CS133
Computational Geometry

Instructor: Ahmed Eldawy

TA: Samriddhi Singla

Welcome back to UCR!

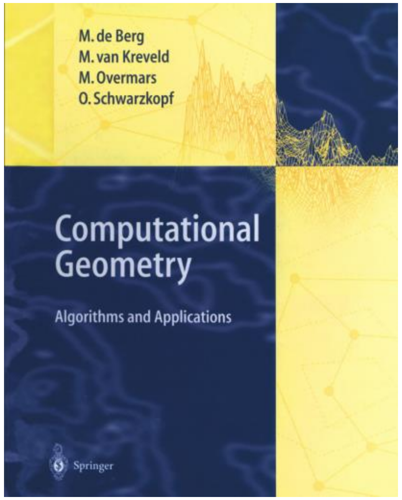
Class information

- Classes: Tuesday and Thursday 8:10 AM – 9:30 AM at WCH 142
- Instructor: Ahmed Eldawy
- TA: Samriddhi Singla
- Office hours:
  Tuesday and Thursday 9:30 AM – 10:30AM @357 WCH
- Conflicts?
  - You can set a meeting by email

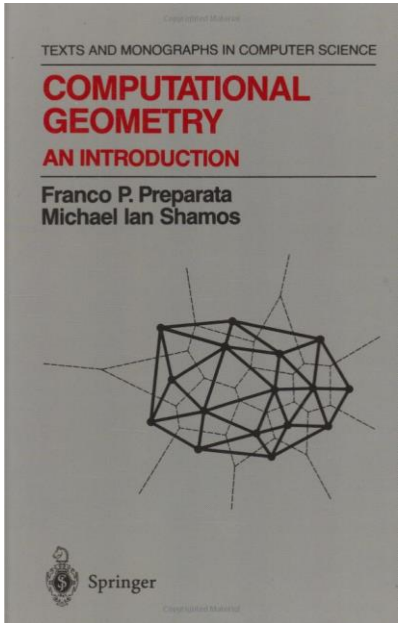
Class Information

- Website: http://www.cs.ucr.edu/~eldawy/19SCS133/
- Email: eldawy@ucr.edu
- Subject: “[CS133] …”

Textbook



Computational Geometry: Algorithms an Applications 3rd Ed, Springer By Mark de Berg, Otfried Cheong, Marc van Kreveld, and Mark Overmars
ISBN: 9783642096815
Free electronic version provided by UCR



(Optional) Computational Geometry: An Introduction, Springer, 2nd Ed By Franco P. Preparata and Michael I. Shamos
ISBN : 0387961313
Available at Orbach Library

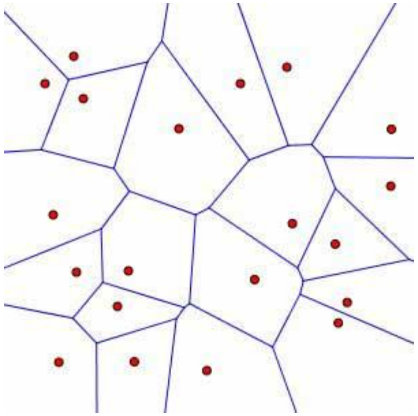
Course work

- (5%) Active participation in the class
- (10%) 5 assignments (Lowest one discarded)
- (30%) 10 labs (Lowest two discarded)
- (10%) First midterm (Tuesday, April 23rd)
- (10%) Second midterm (Thursday, May 23rd)
- (35%) Final exam
  - Date: Saturday, June 8th, 2019
  - Time: 8:00 a.m. - 11:00 a.m.
  - Location: WCH 142

