Date

Now

$A_1(A_2 \times A_3)$

↓

→ $A_1 \times B$
20×10.

→ C
30×10.

# of products for C_j

① $20 \times 40 \times 10 = 12000$

② $30 \times 20 \times 10 = + \ 6000$

$ \qquad\qquad\qquad\quad 24000$

$\therefore A_1(A_2 A_3)$
is more efficient

—— x ——

(1) Big-O estimate for the no. of operations....

$t := 0$ — 1
for $i := 1$ to 3 — 2
  for $j := 1$ to 4 — 3
    $t := t + ij$ — 4

→ Only line no. 4 has operation of sum or product.
→ It contains one product b/w i & j which is then added to t
→ ∴ 2 operations per value of i & j
→ i takes values from 1–3 ; j from 1–4.

Total no. of operation = product of the no. of values for i and j and the no. of operations in line 4.

$\therefore \quad 4 \times 3 \times \cancel{2} = 12$

$12 \times 2 = 24$ $\qquad \therefore O(1)$

Date

2.
$t := 0$
for $i := 1$ to n
for $j := 1$ to n
$t = t + i + j$.

$n \times n \times 2 \Rightarrow 2n^2 \quad \therefore O(n^2)$.

3. imp! *
$m := 0$ — ①
for $i := 1$ to n — ②
for $j := i+1$ to n. — ③
$m := \max(a_i a_j, m)$ — ④

There is only one line that contains multiplication/ ~~n x n / x n x n~~ comparison. b/w the product and m. Thus this line contain 2 operations per value j and i.

i goes upto n
j goes upto n-1 as it starts from i+1.

$\therefore$

$n \times (n-1) \times 2 = 2n(n-1)$.
$2n^2 - 2n$
$\therefore O(n^2)$ Ans!

4.
$i := 1$
$t := 0$
while $i \leq n$ → it would run uptill last power of 2 that is smaller than n.
$t := t + 1$
$i = 2i$

$2^x \leq n$

Date

also

$x = \log_2 (n)$.

$\therefore$ i takes values from $1, 2, 3, 4 \dots \log_2(n)$.

→ which are $[\log_2(n) + 1]$ values. ⟶ (imp!)

$\therefore (\log_2(n) + 1) \times 2 = 2\log_2(n) + 2$.

$\therefore O(\log n)$.

5.

procedure minimum (a_1, a_2, a_n : natural no with $n \geq 1$).
  min := a_1
  for i := 2 to n
    if a_i < min then min := a_i
  return min.

$(n-1)1 \Rightarrow n-1$ comparisons!

6. algorithm to put the first four no.s of a list in increasing order.

procedure sort first four (a_1, a_2, a_n : real no.s with $n \geq 2$).

for i := 2 to min (4, n)
  for j := 1 to i-1.
    if $a_i < a_j$ them
      temp = a_i
      for k := j to i-1

$a_{k+1} := a_k$.

a_j = temp.

return $a_1, a_2 \dots a_n$.