2022-CS-18

Question 1

(Estimated Time 20 mins) **(Each proof contains 5 points)**

Prove all the thetas (using halving technique or any way you like) . Both find out its lower(Omega) and upper(Big O) bound by the same function so that you can declare it to be theta of the given function.

1. $1^2 + 2^2 + 3^2 + 4^2 + \ldots + N^2$ -- $\Theta(N^3)$

Upper bound:-
$\Rightarrow 1^2+2^2+3^2+\ldots+N^2$
$\leq N^2+N^2+N^2+\ldots+N^2$
$\leq N^2 \times N \leq N^3$
$\Rightarrow O(N^3)$

Lower bound:-
$\Rightarrow 1^2+2^2+3^2+\ldots+(\frac{N}{2})^2+(\frac{N}{2}+1)^2+\ldots+N^2$
$\geq (\frac{N}{2})^2+(\frac{N}{2}+1)^2+\ldots+N^2$
$\geq (\frac{N}{2})^2+(\frac{N}{2})^2+\ldots+(\frac{N}{2})^2 \geq (\frac{N}{2})^2(\frac{N}{2})$
$\Rightarrow \Omega(N^3/8)$

2. $1 + 2 + 3 + 4 + \ldots + N^2$ -- $\Theta(N^4)$

Upper bound:-
$\Rightarrow 1+2+3+\ldots+N^2$
$\leq N^2+N^2+\ldots+N^2$
$\leq N^2 \times N^2 \leq N^4$
$\Rightarrow O(N^4)$

Lower bound:-
$\Rightarrow 1+2+3+\ldots+N^2+(N^2+1)+\ldots+N^2$
$\geq \frac{N^2}{2}+(N^2)^2+\ldots+(\frac{N^2}{2})^2$
$\geq N^4/4$
$\Rightarrow \Omega(N^4/4)$

3. $1 + 3 + 5 + 7 + 9 + \ldots + (2N + 1)$ -- $\Theta(N^2)$

Upper bound:-
$\Rightarrow 1+3+5+7+\ldots+(2N+1)$
$\leq (2N+1)+(2N+1)+\ldots+(2N+1)$
$\leq 4N^2+4N+1$
$\Rightarrow O(N^2)$

Lower bound:-
$\Rightarrow 1+3+5+\ldots+(\frac{2N+1}{2})+(\frac{2N+1}{2}+1)+\ldots+(2(2N+1))$
$\geq (2(N+1)+(2N+1)+\ldots+(2N+1)$
$\geq (N/4)(N/4)$
$\Rightarrow \Omega(N^2/16)$

4. $2 + 4 + 6 + 8 + \ldots + 2N$ -- $\Theta(N^2)$

Upper bound:-
$\Rightarrow 2+4+6+\ldots+2N$
$\leq 2N+2N+\ldots+2N$
$\leq 4N^2 \Rightarrow O(N^2)$

Lower bound:-
$\Rightarrow 2+4+6+\ldots+\frac{2N}{2}+(\frac{2N}{2}+1)+\ldots+2N$
$\geq \frac{N}{2}+\frac{N}{2}+\ldots+\frac{N}{2}$
$\geq \frac{N}{2} \times \frac{N}{2} \geq \frac{N^2}{4} \Rightarrow \Omega(N^2/4)$

5. $1 + 2 + 3 + 4 + \ldots + (N/2)$ -- $\Theta(N^2)$

Upper bound:-
$\Rightarrow 1+2+3+\ldots+N/2$
$\leq N/2+N/2+\ldots+N/2$
$\leq N/2 \times N/2 \Rightarrow O(N^2/4)$

Lower bound:-
$\Rightarrow 1+2+3+\ldots+\frac{N}{4}+(\frac{N}{4}+1)+\ldots+\frac{N}{2}$
$\geq N/4+(N/4)^2+\ldots+(N/4)$
$\geq (N/4)(N/4) \Rightarrow \Omega(N^2/16)$

