Computational Geometry

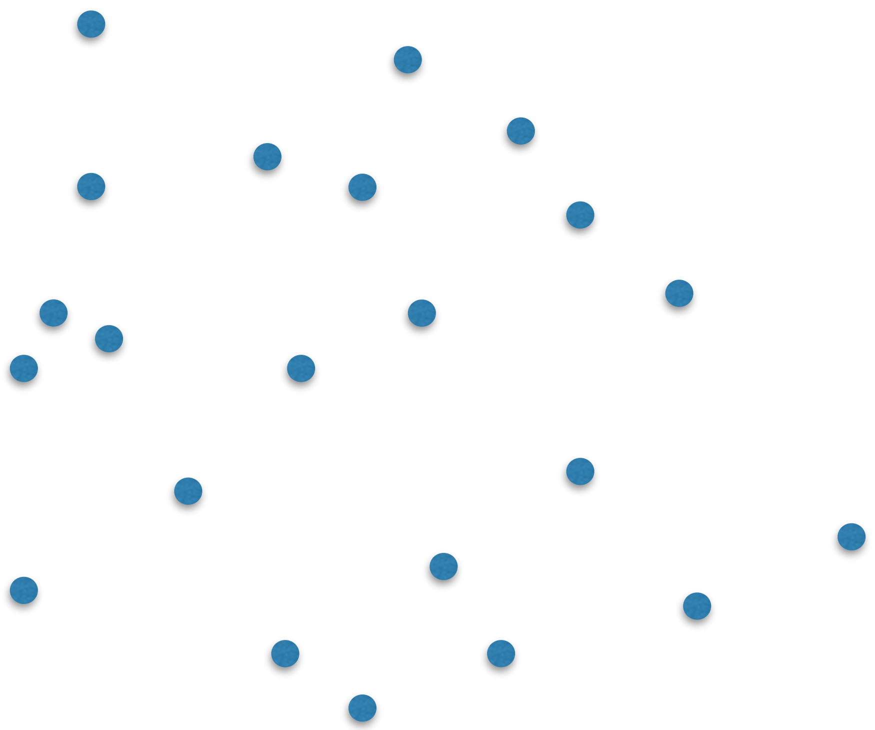
(csci3250)

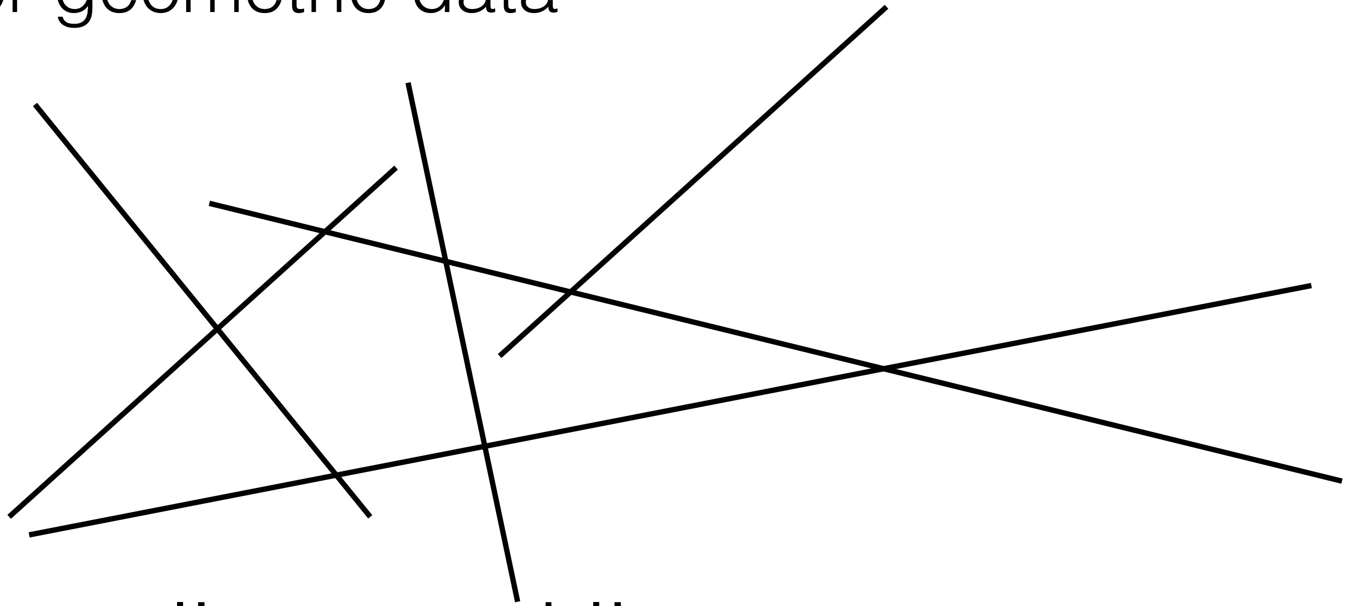
Laura Toma

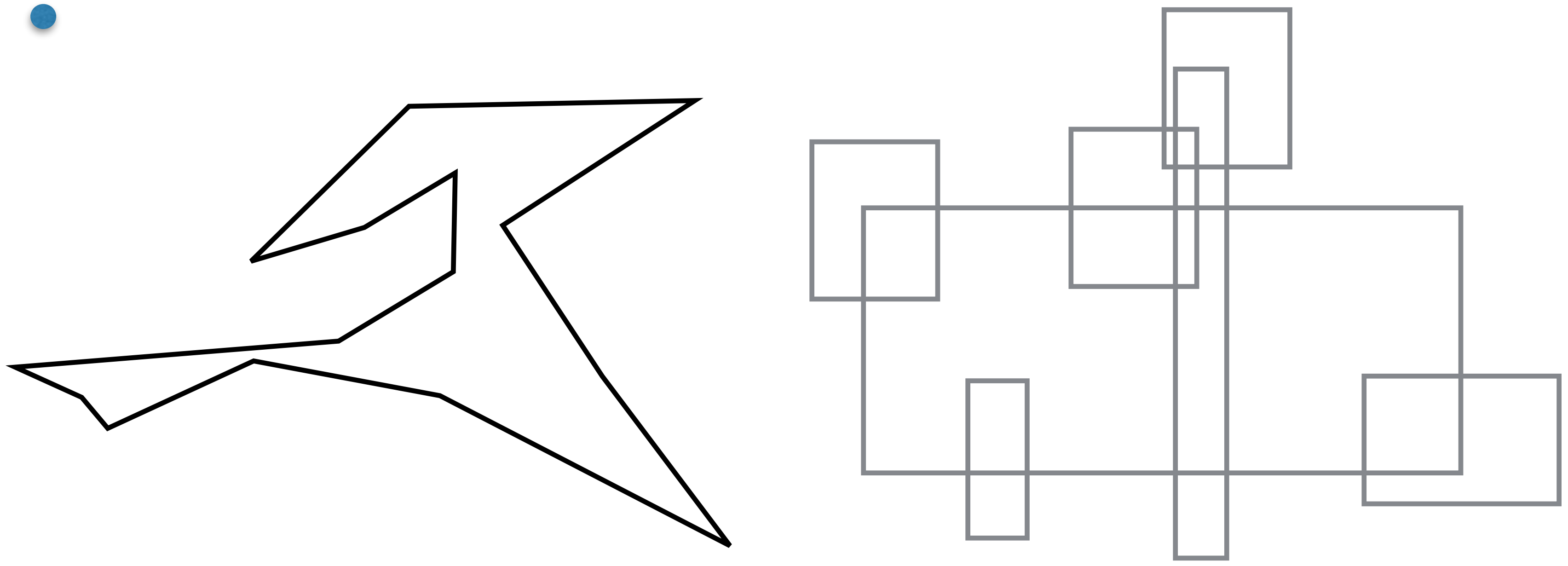
Bowdoin College

What is Computational Geometry?

