csci 3250: Computational Geometry

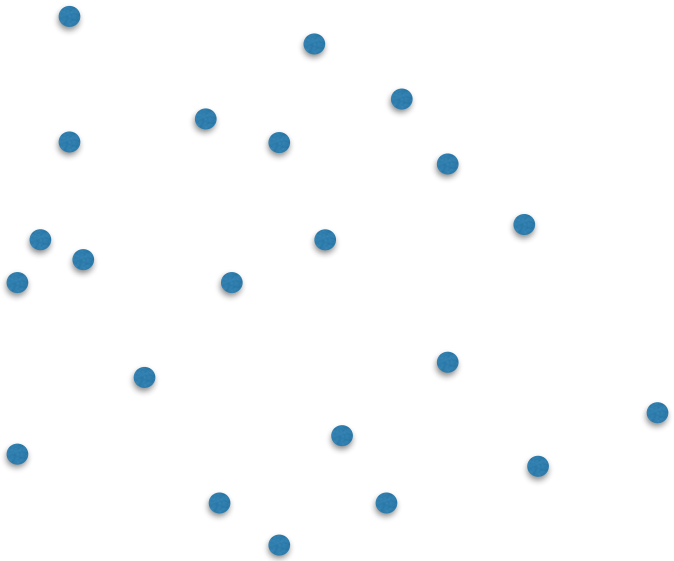
Fall 2024

Laura Toma

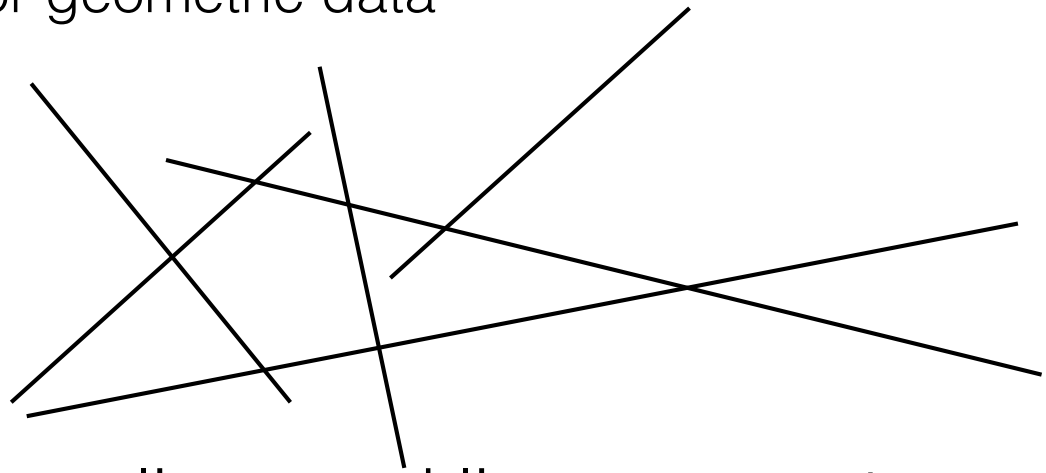
Bowdoin College

What is Computational Geometry?

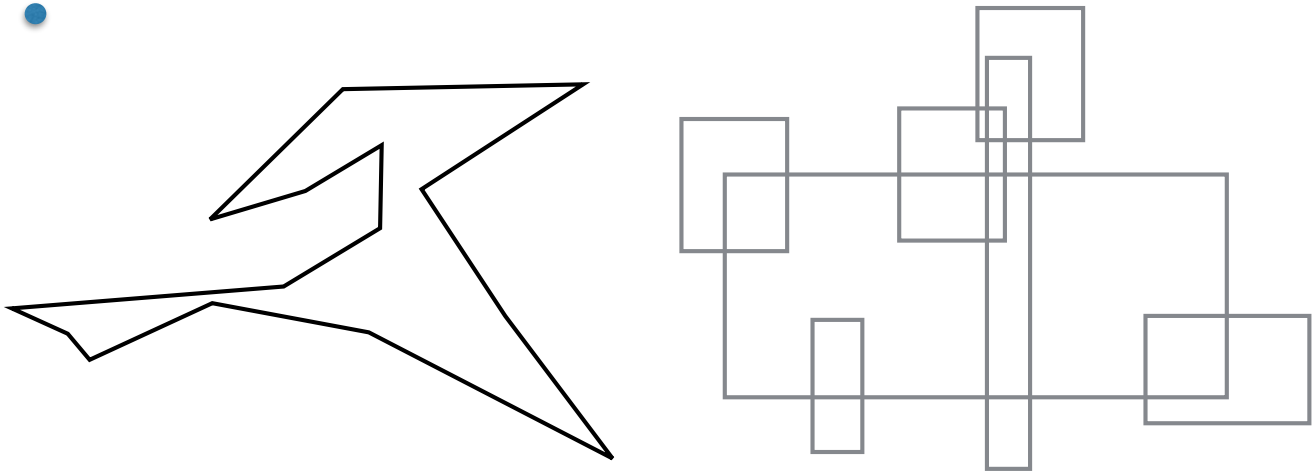
- CG deals with algorithms for geometric data



points



lines and line segments

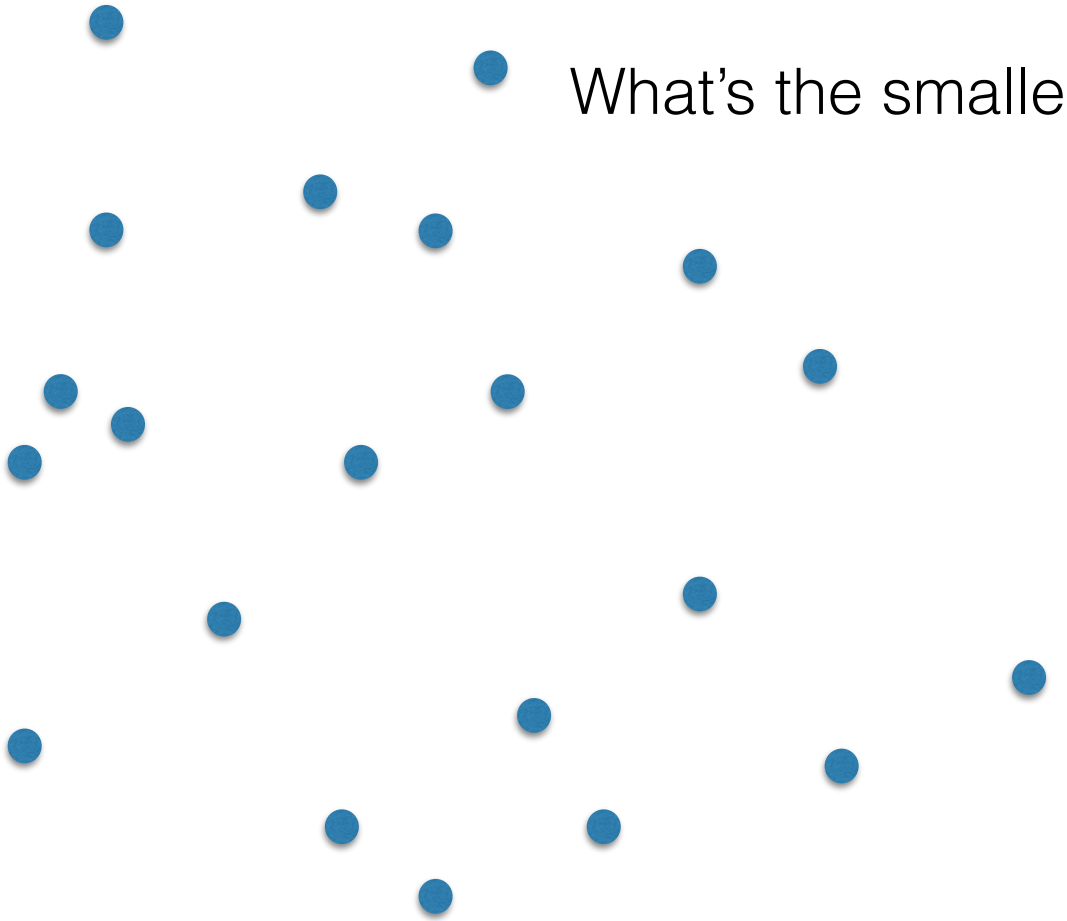


polygons

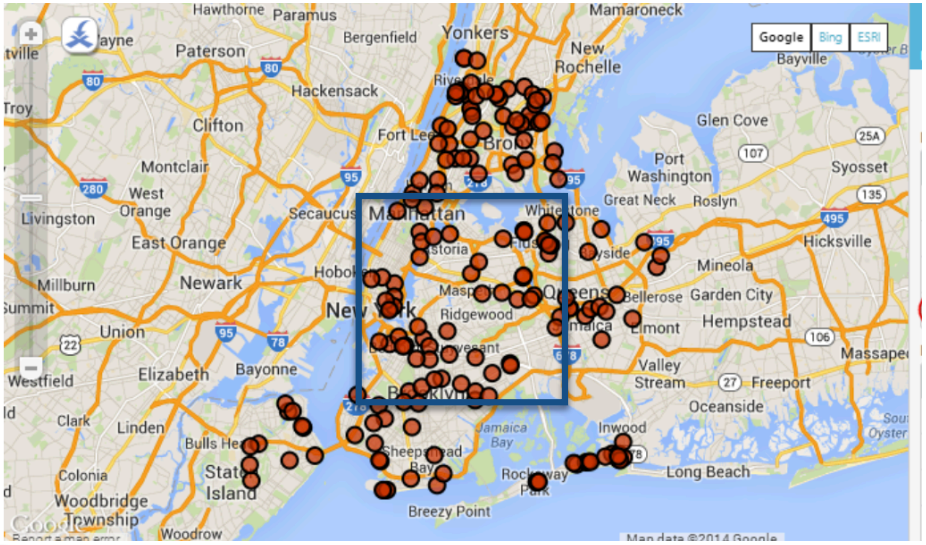
What's the closest pair of points?