6. $1+2+4+8+16+\ldots+N^2$ ---- $\Theta(N^2)$

Upper bound:-
$\Rightarrow 1+2+4+\ldots N^2$
$\leq N^2+N^2+\ldots+N^2$
$\leq N^2 \times \log_2^2 N$
$\leq N^2$
$\Rightarrow O(N^2)$

Lower bound:-
$\Rightarrow 1+2+4+8+\ldots+(N^2)+(\frac{N^2}{2}+1)+\ldots+N^2$
$\geq \frac{N^2}{2}+\frac{N^2}{2}+\frac{N^2}{2}+\ldots+\frac{N^2}{2}$
$\geq \frac{N^2}{2}(\frac{\log_2 N}{2}) \geq \frac{N^2}{4}\log_2 N$
$\Rightarrow \frac{N^2}{4}$

Question 2

(35*2 = 70 Points)

(Estimated Time 100 mins)

What is the algorithm's complexity of the following piece of code - Sample Solution is in RED.

```
int Sum=0;                  // O(1) Time
for(int i=0; i<N; i++)      //(1+1+1+...+1 - - - N Times =O(N)
 for(int j=0; j<N; j++)  Sum++;
 // (1+1+1+...+1) + (1+1 +... +1)+... + (1+1 +... +1) added N times
    // N        +     N      +... +     N       = O(N²)
Overall Complexity : O(1) +O(N) + O(N²) + O(N²) = O(N²)
```

2) What is the algorithm's complexity of the following piece of code

```
int Sum=0;                  // O(1)
for(int i=0; i<N; i++)      // (1+1+...+N) = O(N)
 Sum++;                     // O(N)
for(int j=0; j<N; j++)      // (1+1+...+N) = O(N)
 Sum++;                     // O(N)
```

Overall Complexity : O(N)

3)

What is the algorithm's complexity of the following piece of code

```
int Sum=0;                  // O(1)
for(int i=0; i<N; i++)      // (1+1+...+N) = O(N)
 for(int j=0; j<N; j++)     // (1+1+...+N) = O(N)
  for(int k=0; k<N; k++)    // (1+1+...+N) = O(N)
   Sum++;                   // O(N)

for(int i=0; i<N; i++)      // O(N)
 for(int j=0; j<N; j++)     // O(N)
  for(int k=0; k<N; k++)    // O(N)
   Sum++;                   // O(N)
```

Overall Complexity :: $O(N^3)$

4)

What is the algorithm's complexity of the following piece of code

```
int Sum=0;                  // O(1)
for(int i=0; i<N; i++)      // (1+1+...+N) = O(N)
 Sum++;                     // O(N)
for(int j=0; j<N; j++)      // (1+1+...+N) = O(N)
 Sum++;                     // O(N)
for(int k=0; k<N; k++)      // (1+1+...+N) = O(N)
 Sum++;                     // O(N)
for(int m=0; m<N; m++)      // (1+1+...+N) = O(N)
 Sum++;                     // O(N)
for(int n=0; n<N; n++)      // (1+1+...+N) = O(N)
 Sum++;                     // O(N)
for(int p=0; p<N; p++)      // (1+1+...+N) = O(N)
 Sum++;                     // O(N)
```

Overall Complexity : O(N)

5)

```
int Sum=0;                  // O(1)
for(int i=0; i<N; i++)      // (1+1+...+N) = O(N)
 for(int j=0; j<i; j++)     // (1+1+...+N) = O(N)
  for(int k=0; k<j; k++)    // (1+1+...+N) = O(N)
   Sum++;                   // O(N)
```

Overall Complexity : O(N)

6

```
int Sum=0;                  // O(1)
for(int i=0; i<N; i+=2)     // O(N/2)
 for(int j=0; j<i; j+=2)    // O(N/2)
  for(int k=0; k<j; k+=2)   // O(N/2)
   Sum++;                   // O(N)
```

Overall Complexity: $O(N^3)$

7

```
int Sum=0;                  // O(1)
for(int i=1; i<N; i*=2)     // O(log N)
 for(int j=1; j<N; j*=2)    // O(log N)
  Sum++;                    // O(N)
```