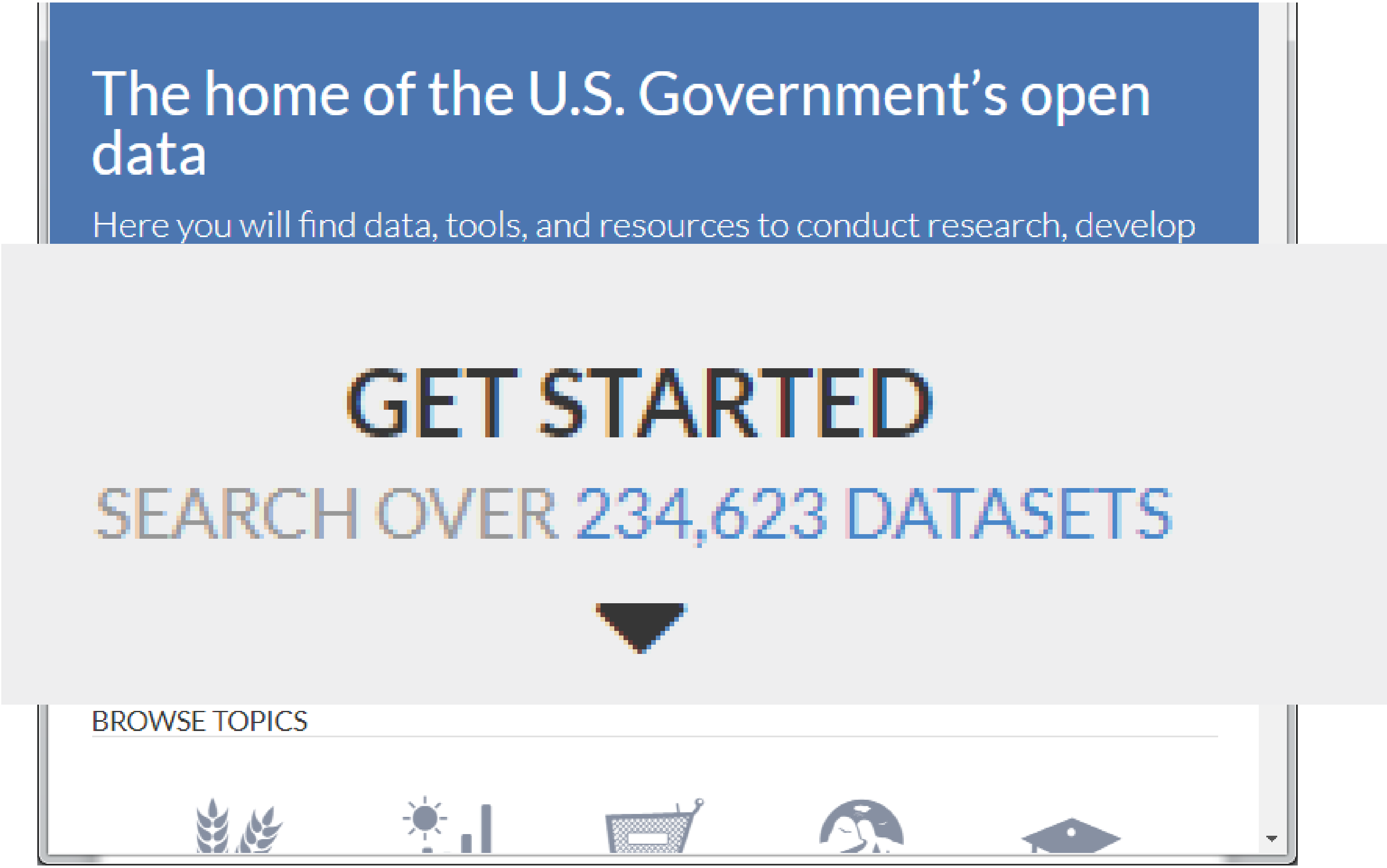
Course goals

- What are your goals?

