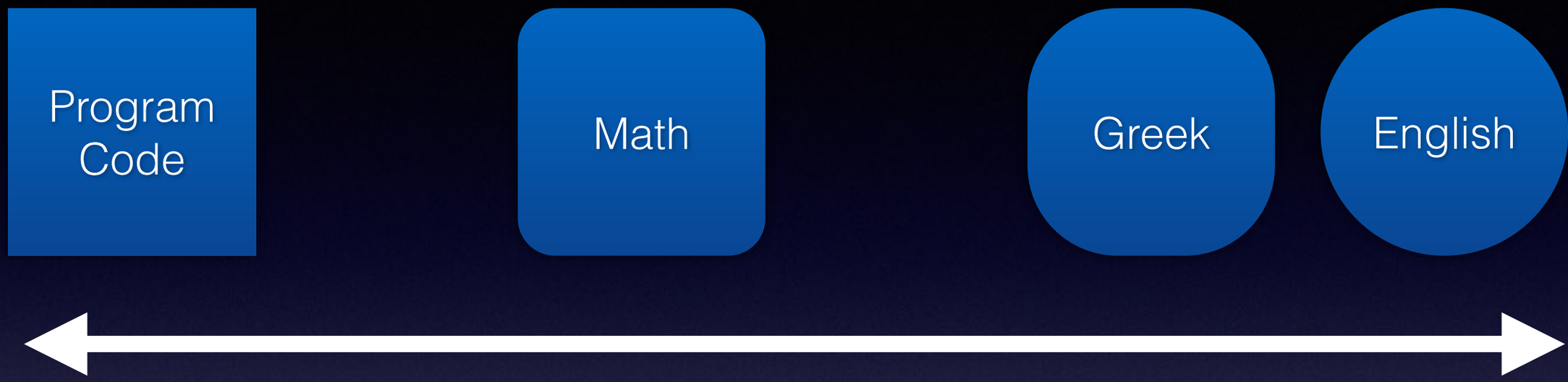
340

algs

JOBS

- Many job interview questions revolve around:
  - Basic OOP programming
  - Basic Algorithm concepts
    - Many from THIS COURSE
- Scenarios (how would you do X ?)
- and stupid pointless BS questions



Code to English

Spectrum

Fibonacci

- A Number sequence people like to refer to
  - if(n = 0) return 0
  - else if(n = 1) return 1
  - else return fib(n-1) + fib(n-2)

$$F_n = \begin{cases} F_{n-1} + F_{n-2} & \text{if } n > 1 \\ 1 & \text{if } n = 1 \\ 0 & \text{if } n = 0\,. \end{cases}$$

Math Translation

- for loop
  - sum = 0;
  - for(i = 1; i < ∞; ++i){
    - sum = sum + 1 / i;
  - }
  - return ∞