- CG deals with algorithms for geometric data

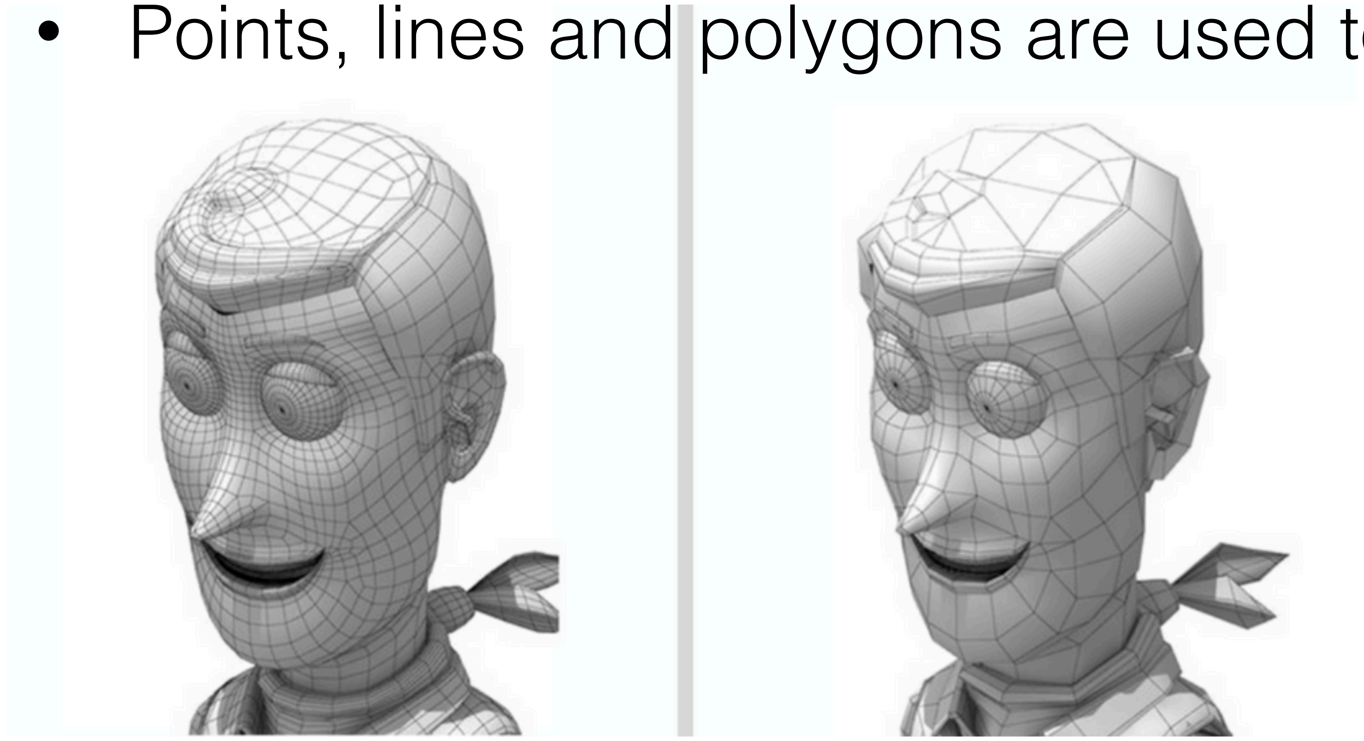

points


lines and line segments

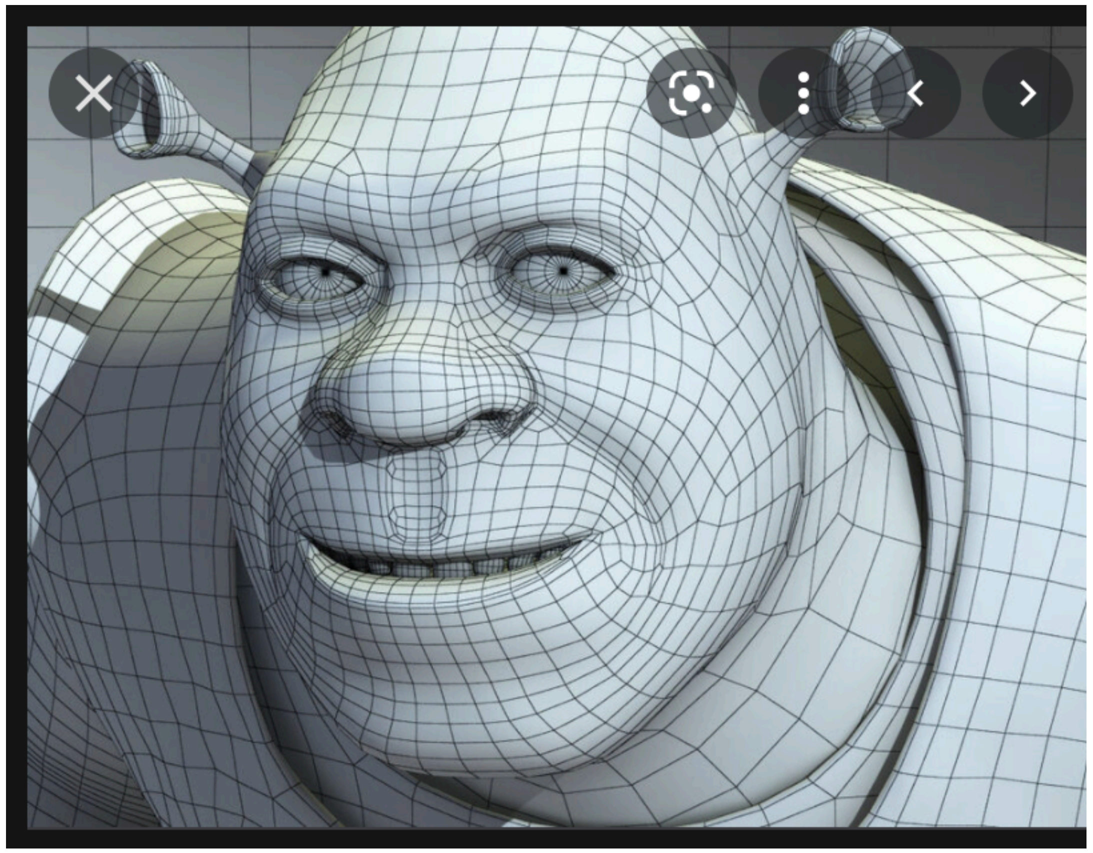

polygons

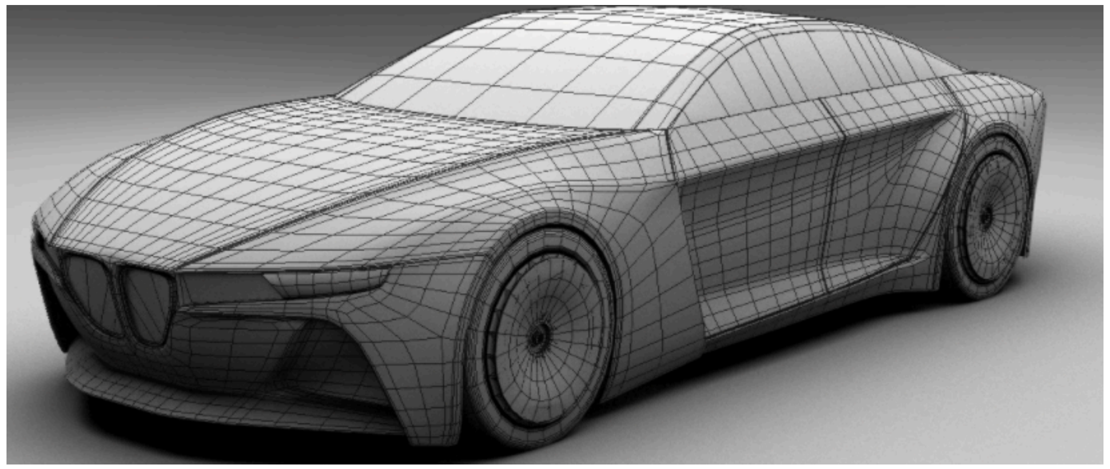
What is Computational Geometry?

- 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.)





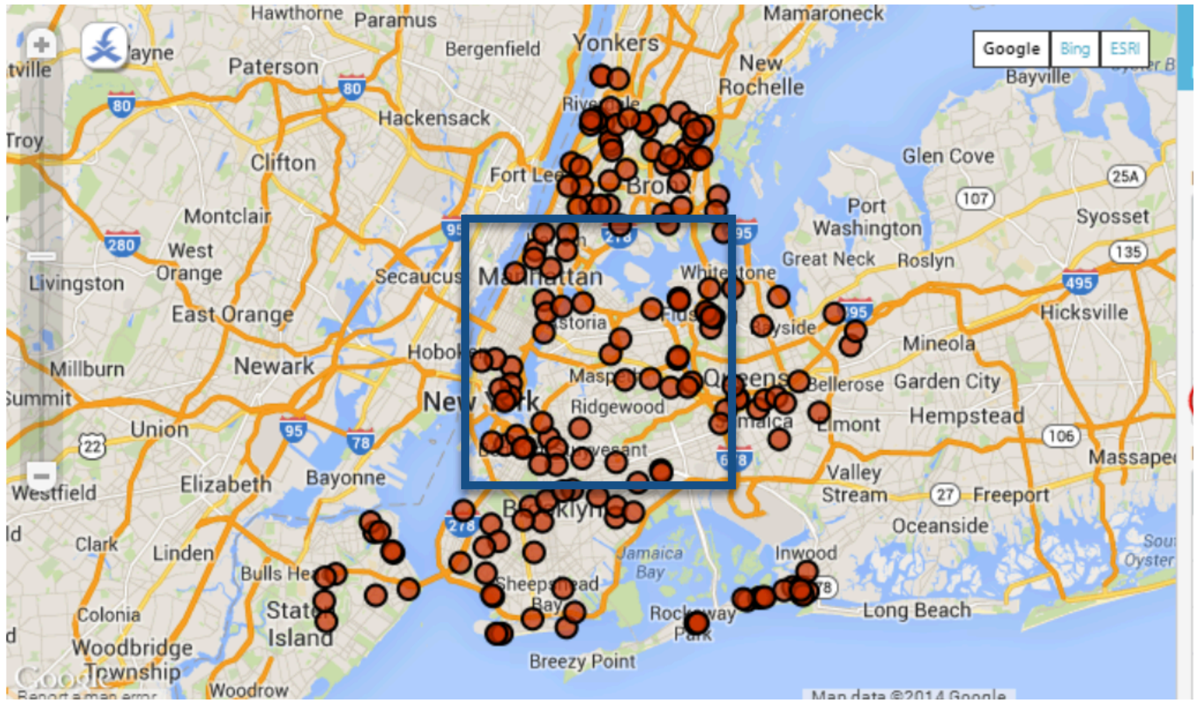
Applications

- Computer graphics and animation
    - rendering, hidden surface removal, lighting, moving, collision detection,..

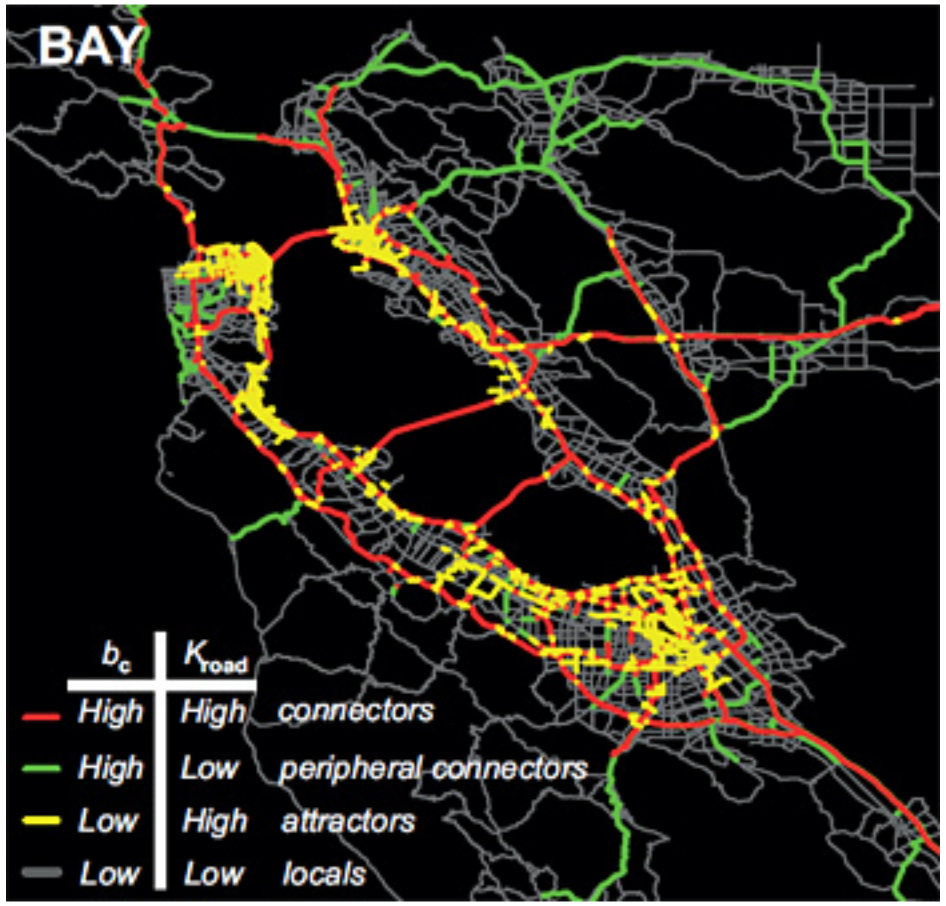
- Robotics and motion planning
- Autonomous vehicles
    - collision detection involves finding intersections