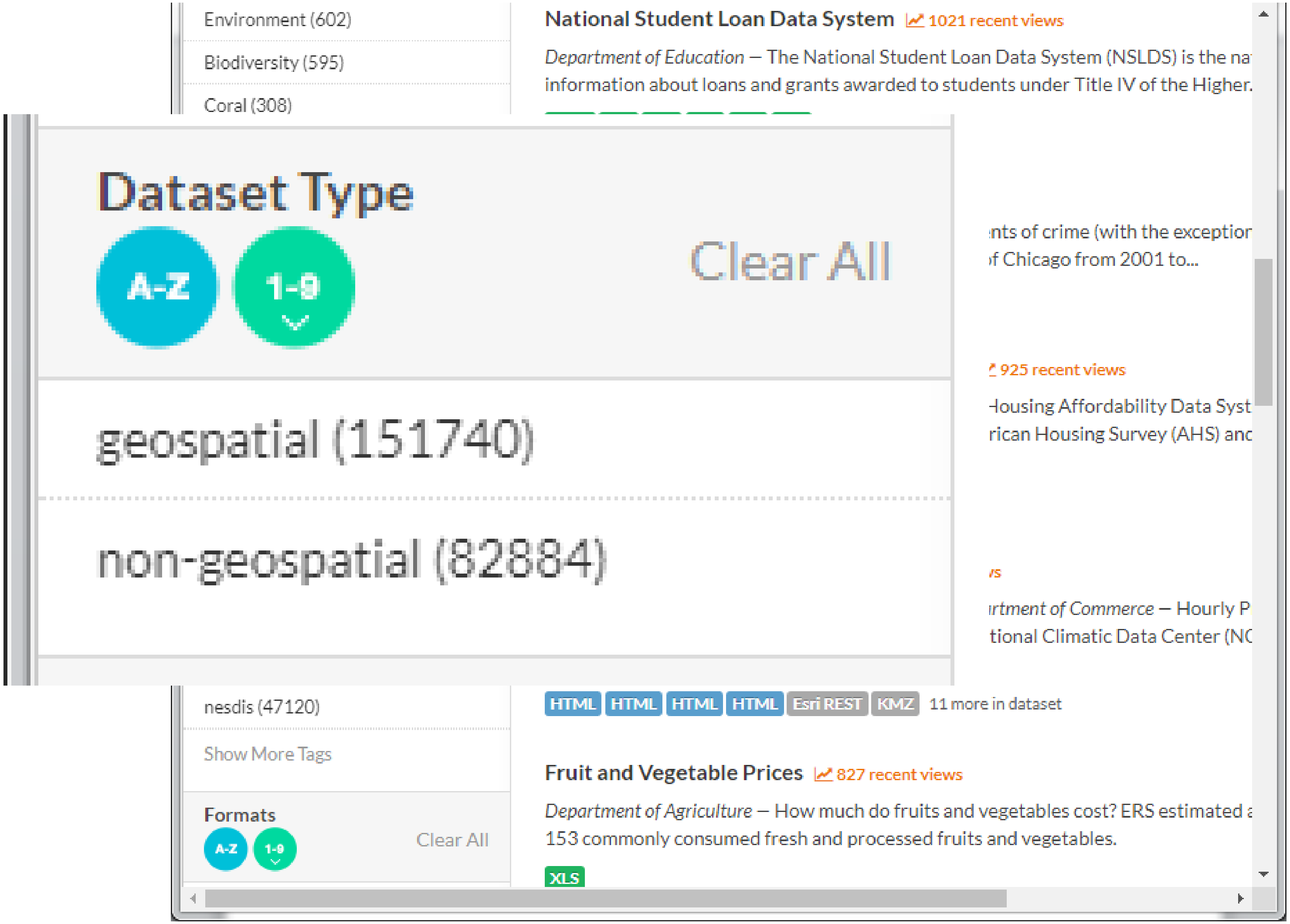
- Sharpen your algorithmic skills
- Understand a new type of algorithms
- Play with points, lines, and polygons
- Generate some nice-looking figures



