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:

1. $f(n) = \Theta(1)$ hence we convert the asymptotic bound to $f(n) = c$
2. For this equation $a = 1$, $b = 2$, $f(n) = c$
3. Computing $n^{\log_b a} = n^{\log_2 1} = n^0 = 1$.
4. We see that $f(n) = c$ "$\approx$" 1 so we try *Case 2* where $f(n) = c = \Theta(1) \Rightarrow$ **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 $<\Rightarrow\not\geq$)
$n^k \neq \Theta(c^n)$ (i.e $<\Rightarrow\neq$)

c. Since $\sin n$ takes on values in the range $[-1, 1]$ periodically $\Rightarrow n^{\sin n}$ takes on values $[n^{-1}, n^1] = [1/n, n]$ periodically However since $1/n \leq \sqrt{n} \leq n$ 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 $>\Rightarrow\not<$)
$2^n \neq O(2^{n/2})$ (i.e $>\Rightarrow\not\leq$)
$2^n \neq \Theta(2^{n/2})$ (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. $=\Rightarrow\not<$) and $n^{\lg c} \neq \omega(c^{lgn})$ (i.e. $=\Rightarrow\not>$)

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) $\lg(n!) \neq o(n \lg n)$ ($=\Rightarrow\not<$) and $\lg(n!) \neq \omega(n \lg n)$ ($=\Rightarrow\not>$)

4-1 b. $T(n) = T(7n/10) + n$

Applying the master theorem:

1. $f(n) = n$ does not contain any asymptotic bounds that need conversion
2. For this equation $a = 1$, $b = 10/7$, $f(n) = n$
3. Computing $n^{\log_b a} = n^{\log_{10/7} 1} = n^0 = 1$.
4. We see that $f(n) = n$ ">" 1 so we try *Case 3* where $f(n) = n = \Omega(n^{0+\epsilon})$ for $\epsilon \leq 1$. Let $\epsilon = 0.5$ (or any other value in the given range) $\Rightarrow n = \Omega(n^{(0+.5)}) = \Omega(n^{0.5})$, then *possibly* **case 3**.

   Checking regularity

$$\begin{aligned} af(n/b) &\leq cf(n) \\ 1\frac{n}{(10/7)} &\leq cn \\ \frac{7}{10}n &\leq cn \\ \Rightarrow \frac{7}{10} &\leq c < 1 \end{aligned}$$

   So select any value of c in this range, e.g. let $c = 4/5$, satisfies regularity.

Hence the solution is given by $T(n) = \Theta(f(n)) = \Theta(n)$.

c. $T(n) = 16T(n/4) + n^2$

Applying the master theorem:

1. $f(n) = n^2$ does not contain any asymptotic bounds that need conversion
2. For this equation $a = 16$, $b = 4$, $f(n) = n^2$
3. Computing $n^{\log_b a} = n^{\log_4 16} = n^2$.
4. We see that $f(n) = n^2$ "$\approx$" n^2 so we try *Case 2* where $f(n) = n^2 = \Theta(n^2)$ $\Rightarrow$ **case 2**.

Hence the solution is given by $T(n) = \Theta(n^{log_b a} \lg n) = \Theta(n^2 \lg n)$.

d. $T(n) = 7T(n/3) + n^2$

Applying the master theorem:

1. $f(n) = n^2$ does not contain any asymptotic bounds that need conversion
2. For this equation $a = 7$, $b = 3$, $f(n) = n^2$

3. Computing $n^{\log_b a} = n^{\log_3 7} = n^{1.77}$.

4. We see that $f(n) = n^2$ ">" $n^{1.77}$ so we try *Case 3* where $f(n) = n^2 = \Omega(n^{1.77+\epsilon})$ for $\epsilon \leq 0.23$. Let $\epsilon = 0.1$ (or any other value in the given range) $\Rightarrow n^2 = \Omega(n^{(1.77+.1)}) = \Omega(n^{1.87})$, then *possibly* **case 3**.

   Checking regularity

$$\begin{aligned} af(n/b) &\leq cf(n) \\ 7(\frac{n}{3})^2 &\leq cn^2 \\ \frac{7}{9}n^2 &\leq cn^2 \\ \Rightarrow \frac{7}{9} &\leq c < 1 \end{aligned}$$

   So select any value of c in this range, e.g. let $c = 8/9$, satisfies regularity.

Hence the solution is given by $T(n) = \Theta(f(n)) = \Theta(n^2)$.

e. $T(n) = 7T(n/2) + n^2$

Applying the master theorem:

1. $f(n) = n^2$ does not contain any asymptotic bounds that need conversion

2. For this equation $a = 7$, $b = 2$, $f(n) = n^2$

3. Computing $n^{\log_b a} = n^{\log_2 7} = n^{2.81}$.

4. We see that $f(n) = n^2$ "<" $n^{2.81}$ so we try *Case 1* where $f(n) = n^2 = O(n^{2.81-\epsilon})$ for $\epsilon \leq .81$. Let $\epsilon = 0.5$ (or any other value in the given range) $\Rightarrow n^2 = O(n^{(2.81-.5)}) = O(n^{2.31})$ so **case 1**.

Hence the solution is given by $T(n) = \Theta(n^{log_b a}) = \Theta(n^{2.81})$.

4-3a. $T(n) = 4T(n/3) + n \lg n$

Applying the master theorem:

1. $f(n) = n \lg n$ does not contain any asymptotic bounds that need conversion

2. For this equation $a = 4$, $b = 3$, $f(n) = n \lg n$

3. Computing $n^{\log_b a} = n^{\log_3 4} = n^{1.26}$.

4. We see that $f(n) = n \lg n$ "<" $n^{1.26}$ so we try *Case 1* where $f(n) = n \lg n = O(n^{1.26-\epsilon})$ for $\epsilon \leq .26$ (since *all* logs are upper bounded by *all* polynomials, *any* polynomial greater than n will upper bound $n \lg n$). Let $\epsilon = 0.1$ (or any other value in the given range) $\Rightarrow n \lg n = O(n^{(1.26-.1)}) = O(n^{1.16})$ so **case 1**.

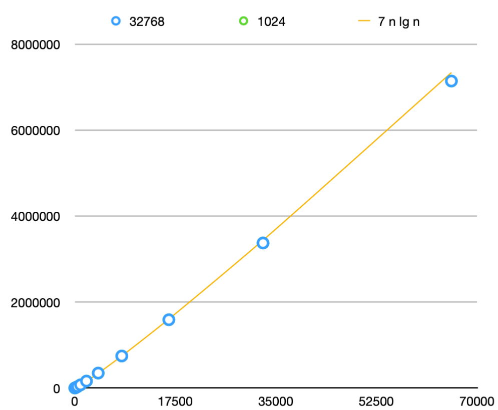
Hence the solution is given by $T(n) = \Theta(n^{log_b a}) = \Theta(n^{1.26})$.

Implementation:

Is there a table showing the empirical data for both element ranges (but it should *not* contain the calculated trend values)?

Is there a graph with:

1. The empirical data for both element ranges plotted as **only** *points*
2. The calculated trend for *each* element range plotted as **only** a *curve*
3. A legend listing each data set and showing the trend curve function with the approximated *constant*
4. Properly labelled graph axes



Constants

- **MergeSort:** $1024 \approx 7n \lg n$, $32768 \approx 7n \lg n$