$$\sum_{i=1}^{\infty} \frac{1}{i} = \infty.$$

number n

- n is a common variable which means **n**umber

- **n**umber of items

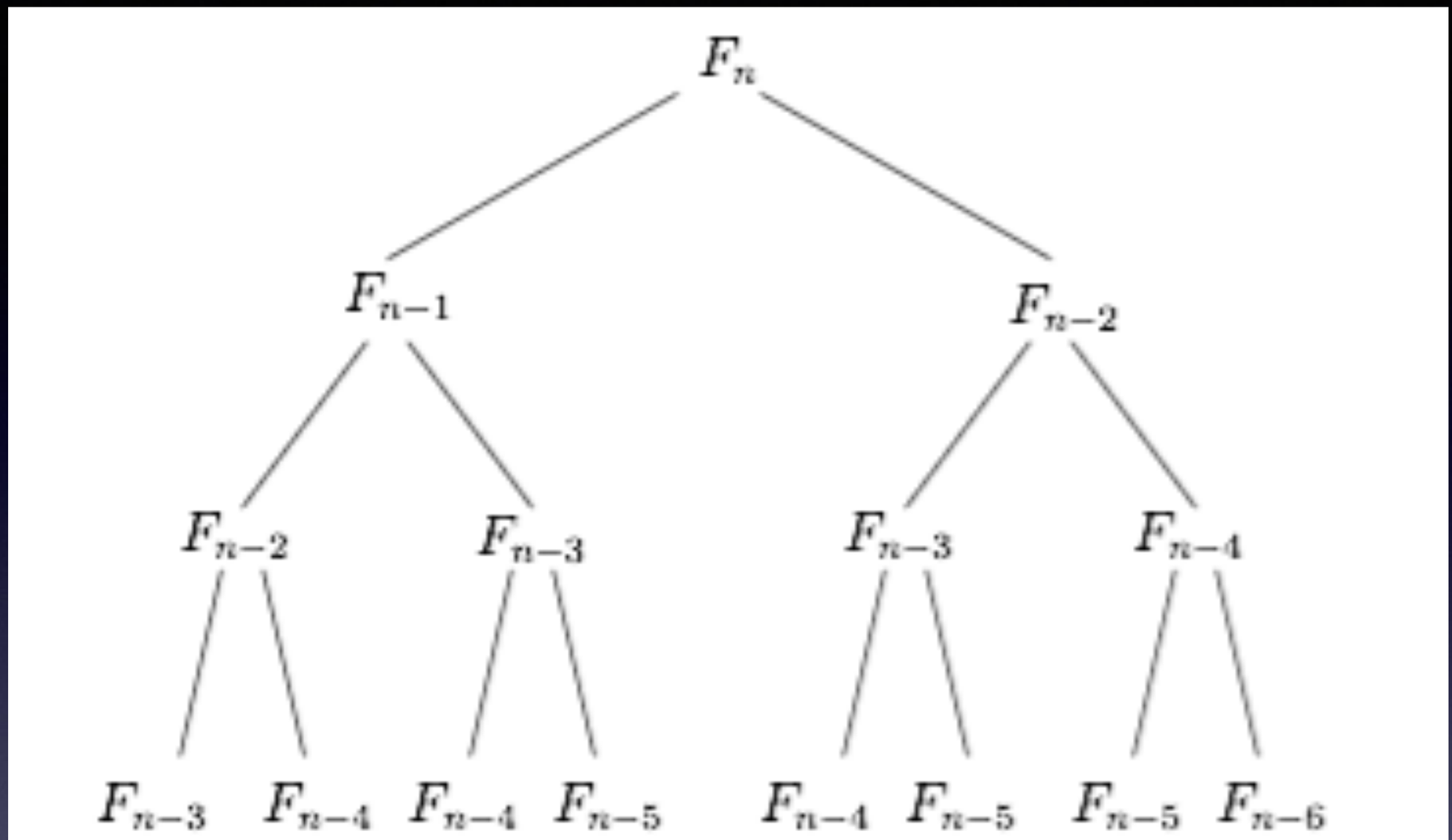
- A linear algorithm takes **n** operations to complete

Fibonacci

- What if you ask for ∞ or better — “n”
  - HOW MUCH WORK WOULD IT TAKE FOR n?
- in code; you can’t. in pseudo code you can
- in math… that is what math does

Speed of Fibonacci

- n=0 or n = 1
  - near instant! done!
- n > 1
  - count “steps” n=0 is one step n=2 is one step
  - n=2 is 3 steps (adding two previous instants)
  - n>2 is 3 + Steps for(n-1) + Steps for (n-2)



exponential growth

don’t worry a whole lot until n becomes gets to 50…
100… then you’ll never calculate it

Big O notation

- A must know concept!
- Job interview question
- The big picture in measuring SPEED of algorithms
- Simple algebra-level math description
- WORST CASE time

SPEED

- My iPod is faster than your PC!
- HOW YOU DO THINGS MATTERS
- BogoSort vs QuickSort
  - The stupid sort (BogoSort) can take (e-1)n!
  - n! is factorial— a HUGE function

Count to 10

- for(i = 1; i<= n; ++i){ // n = 10
  - print line (i);
- }
- Big-O (n) = so it’s linear; graphs as a line
- aka: Order of n

Count

- print line (“1 2 3 4 5 6 7 8 9 10”);
- Big-O (1) = constant speed. 1 operation.
- aka: Order of 1
- Think - string length! Store the length instead of counting it.

Count to 10

- for(i = 1; i<= n; ++i){ // n = 10
  - for(x = 1; x<= n; ++x){ // n = 10
    - print line (i);
    - }
- } // yes this prints 10 of each number it counts
- Big-O (n^2) = exponential
- aka: Order of n squared

Best case Ω

- omega Ω
- For Fibonacci, best case is n=1 or n =0
  - Ω(1) or “omega one”
- Ω is not used a lot because we mostly care about typical or worst case.

