Q1) a) $\lim_{n\to\infty} \dfrac{(n^2-3n)^2}{5n^3+n} = \lim_{n\to\infty} \dfrac{n^4-6n^3+9n^2}{5n^3+n}$) divide n^3

$= \lim_{n\to\infty} \dfrac{n-6+9/n}{5+1/n} = \dfrac{\infty - 6 + \cancel{\frac{9}{\infty}}\, 0}{5+\cancel{\frac{1}{\infty}}\, 0} = \dfrac{\infty}{5} = \infty$

Therefore, $f(n) \in \Omega(g(n))$

---

b) $\lim_{n\to\infty} \dfrac{n^3}{\log_2 n^4} = \lim_{n\to\infty} \dfrac{n^3}{4\log_2 n} = \dfrac{\infty}{\infty}$

$= \lim_{n\to\infty} \dfrac{n^3}{4 \cdot \frac{\ln n}{\ln 2}} = \lim_{n\to\infty} \dfrac{n^3 \cdot \ln 2}{4 \cdot \ln(n)}$

$= \dfrac{\ln 2}{4} \lim_{n\to\infty} \dfrac{n^3}{\ln(n)}$) take derivatives $\to \dfrac{\ln 2}{4} \lim_{n\to\infty} \dfrac{3n^2}{\frac{1}{n}}$

$= \dfrac{\ln 2}{4} \cdot \lim_{n\to\infty} 3n^3 = \infty$

so, $f(n) \in \Omega(g(n))$

---

c) $\lim_{n\to\infty} \dfrac{5n\log_2(4n)}{n \cdot \log_2(5^n)} = 5 \lim_{n\to\infty} \dfrac{n \cdot \log_2(4n)}{n \cdot \log_2 5^n} = 5 \lim_{n\to\infty} \dfrac{\cancel{n} \cdot (\log_2 4)(\log_2 n)}{n^{\cancel{2}} \log_2 5}$

$= 5 \lim_{n\to\infty} \dfrac{2+\log_2 n}{n(\log_2 5)} = \dfrac{5}{\log_2 5}\left(\underbrace{\lim_{n\to\infty} \dfrac{2}{n}}_{0} + \lim_{n\to\infty} \dfrac{\log_2 n}{n}\right)$ $\frac{2}{\infty}=0$

$= \dfrac{5}{\log_2 5} \lim_{n\to\infty} \dfrac{\log_2 n}{n} \Rightarrow \dfrac{5}{\log_2 5} \lim_{n\to\infty} \dfrac{1}{\frac{n}{\log_2 n}} = \dfrac{5}{\log_2 5} \lim_{n\to\infty} \dfrac{1}{\frac{1}{\frac{1}{n\ln 2}}}$

$= \dfrac{5}{\log_2 5} \lim_{n\to\infty} \dfrac{1}{n\ln 2} = 0$ $\quad \frac{1}{\infty} = 0$

So, $f(n) = O(g(n))$

Scanned with CamScanner

d) $\lim_{n\to\infty} \frac{n^n}{10^n} = \lim_{n\to\infty} \left(\frac{n}{10}\right)^n = \lim_{n\to\infty} e^{\ln(n/10)^n}$

$= \lim_{n\to\infty} e^{n \cdot \ln\left(\frac{n}{10}\right)} = \lim_{n\to\infty} e^{\infty} = \infty$

so, $f(n) = \Omega(g(n))$

e) $\lim_{n\to\infty} \frac{8n \cdot \sqrt[5]{2n}}{n \sqrt[3]{n}} = \lim_{n\to\infty} \frac{8n \cdot (2n)^{1/5}}{n \cdot (n)^{1/3}} = \frac{16 n^{\frac{18}{15}}}{n^{\frac{20}{15}}}$

$= \lim_{n\to\infty} 16/n^{2/5} = \frac{16}{\infty} = 0$

so, $f(n) \in O(g(n))$

Q(2) a) $i=0$, $i=1$ - - - $i=n-1$

~~each time~~ c c c ... c

total time $\underbrace{c + c + \cdots + c}_{n \text{ times.}} = n \cdot c$ is constant but n is not. So, c will be disregarded

