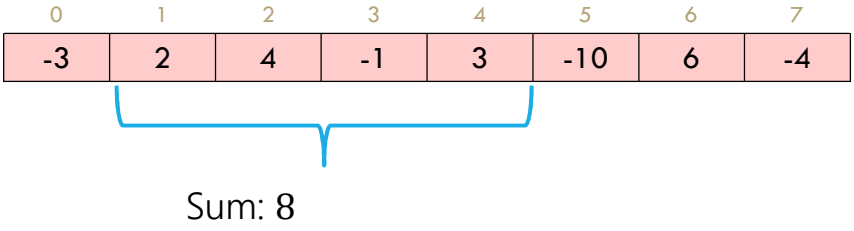
Another divide and conquer

Maximum contiguous subarray sum

Given: an array of integers (positive and negative), find the indices that give the maximum contiguous subarray sum.

Find i, j that maximize $A[i] + A[i+1] + \cdots + A[j]$



Classic Application

Suppose you need to multiply **really** big numbers.

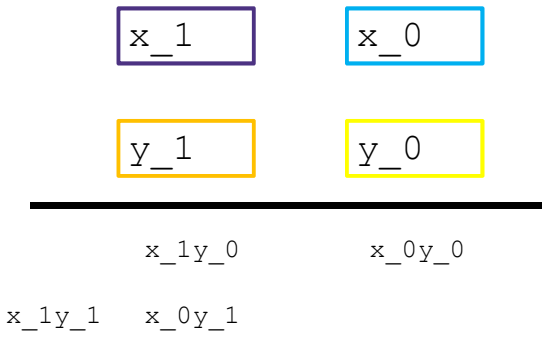
Much bigger than `ints`

Split the n bit numbers in half

Think of them as written in base $2^{n/2}$

What would the "normal" multiplication algorithm do?

4 multiplications, i.e. 4 recursive calls.



Activity

Write a recurrence to describe the running time of this code. What's the big-O?

```
DivConqExponentiation(int a, int b, int n)
  if(n==0) return 1
  if(n==1) return a%b
  int k = divConqExponentiation(a,b,n/2) /*int div*/
  if(n%2==1) return (k*k*a)%b
  return (k*k)%b
```

Go to pollev.com/Robbie so I know how long to explain

Master Theorem

Given a recurrence of the following form, where a, b, c, and d are constants:

$$T(n) = \begin{cases} d & \text{if } n \text{ is at most some constant} \\ aT\left(\frac{n}{b}\right) + f(n) & \text{otherwise} \end{cases}$$

Where $f(n)$ is $\Theta(n^c)$

If $\log_b a < c$ then $T(n) \in \Theta(n^c)$

If $\log_b a = c$ then $T(n) \in \Theta(n^c \log n)$

If $\log_b a > c$ then $T(n) \in \Theta\left(n^{\log_b a}\right)$