CS360 - Assignment 2

2.3-5 *Binary Search*

```
Binary_Search(A, v, low, high)
if low <= high                                  // Still elements to check
     mid = (low+high)/2                         // Get midpoint of remaining elements
     if v = A[mid]
          return mid                            // We found it!
     if v < A[mid]
          return Binary_Search(A, v, low, mid-1)    // Discard lower half
     else
          return Binary_Search(A, v, mid+1, high)   // Discard upper half
return nil                                      // Bummer - element not found!
```

Intuitively we see that the worst case behavior occurs when the element is not in the array. In this case, the recursion will continue to execute cutting the array in half until $\frac{n}{2^k} = 1 \Rightarrow k = \lg n$. Since the function body is $\Theta(1)$, i.e. there are no loops in it, the worst case run time would be $T(n) = (\lg n)\Theta(1) = \Theta(\lg n)$. However this is **not** a *proof* of the run time. Instead, we must construct the recursive equation and solve it via the Master theorem. Since at each pass we cut the number of elements into a *single* half with constant time operations at each pass (simply a comparison and computation of the next midpoint), the recursive equation is given by:
$T(n) = T(n/2) + \Theta(1)$
Applying the master theorem with $a = 1$, $b = 2$ and $f(n) = \Theta(1) = c$ gives $\log_b a = \log_2 1 = 0 \Rightarrow n^{\log_b a} = n^0 = 1$. Therefore we are in **case 2** hence the solution is given by $T(n) = \Theta(n^{\log_b a} \lg n) = \Theta(\lg n)$.

3-2

			O	o	Ω	ω	Θ
a.	$\lg^k n$	n^ϵ	Y	Y	N	N	N
b.	n^k	c^n	Y	Y	N	N	N
c.	$\sqrt{n}$	$n^{\sin(n)}$	N	N	N	N	N
d.	2^n	$2^{n/2}$	N	N	Y	Y	N
e.	$n^{\lg c}$	$c^{\lg n}$	Y	N	Y	N	Y
f.	$\lg(n!)$	$\lg(n^n)$	Y	N	Y	N	Y

a. Since $\lg^k n = o(n^a)$ (*logs* are bounded by *polynomials*) $\Rightarrow \lg^k n = O(n^a)$ (i.e. $< \Rightarrow \leq$)
Also if $\lg^k n = o(n^a) \Rightarrow \lg^k n \neq \omega(n^a)$ (i.e $< \Rightarrow \not>$)
Likewise $\lg^k n = o(n^a) \Rightarrow \lg^k n \neq \Omega(n^a)$ (i.e $< \Rightarrow \not\geq$)
Finally since $\lg^k n \neq \Omega(n^a) \Rightarrow \lg^k n \neq \Theta(n^a)$ (i.e $< \Rightarrow \neq$)

b. Since $n^k = o(c^n)$ (*polynomials* are bounded by *exponentials*)
By similar argument as above $\Rightarrow n^k = O(c^n)$ (i.e. $< \Rightarrow \leq$)
$n^k \neq \omega(c^n)$ (i.e $< \Rightarrow \not>$)

$n^k \neq \Omega(c^n)$ (i.e $< \Rightarrow \ngeq$)
$n^k \neq \Theta(c^n)$ (i.e $< \Rightarrow \neq$)

c. Since $\sin n$ takes on values in the range $[0, 1]$ periodically $\Rightarrow n^{\sin n}$ takes on values $[n^0, n^1] = [1, n]$ periodically However since $1 \leq \sqrt{n} \leq n$ for all n, $\Rightarrow \sqrt{n}$ is *never always* $<$, $>$, or $= n^{\sin n}$ for *all* $n \geq n_0$
$\Rightarrow \sqrt{n}$ is **not** asymptotically bounded by $n^{\sin n}$.

d. Using the limit test for ω on pg. 48, we can easily show that
$\lim_{n\to\infty} \frac{2^n}{2^{n/2}} = \lim_{n\to\infty} 2^{n/2} = \infty$
$\Rightarrow 2^n = \omega(2^{n/2})$
Using a reverse argument as (a) gives $\Rightarrow 2^n = \Omega(2^{n/2})$ (i.e. $> \Rightarrow \geq$)
$2^n \neq o(2^{n/2})$ (i.e $> \Rightarrow \nless$)
$2^n \neq O(2^{n/2})$ (i.e $> \Rightarrow \nleq$)
$2^n \neq \Theta(2^{n/2})$ (i.e $> \Rightarrow \neq$)

e. By equation 3.15, $n^{\lg c} = c^{\lg n}$, so by the reflexive property
$n^{\lg c} = \Theta(c^{\lg n})$ $n^{\lg c} = \Omega(c^{\lg n})$ $n^{\lg c} = O(c^{\lg n})$
Since $n^{\lg c} = \Theta(c^{\lg n})$
$\Rightarrow n^{\lg c} \neq o(c^{lgn})$ (i.e. $= \Rightarrow \nless$) and $n^{\lg c} \neq \omega(c^{lgn})$ (i.e. $= \Rightarrow \ngtr$)

f. Since $\lg(n^n) = n \lg n$ from equation 3.18 $\Rightarrow \lg(n!) = \Theta(n \lg n)$ which by the theorem gives $\lg(n!) = O(n \lg n)$ $(= \Rightarrow \leq)$ and $\lg(n!) = \Omega(n \lg n)$ $(= \Rightarrow \geq)$
Similarly to part (e) $\lg(n!) \neq o(n \lg n)$ $(= \Rightarrow \nless)$ and $\lg(n!) \neq \omega(n \lg n)$ $(= \Rightarrow \ngtr)$