Applications

- Spatial database engines
  - store data and its geometry
  - contain specialized data structures for answering queries on geometric data
  - e.g.: find all restaurants in a range

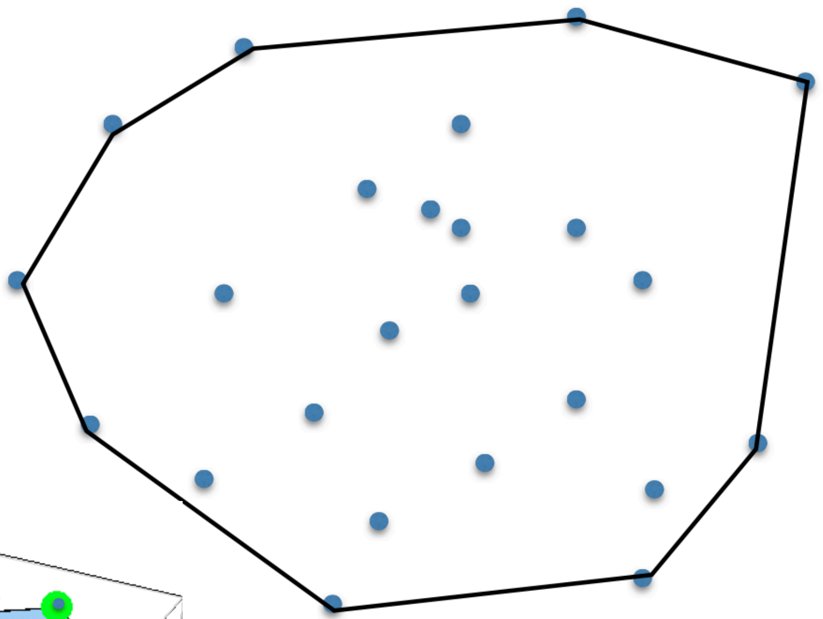
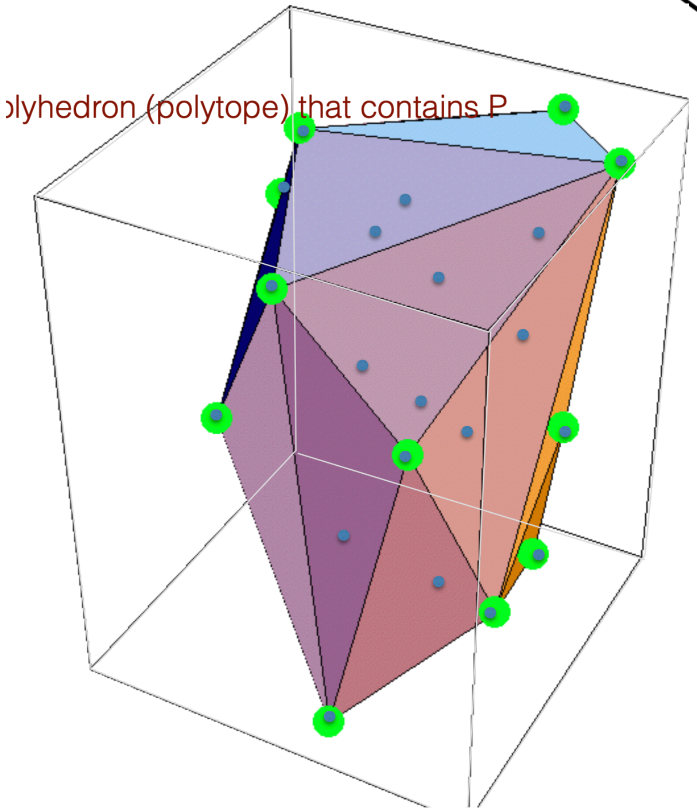


- Traffic analysis based on cell phone data
  - Use location data
  - Model real-time traffic conditions, find congestion patterns



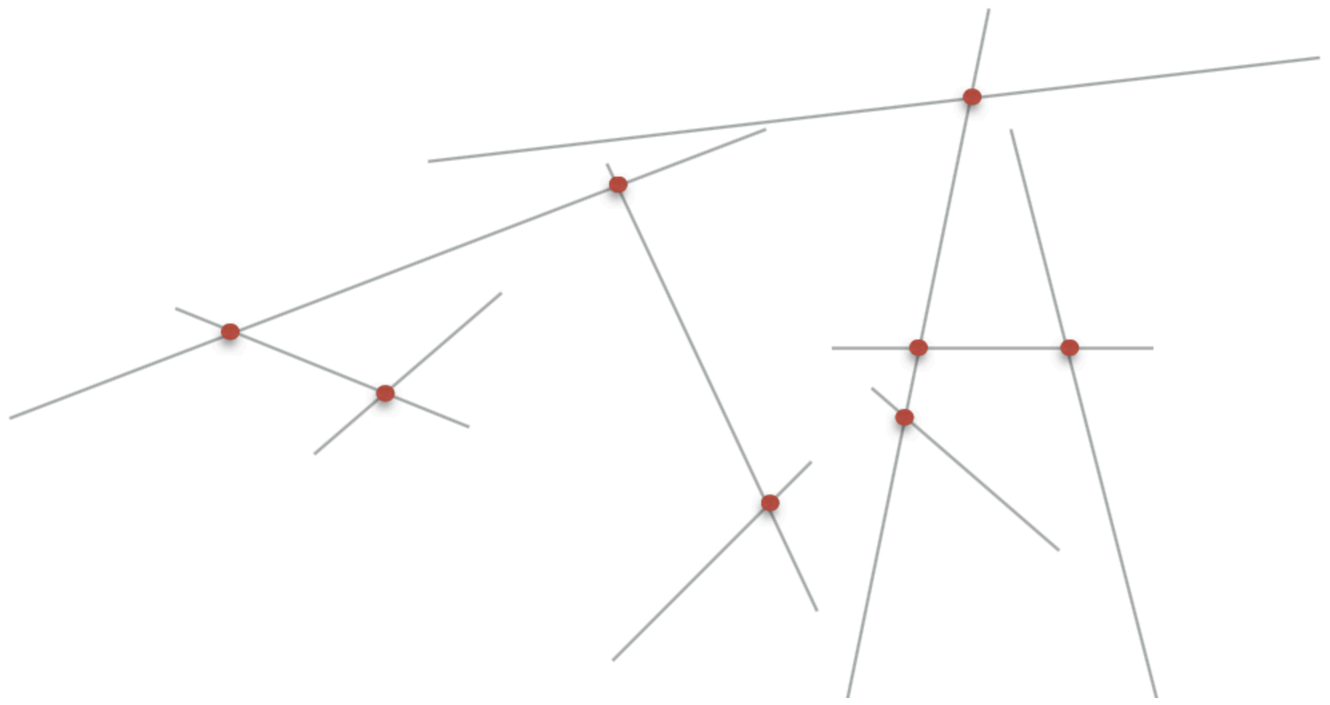
Syllabus overview

- Introduction and setup
- Geometric primitives
    - point leftOf segment, segment intersection
- 2D Convex hull
    - Gift wrapping, incremental. Quick hull, Graham scan
    - lower bound
- 3D convex hull
    - incremental algorithm