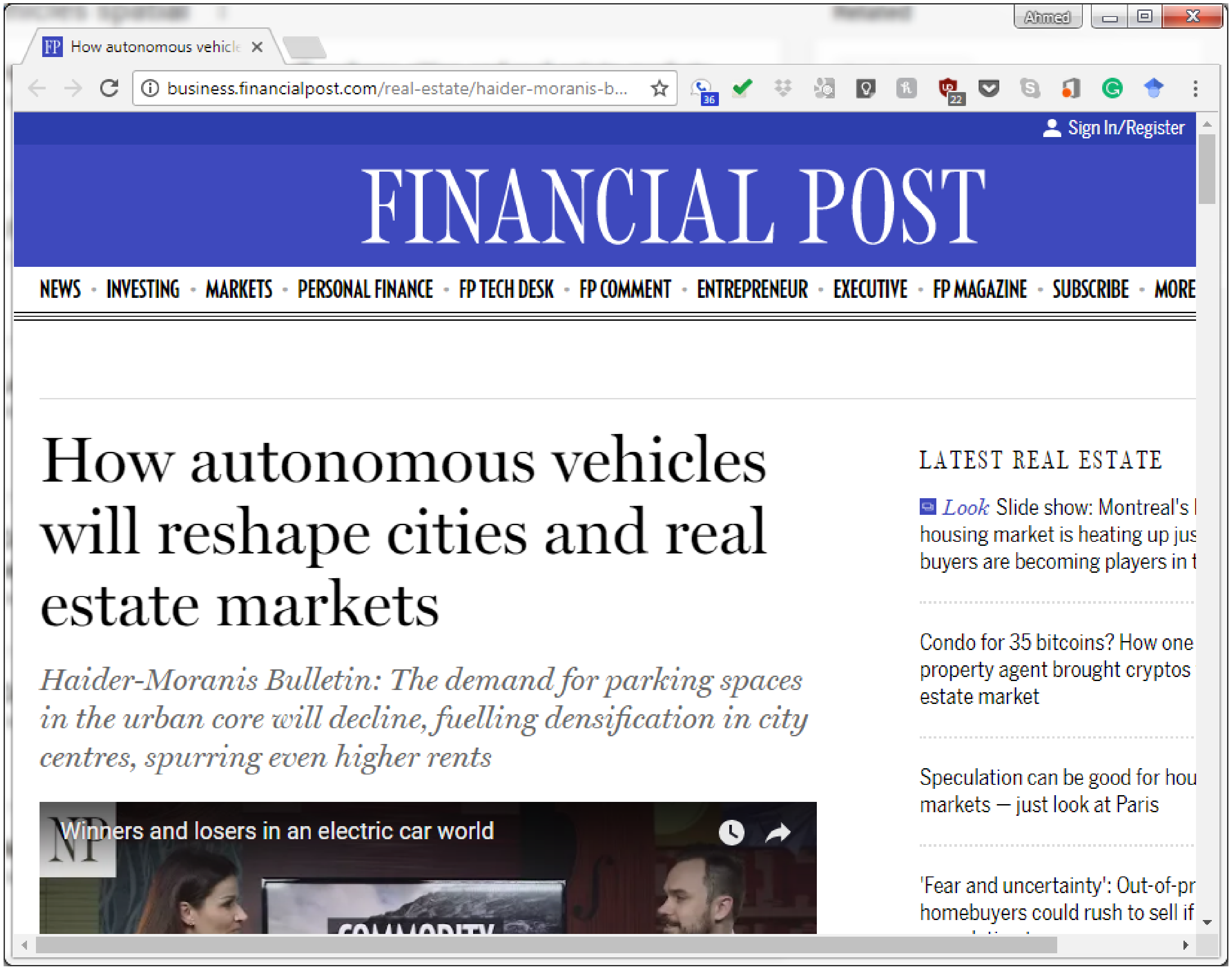
The Rise of Spatial Data



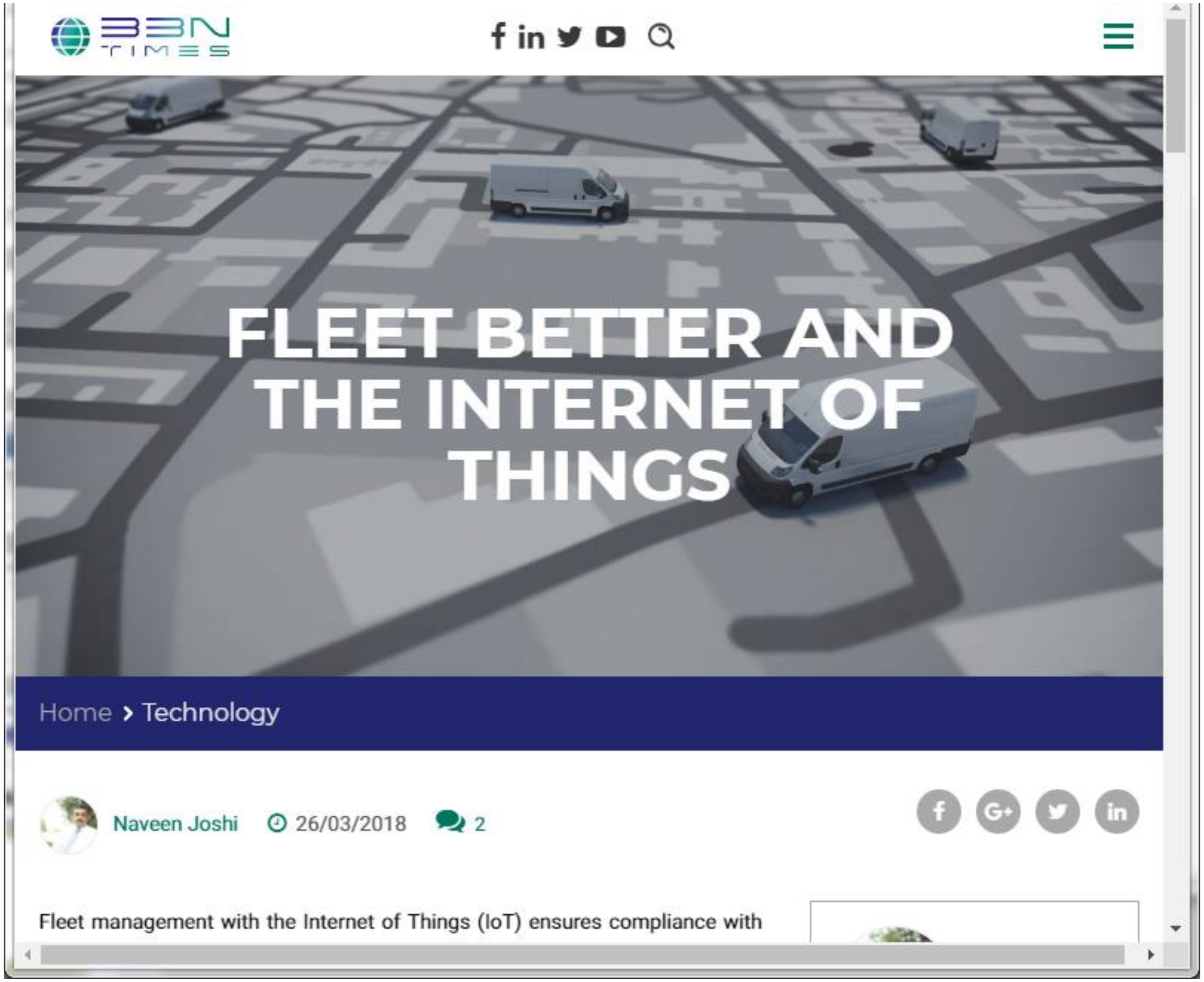
The Rise of Spatial Data



Autonomous Vehicles



Internet of Things (IoT)

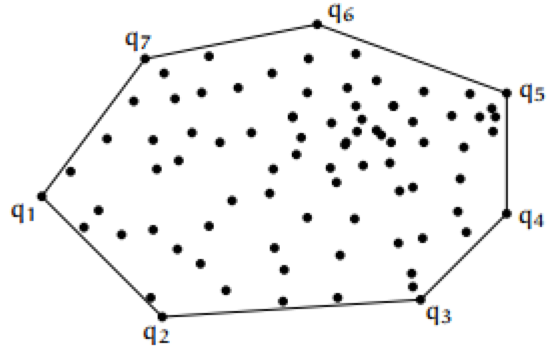


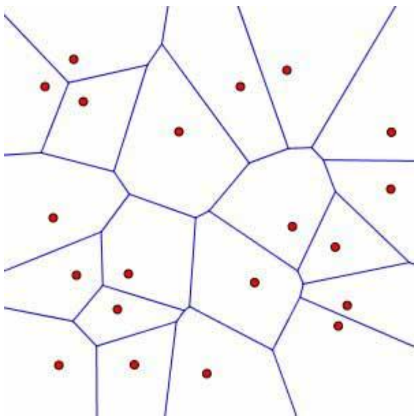
Satellites

Make a contribution
The Guardian
News
Opinion
Sport
Culture
Lifestyle
US World Environment Soccer US politics Business Tech More
Elon Musk's SpaceX gains formal approval for satellite broadband network

Course Overview

- Background on algorithms, floating point calculations, and linear algebra
- Computational geometry primitives
- Convex hull algorithms
- Search problems
- Intersection problems
- Polygon simplification
- Voronoi diagram
- Delaunay triangulation

