**Q1**.

double sum\_skip7 (double array[], int n)

//n: size of the array. Assume n is divisible by 7, i.e., n=7\*k, where k is a

positive integer

{

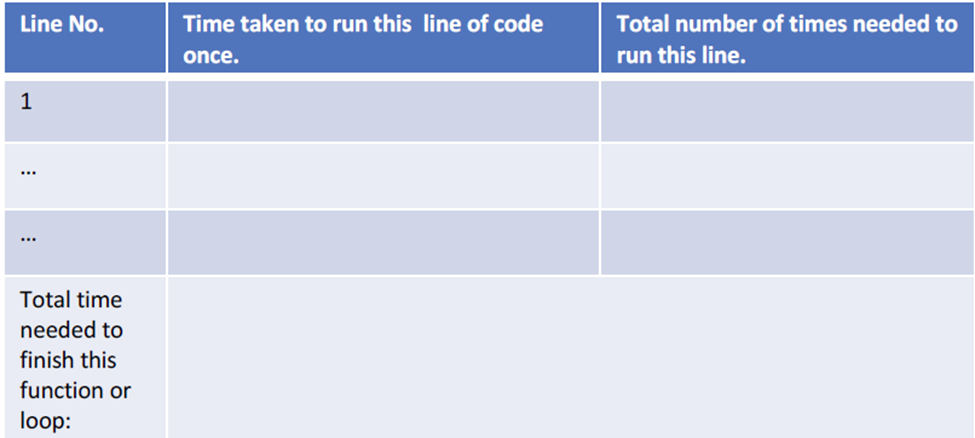
double sum=0;

for (int i=0; i<n; i=i+7)

sum = sum + array[ i ];

return sum;

}



C1 1

2 C2 n/7 +1

3 C3 n/7

4 C4 1

T = C1+C2(n/7+1)+C3(n/7)+C4

= n(C2+C3)/2+(C1+C2+C3)

**Q2**.

double sum\_exponentials(int n)

//n is a power of 5, i.e., n=5k or k=log3n, where k is a positive integer

{

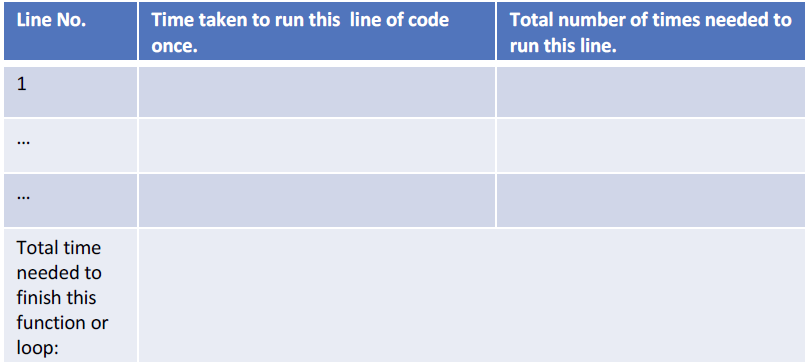
int sum=0;

for (int i=1; i<n; i=i\*5)

sum = sum + i;

return sum;

}



C1 1

2 C2 log5n+1

3 C3 log5n

4 C4 1

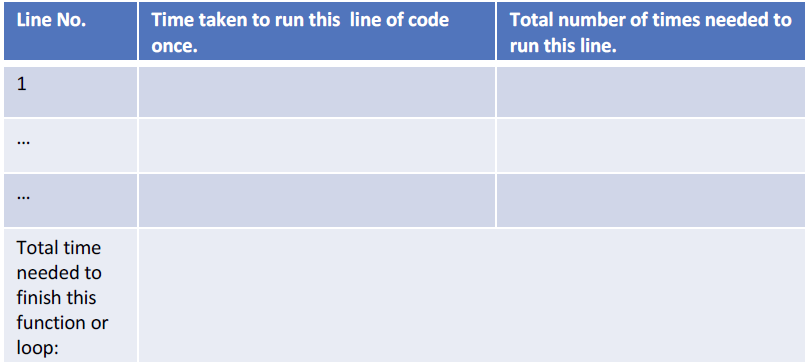
T = C1+C2(log5n+1)+C3log5n+C4

T = C1+C2+C4+log5n(C2+C3)

**Q3.**

for (int i=0; i<n; i++)

for (int j=i; j<=n; j++)

cout << i << “,” << j <<endl;

C1 n+1

C2 n(n+1)

C3 n^2

T = C1(n+1)+C2(n(n+1))+C3(n^2)

T = C1n+C1+C2n^2+C2n+C3(n^2)

T = n^2(C3+C2)+n(C1+C2)+C1