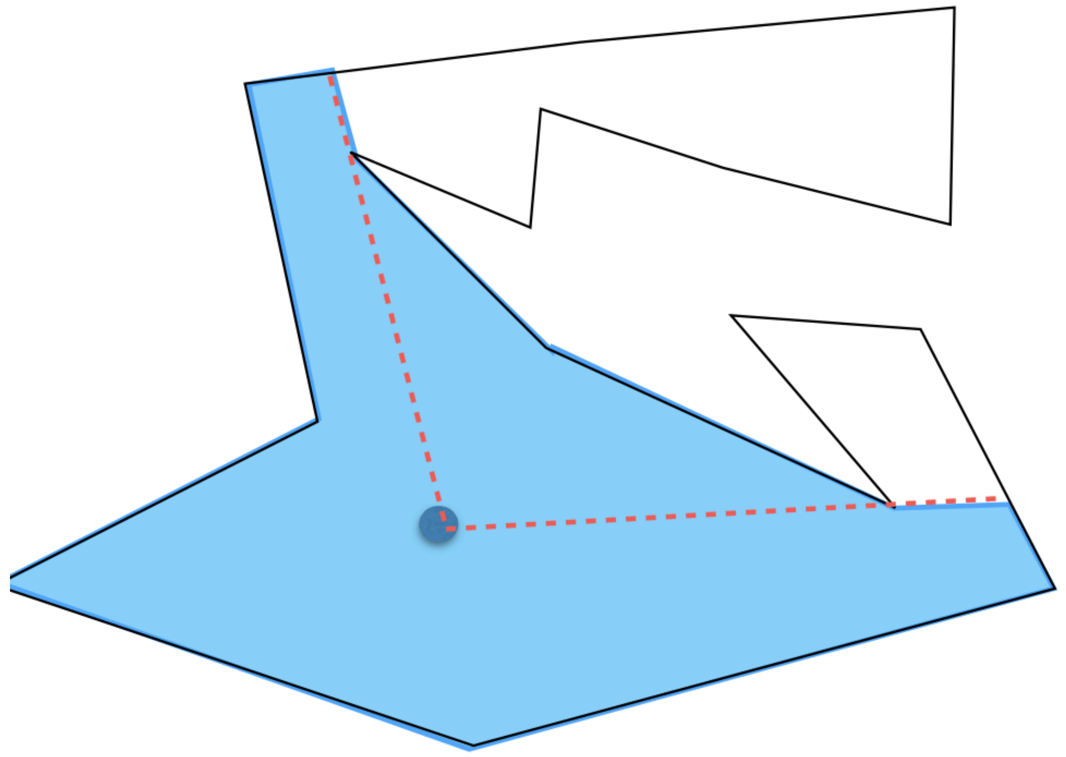



Syllabus overview

- Segment intersection
    - Bentley-Ottman sweep

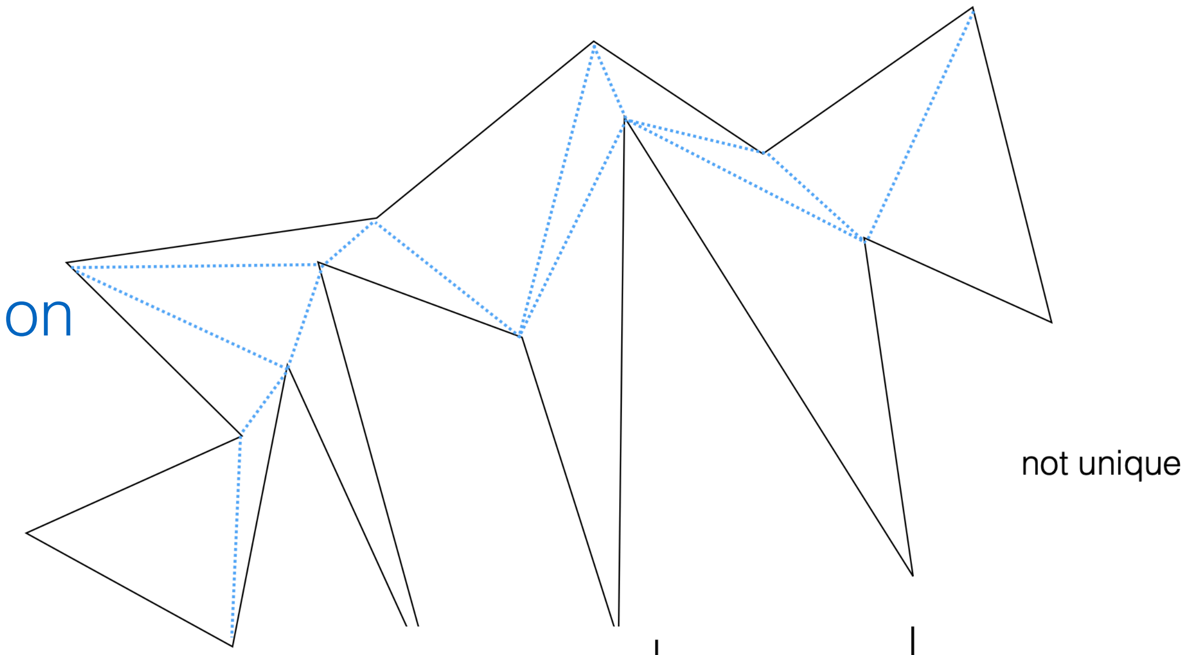


- The art gallery problem

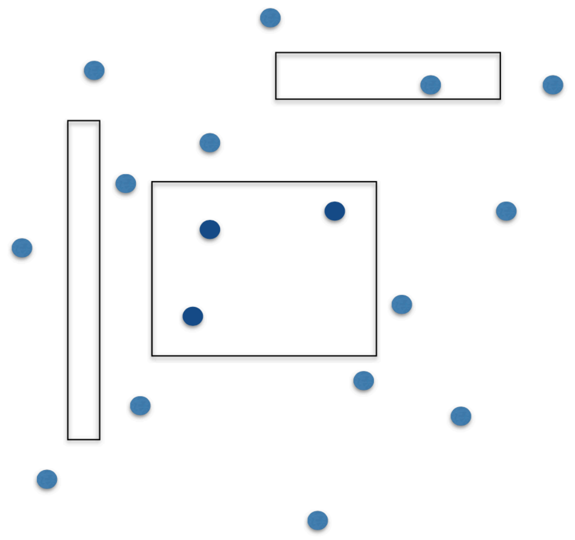


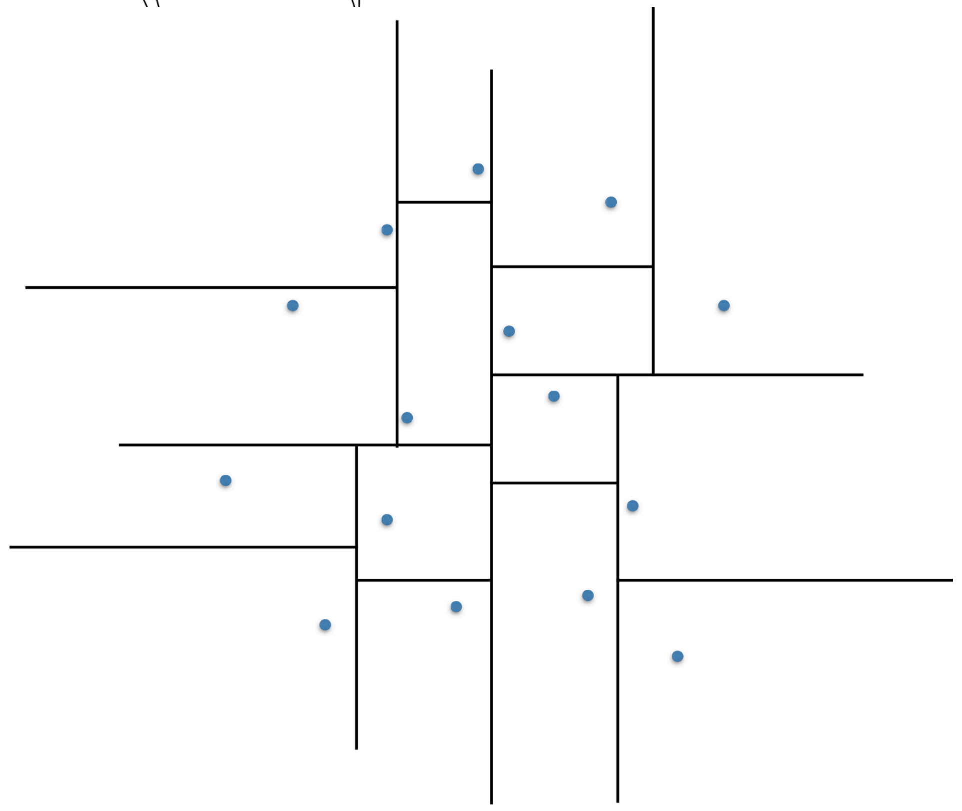
Syllabus overview

- Polygon triangulation
  - O(n lg n) algorithm via trapezoidalization



- Orthogonal range searching
  - kd-trees and range trees

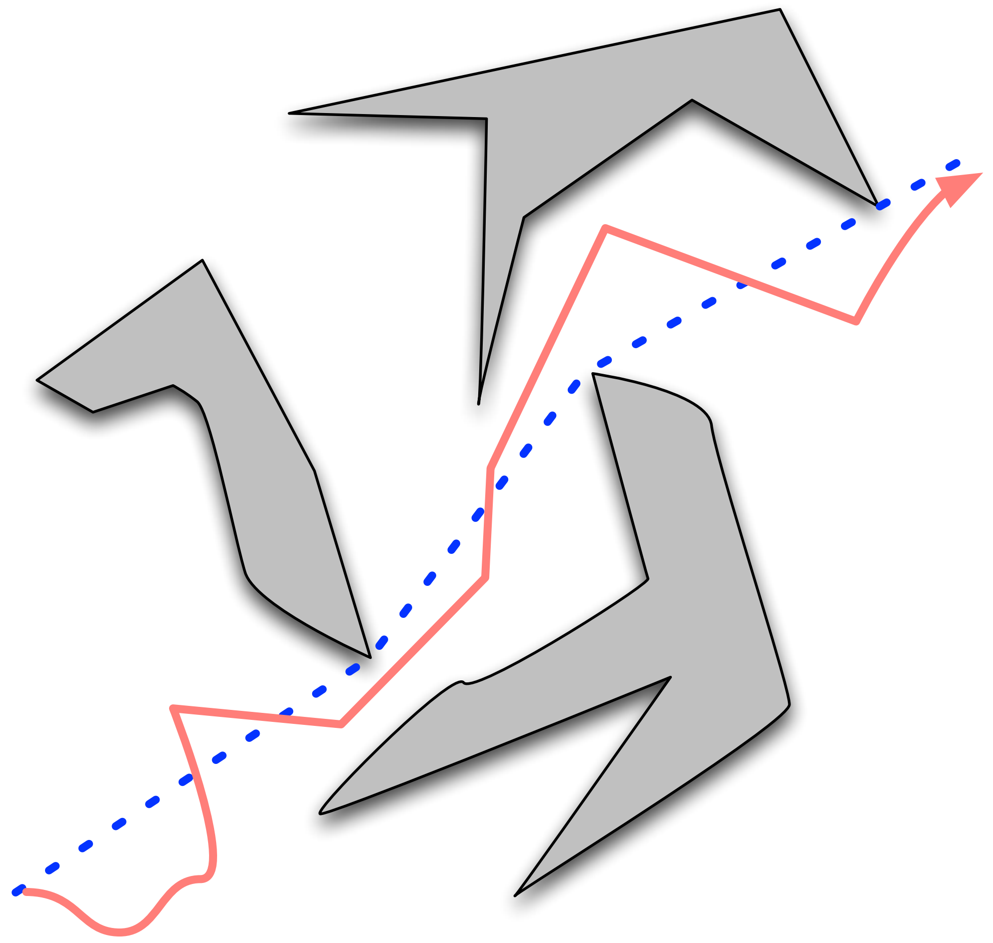




- ~~Voronoi diagrams and Delaunay triangulations~~
  - if time permits

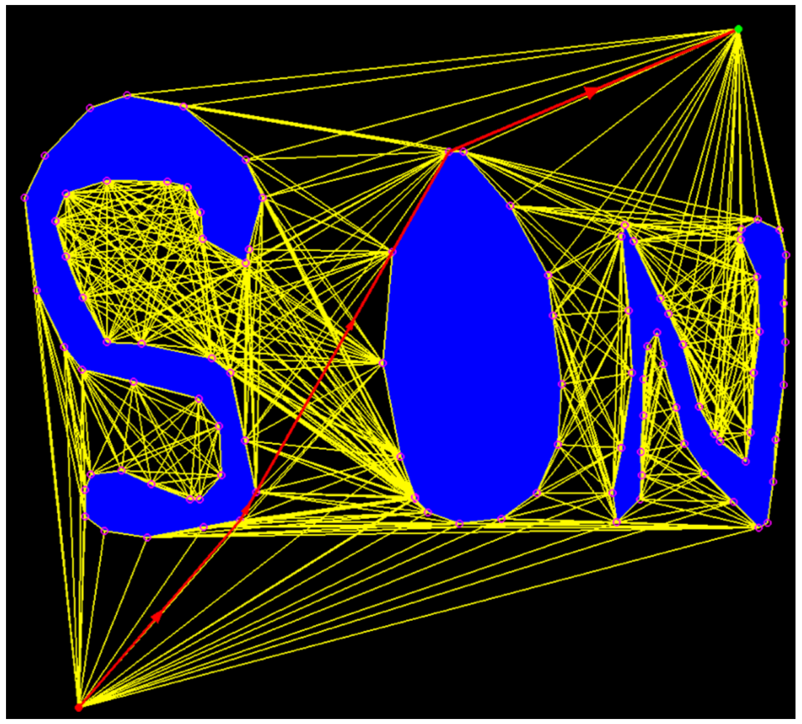
Syllabus overview

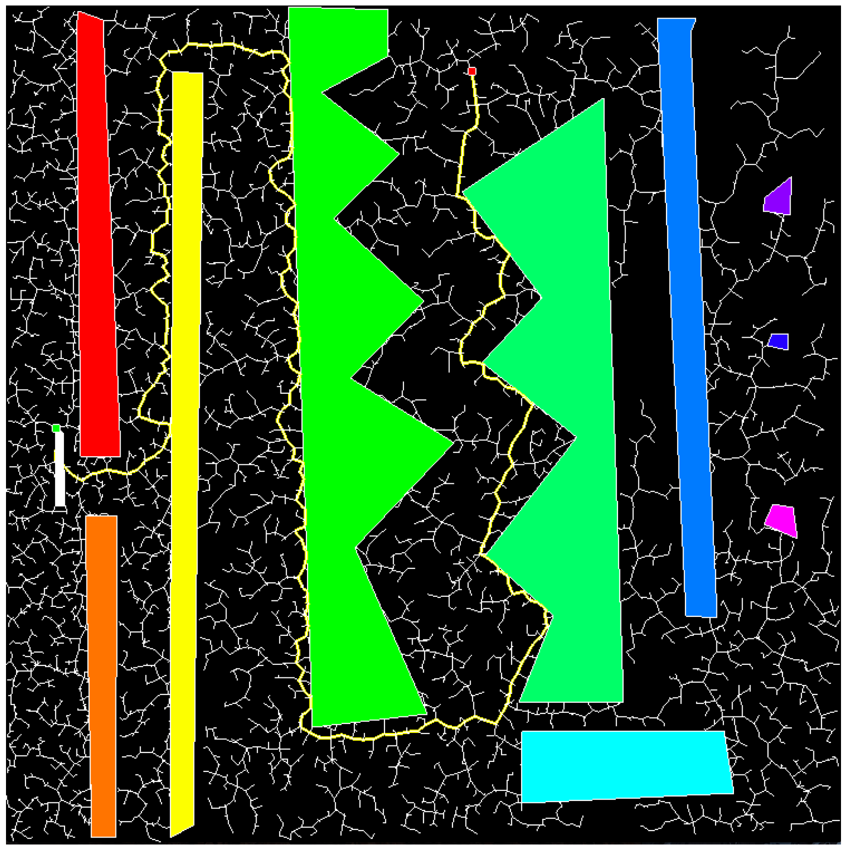
- Path planning : find collision-free path from start to end



Syllabus overview

- Path planning in 2D
  - shortest path in a simple (non-convex) polygon with the Funnel algorithm.
  - shortest paths among polygonal obstacles via visibility graph (VG)
  - computation of the VG with plane sweep.





- Path planning in 3D
  - combinatorial planning