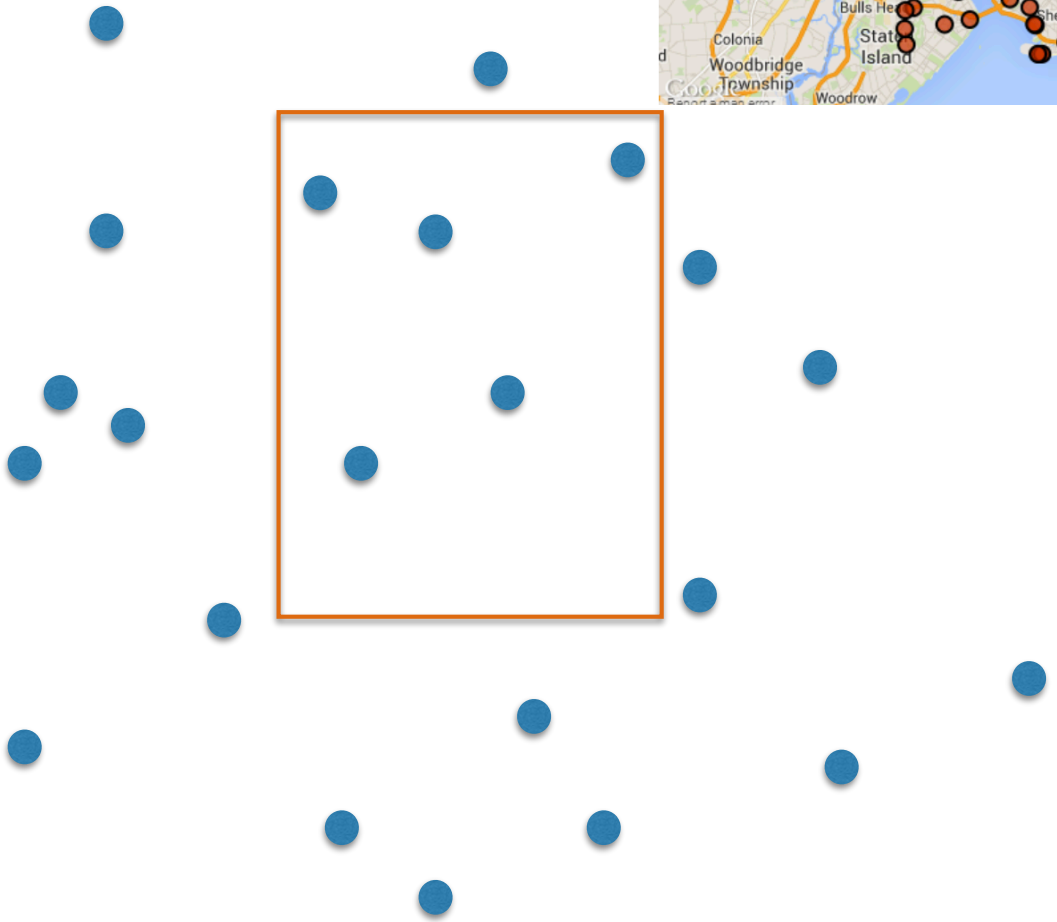
What's the diameter of this set?

What's the smallest enclosing disk?



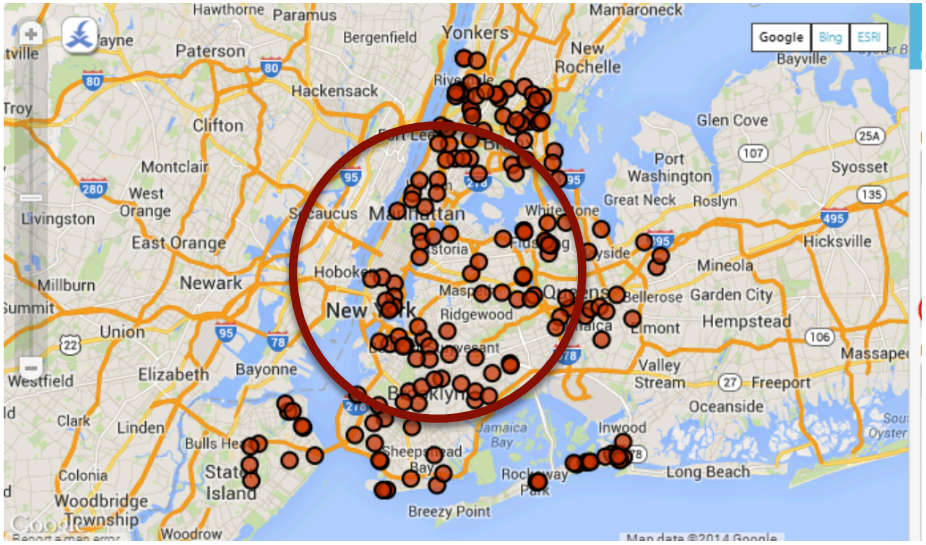
What points fall in this range?