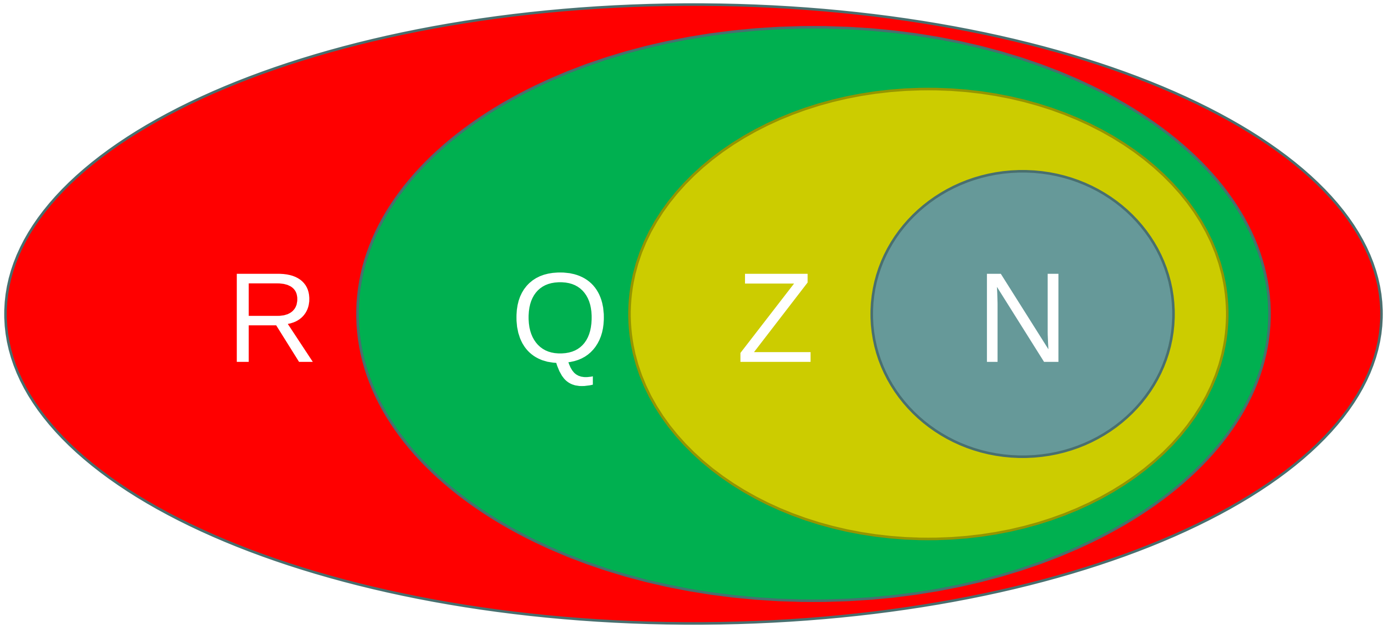




Number Representation

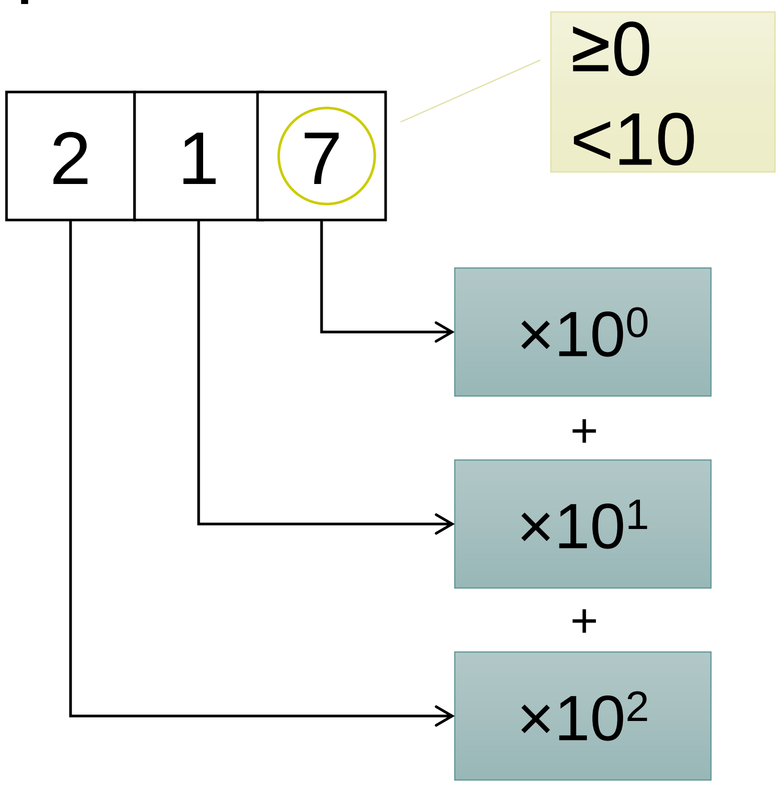
Number Representation

- Number sets



Natural Numbers (N)

- Decimal representation



Binary Representation

- Base-2 representation

1	1	0	1	1	0	0	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

$$(11011001)_2 = (217)_{10}$$

Integer Numbers (Z)