  - sampling-based planning
    - probabilistic roadmaps (PRM)
    - RRT

Syllabus overview

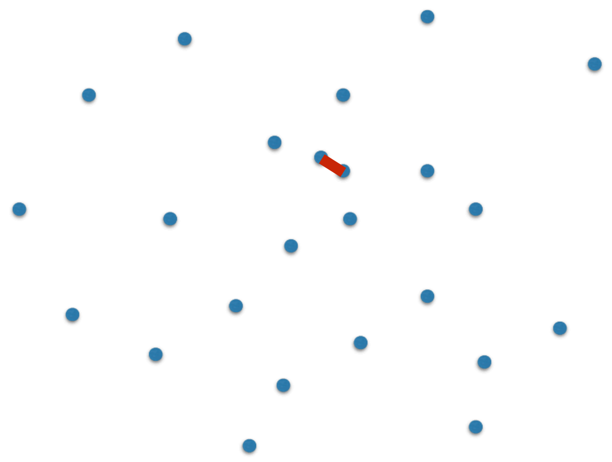
- We'll explore algorithms
- The usual questions
    - Properties of the solution?
    - Complexity of the result?
    - Worst-case running time?
    - Can we do better?
    - Lower bound for the problem?
    - Is the algorithm practical?
    - Handle degeneracies in the input?
    - Can we make some practical assumptions about the data?

How will this class work?

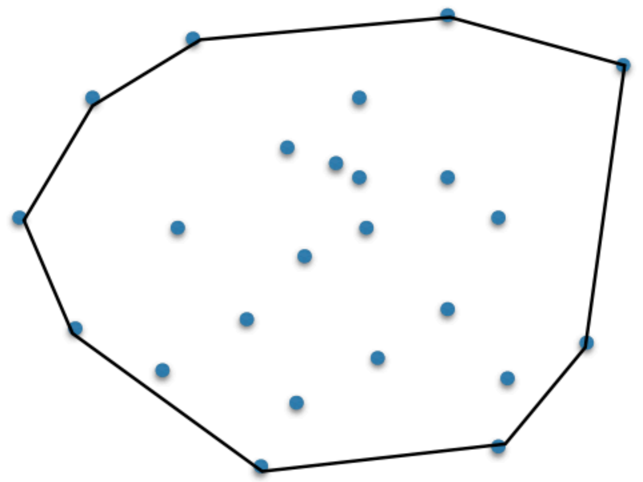
- Style: Lectures and in-class group work
- Material is theoretical, assignments are programming
- All work comes from programming assignments
    - 7 assignments, one every two weeks
    - Language: C/C++
    - We'll use GitHub
    - pair-programming encouraged (you must follow pair-programming guidelines posted on class website)

Assignments

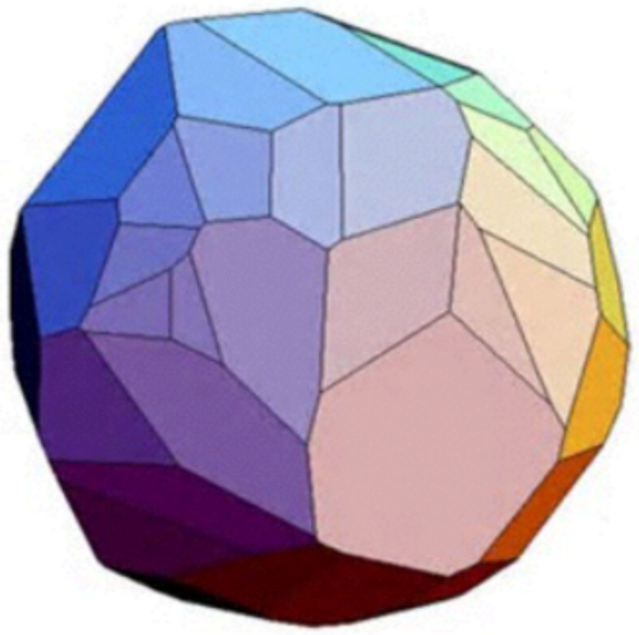
- A1: Finding the closest pair of points in a set



