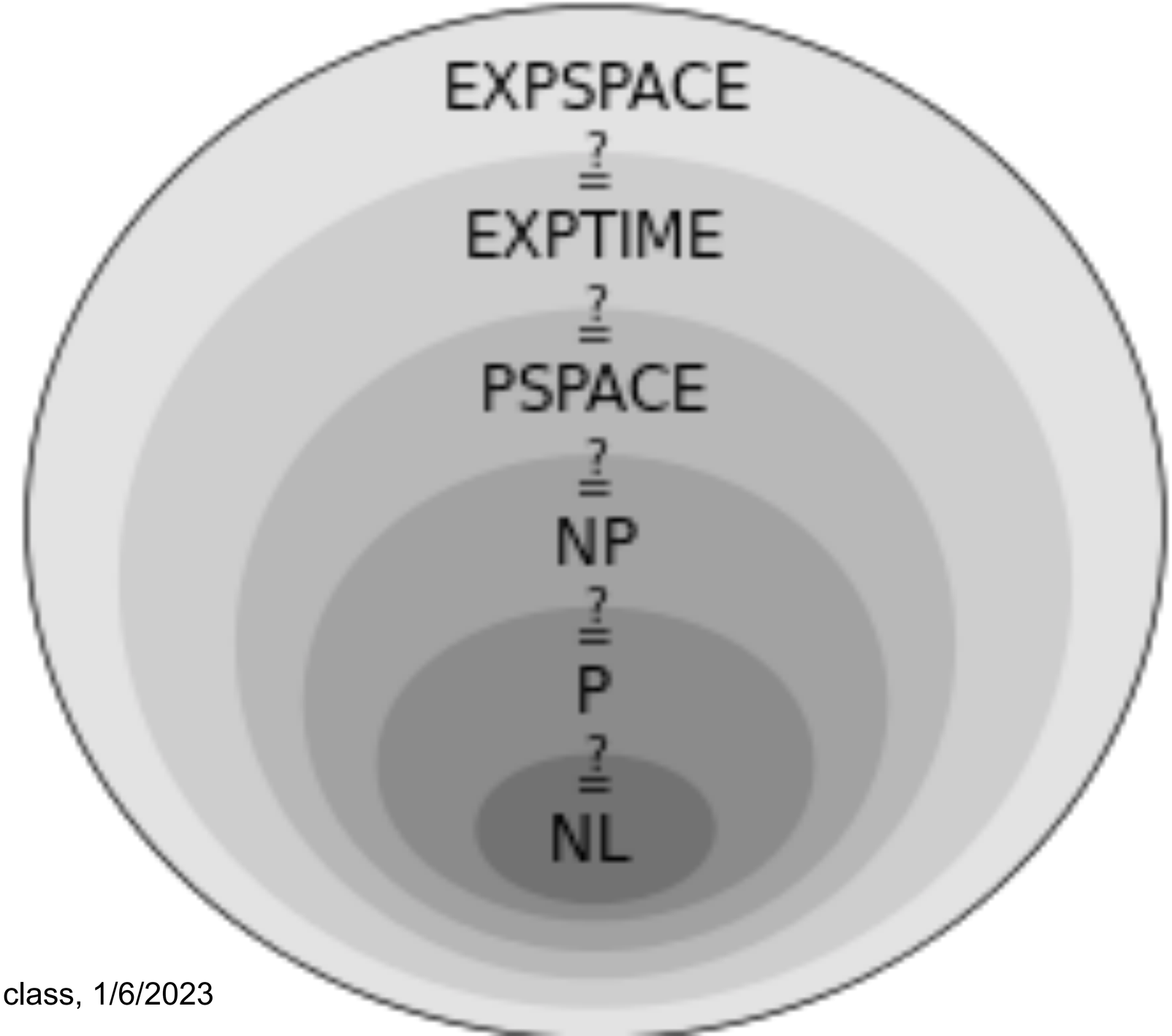
Speed: 10^7 instructions/sec
Running O($n \log n$) sort
Requires $50\ n \log n$ instructions

$$\frac{50 * 10^7 * log 10^7}{10^7} \approx 1163s$$

under 20 minutes!