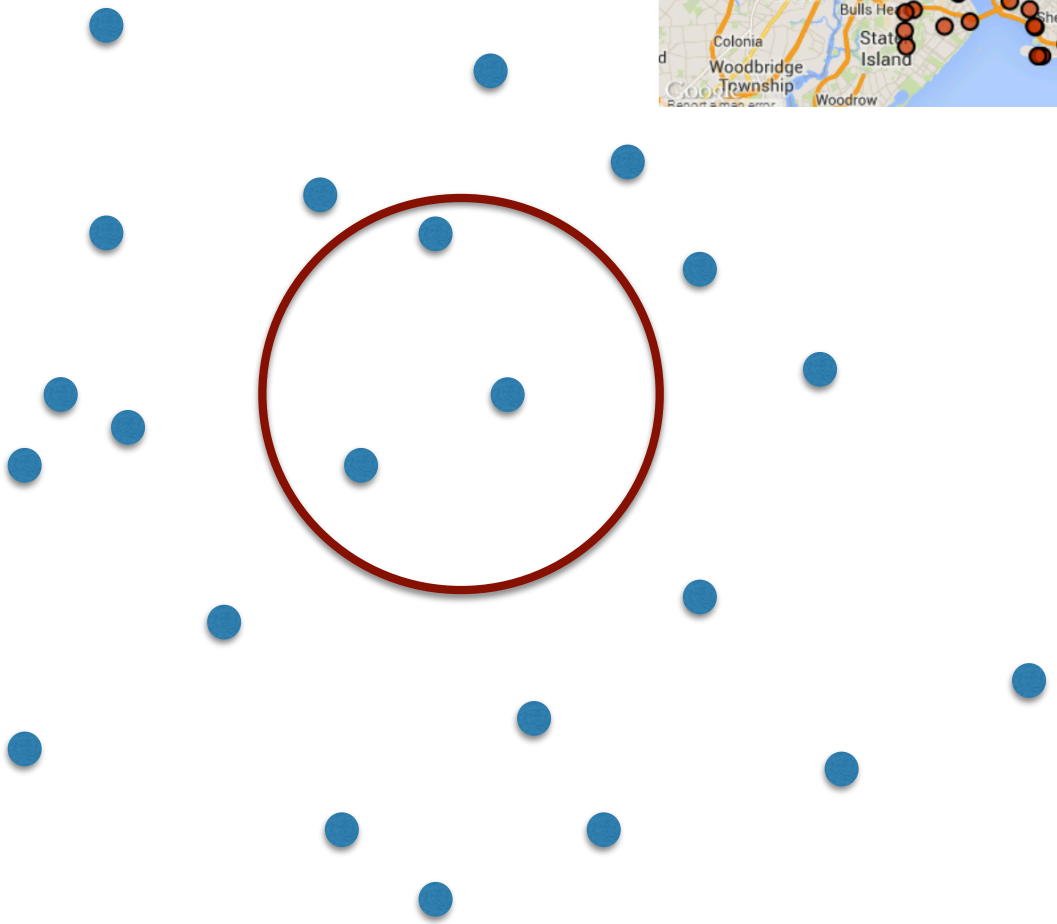




geometric search

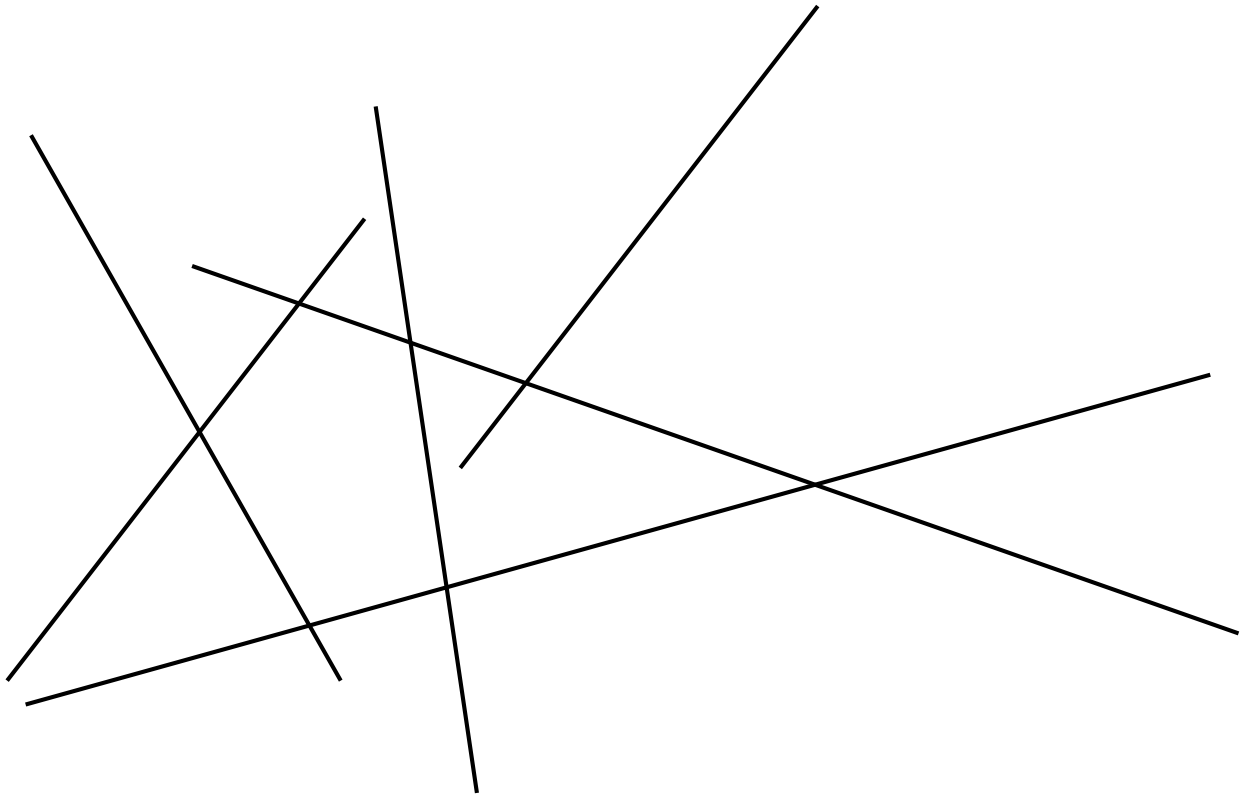
What points fall in this range?



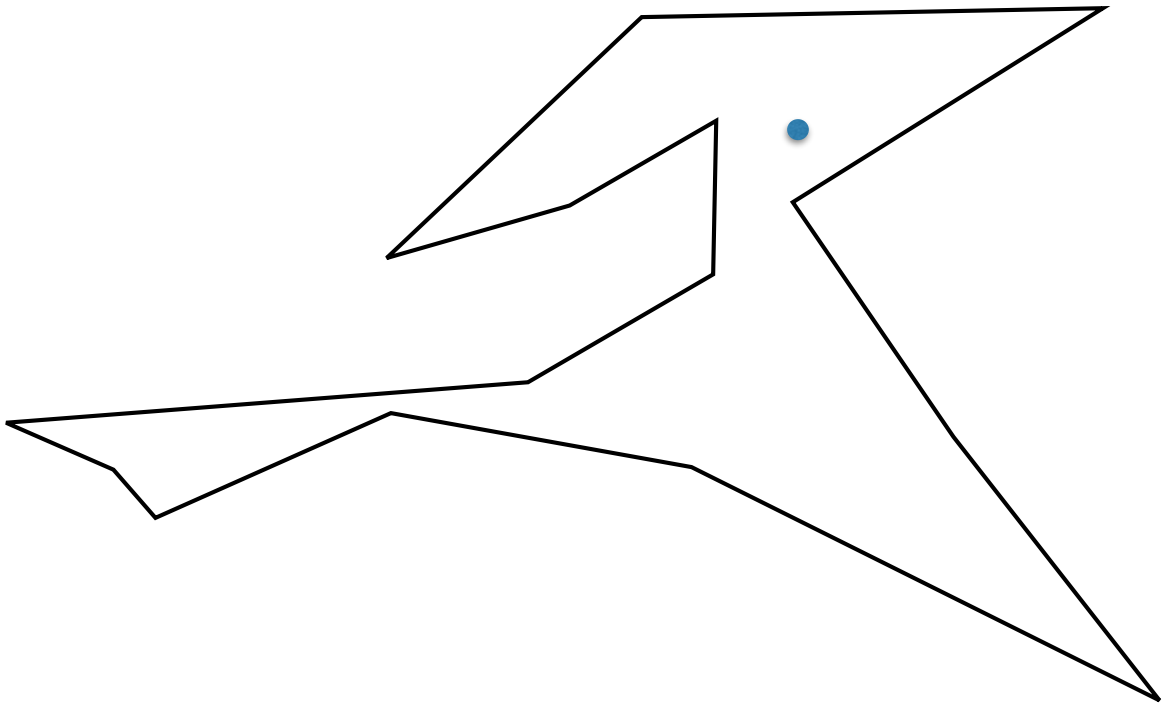


geometric search

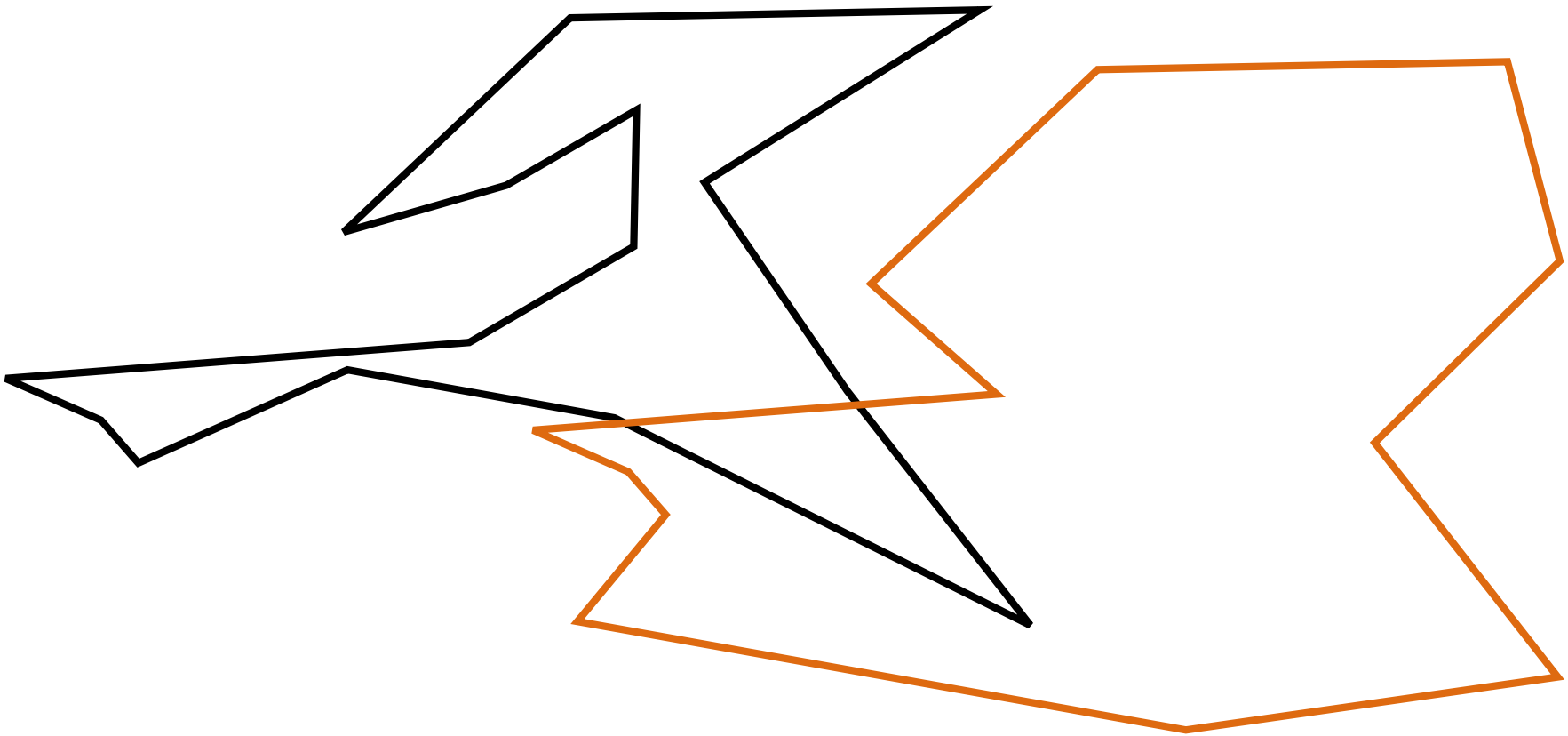
What are the intersections ?



