Ass01.

1. $f(n) = 100n^3 + n^2$ and $g(n) = 5^n$ are increasing positive function for all $n > 0$.

~~we use~~

For $f(n) = 100n^3 + n^2$ and $g(n) = 5^n$, we use L'Hôpital's Rule

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{f'(n)}{g'(n)} = \lim_{n\to\infty} \frac{300n^2+2n}{5^n \ln 5} = \lim_{n\to\infty} \frac{600n+2}{5^n \ln 5 \ln 5}$$

$$= \lim_{n\to\infty} \frac{600}{5^n \ln^3 5}$$

$$= 0$$

That is, $0 \le \frac{f(n)}{g(n)}$ and $\exists n_0 > 0, \forall c > 0, \forall n \ge n_0, \frac{f(n)}{g(n)} < c$

That is,

$$\forall c > 0, \underbrace{\forall n \ge n_0}_{\exists n_0 > 0,}, 0 \le \frac{f(n)}{g(n)} < c.$$

Let $n_0 = 1$, we have $\forall c > 0, n \ge n_0$, $0 \le f(n) < c \cdot g(n)$.

Following the little oh notation,

Thus, $100n^3 + n^2 = o(5^n)$.

2. (a) This is true.

$f(n) = O(g(n)) \Rightarrow \exists\, c_1>0, n_0>0$, for any $n \geq n_0$,
$0 \leq f(n) \leq c_1 \cdot g(n)$

$g(n) = o(h(n)) \Rightarrow \forall c_2>0, \exists\, n_1>0, \forall n > n_1$,
$0 \leq g(n) < c_2 \cdot h(n)$

$\Rightarrow 0 \leq c_1 \cdot g(n) < c_1 \cdot (c_2 \cdot h(n))$

Thus, $0 \leq f(n) \leq c_1 \cdot g(n) < c_1 \cdot c_2 \cdot h(n)$

$\exists c_1>0, n_0>0, n_1>0, \forall c_2>0, n>n_1, n>n_0$.

In other words, there exists constants $c' = c_1 \cdot c_2 > 0$, and $n_2 = \max\{n_0, n_1\}$. such that the last inequality is held. Follow the big-O notation, we have $f(n) = O(h(n))$.

(b) This is false.

Consider that $f(n) = n^2$, $g(n) = n$, $h(n) = n\log n$.

$f(n), g(n)$ satisfy with $f(n) = \Omega(g(n))$

$g(n), h(n)$ satisfy with $g(n) = O(h(n))$.

But $f(n) \neq \Theta(h(n))$.

(c). This is false.

Consider $a = \frac{1}{2}$.

And because $f(n) < g(n)$, $a^{f(n)} > a^{g(n)}$, which doesn't satisfy with $a^{f(n)} = o(a^{g(n)})$.

3.

(a). $T(n) = T(\frac{n}{4}) + n\log n.$

we apply the rescursive - tree method.

$(\frac{1}{4})^k \cdot n = 1 \Rightarrow k = \log_4 n$

$$T(n) = T(\frac{n}{4}) + n\log n$$

$$= T(\frac{1}{4^2}n) + \frac{1}{4}n\log\frac{1}{4}n + n\log n$$

$$\leq T(\frac{1}{4^2}n) + \frac{1}{4}n\log n + n\log n.$$

$$\leq T(\frac{1}{4^3}n) + \frac{1}{4^2}n\log n + \frac{1}{4}n\log n + n\log n$$

...

$$\leq T(1) + n\log n \sum_{i=0}^{k-1} (\frac{1}{4})^i$$

$$= T(1) + n\log n\, \Theta(1)$$

$$= \Theta(n\log n)$$

Use the fact that a geometric sum is Θ(largest term).

So, we obtain the asymptotic upper bound of $T(n) = O(n\log n)$

(b) $T(n) = 2T\left(\frac{3}{4}n\right) + n^2$.

We apply the iteration method.