4-1 b $T(n) = T(7n/10) + n$ choose $a = 1$ $b = 10/7$ $f(n) = n$
Then $log_b a = log_{10/7} 1 = 0 \Rightarrow n^{log_b a} = n^0 = 1$
$f(n) = n = \Omega(n^{0+\epsilon})$ for $\epsilon \leq 1$ so *possibly* **case 3**. Checking regularity
$1(\frac{n}{(10/7)}) = \frac{7}{10} n \leq cn$ for $\frac{7}{10} \leq c < 1$ so choose $c = \frac{4}{5}$ (or any other value in the given range.)
$\Rightarrow T(n) = \Theta(n)$

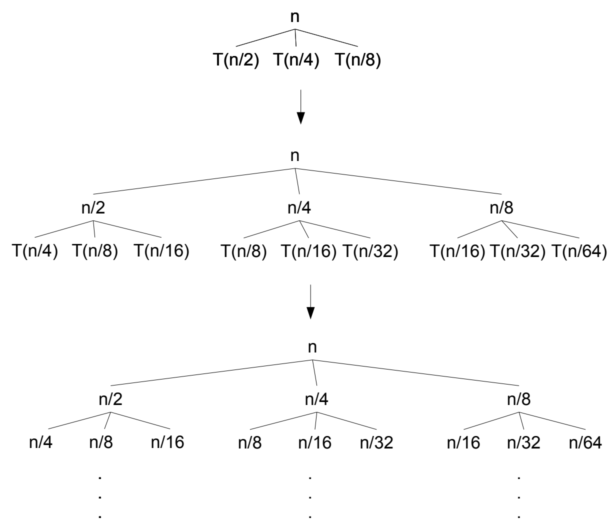
c. $T(n) = 16T(n/4) + n^2$ choose $a = 16$ $b = 4$ $f(n) = n^2$
Then $log_b a = log_4 16 = 2 \Rightarrow n^{log_b a} = n^2$
$f(n) = n^2 = \Theta(n^2)$ so **case 2**.
$\Rightarrow T(n) = \Theta(n^2 \lg n)$

d. $T(n) = 7T(n/3) + n^2$ choose $a = 7$ $b = 3$ $f(n) = n^2$
Then $log_b a = log_3 7 = 1.77 \Rightarrow n^{log_b a} = n^{1.77}$
$f(n) = n^2 = \Omega(n^{1.77+\epsilon})$ for $\epsilon \leq .23$ so *possibly* **case 3**. Checking regularity
$7(\frac{n}{3})^2 = \frac{7}{9} n^2 \leq cn^2$ for $\frac{7}{9} \leq c < 1$
$\Rightarrow T(n) = \Theta(n^2)$

e. $T(n) = 7(n/2) + n^2$ choose $a = 7$ $b = 2$ $f(n) = n^2$
Then $log_b a = log_2 7 = 2.81 \Rightarrow n^{log_b a} = n^{2.81}$
$f(n) = n^2 = O(n^{2.81-\epsilon})$ for $\epsilon \leq .81$ so **case 1**.
$\Rightarrow T(n) = \Theta(n^{2.81})$

4-3f $T(n) = T(n/2) + T(n/4) + T(n/8) + n$

This recurrence cannot be solved using the master theorem so construct a recursion tree



Level 0 $= n$
Level 1 $= (n/2) + (n/4) + (n/8) = (7/8)n$
Level 2 $= (n/4) + 2(n/8) + 3(n/16) + 2(n/32) + (n/64) = 49n/64 = (7/8)^2 n$
...
Level $i = (7/8)^i n$
The total number of levels will be due to the T(n/2) term giving a maximum $\lg n$ levels. Hence the runtime will be

$$\begin{aligned}
T(n) &\leq \sum_{i=0}^{\lg n} \left(\frac{7}{8}\right)^i n \\
&\leq n \sum_{i=0}^{\infty} \left(\frac{7}{8}\right)^i \\
&\leq n \left(\frac{1}{1-\frac{7}{8}}\right) \\
&\leq 8n = O(n)
\end{aligned}$$

Hence we *guess* that $T(n) = O(n)$ and *verify* using the substitution method.

Lower Bound
Clearly $T(n) = T(n/2) + T(n/4) + T(n/8) + n \geq cn$ for $c = 1$.
$\Rightarrow T(n) = \Omega(n)$

Upper Bound
Assume $T(n) \leq cn$

$$\begin{aligned}
\Rightarrow T(n) &\leq c(n/2) + c(n/4) + c(n/8) + n \\
&\leq \frac{7}{8}cn + n \\
&\leq \left(1 + \frac{7}{8}c\right)n \\
&\leq cn \quad \text{for } c \geq 8
\end{aligned}$$

$\Rightarrow T(n) = O(n)$

Hence, since $T(n) = \Omega(n)$ **and** $T(n) = O(n)$ gives $\Rightarrow T(n) = \Theta(n)$.