The nines of reliability

Pct Reliable	number of nines	time out per year
90%	1	36.5 days
99%	2	3.65 days
99.9	3	8.7 hours
99.99	4	52 minutes
99.999	5	31 seconds
99.9999	6 (six sigma)	3 seconds

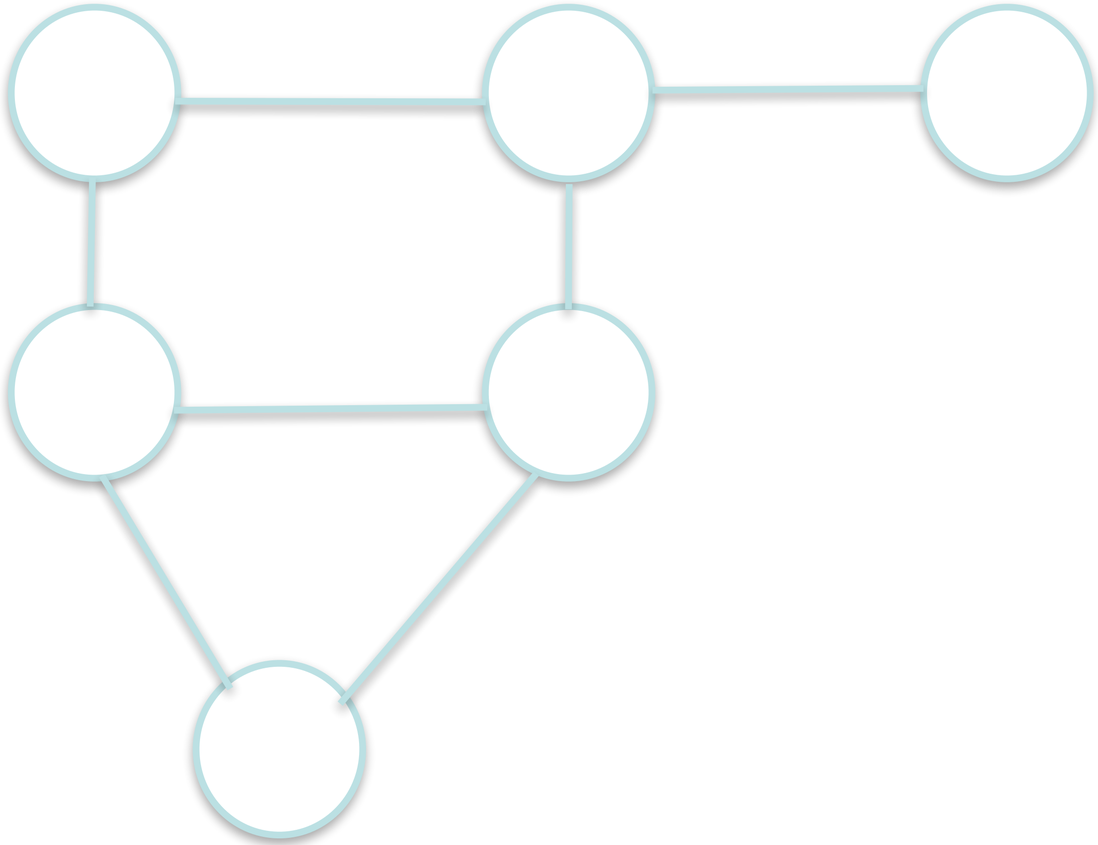


wikipedia, complexity class, 1/6/2023

NP-Complete

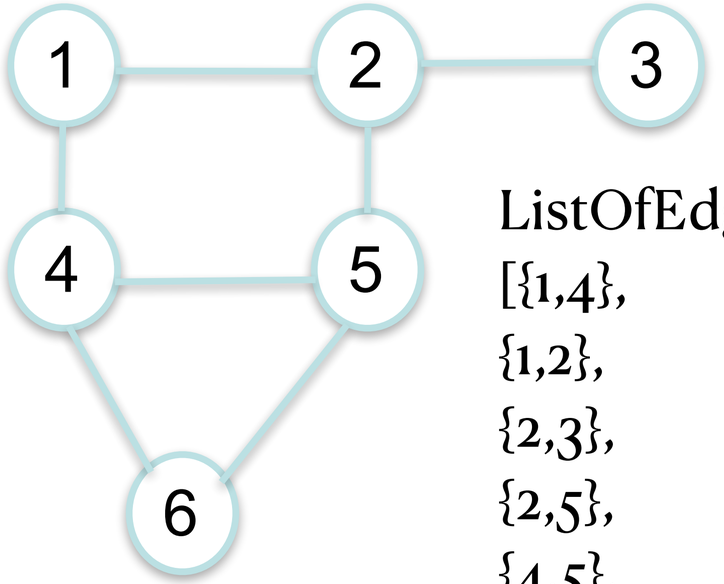
- NP = Non-determinitic Polynoimial
- in NP == Solution is verifiable in P time
- problem is provably equivalent to other NP complete problems

vertex cover of a graph is a set of vertices that includes at least one endpoint of every edge.