$\left(\frac{3}{4}\right)^k \cdot n = 1$ $\therefore$ $k = \log_{\frac{4}{3}} n$.

$$
\begin{aligned}
T(n) &= 2T\left(\tfrac{3}{4}n\right) + n^2 \\
&= 2^2 T\left(\left(\tfrac{3}{4}\right)^2 n\right) + 2\left(\tfrac{3}{4}n\right)^2 + n^2. \\
&= 2^3 T\left(\left(\tfrac{3}{4}\right)^3 n\right) + 2^2 \cdot \left(\left(\tfrac{3}{4}\right)^2\right)^2 \cdot n^2 + 2\left(\tfrac{3}{4}\right)^2 n^2 + n^2. \\
&\ \ \cdots \\
&= 2^k T(1) + n^2 \sum_{i=0}^{k-1} 2^i \cdot \left(\left(\tfrac{3}{4}\right)^i\right)^2 \\
&= 2^{\log_{\frac{4}{3}} n} T(1) + n^2 \, \Theta\left(\frac{9^k}{8^k}\right). \\
&= n^{\log_{\frac{4}{3}} 2} T(1) + n^2 \Theta(1) \\
&= \Theta\left(n^{\log_{\frac{4}{3}} 2}\right).
\end{aligned}
$$

Use the fact that a geometric sum is Θ(largest term).

So we obtain the asymptotic upper bound of $T(n) = O\left(n^{\log_{\frac{4}{3}} 2}\right)$.

4.

(a) $\sum_{k=1}^{n} k^{\frac{9}{5}}$

upper bound: $\sum_{k=1}^{n} k^{\frac{9}{5}} \le \sum_{k=1}^{n} n^{\frac{9}{5}} = n \cdot n^{\frac{9}{5}} = n^{\frac{14}{5}}$

$= O(n^{\frac{14}{5}})$.

lower bound: $\sum_{k=1}^{n} k^{\frac{9}{5}} = \sum_{k=1}^{\frac{n}{2}-1} k^{\frac{9}{5}} + \sum_{k=\frac{n}{2}}^{n} k^{\frac{9}{5}}$ ①

$\geq \sum_{k=\frac{n}{2}}^{n} k^{\frac{9}{5}}$ ②

$\geq (n-\frac{n}{2}+1) \cdot (\frac{n}{2})^{\frac{9}{5}}$ ③

$\geq (n-(\frac{n}{2}+1)+1) \cdot (\frac{n}{2})^{\frac{9}{5}}$ ④

$\geq \frac{n}{2} \cdot (\frac{n}{2})^{\frac{9}{5}}$ ⑤.

$= (\frac{n}{2})^{\frac{14}{5}}$ ⑥

$= (\frac{1}{2})^{\frac{14}{5}} \cdot n^{\frac{14}{5}}$ ⑦

$= \Omega(n^{\frac{14}{5}})$ ⑧.

where the lower bound in ⑤ follows from the fact that there more than $\frac{n}{2}$ additive terms in ② and the smallest term is $(\frac{n}{2})^{\frac{9}{5}}$.

Thus, the ~~asymptotic~~ asymptotic upper bound and lower bound together imply $\sum_{k=1}^{n} k^{\frac{9}{5}} = \Theta(n^{\frac{14}{5}})$.

(b). $\sum_{k=1}^{n} \frac{k^6}{3^k}$.

lower bound:

First, we define $a_k = \frac{k^6}{3^k}$. Thus, $\sum_{k=1}^{n} \frac{k^6}{3^k} = \sum_{k=1}^{n} a_k$.

Now we have

$$\frac{a_{k+1}}{a_k} = \frac{\frac{(k+1)^6}{3^{k+1}}}{\frac{k^6}{3^k}} = \frac{1}{3}\left(\frac{k+1}{k}\right)^6 \leq \frac{1}{3}\cdot\left(\frac{11}{10}\right)^6 \approx 0.591 < 1.$$

when $k \geq 10$.

We assume $r = \frac{1}{3}\cdot\left(\frac{11}{10}\right)^6$ when $k \geq 10$.

We have $a_{k+1} \leq r \cdot a_k$

That is, $a_{k+1} \leq r\cdot a_k \leq r^2 \cdot a_{k-1} \leq r^3 \cdot a_{k-2} \leq \cdots \leq r^{k-9}\cdot a_{10}$

for $k \geq 10$.

