1

Code:

```
sum = 0;
for (i=1; i <=n; i++)
    sum += n;
```

sum += n; → dominating instruction (i)

values of i	(i) executed
1	1
2	1
3	1
⋮	⋮
n	1

sum = n

$O(n)$

2

```
sum1 = 0;
for (i=1; i<=n; i++)
    for (j=1; j<=n; j++)
        sum1++; ★
```

value of i	☆ executes
1	n
2	n
⋮	⋮
n	n

sum: $n \times (n)$ → total values of i

$= n^2$

$\boxed{O(n^2)}$

(3)

```
sum2 = 0;
for (i=1; i<=n; i++)
   for (j=1; j<=i; j++)
      sum2++; ☆
```

values of i	☆ executes
1	1
2	2
3	3
⋮	⋮
n	n

sum $= 1+2+3+\cdots+n = \dfrac{n(n+1)}{2}$

$= \dfrac{n^2}{2} + \dfrac{n}{2}$

$O(n^2)$ → dominating term.

(4)

```
sum = 0;
 for (j=1; j<=n; j++)
    for (i=1; i<=j; i++)
       sum++;
 for (k=0; k<n; k++)
    A[k] = k;
```

sum++; → $O(n^2)$

A[k] = k; → $O(n)$

$O(n^2) + O(n) \Rightarrow O(n^2)$.

(5)

```
sum1 = 0;
for (k=1; k<=n; k*=2)
   for (j=1; j<=n; j++)
      sum1++;  ☆
```

values of i	☆ executed
$1 = 2^0$	n
$2 = 2^1$	n
$4 = 2^2$	n
$8 = 2^3$	n
$16 = 2^4$	n
⋮ ⋮	
$n = 2^k$	n

$2^k = n$

$k = \log_2 n$

$k \times n = (\log_2 n) \times n$

so

total values of i, $k = \log_2 n$

$= O(n \log n)$

⑥

```
sum2 = 0;
for (k=1; k<=n; k*=2)
   for (j=1; j<=k; j++)
      sum2++; ☆
```

Values of i

$1 = 2^0$

$2 = 2^1$

$4 = 2^2$

$\vdots$

$n = 2^k$

$2^k = n$

$\boxed{k = \log_2 n}$

☆ executed

$1 = 2^0$

$2 = 2^1$

$4 = 2^2$

$\vdots$

$n = 2^k$

$$\text{total} = 2^0 + 2^1 + 2^2 + \cdots + 2^{\log_2 n}$$

$$= 1 + 2 + 2^2 + \cdots + 2^{\log n}$$

number of terms $= \log n$

geometric series

$$\text{sum} = \frac{a(r^k - 1)}{r - 1}$$

a = first term

r = ratio

k = total terms

$$= \frac{1 \times (2^k - 1)}{2 - 1}$$

$$= 2^k - 1$$

$$= 2^{\log_2 n} - 1$$

$$= n - 1$$

$2^{\log_2 n} = x$

$\log_2 x = \log_2 n$

$x = n$

Complexity $= O(n)$