Vertex Cover Algorithm

- Find the minimum vertex cover of a graph
  - Given a list of edges
  - Return a list of nodes in a minimal (or close to minimal) vertex cover

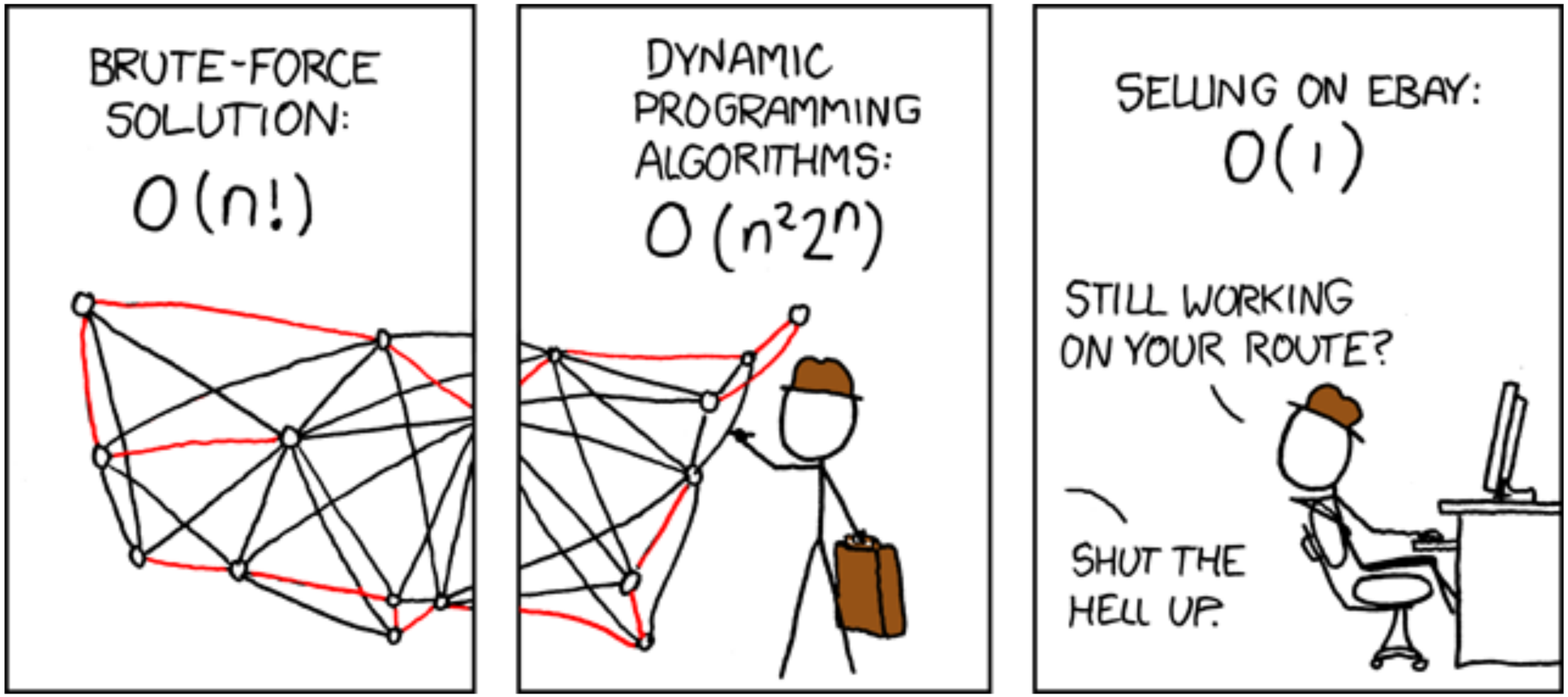


ListOfEdges
[{1,4},
{1,2},
{2,3},
{2,5},
{4.5},
{4,6},
{5,6}
]

Return
[1,2,4,6]

There are other solutions

xkcd??