Thus, $a_k \leq r^{k-10} a_{10}$.

when $k \geq 10$

$$\sum_{k=1}^{n} a_k = a_1 + a_2 + a_3 \cdots + a_9 + \sum_{k=10}^{n} a_k$$

$$= a_1 + a_2 \cdots + a_9 + \sum_{k=10}^{n} r^{k-10}\cdot a_{10}$$

$$= a_1 + a_2 \cdots + a_9 + a_{10}\sum_{k=10}^{n} r^{k-10}$$

$$= \Theta(1) + \Theta(a_{10})\cdot\Theta(1) = \Theta(1) + \Theta(1)\cdot\Theta(1) = \Theta(1).$$

~~$= \Theta(1) + \Theta(a_{10}) = \Theta(1) + \Theta(1) = \Theta(1)$~~

Use the fact that a geometric sum is Θ(largest term).

upper bound: $\sum_{k=1}^{n} a_k \geq a_1 = \frac{1}{3} = \Theta(1)$

Thus, $\sum_{k=1}^{n} \frac{k^6}{3^k} = \Theta(1)$.

5.

(a)

step 1. Let $w(i)$ be the minimum sum of all placement costs and access costs for servers $S_1, S_2 \dots S_i$. $(1 \le i \le n)$.
We assume that we place a copy of the file at server i.
So the placement cost is C_i.

If we want to ~~find a~~ search over the possible places to put the highest copy of the file before i, we need a optimal solution at the position j, $(0 \le j < i)$.

Then the cost for ~~position~~ all servers up to S_j is $W(j)$.

~~$W(j)$ is equal to C_j~~

Then $w(i)$ is equal to $\underbrace{C_i}_{\text{placement cost}} + \min\{W(j) + \underbrace{\text{access cost from server } j+1 \text{ to } i}_{(i-(j+1))}\}$

$$0+1+\dots+(i-(j+1)) = \frac{(i-j-1)\cdot(i-j)}{2}.$$

step 2.

In the optimal solution, we have

$$W(i) = C_i + \min_{0 \le j < i}\left\{W(j) + \frac{(i-j-1)(i-j)}{2}\right\}.$$

step 3.

The value of the optimal solution:

The value of $W(i)$ can be built up in order of increasing i, in time $O(i)$ for iteration i.

Thus $\sum_{i=1}^{n} O(i) = O(n^2)$. for $i > 0$.

for $i = 0$, we have $W(0) = C_0 = \Theta(1)$.

Thus $$T_n = \begin{cases} \Theta(1) & n = 0 \\ O(n^2). & n > 0. \end{cases}$$

step 4

Because this problem is to find an optimal solution, it does not require to construct the optimal solution. step 4 is omitted.

(b).

step 1.

~~Let C(i) be the~~

Let $C(i)$ be the minimum penalty ~~during all travel days~~ in which hotel (i) to break the travel up and we assume that the last stop is hotel i.

So $C(0)=0$ because we start from the road at 0 mile.

We also need to consider all possible hotels j, $(0 \le j < i)$ which we stopped before the ith hotel.

Thus, the penalty for the distance from hotel j to hotel i is $(200-(a_i-a_j))^2$

step 2.

In the optimal solution, we have

$$C(i) = \min_{0 \le j < i} \{ C(j) + (200-(a_i-a_j))^2 \}.$$

step 3:

For finding minimum cost of $C(i)$, it takes linear time to calculate each value of $C(j)$ $(0 \le j < i)$ because in every recurrence, we consider the optimal route between $a_{i-1}-a_j$, $a_{i-2}-a_j$ ... till a_0-a_j.

Thus $\sum_{i=1}^{n} O(i) = O(n^2)$

$$T_n = \begin{cases} \Theta(1) & n=0 \\ O(n^2) & n>0 \end{cases}$$

step 4:

Because this problem is to find an optimal Solution, it doesn't require to construct the optimal solution. step 4 is omitted.

Bonus

BQ 1.

$$T(n) = \begin{cases} \Theta(1) & n < 100 \\ \sum_{i=100}^{n} \sum_{j=0}^{-\log_{\frac{5}{13}} i} \Theta\left([(\frac{5}{13})^j \cdot i]^2 \cdot \log^2[(\frac{5}{13})^j \cdot i] \right) & \text{otherwise} \end{cases}$$

Because

$$\sum_{j=0}^{-\log_{\frac{5}{13}} i} \Theta\left([(\frac{5}{13})^j \cdot i]^2 \cdot \log^2[(\frac{5}{13})^j \cdot i] \right)$$

$$\leq \sum_{j=0}^{-\log_{\frac{5}{13}} i} O\left([(\frac{5}{13})^j \cdot i]^2 \cdot \log^2 i \right) = O(i^2 \log^2 i).$$

Thus

$$T(n) = \Theta(1) + \sum_{i=100}^{n} O(i^2 \log^2 i) = O(n^3 \log^2 n).$$