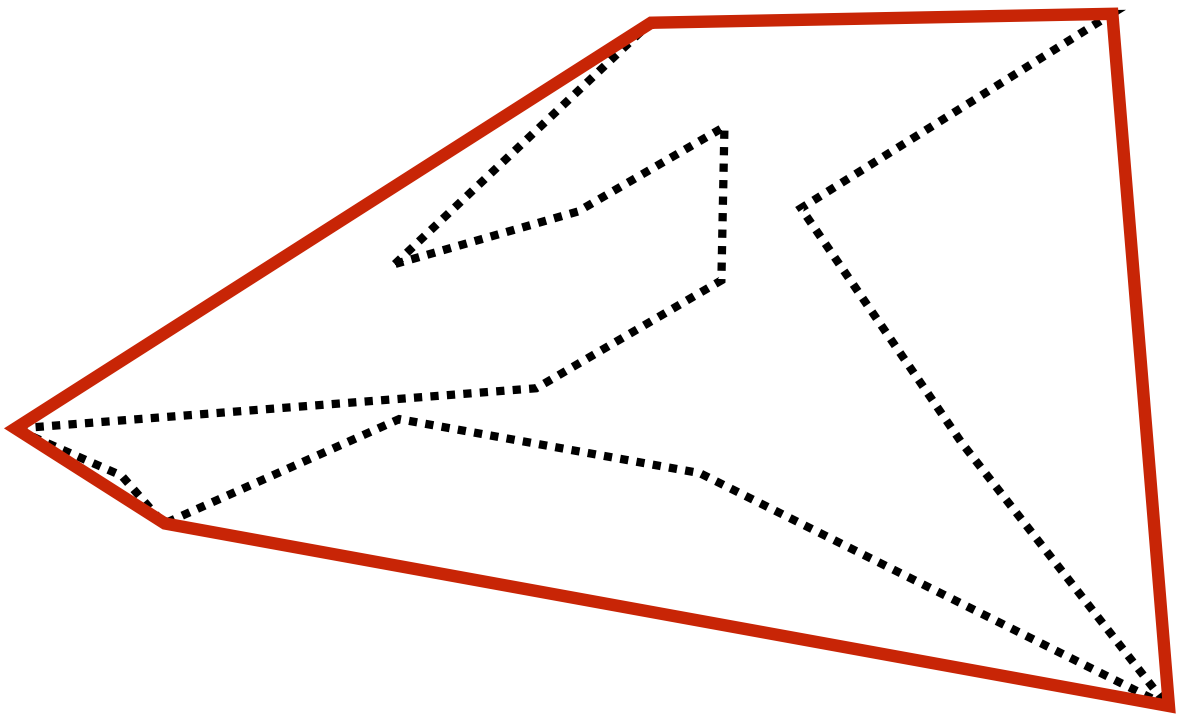
Is a point inside a polygon?



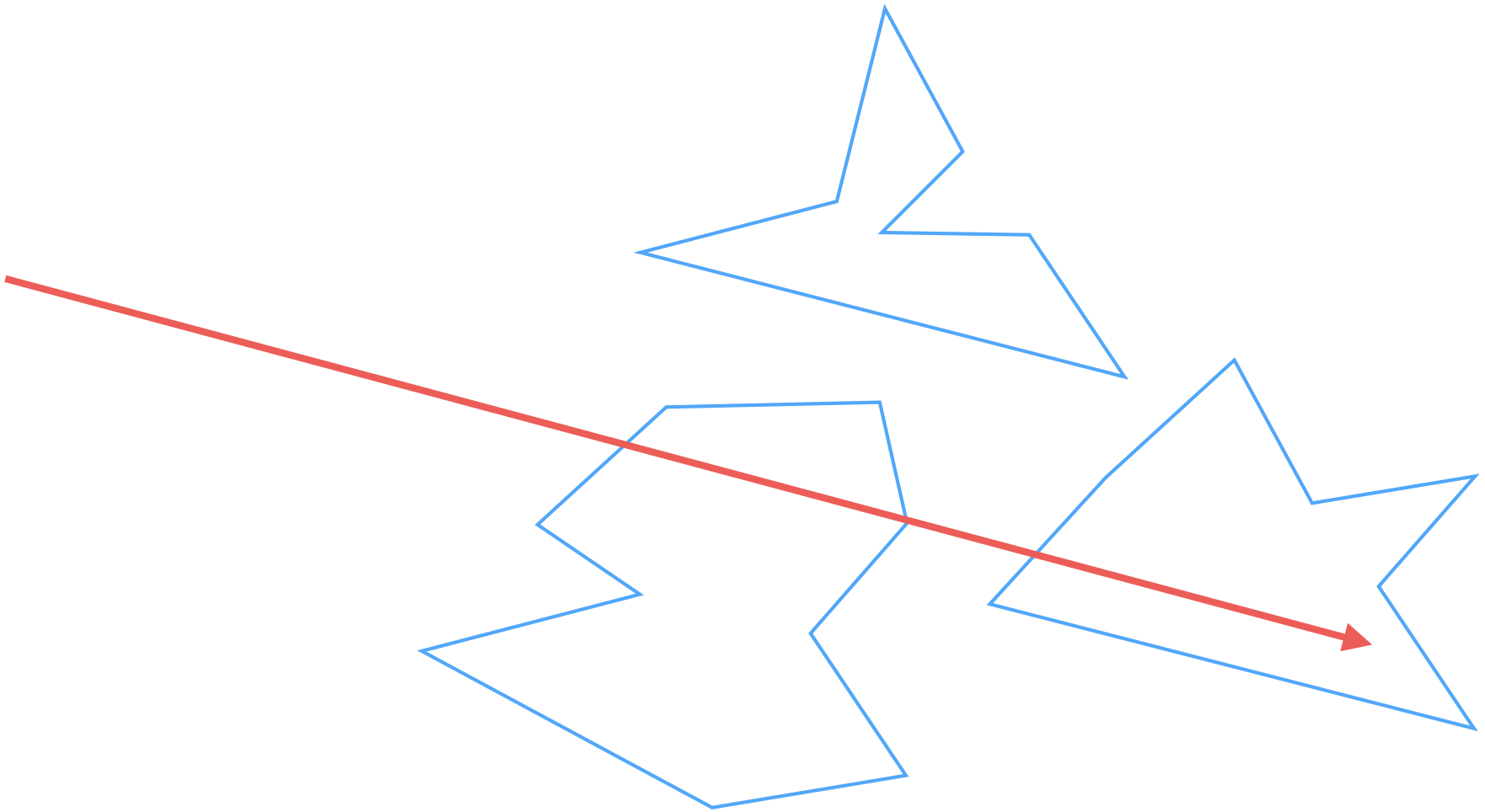
Do two polygons intersect?