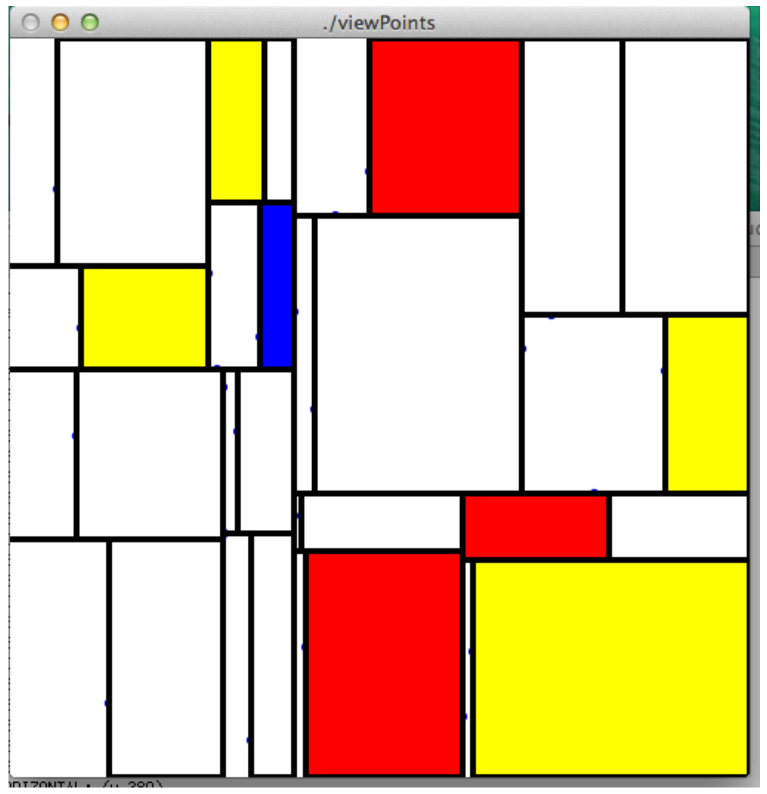
- A2: 2D convex hull via Graham scan



- A3: 3D convex hull

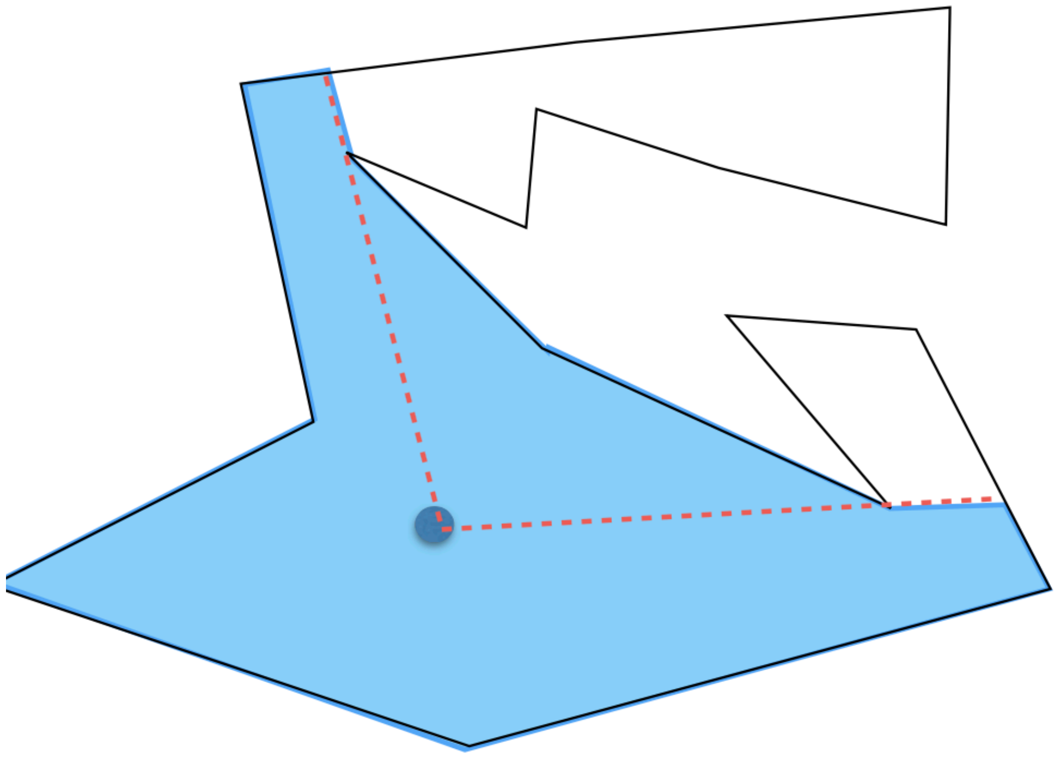


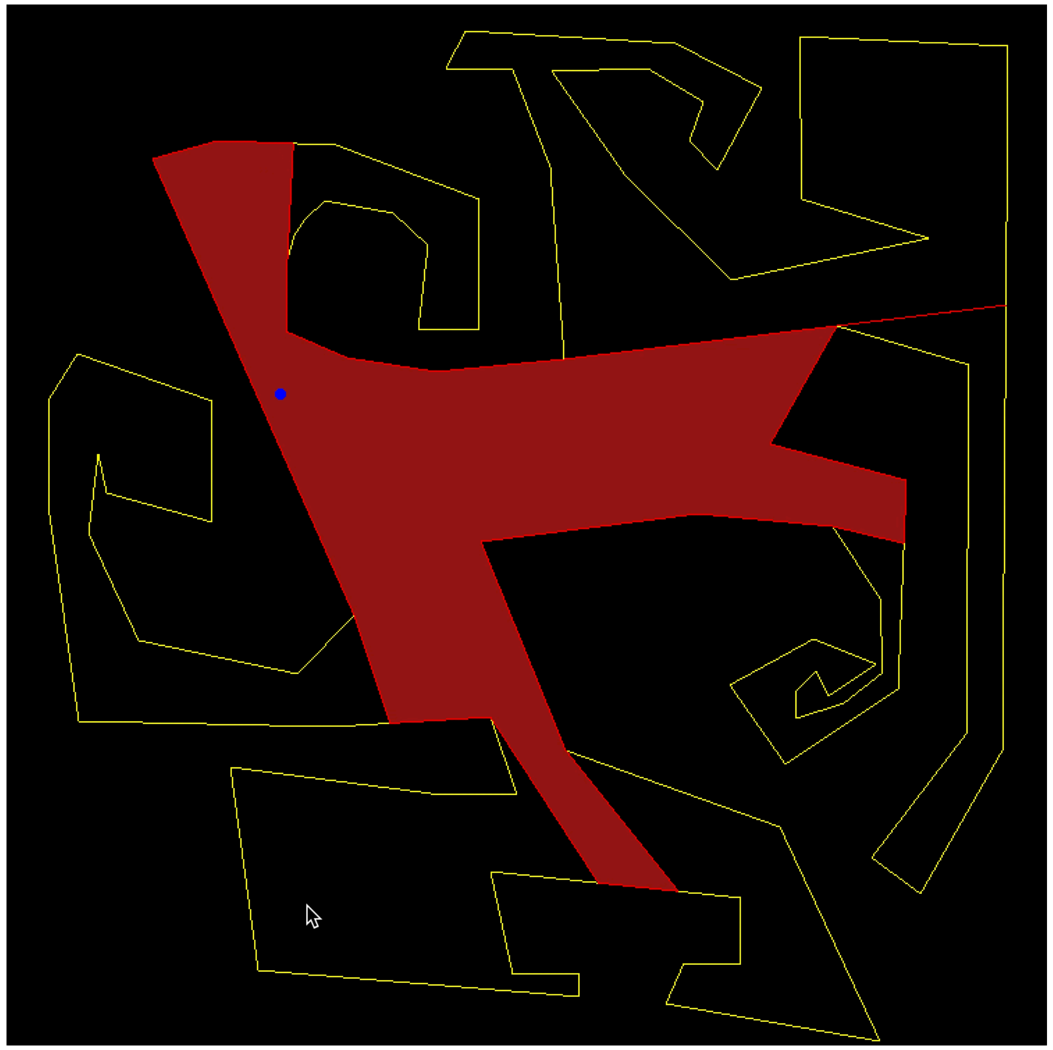
- A4: Mondrian art via building a kd-tree tree



Assignments

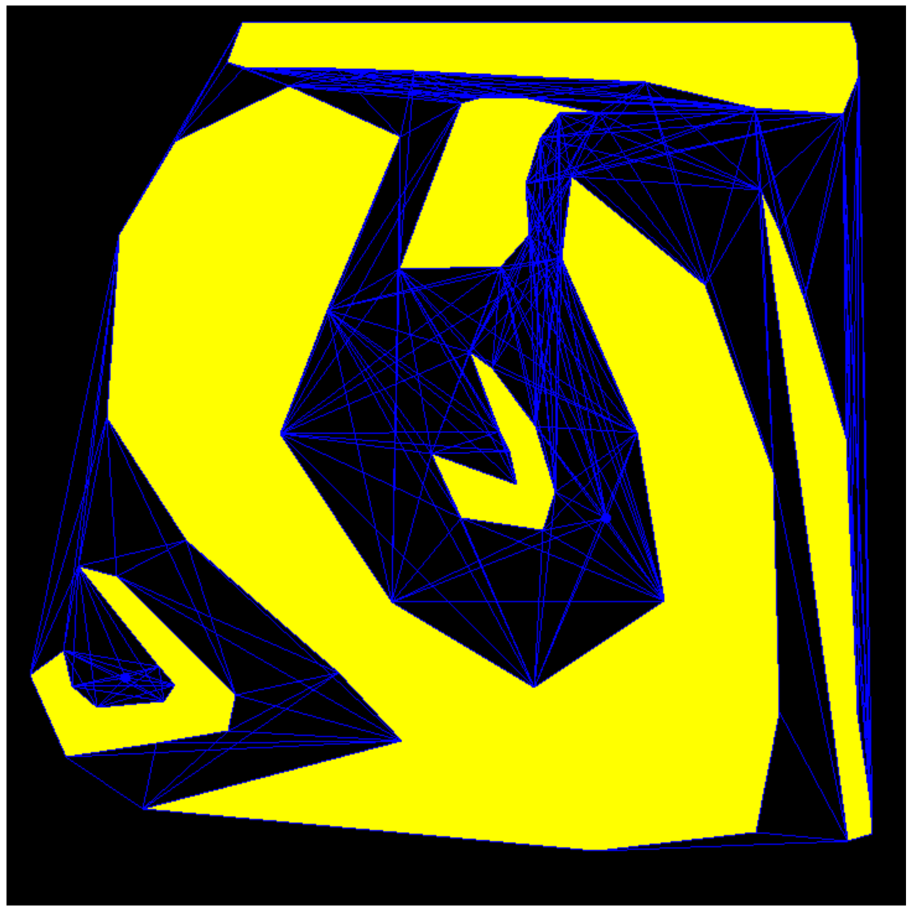
- Assignment 5: Guarding a non-convex polygon