Overall Complexity : $O(\log_2^2 N)$

8

```
int Sum=0;                  // O(1)
for(int i=1; i<N; i*=2)     // O(log N)
 Sum++;                     // O(N)
for(int j=1; j<N; j*=2)     // O(log N)
 Sum++;                     // O(N)
```

Overall Complexity: $O(\log_2 N)$

9

```
for(int i=1; i<=N*N; i+=2)
  for(int j=1; j<N*N; j*=2)
    Sum++;
```

// $O(N^2/2)$
// $O(\log_2 N^2)$
// $O(N^2)$

Overall Complexity: $O(N^2 \log_2 N)$

10

```
for(int i=1; i<=N*N; i+=2)
  Sum++;

for(int j=1; j<N*N; j*=2)
    Sum++;
```

// $O(N^2/2)$
// $O(N^2)$
// $O(\log_2 N^2)$
// $O(N^2)$

Overall Complexity: $O(N^2)$

11

```
for(int i=1; i<=N*N; i*=2)
  for(int j=1; j<N*N; j*=2)
    Sum++;
```

// $O(\log_2 N^2)$
// $O(\log_2 N^2)$
// $O(N^2)$

Overall Complexity: $O(\log_2 N^4)$

12

```
for(int i=1; i<=N*N; i*=2)
    Sum++;

for(int j=1; j<N*N; j*=2)
    Sum++;
```

// $O(\log_2 N^2)$
// $O(N^2)$
// $O(\log_2 N^2)$
// $O(N^2)$

Overall Complexity: $O(\log_2 N)$

13

```
int Sum=0;
for(int i=1; i<=N; i*=2)
 for(int j=1; j<=N; j*=2)
   for(int k=1; k<=N; k*=2)
     Sum++;
```

// $O(1)$
// $O(\log_2 N)$
// $O(\log_2 N)$
// $O(\log_2 N)$
// $O(N)$

Overall Complexity: $O(\log_2^3 N)$

14

```
int Sum=0;
for(int i=1; i<=N; i*=2)
Sum++;
for(int j=1; j<=N; j*=2)
    Sum++;
for(int k=1; k<=N; k*=2)
    Sum++;
```

// $O(1)$
// $O(\log_2 N)$
// $O(N)$
// $O(\log_2 N)$
// $O(N)$
// $O(\log_2 N)$
// $O(N)$

Overall Complexity: $O(\log_2 N)$

15

```
int sum,i,j;
sum = 0;
for (i=1;i<n;i=i*2)
{
  for (j=0;j<n;++j)
  {
      sum++;
  }
}
```

// $O(1)$
// $O(1)$
// $O(\log_2 N)$
// $O(N)$
// $O(N)$

Overall Complexity: $O(N \log_2 N)$

16

BE CAREFUL GEOMETRIC SERIES

```
int sum,i,j;
sum = 0;
for (i=1; i<n; i=i*2)
{
  for (j=0; j <i ; ++j)
  {
      sum++;
  }
}
```

// $O(1)$
// $O(1)$
// $O(\log_2 N) = 1+2+4+8+\ldots$
// $O(N)$
// $O(N)$

Overall Complexity: $O(N \log_2 N)$

17

BE CAREFUL GEOMETRIC SERIES

```
int sum,i,j;
sum = 0;
for (i=1; i<n; i=i*5)
{
  for (j=0; j<i; j+=2)
  {
      sum++;
  }
}
```

// $O(1)$
// $O(1)$
// $O(\log_5 N) = 1+5+25+\ldots$
// $O(N/2)$
// $O(N)$

Overall Complexity: $O(N \log_5 N)$

18

```
int sum,i,j;
sum = 0;
for (i=1; i<n;  i=i*4)
{
  for (j=0 ; j<n ; j+=3)
  {
      sum++;
  }
}
```

// $O(1)$
// $O(1)$
// $O(\log_4 N) = 1+4+16+64+\ldots$
// $O(N)$
// $O(N)$

Overall Complexity: $O(N \log_4 N)$

19 What will be the output (the value of Sum) of the program asymptotically in BIG-O notation, I am not asking here the complexity of loop rather the asymptotic bound on the value of Sum:

```
int Sum = 0;          // O(1)
for(int i=1; i<=n; i+=1)   // O(N)
{
 Sum+=i;              // O(N)
}
cout<<Sum<<endl;      // O(N)
```