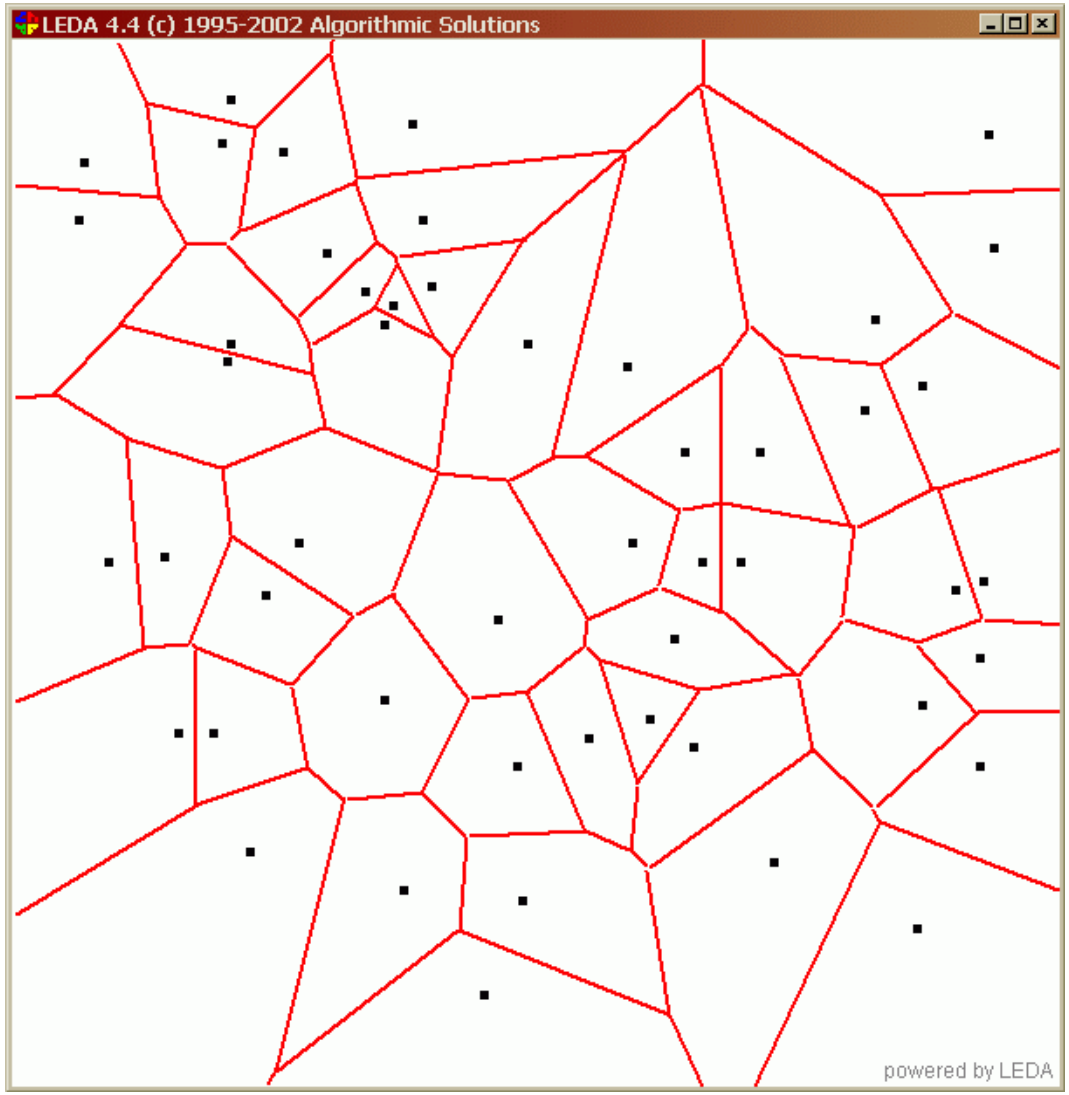
What's the smallest enclosing convex polygon?



What's the first polygon intersected by this ray?

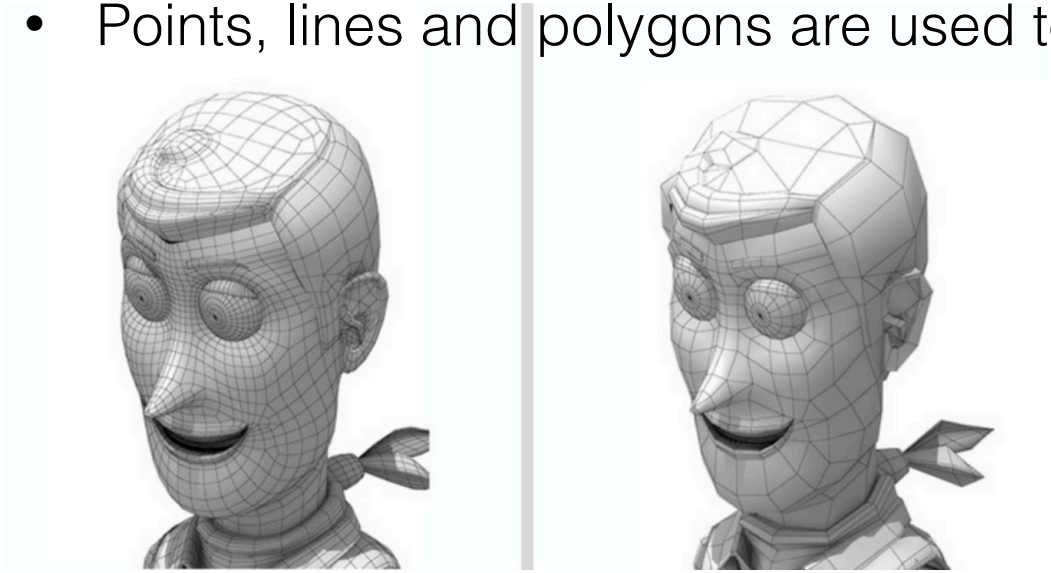


Voronoi diagrams

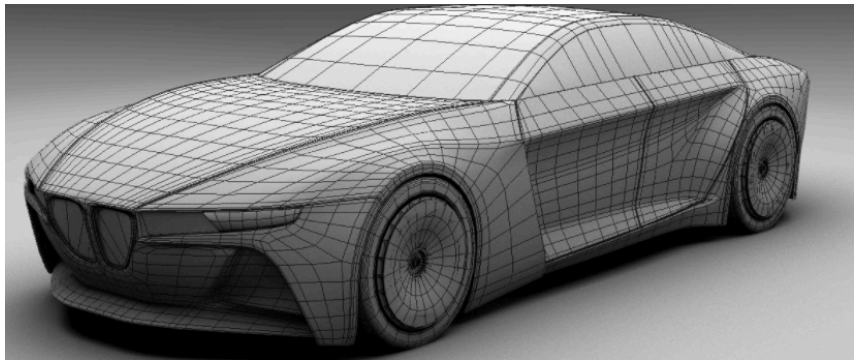


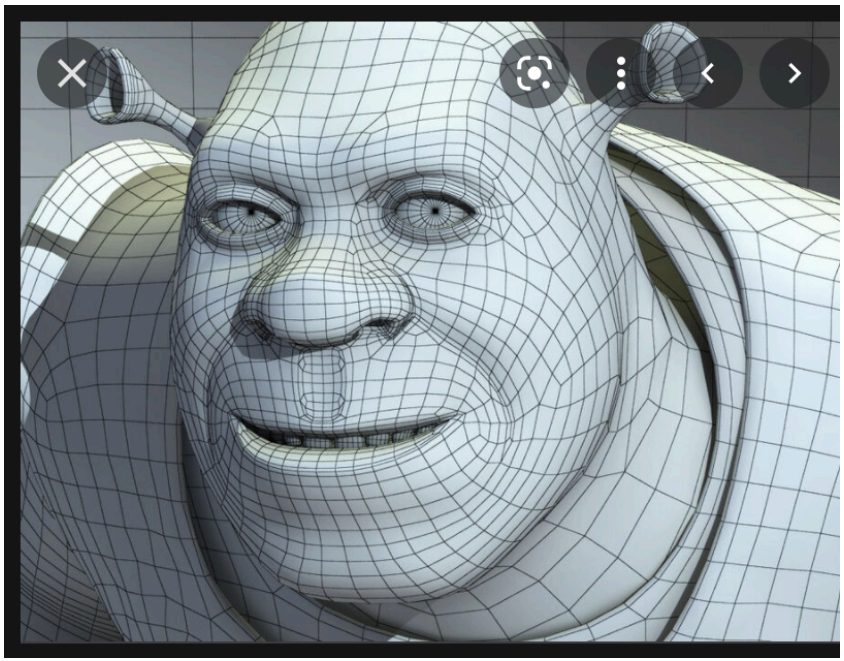
Driven by applications in Graphics

- Points, lines and polygons are used to model complex shapes



(Woody, from Pixar's Toy Story series, as a high and low polygon mesh. Image from fabelar.)