- More on xkcd.com

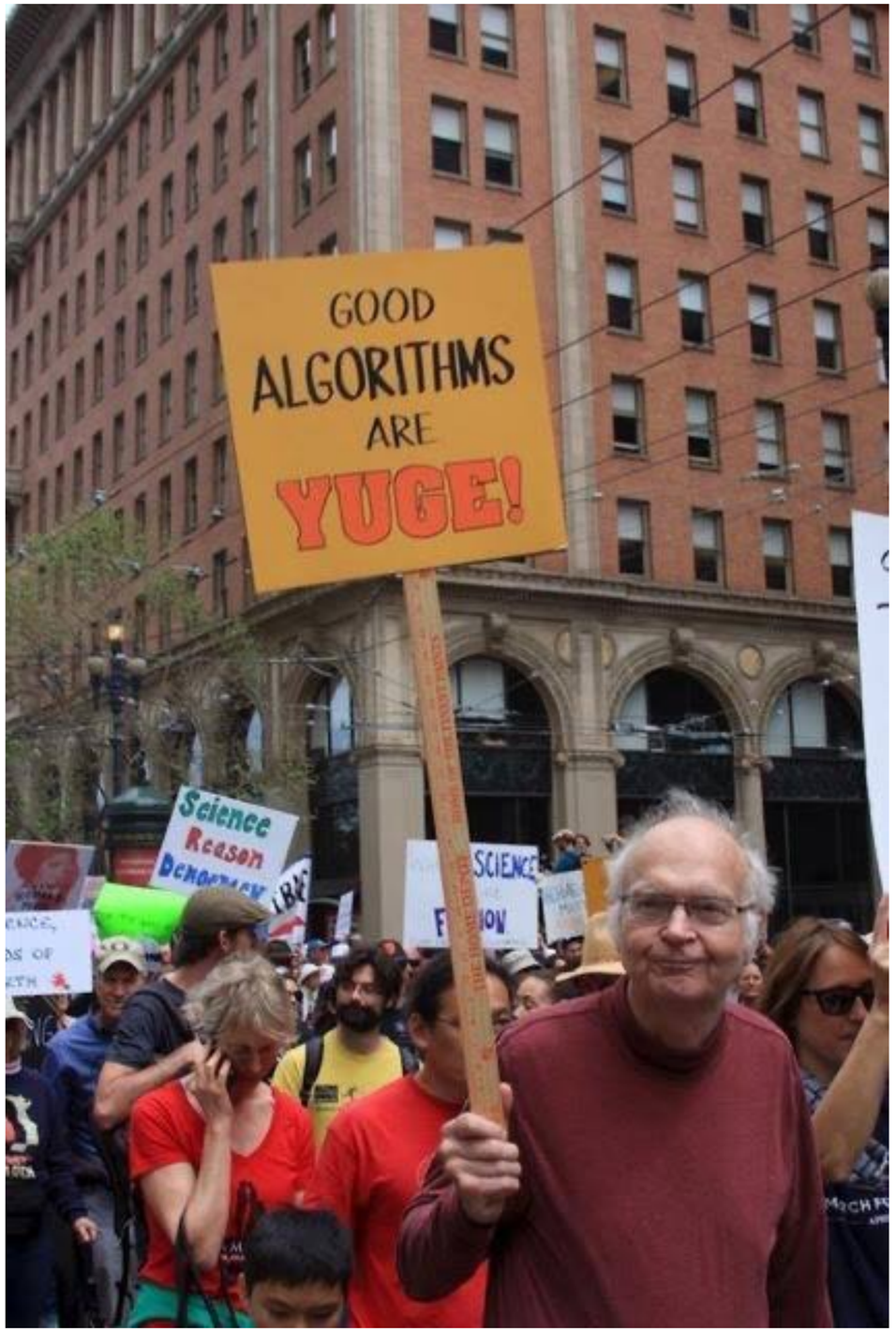
Algorithm for Algorithm Development

```
def algorithmDevelopment(problemSpec):
    correct = false
    while not correct or not fastEnough(runningTime):
        algorithm = deviseAlgorithm(problemSpec)
        correct = analyzeCorrectness(algorithm)
        runningTime = analyzeEfficiency(algorithm)
return algorithm
```

Algorithm for Program Development

```
def programDevelopment(algorithm, testSuite):
    language = pickLanguage(algorithm)
    program = code(algorithm, program)
    do:
        check = false
        while not check:
            program = debug(program)
            check = verifyProgram(program, testSuite)

        performance = measure(performance)
    while not acceptable(performance)
```


GOOD
ALGORITHMS
ARE
YUGE!
Science
Reason