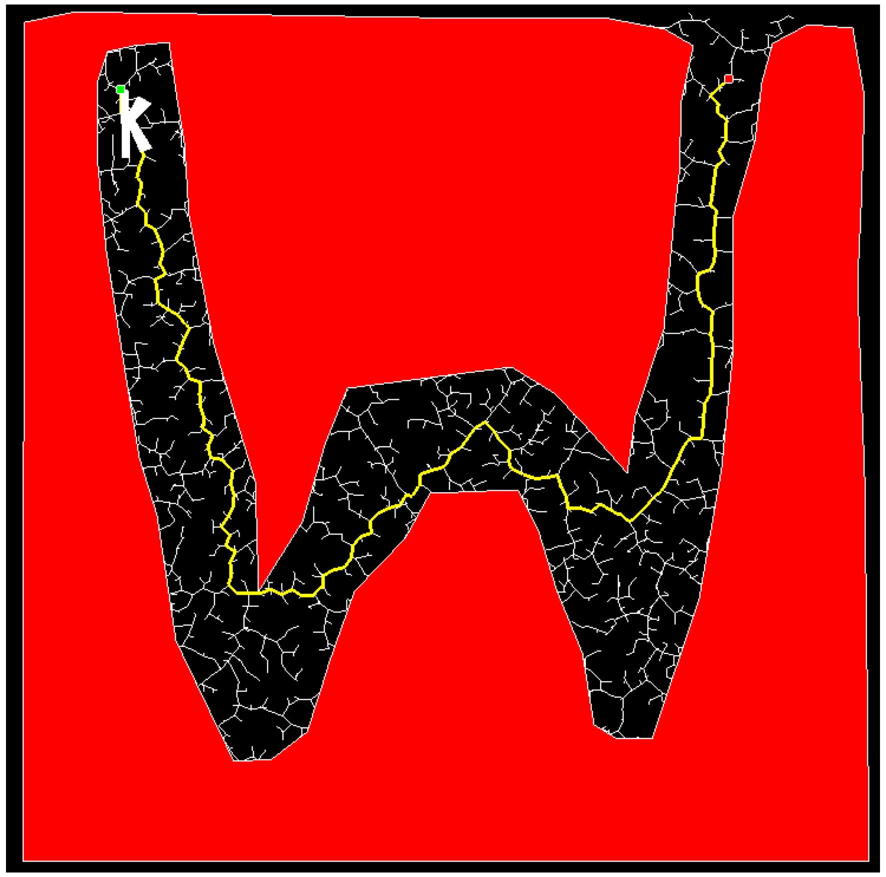


Assignments

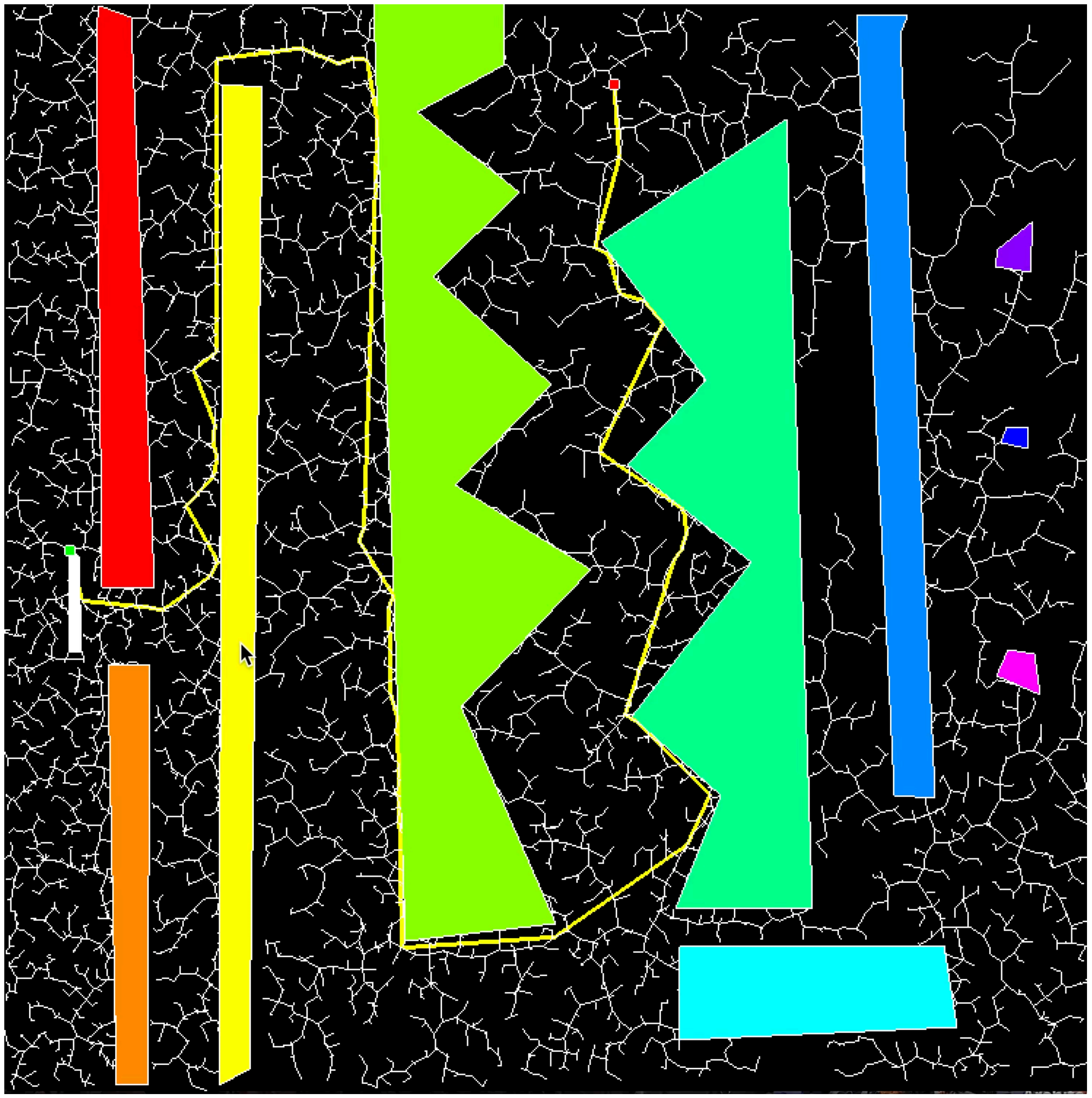
- Assignment 6: Motion planning via the VG for a point robot moving among polygonal obstacles in 2D



- Assignment 7: Heuristical motion planning for a polygonal robot moving in 2D via PRM or RRT



- Assignment 7: Heuristical motion planning for a polygonal robot moving in 2D via PRM or RRT

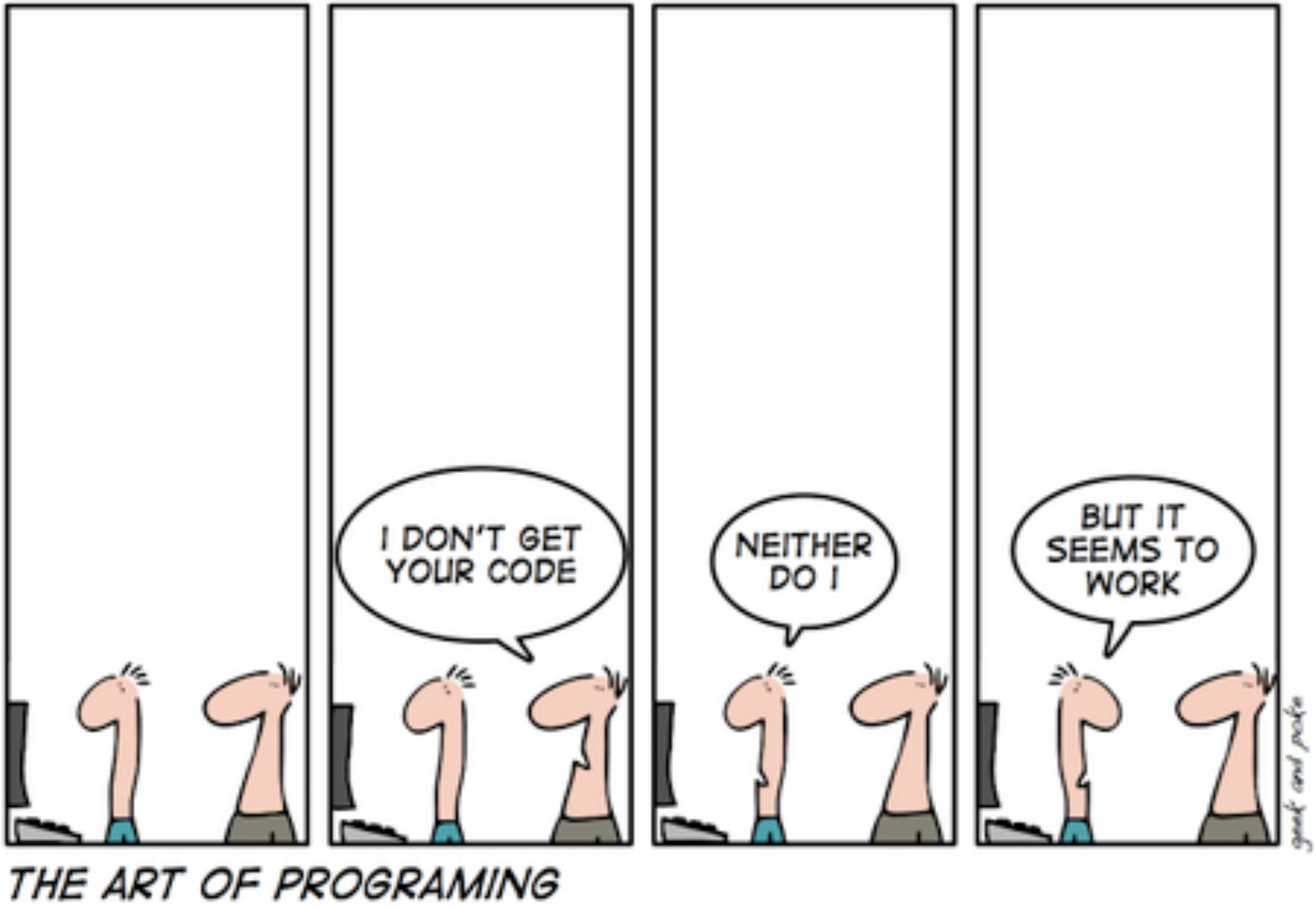


How will this class work?

- Office hours
    - tbd , will announce next week
- TAs
    - n/a

How will this class work?

- Grading
  - 7 assignments, each weighted equally
  - class participation for tie breaking
  - No exams
- Work can get intense!
  - plan accordingly..
- If you don’t like or don’t want to learn programming
  - seriously consider a different class and send me an email to discuss alternatives



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

Programming is a craft

- Systems is a prereq for this class, so everyone should be familiar with C/C++
- Assignments are increasing in difficulty
  - The first assignments are easier and skeleton code is provided
  - Assignments 5, 6, 7 are harder and from scratch
- Programming is not a science, it’s a learnt craft
  - We all grow as programmers by practicing. No exceptions.
  - Start wherever you are and move forward!

Programming is a craft

- For starters, expect to spend most of your time debugging your code
    - From here, 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 code => high quality code ==> easy to debug

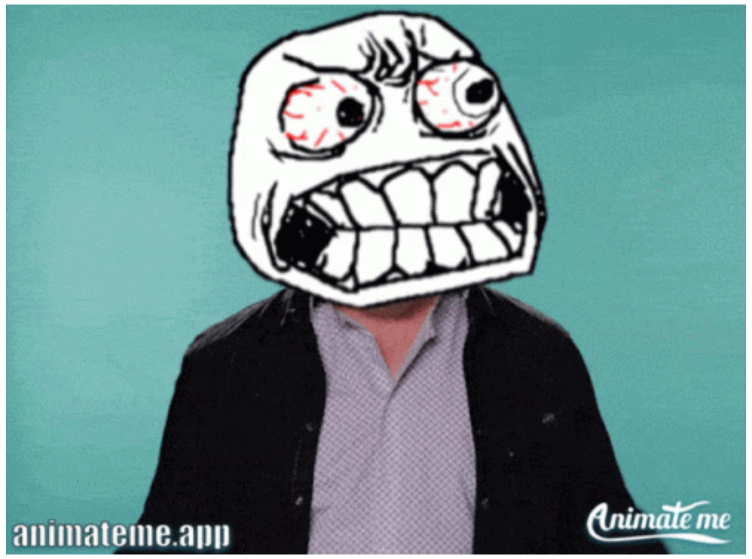
Growing as a programmer: the cycle we all go through

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!