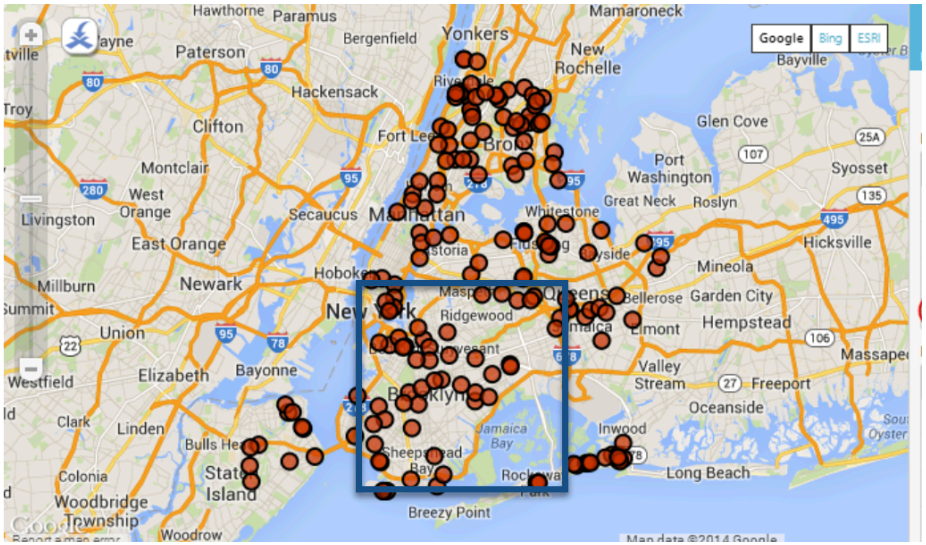
Driven by applications in Graphics

- Rendering, hidden surface removal, lighting, moving, collision detection, ….. involve geometric algorithms.

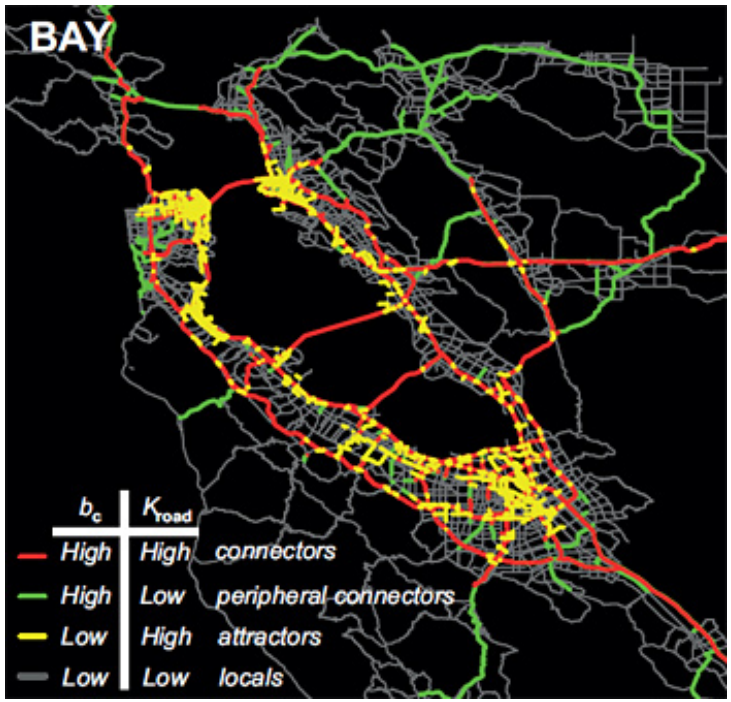
Autonomous vehicles, robotics and motion planning

- find a collision-free path
- collision detection involves finding intersections

- Database engines
  - store data and its geometry
  - contain specialized data structures for answering searches
  - e.g.: find all restaurants in a range

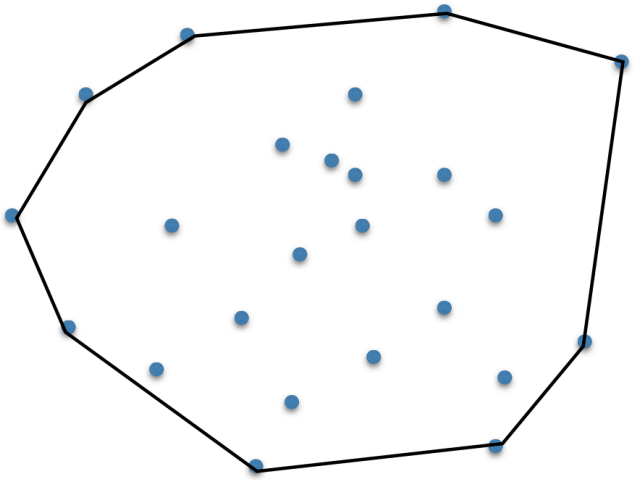


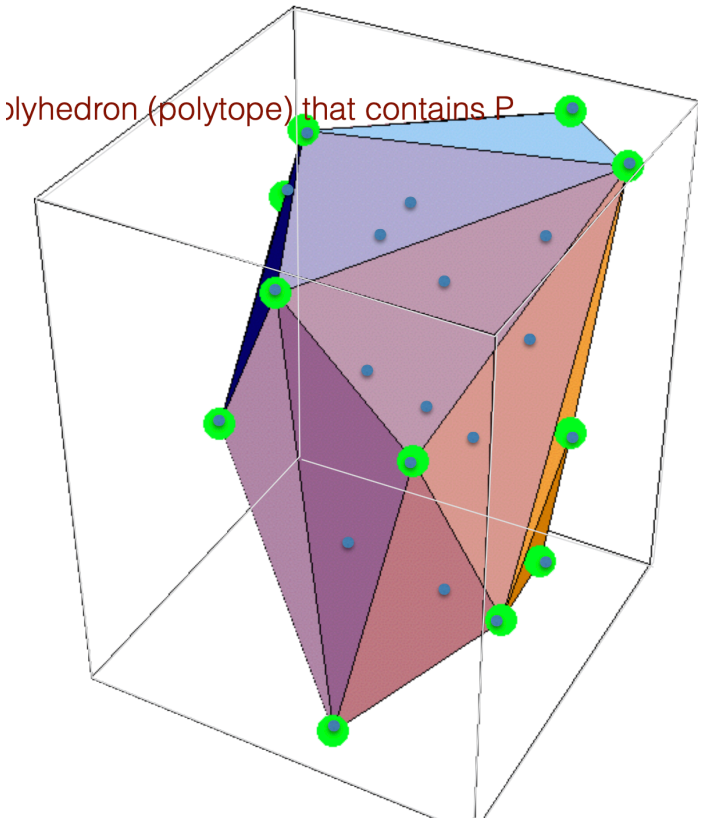
- Traffic analysis based on cell phone data
  - Use location data
  - Model real-time traffic conditions, find congestion patterns



Syllabus overview

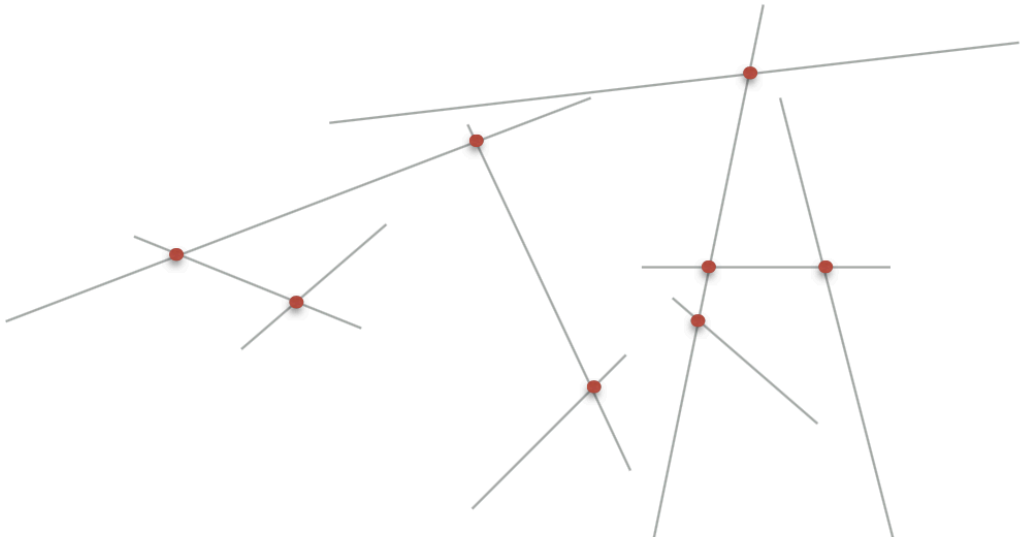
- Introduction and setup
- Geometric primitives
- 2D Convex hull
- 3D convex hull