20 What will be the output(the value of Sum) of the program asymptotically in BIG-O notation:

```
int Sum = 0;          // O(1)
for(int i=1; i<=n; i*=2)   // O(log2N)
{
 Sum+=i;              // 2^k - 1 = 2^(log2N) - 1
}
cout<<Sum<<endl;      // O(N)
```

21 What is the time complexity of the algorithm:

```
int Sum = 0;          // O(1)
for(int i=1; i<=n; i+=1)   // O(N) = 1+1+...+N
{
 for(int j=1; j<=i; j++)   // O(N^2)
 {
    Sum++;            // O(N)
 }
}
cout<<Sum<<endl;
```

O(N) + O(N^2) = O(N^2)

22 What is the time complexity of the algorithm:

```
int Sum = 0;          // O(1)
for(int i=1; i<n; i*=2)    // O(log2N)
{
 for(int j=1; j<=i; j++)   // O(N)
 {
    Sum++;
 }
}
cout<<Sum<<endl;      // O(NlogN)
```

40* Complexity of primeNumber function.

```
int sqrt(int N)
{
 int d;                   // O(1)
 for(d=0; d*d<=N; d++) { }   // O(√N)
 return d-1;              // O(1)
}
bool primeNumber(int n)
{
        bool isPrime = true;     // O(1)
        int lmt = (sqrt(n));     // O(√N)
        for (int d=2; d <=lmt ;++d)   // O(√N)
        {
                if (n%d==0)
                        return false;   // O(1)
        }
        return true;             // O(√N)
}
```

41* Complexity of primeNumber function.

```
int sqrt(int N)
{
 int d;                   // O(1)
 for(d=0; d*d<=N; d++){}  // O(√N)
 return d-1;
}
bool primeNumber(int n)
{
        bool isPrime = true;     // O(1)
        for (int d=2; d <= sqrt(n) ;++d)   // O(N)
        {
                if (n%d==0)
                        return false;
        }
        return true;
}                                // O(N)
```

23 What is the time complexity of the algorithm:

```
int f1(int n)
{
        int K=0;                          // O(1)
        for(int j=0; j*j<=n*n; j++) K++;  // O(N)
        return K;                         // O(1)
}
int main()
{
        int Sum = 0, n;                   // O(1)
        cin>>n;
        for(int i=1; i<=f1(n); i+=1)      // O(N^2)
          for(int j=1; j<=i; j++)  Sum++; // N^2(N^2+1)/2
        cout<<Sum<<endl;
}
```

$\Rightarrow O(N^4)$

24

What is the time complexity of the algorithm:

```
int f1(int n)
{
        int K=0;                          // O(1)
        for(int j=1; j*j<=n; j*=2)   K++; // O(log2 √N)
        return K;                         // O(1)
}
int main()
{
        int Sum = 0;                      // O(1)
        int n;
        cin>>n;                           // O(1)
        for(int i=1; i<=f1(n); i+=1)      // O(N log2 √N)
            for(int j=1; j<=i; j++)  Sum++; // O(N^2 log2 √N)
        cout<<Sum<<endl;
}
```

$\Rightarrow O(N^2 \log_2 \sqrt{N})$

25

What is the time complexity of the algorithm:

```
int f1(int n)
{       int K=0;                          // O(1)
  for(int j=1; j*j<=n; j++)               // O(√N)
    K++;                                  // O(√N)
 return K*K;                              // O(1)
}
int main()
{
int Sum = 0;                              // O(1)
int n;
cin>>n;                                   // O(1)
int Terminator = f1(n);
for(int i=1; i<= Terminator; i+=1)        // O(N)
{
 for(int j=1; j<=i; j++)                  // N(N-1)/2  (N^2)
 {
    Sum++;
 }
}
cout<<Sum<<endl;                          // O(1)
}
```

