test #1, assignment #1: results posted RSN....

CSC236 fall 2018

recursive time complexity

difficult road to laziness...
...we'll get there next week...

Danny Heap
heap@cs.toronto.edu / BA4270 (behind elevators)
http://www.teach.cs.toronto.edu/~heap/236/F18/
416-978-5899

Using Introduction to the Theory of Computation, Chapter 3

binary search

Recursive $T(n)$

correctness.... later
worst-case time complexity

n = len(A[b:e+1]) = e-b+1

x: value to be searched for
A: array
b: beginning index
e: end index

```
def recBinSearch(x, A, b, e) :
  if b == e :
    if x <= A[b] :
      return b
    else :
      return e + 1
  else :
    m = (b + e) // 2 # midpoint
    if x <= A[m] :
      return recBinSearch(x, A, b, m)
    else :
      return recBinSearch(x, A, m+1, e)
```

c1 \in \R^+

c2

c3 = max

sum = c'

$$T(n) = \begin{cases} 1 & \text{if } n = 1 \\ 1 + \max(T(\lceil n/2 \rceil), T(\lfloor n/2 \rfloor)) & \end{cases}$$

c'' .... = 1 (choosing units of c'').

exercise to reader: show that m-b+1 = ceiling(n/2), and e-(m+1)+1 = floor(n/2)
show that T is nondecreasing

guess bound on $T(n)$... by unwinding/substitution

suppose n = 2^k, for some natural number k (bigger than 0 , for now).

```
T(n) = T(2^k) = 1 + T(2^{k-1})
        = 1 + 1 + T(2^{k-2}) = 2 + T(2^{k-2})
        = 3 + T(2^{k-3})
        ...   intuition happens here!
       = k + T(2^{k-k}) = lg(n) + c'
```

conjecture: T \in \Theta(lg)

want to prove T \in \Omega(lg) [then big-Oh later...]

prove lower bound on $T(n)$

Let c = ??? 1. Then c \in \R^+. Let B = ??? 2. Then B \in \R^+.

(complete induction)

Let n be an arbitrary natural number no smaller than B. Assume \forall B <= i < n, T(i) >= c lg(i). I will show that T(n) >= c lg(n).

```
case n >= 3: T(n) = 1 + T(ceiling(n/2))         # since n >= B > 1
                  >= 1 + c lg(ceiling(n/2))    # by IH, since B <= ceiling(n/2) < n,  since n >= 3
                  >= 1 + c lg(n/2)             # since lg nondecreasing
                  = 1 + c(lg(n) - lg(2)) = 1 - c + c lg(n)
                  >= c lg(n)                   # since c <= 1

base case n = 2: Then T(2) = 1 + T(1) = 1 + c' >= c lg(2) = c         # since c = 1
```

try to prove upper bound on $T(n)$

trouble!?!

Let c = ???. Then c \in \R^+. Let B = n???. Then B \in \R^+.

(complete induction)

Let n be an arbitrary natural number no smaller than B. Assume (IH) \forall B <= i < n, T(i) <= c lg(i). I will try to show that T(n) <= c lg(n).

```
    case n ???:  T(n) = 1 + T(ceiling(n/2))      # since n >= B > 1
                      <= 1 + c lg(ceiling(n/2)) # by IH, since B <= ceiling(n/2) < n,  since n > 2
                      <= 1 + c lg((n+1)/2)       # since lg is nondecreasing
                      = 1 + c(lg(n+1) - 1) = 1 - c + c lg(n+1)
                      <= c lg(n).......................darn!
```

strengthen the claim: T(n) <= c lg(n-1)

```
case n ???:  T(n) = 1 + T(ceiling(n/2))      # since n >= B > 1
                  <= 1 + c lg(ceiling(n/2)-1) # by IH, since B <= ceiling(n/2) < n,  since n > 2
                  <= 1 + c lg((n+1)/2 -1)       # since lg is nondecreasing
                  = 1 + c lg((n+1-2)/2) = 1 + c lg((n-1)/2) = 1 + c(lg(n-1) - 1)
                  = 1 - c + c lg(n-1)
                  <= c lg(n-1)                          # since c > = 1

case n = 2: T(2) = 1 + c' <= c lg(1)...........won't work
case n = 3: T(3) = 1 + T(2) = 2 + c' <= c lg(2) = c                    # true since c = 2+c'
```

Notes