- We use a negative sign
- The computer can only represent 0 or 1 (no signs)
- Sign-magnitude representation (not used)
  - Reserve a bit for sign (0: +ve, 1: -ve)
  - Advantage: Simplicity of representation
  - Drawbacks: Two representations for the zero, and complexity of addition and subtraction operations
- Two's complement (Designer's choice)
  - $-x = \sim x + 1$

Rational Numbers (Q)

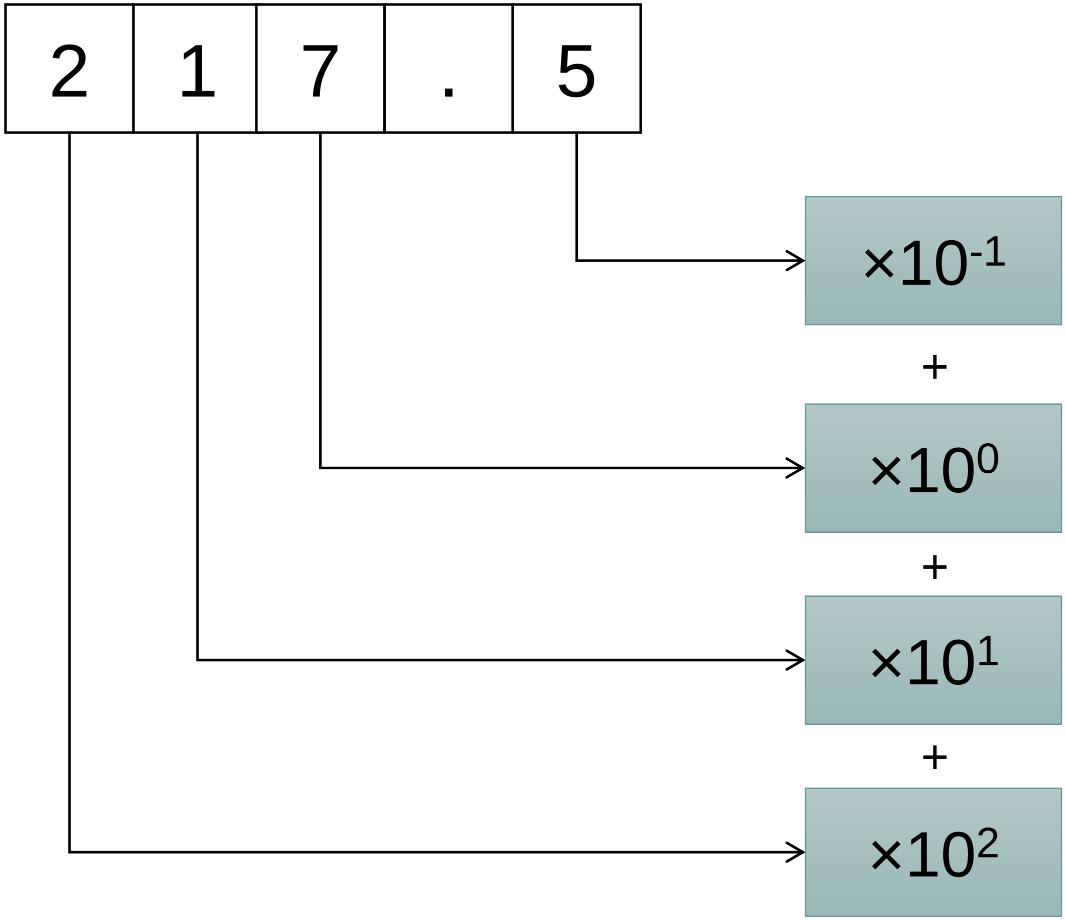
- $q = \frac{a}{b}$
- Where a and b are integers and $b > 0$
- Advantages
  - Simple representation
  - Simple calculations
  - Closed under most operations
  - Can be 100% accurate
- Disadvantages
  - Not closed under certain operations

Operations

- Addition/subtraction: $\frac{a}{b} \pm \frac{c}{d} = \frac{ad \pm bc}{bd}$
- Multiplication: $\frac{a}{b} \cdot \frac{c}{d} = \frac{ac}{bd}$
- Reciprocal: $\left(\frac{a}{b}\right)^{-1} = \frac{b}{a}$
- Division: ${}^{\frac{a}{b}}\big/_{\frac{c}{d}} = \frac{ad}{bc}$
- All the above operations are accurate
- Some might produce +/- ∞ or NaN

Real Numbers (R)

