CSE 321 HW1

Yavuz Selim İkizler 1901042617

$f(n) = O(g(n)) \Rightarrow \lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$

$f(n) = \Omega(g(n)) \Rightarrow \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$

$f(n) = \Theta(g(n)) \Rightarrow 0 < \lim_{n\to\infty} \frac{f(n)}{g(n)} < \infty$

| | $f(n)$ | $g(n)$ | |
|---|---|---|---|
| a) | $2^n$ | $2^{2n}$ | $\lim_{n\to\infty} \frac{2^n}{2^{2n}} = 2^{-1} = \frac{1}{2}$ iso $f(n) \in \Theta(g(n))$ |
| b) | $n^2$ | $n^3$ | $\lim_{n\to\infty} \frac{n^2}{n^3} = \frac{2n}{3n^2} = \frac{2}{6n} = \frac{1}{\infty} = 0$ $f(n) \in O(g(n))$ |
| c) | $3n+1$ | $2n-5$ | $\lim_{n\to\infty} \frac{3n+1}{2n-5} = \frac{3}{2} = f(n) \in \Theta(g(n))$ |
| d) | $4n^2$ | $n^2$ | $\lim_{n\to\infty} \frac{4n^2}{n^2} = 4 = f(n) \in \Theta(g(n))$ |
| e) | $\log_2(n)$ | $\log_{10}(n)$ | $\lim_{n\to\infty} = \frac{\log_2(n)}{\log_{10}(n)} = \frac{\frac{1}{n \cdot \ln 2}}{\frac{1}{n \cdot \ln 10}} = \frac{\ln 10}{\ln 2}$ $\log_2 10$ $f(n) \in \Theta(g(n))$ |
| f) | $2^n$ | $3^n$ | $\lim_{n\to\infty} \frac{2^n}{3^n} = \frac{2}{3} \Rightarrow f(n) \in \Theta(g(n))$ |
| g) | $n^3$ | $1000n^2$ | $\lim_{n\to\infty} \frac{n^3}{1000n^2} = \frac{3n^2}{2000n} = \frac{6n}{2000} = \infty$ $f(n) \in \Omega(g(n))$ |
| h) | $5n+4$ | $2n+2$ | $\lim_{n\to\infty} \frac{5n+4}{2n+2} = \frac{5}{2}$ $f(n) \in \Theta(g(n))$ |
| i) | $\sqrt{n}$ | $\log_2(n)$ | $\lim_{n\to\infty} \frac{\sqrt{n}}{\log_2 n} = \frac{\frac{1}{2\sqrt{n}}}{\frac{1}{n \ln 2}}$ |
| j) | $2^n$ | $2^{n+1}$ | $\lim_{n\to\infty} \frac{2^n}{2^{n+1}} = \frac{1}{2}$ $f(n) \in \Theta(g(n))$ |

2-) $1/2n < \log n < \sqrt{n+5} < n+1 < n^2/\log(n) < 2^n < 10^n < n! < n^{2n}$

$O(1/n)$ $O(\log n)$ $O(\sqrt{n})$ $O(n^2\log(n))$ $O(10^n)$ $O(2^n)$ $O(n!)$ $O(n^{2n})$

in general rule

$O(1) \subseteq O(\log\log n) \subset O(\log n) \subset O(n^a) \subset O(n) \subseteq O(n\log n) \subset O(n^a)$
$a<1$ $\quad$ $a>1$

$\subset O(a^n) \subset O(n \cdot a^n) \subset O(n!) \subset O(a^{n^s})$

$O(1/2n) = O(1/n)$
for $n \geq 1$

$\lim_{n\to\infty} \frac{\frac{1}{n}}{1} = 0$

$\lim_{n\to\infty} \frac{10^n}{2^n} = 5^n = \infty$

$2^n \subseteq 10^n$

So, $O(\frac{1}{n}) \subseteq O(1) \subseteq O(\log n) \subseteq O(\sqrt{n}) \subseteq O(n) \subseteq O(n\log n) \subseteq O(n^2\log n) \subseteq O(a^n) \subseteq O(n!)$
$\subseteq O(n^{a n})$

$O(\sqrt{n+5}) = O(\sqrt{n})$
$\sqrt{n} = n^{\frac{1}{2}}$
$\frac{1}{2} < a$

$\lim_{n\to\infty} \frac{n\log n}{n^2\log n} = \frac{1}{n} = \infty$

4-)
```
i=2
while i<=n:
    if i%2 != 0      O(1)
        i=i-1        O(1)
    else:
        i=i*i        O(1)
        i=i+1        O(1)
    print(i)
```

$\}(O(n))$ $\quad$ $\}O(\log n)$

step: 1 $\quad$ else
$2\%2 = 0$
$i = 2 * 2$
$i = i+1$
$i = 5$

step2 $\quad$ if
$i\%2 \neq 0$
since $i=5$
so
$i = i-1$
$i = 4$

step3: $\quad$ else
$i\%2 = 0$
since $i=4$
$i = i*i$
$i = i+1$
$i = 17$

step 4 $\quad$ if
$i\%2 \neq 0$
since $i=17$
$i = i-1$
$i = 16$

%50 percentage of step3 inside the loop get into if statement and %50 percentage get into else so,

$ATC = P_1 \cdot T_1 + P_2 \cdot T_2$

$O(\frac{1}{2} \cdot \log(n) + \frac{1}{2} \cdot O(n))$

$O\left(\frac{\log n + O(n)}{2}\right) = O(n)$

5-) The algorithm can work like this. Search the array one by one then if its find the even element it returns otherwise continues,

$$T(n) = \sum_{i=1}^{3} P_i \times T(n) \Rightarrow P_1 \cdot O(1) + P_2\, O(1) + P_3\, O(n)$$

$P_1 = 0{,}2 \times O(1)$

$P_3 = (0{,}8)^n \cdot c$

$P_2 = 1 - P_1 - P_3$

$\cong 0{,}8$

$\lim_{n \to \infty} (0{,}8)^n \cdot O(n)$

$0 \cdot O(n)$

$= 0$

$$T(n) = 0{,}2 \cdot O(1) + 0{,}8\, O(1) + 0 \cdot O(n)$$

$$= O(1)$$

3-) Time complexity analysis given both on paper and on the py code

a-) adding items to the list by using inorder traversal takes $O(n)$ time,

merging the two bst list takes $O(n+m) \Rightarrow O(n)$ time

Creating a node from the list items takes $O(n)$ time

$$O(n) + O(m) + O(n+m) + O(n) \Rightarrow O(n)$$

b-) used inorder score algorithm to achive smallest element on binary search tree

visits all the nodes after even it finds the correct data, so it takes $O(n)$ time,

d-) Search all the nodes from bst and compare if its between low and high variable,