Both cases Θ

- Θ Theta (yes more greek)
- not used much but means the Best and Worst are the same.
- Saves some writing adds more messy symbols…

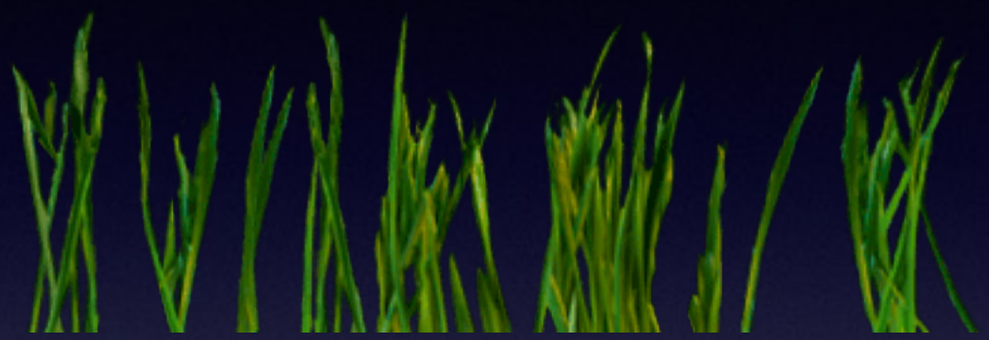
Modulo

- REALLY USEFUL
- % in many languages
- Division remainder
- 130 minutes / 60 = (floor or int) 2 hours
- 130 minutes % 60 = remainder of 10 minutes

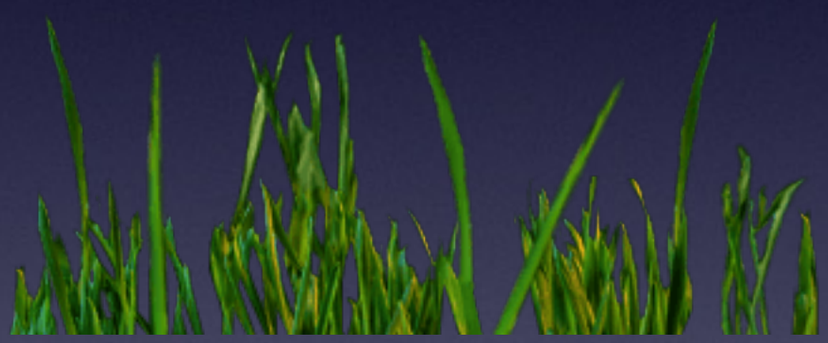
Primes

- Prime numbers are usually useful numbers
- Encryption uses them
- factors: 1,itself
- testing algs..

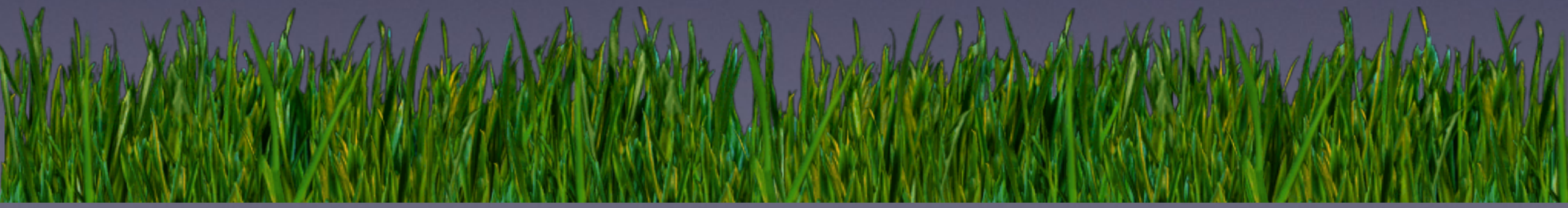
510 = 17 x 30



390 = 13 x 30



330 = 11 x 30



Hashing

- Generate a representative number for something
- used extensively on strings in scripting
- Similar to an ID … techID #, social security # , ip address, barcode
- Many schemes to generate them — not much use if hash #s are duplicated.

Hash techniques

- Security - encryption - MD5 etc
- Speed - simple checksum or character total
- Goal: as few collisions as possible when searching the storage.
- IP addresses: NAT and subnets… 1 hash for a group of people - not as good as individual IPs

Hash Tables

- Best case - Ω(1)
- Worst case - O(n)

- Indexed storage (array) use hash to jump to that spot in the storage.
- Check the spot if wrong data, jump to a predetermined RELATIVE position (like the next spot open spot)
- Worst case, storage is FULL and hashes come out the same

Binary Search

- Divide and conquer
- 0 1 2 3 4 5 6 7 8 9 10