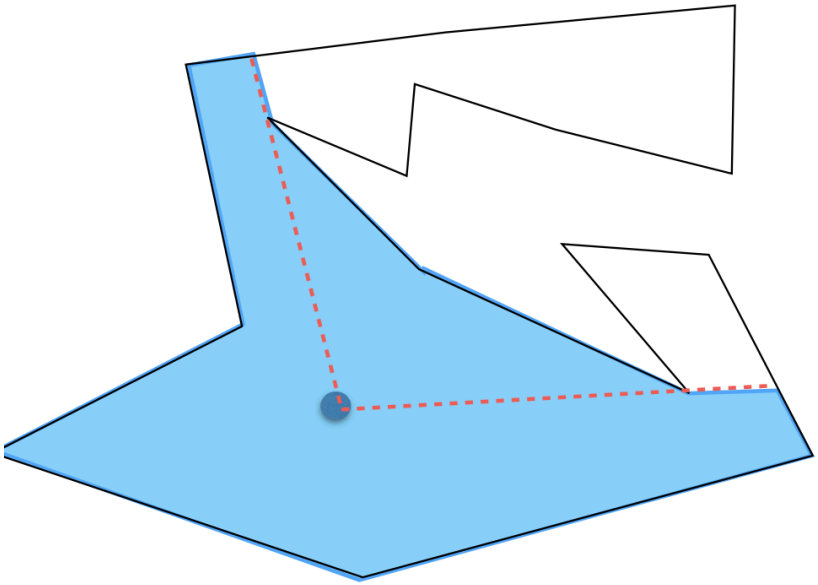


Syllabus overview

- Segment intersection

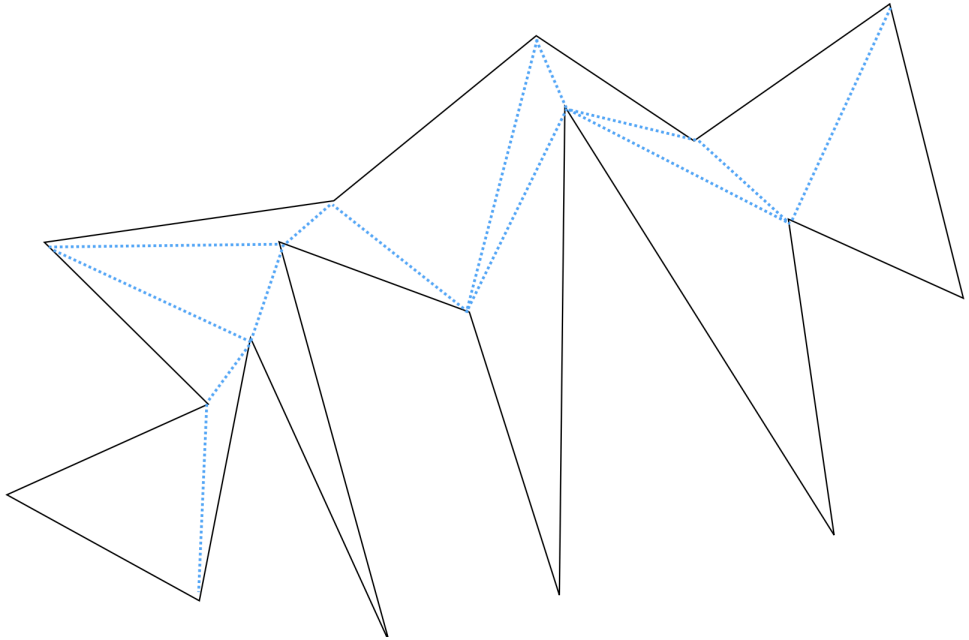


- The art gallery problem

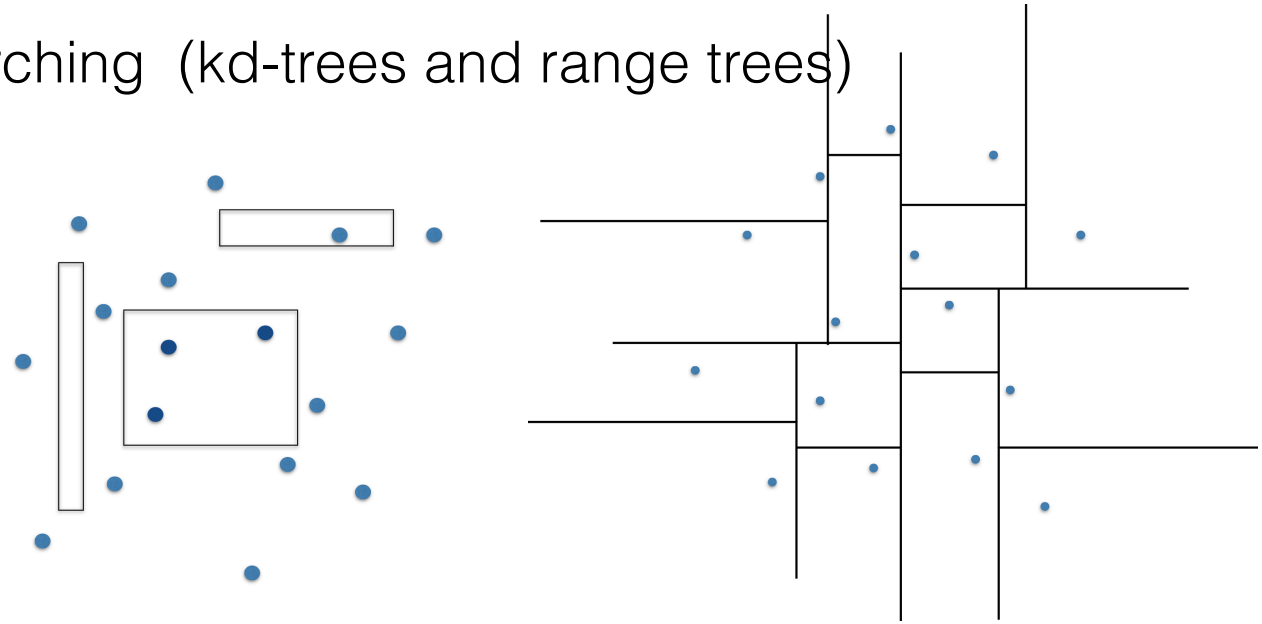


Syllabus overview

- Polygon triangulation

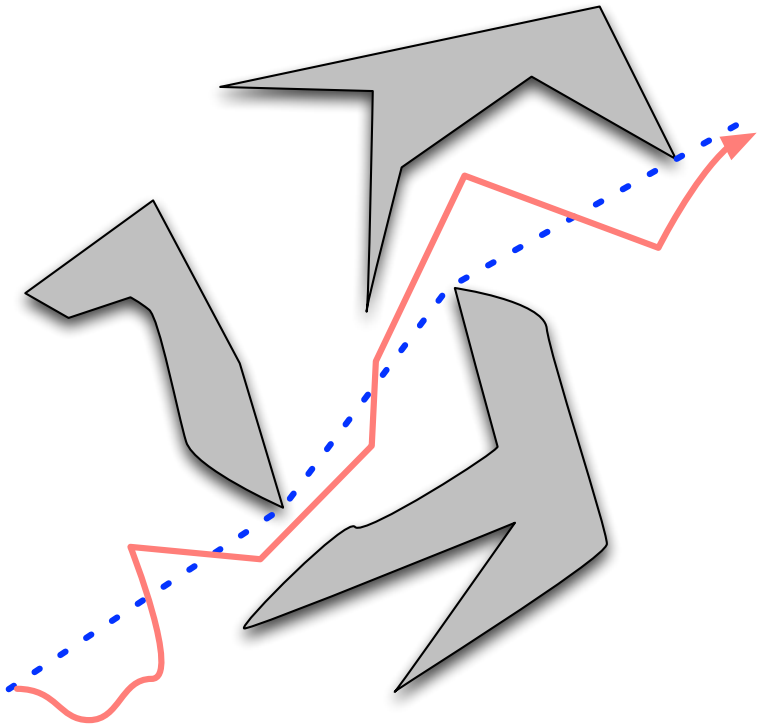


- Orthogonal range searching (kd-trees and range trees)



Syllabus overview

- Path planning : find collision-free path from start to end



Syllabus overview

- We'll explore algorithms for these problems and ask the usual questions
  - Complexity of the result?
  - Worst-case running time?
  - Is the algorithm practical?
  - Handle degeneracies in the input?
  - Can we make some practical assumptions about the data in order to get a simplified algorithm?
  - Lower bound for the problem?
  - Can we do better?

How will this class work?

https://bowdoin-compgeom-f24.github.io/

- Will not use Canvas
- Class syllabus and resources maintained on a public github site
- Slack: for all communication and resource sharing

Projects

- Programming is not a science, it's a learnt craft
    - We all grow as programmers through practice. No exceptions.
    - Start wherever you are and move forward
- For starters, expect to spend most of your time debugging your code
    - You'll start writing code expecting you'll debug
    - This will change the way you approach coding
    - The pain of debugging will teach you (eventually!) to develop your code structured, well documented => simple, elegant, easy to understand => high quality ==> easy to debug

Learning goals

YOU need to get here

The faster the better!

