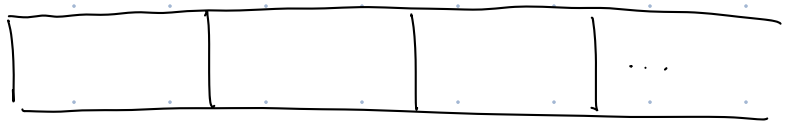
DIVIDE + CONQUER: CLOSEST POINTS

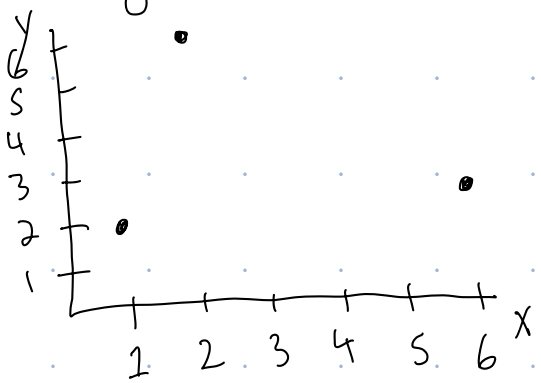
Learning Goals

- Analyze runtime of D&C alg [D&C 1]
- Create a D&C alg [D&C 2]
- Benchmark algorithm with brute force / easier problem
- Build Intuition with easier problems
- Analyze ethics of alg using ethical matrix [Eth 1]

Closest Points Problem

Input: Array of 2-D points:

P = 



Output:

Applications:

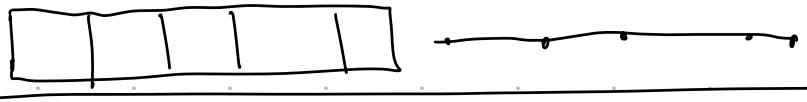
Algorithms + Ethics

Algorithm is essentially a mathematical object.
But once it gets implemented for a particular task, has ethical implications

Closest Points 2D

Before designing a sophisticated algorithm, try to benchmark

- Want better than "Brute Force"
- Can't do better than 1-D

Input: P= 

3. Runtime

BruteForce2D(P) (check every pair)

.
.
.
.

ClosestPts1D(P) (For loop)

2. ↳

.
.
.
.

1. go/cs302groups

DC_CP_1D(P) (Divide + Conquer)

Input : Integer array A of length n
Output: Sorted array

```
// Base Case
if n == 1 then
    return A;
end
// Divide and Conquer
A_1 = MergeSort(A[1 : n/2]);
A_2 = MergeSort(A[n/2 + 1 : n]);
// Combine
```

← Base Case (When too small to divide further?)

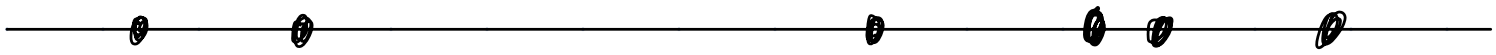
← Divide input into equal sized pieces + Recursively solve each piece

* How solve whole problem if can solve parts

Base Case!

What size set of pts should trigger base case?

A) 0 B) ≤ 1 C) ≤ 2 D) ≤ 3



DC_CP_1D (P) (Divide + Conquer)

Runtime?

Strategy to improve runtime:

1. find slow parts in recursive call
2. Remove from recursion or improve

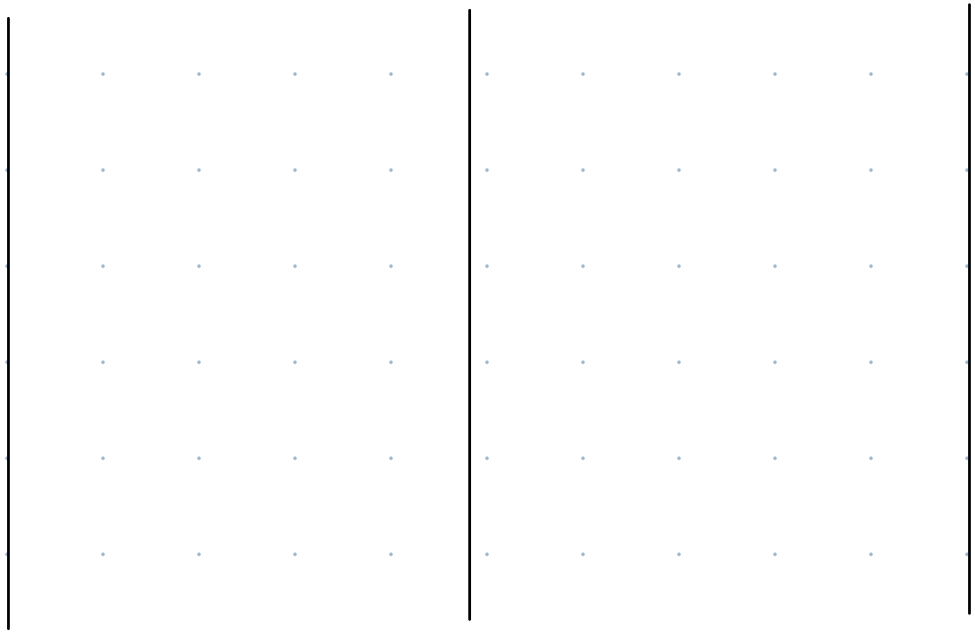
DC_CP_1D_preSort(P)

Ethical Matrix (O'Neil + Gunn)

Where travel + why?

Air Traffic Control Improvement

Stakeholders	Well-Being ↑	Autonomy ↑	Justice ↑



Ethical matrix does not tell you what to do!

Tool for thinking about consequences, both + and −.

→

→