$\Rightarrow O(N^2)$

26

What is the time complexity of the algorithm:

```
int f1(int n)
{       int K=0;                          // O(1)
        for(int j=0; j*j<=n; j++)         // O(√N)
          K++;                            // O(√N)
        return K;                         // O(1)
}
int main()
{
        int Sum = 0;                      // O(1)
        int n;
        cin>>n;                           // O(1)
        int Terminator = f1(n);           // O(√N)
        for(int i=1; i<=Terminator; i+=1) // O(√N)
        {
          for(int j=1; j<=i; j++)         // O(N)
          {
            Sum++;                        // O(N)
          }
        }
        cout<<Sum<<endl;                  // O(1)
}
```

$\Rightarrow O(N)$

27

```
for (i=1;i<n;i=i*4)                       // O(log4 N)
{
      cout << i;
       for (j=0;j<n;j=j+2)                // N/2 log4 N
       {
              cout << j;
              sum++
       }
       cout << sum;
}
```

$\Rightarrow N \log_4 N$

28

```
for (i=1;i<n;i=i*4)                       // O(log4 N)
{
      cout << i;                          // 2^k - 1 = 2^(log4 N) - 1
      for (j=0;j<i; j=j+2)
      {
              cout << j;
              sum++
      }
      cout << sum;                        // O(1)
}
```

$\Rightarrow O(\log_4^2 N)$

29

```
for (i=1;i<=n*n;++i)
{
    cout << i;
      Sum=0;
      for (j=1; j<=i; ++j)
      {
            Sum++;
            cout << i;
      }
      cout << Sum;
}
```

// $O(N^2)$
// $O(N^2)$
// $O(N^2)$
// $O(N^2)$
// $O(1)$
$\Rightarrow O(N^2)$

30

```
for (i=1;i<=n*n*n;++i)
{
    cout << i;
      Sum=0;
      for (j=1; j<=i; ++j)
      {
            Sum++;
            cout << i;
      }
      cout << Sum;
}
```

// $O(N^3)$
// $O(N^3)$
// $O(N^3)$
// $O(N^3)$
// $O(1)$
$\Rightarrow O(N^3)$

31

```
for (i=1;i<=n*n*n; i*=2)
{
    cout << i;
      Sum=0;
      for (j=1;j<=i; j++)
      {
            Sum++;
            cout << i;
      }
      cout << Sum;
}
```

// $O(\log_2 N)$
// $O(\log_2 N)$
// $O(\log_2 N)$
// $O(\log_2 N)$
// $O(1)$
$\Rightarrow O(\log_2^2 N)$

32

```
for (i=1;i<=n*n*n; i*=2)
{
    cout << i;
      Sum=0;
      for (j=1;j<=n; j++)
      {
            Sum++;
            cout << i;
      }
      for (k=1;k<=n; k++)
      {
            Sum++;
            cout << i;
      }
      cout << Sum;
}
```

// $O(\log_2 N)$
// $O(\log_2 N)$
// $O(N \log_2 N)$
// $O(N \log_2 N)$
// $O(N \log_2 N)$
$\Rightarrow O(N \log_2 N)$

33

```
for (i=1;i<=n*n*n; i*=2)
{
    cout << i;
      Sum=0;
      for (j=1;j<=i; j++)
      {
            Sum++;
            cout << i;
      }

      for (j=1;j<=n; j*=2)
      {
            Sum++;
            cout << i;
      }

      cout << Sum;
}
```

// $O(\log_2 N)$
// $O(\log_2 N)$
// $O(\log_2^2 N)$
// $O(\log_2^2 N)$
// $O(\log_2^2 N)$
// $O(\log_2^2 N)$
$\Rightarrow O(\log_2^2 N)$

34

```
for (i=1;i<=n*n*n; i*=2)
{
    cout << i;
      Sum=0;
      for (j=1;j<=i; j++)
      {
            Sum++;
            cout << i;
      }

      for (j=1;j<=n; j++)
      {
            Sum++;
            cout << i;
      }

      cout << Sum;
}
```

// $O(\log_2 N)$
// $O(\log_2 N)$
// $O(\log_2^2 N)$
// $O(N \log_2^2 N)$
// $O(N \log_2 N)$
// $O(N \log_2 N)$
$\Rightarrow O(\log_2 N)$