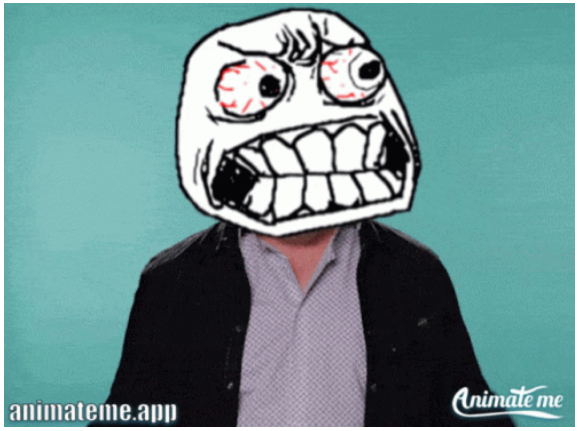
Growing as a programmer: the cycle we all go through

I'll just do it quick and dirty for now..

Comments are for wimps

Noone will read this code but me so why does it matter? code is code. I'll make it pretty later

Why bother writing another function? i'll do that later



DEBUGGING

I can't debug my code! Help!

I don't even know where the error starts!

The screen is blank!

It ran just fine on that different input!

small functions

develop incrementally

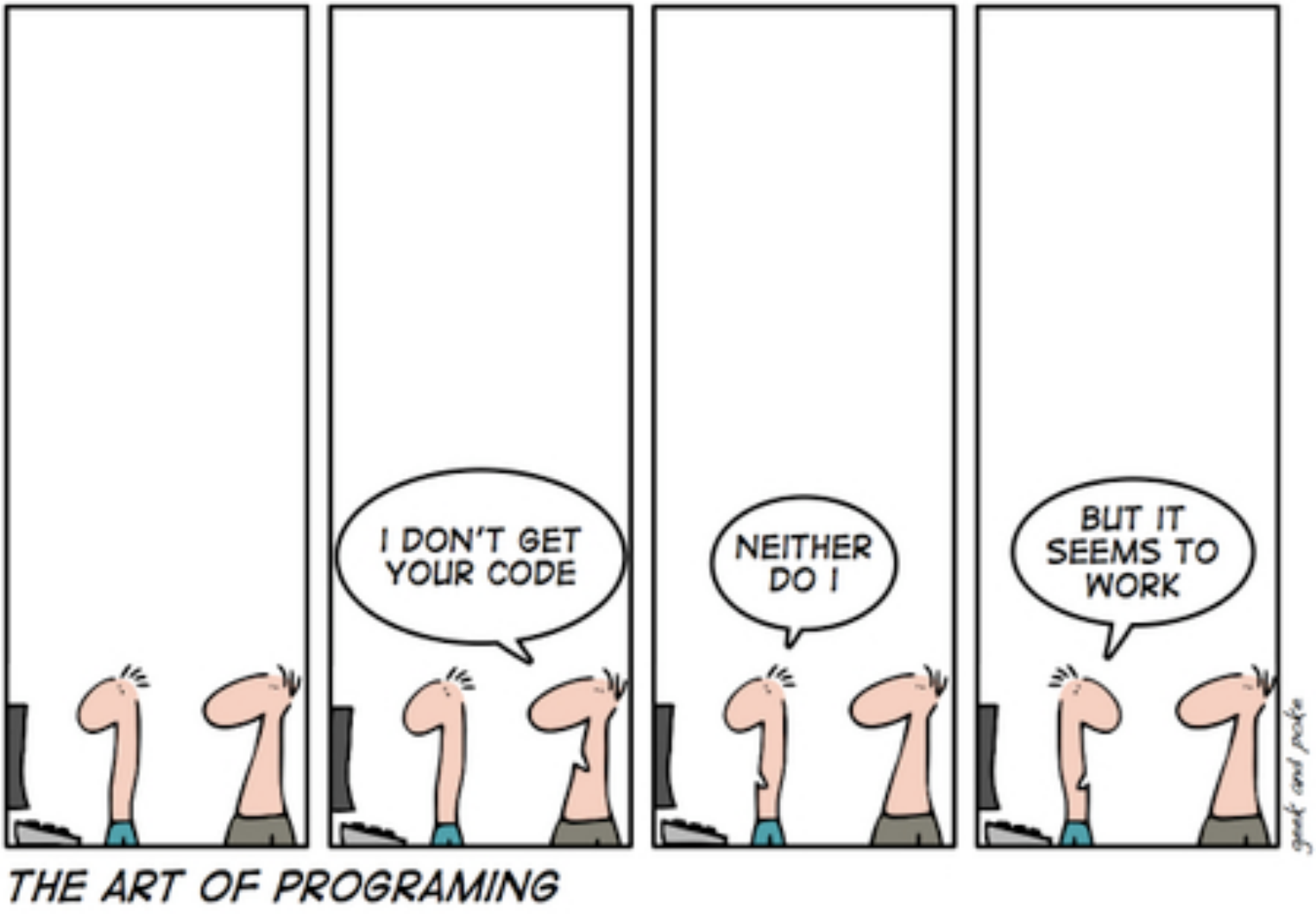
good comments

test early and often

YOU need to debug YOUR code.

Good quality coding practices

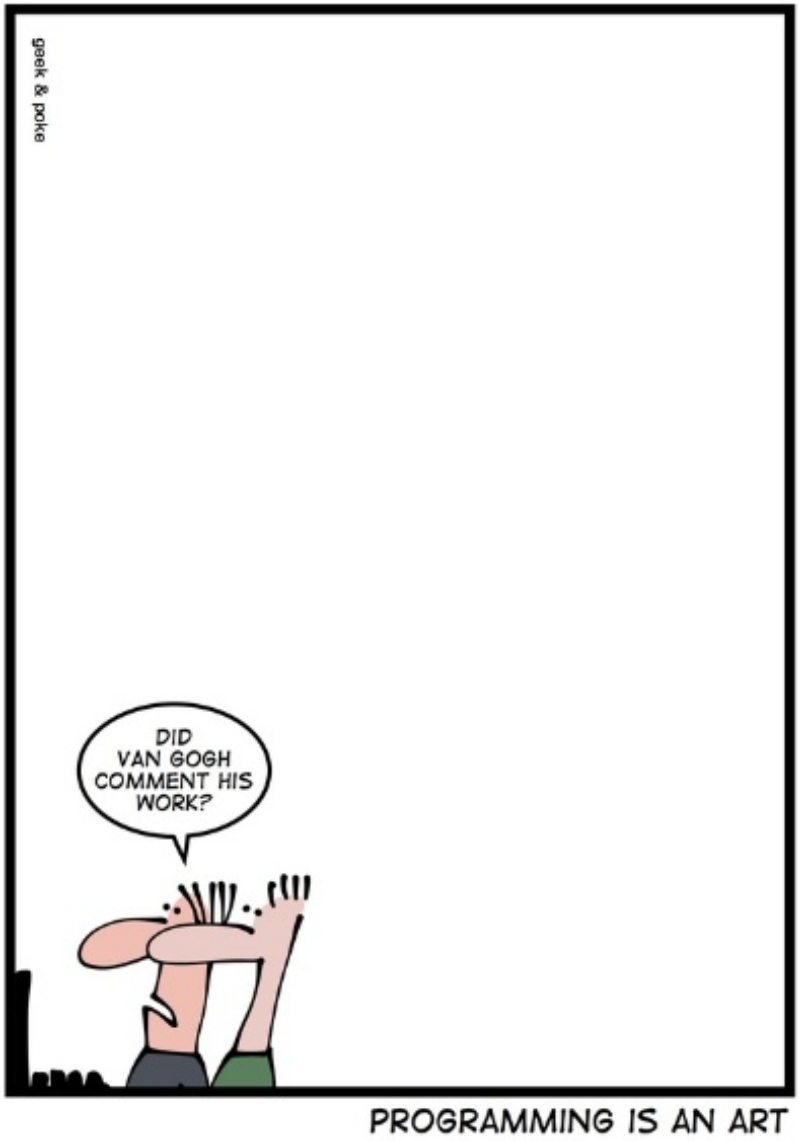
- Break the functionality in separate blocks/functions
- Develop your code incrementally, one function at a time
- Use meaningful names
  - Give functions and variables meaningful names
  - Bad names may seem short and easy, but they make code harder to understand, debug and maintain
- Testing
  - Add testing and test cases.
  - Test early and often. Make sure one function passes the testing before you move to the next one
  - Don’t write everything and then start the testing



CREDIT: https://www.castsoftware.com/images/imported-images/poor-software-quality.jpg

Good quality coding practices

- Write **comments**
  - you will forget what the code does
  - other people read your code
  - debugging bad code is time consuming and frustrating
  - comments help the reader understand the code
  - Bad comments: e.g. repeat what the code does
  - Good comments: e.g. summarize what each function does and its parameters. Summarize blocks of code. State logical invariants.



LAs

Kaylie Harlin(she/her)

New York
loves skiing and coffee

Danielle Simon(she/her)

Chicago
Bowdoin running club, figure skating

Office hours

- LAs
    - tbd
- Laura:
    - Tue 1-3pm
    - Wed 3-5pm (algorithms students have priority)