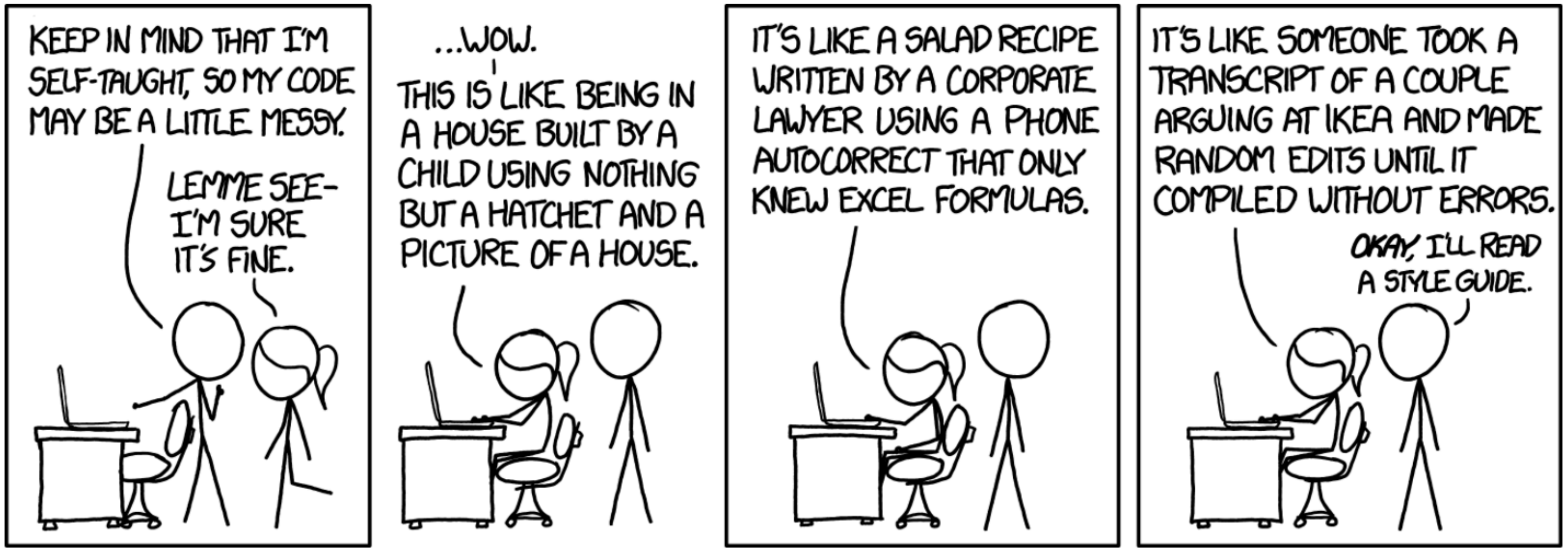
small functions

develop incrementally

good comments

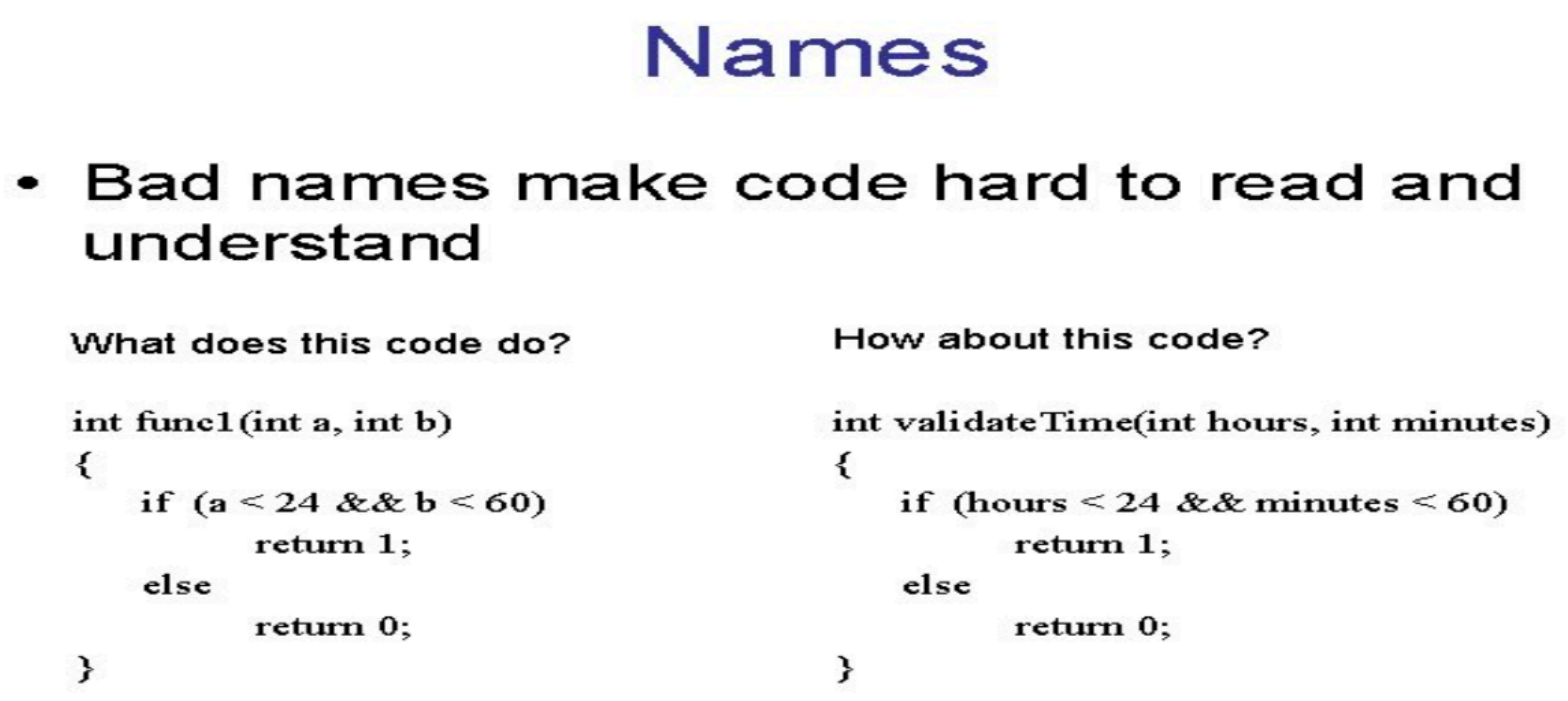
test early and often

In this class…YOU need to debug YOUR code.


KEEP IN MIND THAT I'M SELF-TAUGHT, SO MY CODE MAY BE A LITTLE MESSY.
LEMME SEE– I'M SURE IT'S FINE.
...WOW.
THIS IS LIKE BEING IN A HOUSE BUILT BY A CHILD USING NOTHING BUT A HATCHET AND A PICTURE OF A HOUSE.
IT'S LIKE A SALAD RECIPE WRITTEN BY A CORPORATE LAWYER USING A PHONE AUTOCORRECT THAT ONLY KNEW EXCEL FORMULAS.
IT'S LIKE SOMEONE TOOK A TRANSCRIPT OF A COUPLE ARGUING AT IKEA AND MADE RANDOM EDITS UNTIL IT COMPILED WITHOUT ERRORS.
OKAY, I'LL READ A STYLE GUIDE.

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 and debug and maintain
- Testing
  - Add testing and test cases. Make sure one function passes the testing before you move to the next one
  - Test early and often. Don’t write everything and then start the testing



Names

- Bad names make code hard to read and understand

What does this code do?

```
int func1(int a, int b)
{
     if  (a < 24 && b < 60)
              return 1;
     else
              return 0;
}
```

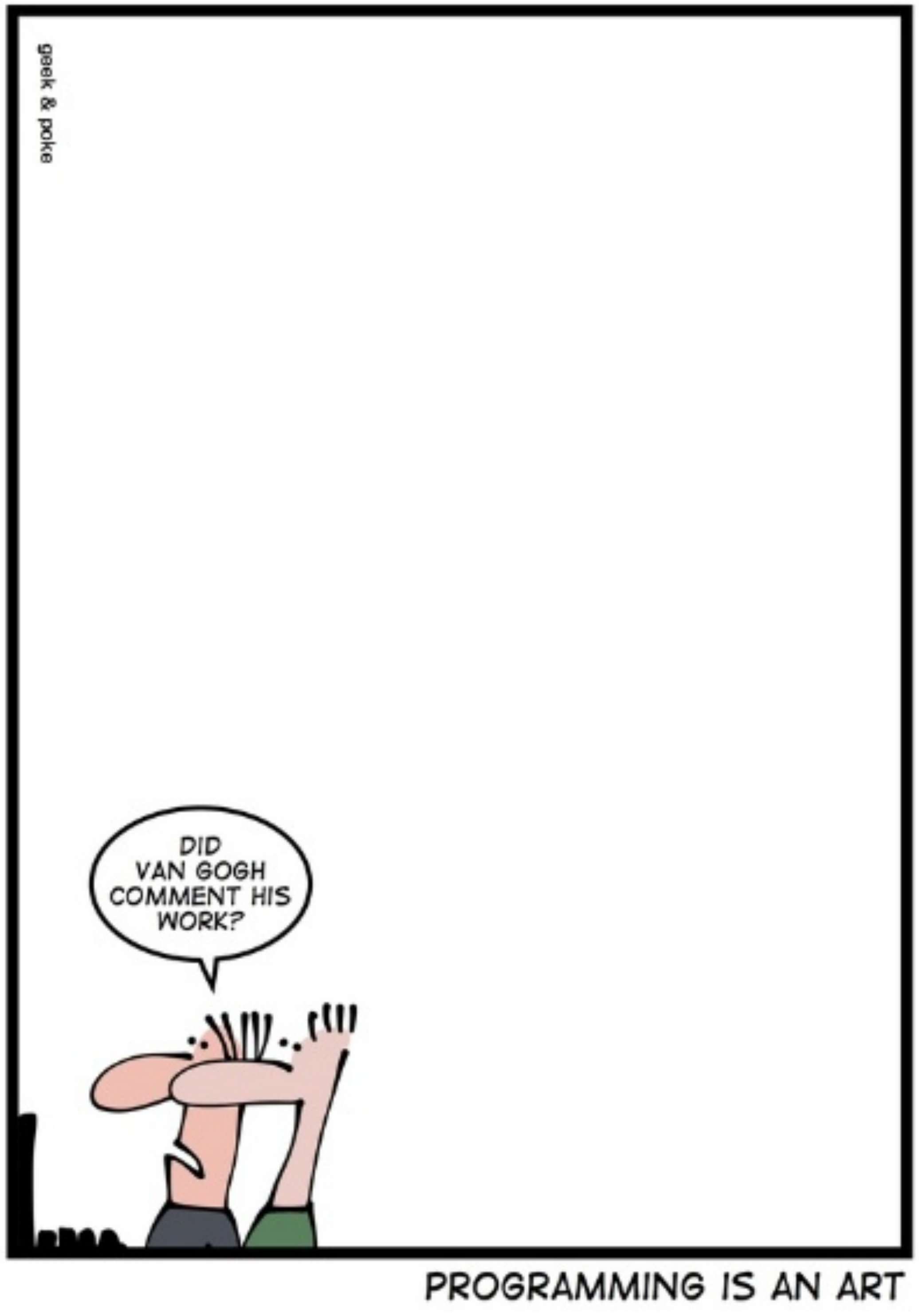
How about this code?

```
int validateTime(int hours, int minutes)
{
     if  (hours < 24 && minutes < 60)
              return 1;
     else
              return 0;
}
```

Credit:

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.



YOU need to get here

I will not write any more bad code
I will not write any more bad code
I will not write any more bad code
I will not write any more bad code
I will not write any more bad code
I will not write any more bad code
I will not write any more bad code
I will not write any more bad code
I will not write any more bad code
I will not write any more bad code
I will not write any more bad code

The faster the better!

Teaching philosophy

How will this class work?

- For algorithmic work, group problem solving is crucial to understanding the material
- I count on you to engage
- Learning will be a lot better for everyone if we create a community
- The classroom is a **friendly space** to ask questions and to “not know”
  - Don’t feel bad to look *dumb*!
  - If someone thinks a question is dumb, then they are dumb
  - Questions and mistakes lead to learning
- Respect the class by preparing and staying on track

Flexibility

- This is still a weird semester, and I am committed to being flexible
- We may need to drop material and assignments
    - that's ok!
- If anyone of us gets sick (with covid), work won't be expected while we recover
    - that's ok!
- Send me an email with any concerns or if you have circumstances that make your learning difficult