35-36

```
for (int i=1; i <= n ; i = i * 2)
{
        for ( j = 1 ; j <= i ; j = j * 2)
                cout<<"*";
}
```
// $O(\log_2 N)$ // $O(\log_2 N)$ $\Rightarrow O(\log_2^2 N)$

```
for (int i=1; i <= n ; i = i * 2)
        for ( j = 1 ; j <= i ; j = j * 2)
                cout<<"*";

for (int i=1; i <= n ; i = i * 2)
        for ( j = 1 ; j <= i ; j = j * 2)
                cout<<"*";
```
// $O(\log_2 N)$ // $O(\log_2 N)$ // $O(\log_2 N)$ // $O(\log_2 N)$ $\Rightarrow O(\log_2^2 N)$

37

```
for (i=0; i<n; i=i+3)
{
        cout << i;
        for (j=1; j<n; j=j*3)
        {
                cout << j;
                sum++
        }
        for (k=1;k<n;k=k*3)
        {
                cout << j;
                sum++
        }

        cout << sum;
}
```
// $O(N/3)$ // $O(N/3)$ // $O(\log_3 N)$ // $O(\log_3 N)$ // $O(\log_3 N)$ // $O(1)$ $\Rightarrow O(N \log_3 N)$

38

```
for (int i=1; i <= n ; i = i * 2)
{
        for ( j = 1 ; j <= i ; j = j * 2)
        {
                cout<<"*";
        }
}

for(int i=0; i<=N; i++)
{
 Sum++;
}
```
// $O(\log_2 N)$ // $O(\log_2 N)$ // $O(\log_2 N)$ // $O(N)$ // $O(N)$ $\Rightarrow O(N)$

39

```
for (i=0; i<n; i=i+3)
{
        cout << i;
        for (j=1; j<n; j=j*3)
        {
                sum++
        }
}
for (k=1;k<n;k=k*3)
{
        cout << j;
        sum++
}
cout << sum;
```
// $O(N/3)$ // $O(N/3)$ // $O(\log_3 N)$ // $O(\log_3 N)$ // $O(\log_3 N)$ // $O(1)$ $\Rightarrow O(N \log_3 N)$

Question 3 Analyze the complexity Θ of the following functions in terms of N. 1*5+5 = 10 points

```
int f1(int N)
{
int Count = 0;
for(int i = 1; i<=N ; i*= 2)
  for(int j=1; j<= i ; j++)
     Count++;
return Count;
}
```
// $O(1)$ // $O(\log_2 N)$ // $(2^k - 1)$ $\Rightarrow O(N)$

```
int f2(int N)
{
  int Count=0;
  int C = f1(N);
  for(int i=0; i<C; i++)
      Count++;
  return Count;
}
```
// $O(1)$ // $O(N)$ // $O(N)$ $\Rightarrow O(N)$

```
int f5(int N)
{
  int Count=0;
  for(int i=0; i<sqrt(f1(N) * f1(N)); i++)
     Count++;
  return Count;
}
```
// $O(1)$ // $O(N^2 \times N)$ $\Rightarrow O(N^3)$

```
int f3(int N)
{
  int Count=0;
  int C = sqrt(f1(N));
  for(int i=1; i<C; i*=2)
     Count++;
  return Count;
}
```
// $O(1)$ // $O(N)$ // $O(\log_2 N)$ $\Rightarrow O(\log_2 N)$

```
int f4(int N)
{
  int Count=0;
  for(int i=0; i<f1(N) * f1(N); i++)
        Count++;
  return Count;
}
```
// $O(1)$ // $O(N^2 \times N)$ $\Rightarrow O(N^3)$

```
Int Sum = 0;
int f6(int N)
{
  if(N==1)
    return 1;

  Sum +=f1(N);   Sum +=f2(N);
  Sum +=f3(N); Sum +=f4(N);   Sum +=f5(N);
  return Sum;
}
```
// $O(1)$ // $O(1)$ // $O(N)$ $\Rightarrow O(N)$