- A decimal (or radix/fraction) point



Fixed-point Representation

- Always assume that the n-right-most digits are after the radix point

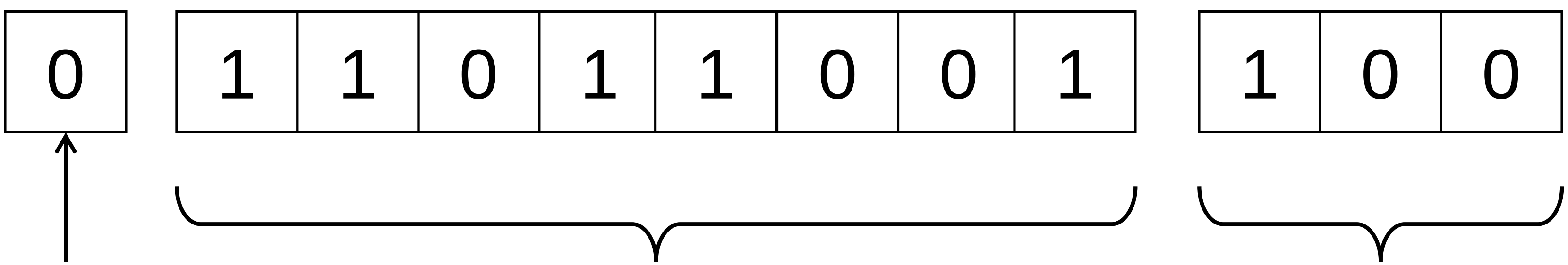
1	1	0	1	1	0	0	1	1	0	0

$=(217.5)_{10}$

- The radix point is *fixed* at that position
- Advantages: Simplicity of representation and +/- operations
- Disadvantages: Cannot represent very large or very small numbers

Floating-point Representation

- The position of the radix point is variable (that point can float around)



Sign bit

All the significant digits
(Mantissa)

Position of the point
(exponent)

$$\text{Value} = (-1)^S \times \text{Mantissa} \times 2^{\text{exponent}}$$

IEEE 754 Standard

Single-precision floating point (32-bits)
1-bit: sign
8-bits: exponent
23-bits: Mantissa

S	E	E	E	E	E	E	E	E	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M

S: 0 for +ve and 1 for –ve numbers

E: 8 bits can represent 256 different exponents
To represent both +ve and –ve exponents, these 8-bits store the exponent plus 127
E=127 indicates an exponent of zero
E=200 indicates an exponent of 200-127=73

Normalization

- If we are not careful, we might end up with redundant representations
  - E.g., $1.5 \times 10^2 = 15 \times 10^1 = 150 \times 10^0$
- In IEEE standard, the fraction point is always placed right next to the first significant (binary) digit
- Since the left-most digit is always one, it is not stored
- This is called normalization

Normalization Examples

- x_1=001011001.110
- $x_1=00\boxed{1.011001110}\times10^{\boxed{110}}$ Exponent

  Mantissa

- x_2=0.0000001110
- $x_2=\boxed{1.11}\times10^{\boxed{-111}}$ Exponent

  Mantissa

32 Floating Point Example

- $x=125.375$
- $x=1111101.011$
- $x=1.111101011\times10^{110}$ (That's 2^6)
- Fraction=111101011
- Fraction=11110101100000000000000
- Exponent=6+127=133=10000101
- Sign=0

0	1	0	0	0	0	1	0	1	1	1	1	1	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Special Cases

- Zero
  - Represented by all zeros in the exponent and fraction
  - Two distinct but equivalent representations: +0 and -0
- Infinity
  - Exponent of all 1’s and fraction of all 0’s
  - Two distinct representations of $+\infty$ and $-\infty$
- Not-a-number (NaN)
  - Exponent is all 1’s and fraction is non-zero

Denormalized Numbers

- $X=0.00001\times10^{-126}$
- Normalized = 1.0×10^{-131}
  - ✘ We cannot represent an exponent of -131
- Exponent=0 (Special marker for denormalized numbers)
- Fraction=00001000000000000000000

Arithmetic

- Multiplication
  - Multiply the signs (XOR)
  - Multiply the mantissas
  - Add up the exponents
- Division
  - Similar to multiplication but can produce infinity or NaN
- Addition/Subtraction
  - Aligns the two mantissas and add/subtract them
  - Adjust the exponent to the result

Summary

- Floating points cannot represent all possible numbers
- It can represent both very small and very large numbers
- Number of significant digits is upper-bounded
- We can represent zero, ∞, and NAN
- The result of any arithmetic operation can produce some error