$\Theta(n)$

---

b) $i=0$ $i=1$ . . . $i=n-1$

each time (because of method a) $\underbrace{n.c + nc + \cdots + nc}_{n \text{ times}} \Big\} (n.c) \cdot n = n^2 . c$ c is constant but n is not. so, c will be disregarded.

first loop in method B is $\Theta(n^2 . c) = \Theta(n^2)$

second loop $j=0$ $j=1$ $j=n-1$

$\underbrace{c \quad c \quad c}_{n \text{ times}} = c\,n.$ so $\Theta(n)$

$\Theta(n^2 + n) = \Theta(n^2)$

Q2) c)

$i=0$ $\underbrace{j=0 \;\ldots\; j=n-1}_{n\text{ time}}$ $i=1$ $\underbrace{j=0 \;\ldots\; j=n-1}_{n\text{ time}}$ ... $i=n-1$ j also n time

$\underbrace{n^2+n^2\ldots}_{n^2}$ $\underbrace{n^2+n^2\ldots}_{n^2}$

$$\underbrace{n^2 + n^2 + \ldots + n^2}_{n \text{ times}}$$

n times because of i starts n times (outer loop)

$O(n^4)$

---

d)

```
for ( int i=0 ; i< str_array.length ; i++)
{
    System.out.println ( str_array [i] );   -> c1 times   O(1)
    str_array [i--] = " ";                  -> c2 time    O(1)
}
```

$O(1+1) = O(2) = O(1)$

total time / $\underbrace{\underset{O(1)}{i=0} \quad \underset{O(1)}{i=1} \quad \ldots \quad \underset{O(1)}{i=n-1}}_{n\text{ time}}$

$O(1 \cdot n) = O(n)$ because n is not constant. (n is length of the array)

---

e) inside of the loop is $O(1)$. It's constant time.

but when total time / $\underbrace{\underset{c}{i=0} \;+\; \underset{c}{i=1} \;+\; \ldots \;+\; \underset{c}{i=n-1}}_{n\text{ time}} = n \cdot c$ c is constant. It will be reported

so $O(n)$

Q3) a) Assuming the array is sorted in ascending order.

__input__:

array $A = [a_0, a_1 \ldots a_{n-1}]$ (integer array

n //length of array.

maximum-difference = 0 //initially 0 //the maximum difference is assigned to this variable

__output__

maximum_difference

__step 1__

//because of ascending order, //maximum difference is finded by //substracting last element - first element

maximum difference $\leftarrow a_{n-1} - a_0$.

return maximum difference. //returning the //maximum diffe- //rence

b) Assuming the array is not sorted.

__input__

array $A = [a_0, a_1, \ldots a_{n-1}]$ //integer array

n //length of the array.

maximum-difference = 0.

__output__

maximum-difference. //maximum difference stored in this variable

step 1

max_element $\leftarrow a_0$ //first index assigned to max element

min-element $\leftarrow a_0$ //first index assigned to min element to compare the others

step 2

$i \leftarrow 1$

while $i < n$ (loop for comparing elements)

if (a_i > max_element)

max-element $\leftarrow a_i$

if (a_i < min-element)

min-element $\leftarrow a_i$

$i \leftarrow i+1$ //increment the i

maximum-difference $\leftarrow$ max-element - min-element

return maximum-difference.

// if a_i is bigger than max_element, then a_i assigned to max-element

// if a_i is smaller than min-element, then a_i assigned to min-element

//To find maximum difference substracting min element from max-element

In a)

initialize array

initialize maximum_difference

then last element of array substracted by first element of array.

These are occurred in constant time. 3 operation

$O(1+1+1) = O(3) = O(1)$

b) first initialize the variables

then loop is run.

$i \leftarrow 1$ $\quad i = 2$ $\quad$ ...... $\quad i = n-1$

total time / c time $\quad$ c time $\quad$ c time

$\underbrace{\qquad\qquad\qquad\qquad}_{n-1 \text{ time (because } i \text{ starts from } 1)}$

$(n-1)c = cn - c$ $\quad$ c is constant, so c is disregarded

$= \underline{O(n)}$

Scanned with CamScanner